



EPS

Escola Politècnica
Superior

Projecte/Treball Fi de Carrera

ENG. TÈCN. INFORMÀTICA DE GESTIÓ. Pla 2001

DESENVOLUPAMENT D'UNA APLICACIÓ WEB PER A LA GESTIÓ D'UN TALLER MECÀNIC

Memòria

Alumne: Anna Vilar Ribas

Director/Tutor: Gustavo Ariel Patow
Departament: Informàtica i Matemàtica Aplicada
Àrea: LSI

Convocatòria: Setembre 2014

Contingut

1. INTRODUCCIÓ, MOTIVACIONS, PROPÒSIT I OBJECTIUS DEL PROJECTE.....	7
1.1. INTRODUCCIÓ	7
1.2. MOTIVACIONS PERSONALS	8
1.3. PROPÒSIT I OBJECTIUS	8
1.4. ESTRUCTURA DEL DOCUMENT	8
2. ESTUDI DE VIABILITAT	10
2.1. RECURSOS NECESSARIS PER DESENVOLUPAR EL PROJECTE	10
2.2. AVALUACIÓ PRÈVIA DE COSTOS I MEDIS	11
2.2.1. <i>Estudi de viabilitat tecnològica</i>	11
2.2.2. <i>Estudi de viabilitat econòmica</i>	11
2.2.2.1 Despesa de recursos humans.....	11
2.2.2.2 Despesa de maquinària	12
3. METODOLOGIA.....	13
3.1. METODOLOGIA DE MÈTRICA 3.....	13
4. PLANIFICACIÓ.....	16
4.1. PLA DE TREBALL I TASQUES PLANIFICADES.....	16
4.1.1 <i>Planificació del Sistema d'Informació</i>	17
4.1.2 <i>Desenvolupament dels Sistemes d'Informació</i>	17
4.1.2.1 Estudi de Viabilitat del Sistema	17
4.1.2.2. Anàlisi del Sistema d'Informació	17
4.1.2.3 Disseny del Sistema d'Informació.....	18
4.1.2.4 Construcció del Sistema d'Informació	18
4.1.2.5 Implantació i Acceptació del Sistema	18
4.1.3 <i>Manteniment dels Sistema d'Informació</i>	19
4.2 TEMPS ESTIMAT	19
5. MARC DE TREBALL I CONCEPTES PREVIS	21
5.1.MARC DE TREBALL	21
5.2.CONCEPTES PREVIS	23
5.2.1. <i>Patrons de disseny</i>	23
5.2.1.1.MVC (Model Vista Controlador).....	23
5.2.2. <i>Marc legal</i>	25
5.2.2.1.Requisits del pressupost	26
5.2.2.2.Requisits del resguard de dipòsit/recepció	26
5.2.2.3.Requisits de la factura	26
5.2.2.4.Requisits comptables.....	27
5.2.2.5.Protecció de dades dels clients	28
5.2.2.6.Reciclatge d'olis usats.....	29
5.2.2.7.Reciclatge de pneumàtics usats	30
5.2.2.8.Normativa utilització gasos fluorats	32
6. REQUISITS DEL SISTEMA.....	34
7. ESTUDIS I DECISIONS.....	38
7.1. MAQUINARI	38

7.2. LENGUATGES DE PROGRAMACIÓ, FRAMEWORKS I LLIBRIES	38
7.2.1. Llenguatges de programació	39
7.2.1.1. PHP	39
7.2.1.2. ASP	40
7.2.1.3. Perl	40
7.2.1.4. Ruby	41
7.2.1.5. Python	41
7.2.1.6. Conclusió	42
7.2.2. Frameworks	42
7.2.2.1. CakePHP	43
7.2.2.2. Smarty	43
7.2.2.3. CodeIgniter	44
7.2.2.4. Symfony	44
7.2.2.5. Conclusió	45
7.2.3. Llibries	45
7.2.3.1. Doctrine 2	45
7.2.3.2. CRUD de Doctrine	51
7.2.3.3. PdfBundle	52
7.3. PROGRAMARI	53
7.3.1. Sistemes Operatius	54
7.3.2. Entorns de treball	54
7.3.2.1. Notepad ++	54
7.3.2.2. XAMPP	55
7.3.2.3. phpMyAdmin	55
8. ANÀLISI I DISSENY DEL SISTEMA	57
8.1. ANÀLISI	57
8.1.1. Parts de l'aplicació	57
8.1.2. Diagrama de cas d'ús general	59
8.1.3. Diagrames i fitxes de cas d'ús de cada mòdul	65
8.1.3.1. Mòdul taller	65
8.1.3.1.1. Gestió recepció	65
8.1.3.1.2. Gestió ordre de reparació (O.R.)	66
8.1.3.1.3. Gestió pressupost	67
8.1.3.2. Mòdul històric	68
8.1.3.2.1. Consultar històric vehicle	68
8.1.3.2.2. Vehicles d'un client	69
8.1.3.3. Mòdul caixa	70
8.1.3.3.1. Marcar factura com a pagada	70
8.1.3.4. Mòdul informes	71
8.1.3.4.1. Imprimir duplicat factura	71
8.1.3.4.2. Imprimir duplicat albarà	72
8.1.3.4.3. Imprimir duplicat pressupost	73
8.1.3.5. Mòdul consultes	74
8.1.3.5.1. Llistar clients	74
8.1.3.5.2. Llistar factures	75
8.1.3.5.3. Llistar vehicles	76
8.1.3.5.4. Llistar factures pendents	77
8.1.3.6. Mòdul base de dades	78
8.1.3.6.1. Gestió client	78
8.1.3.6.2. Gestió vehicle	79
8.1.3.6.3. Gestió article	80
8.1.3.6.4. Gestió preu ma d'obra	81
8.1.3.6.5. Gestió quota de reciclatge	82

8.1.3.7. Mòdul configuració	83
8.1.3.7.1. Gestionar empresa.....	83
8.1.3.7.2. Gestionar usuaris	84
8.1.3.7.3. Tancament any comptable	85
8.1.3.7.4. Gestionar formes de pagament	86
8.1.3.7.5. Gestionar IVA.....	87
8.2. DISSENY DEL SISTEMA	88
8.2.1 Estructura de la base de dades.....	88
8.2.1.1. Model Entitat-Relació	88
8.2.1.2. Explicació de les entitats de la base de dades	90
8.2.1.3. Model Relacional.....	91
8.2.2 Diagrama de classes	93
8.2.3 Explicació de les classes i relacions.....	95
8.2.3.1 Condició pagament	95
8.2.3.2 Client	97
8.2.3.3 Vehicle.....	98
8.2.3.4 Recepció	100
8.2.3.5 Treballador.....	101
8.2.3.6 Rol	103
8.2.3.7 Ordre de reparació (O.R.).....	103
8.2.3.8 Línia ordre de reparació.....	106
8.2.3.9 Taxa	107
8.2.3.10 Recanvi	108
8.2.3.11 Ma d'obra	109
8.2.3.12 Treball extern.....	110
8.2.3.13 Mètode pagament.....	111
8.2.3.14 Factura.....	112
8.2.3.15 Factura única mensual	113
8.2.3.16 Factura immediata	115
8.2.3.17 Pressupost	116
8.2.3.18 Albarà	117
8.2.3.19 Impost.....	119
8.2.3.20 Configuració.....	119
8.2.4 Diagrames d'activitat	120
8.2.4.1. Alta d'una recepció.....	120
8.2.4.2. Gestió de l'escriptori.....	122
8.2.5 Diagrames de seqüència	124
8.2.5.1. Consultar històric d'un vehicle	125
8.2.5.2. Gestionar OR.....	126
8.2.5.3. Facturar OR.....	128
8.2.5.4. Facturar albarans.....	129
8.2.6 Descripció de l'interfície de l'aplicació.....	131
8.2.6.1. Pantalla de login i logout.....	132
8.2.6.2. Pantalla amb la barra d'opcions per tota l'aplicació	132
8.2.6.3. Pantalla escriptori.....	133
8.2.6.4. Pantalla de la pestanya Consultes	134
9. IMPLEMENTACIÓ I PROVES	136
9.1. AUTENTIFICACIÓ D'USUARIS	136
9.2. PANTALLA PRINCIPAL, L'ESCRITORI.....	139
9.2.1. Pressupostos pendents d'acceptar	140
9.2.2. Mostrar vehicles al taller	142
9.3. PANTALLES DE TIPUS CRUD	145
9.4. ASSIGNAR UN VEHICLE A UN CLIENT	147

9.5. LLISTAR TREBALLADORS	149
9.6. MARCAR FACTURA COM A PAGADA	153
9.7. IMPRIMIR LLISTATS.....	154
9.8. RELACIONAR 2 ENTITATS MITJANÇANT EL FORMULARI D'ALTA.....	157
9.9. AFEGIR RECANVIS, MA D'OBRA O TREBALLS EXTERNS A UNA ORDRE DE REPARACIÓ	159
9.9.1. Afegir ma d'obra a l'ordre de reparació.....	161
9.9.2. Crear ma d'obra a l'ordre de reparació.....	162
9.10. TANCAR ORDRE DE REPARACIÓ	164
10. IMPLANTACIÓ I RESULTATS.....	168
10.1. IMPLANTACIÓ	168
10.1.1. Muntatge/instal·lació al taller.....	168
10.1.2. Migració de dades	170
10.1.3. Conceptes per al desenvolupament de l'aplicació	170
10.2. RESULTATS.....	170
10.2.1. Login/logout.....	171
10.2.2. Escriptori.....	172
10.2.3. Taller.....	180
10.2.4. Històric.....	181
10.2.5. Caixa	182
10.2.6. Informes.....	184
10.2.7. Consultes.....	184
10.2.8. Base de dades.....	185
10.2.9. Configuració.....	188
11. CONCLUSIONS.....	191
12. TREBALL FUTUR.....	193
13. BIBLIOGRAFIA	194
14. ANNEXOS.....	196
14.1. MIGRACIÓ DE DADES	196
15. MANUAL D'USUARI I/O INSTAL·LACIÓ	202
15.1. MANUAL D'USUARI.....	202
15.1.1. Pantalla d'identificació	202
15.1.2. Pantalla escriptori.....	203
15.1.3. Pantalla taller.....	204
15.1.4. Pantalla històric.....	205
15.1.5. Pantalla caixa	207
15.1.6. Pantalla informes	208
15.1.7. Pantalla consultes.....	208
15.1.8. Pantalla base de dades.....	210
15.1.9. Pantalla configuració.....	213
15.1.10. Pantalla tancar sessió	214
15.2. MANUAL D'INSTAL·LACIÓ	215

1. Introducció, motivacions, propòsit i objectius del projecte

1.1. Introducció

Collsaplana Motor SL, és un taller mecànic situat a Vilablareix, just al costat de Girona. El taller, a part de fer tot tipus de reparacions mecàniques, està especialitzat en la reparació de caixes de canvis automàtiques i manuals, vehicles clàssics, competició. Els clients són particulars, empreses o altres tallers mecànics, normalment de la província.

Una aplicació per aquest negoci ha de ser senzilla i pràctica, per a què qualssevol mecànic, sense gaires coneixements d'informàtica, la pugui utilitzar i portar una bona gestió del taller.

Algunes de les mancances trobades a l'aplicació actual són:

- Aplicació molt poc intuïtiva. És complicat trobar un pressupost ja fet per poder començar la reparació.
- Espai molt limitat per a poder posar comentaris als clients.
- Només es pot tenir definit un preu de mà d'obra. En el cas d'aquesta empresa, depenent de la feina a realitzar, tenen diferents preus.
- Gestió de les dades molt dolenta. No es poden donar d'alta diferents usuaris perquè utilitzin l'aplicació. Això fa que qualssevol persona pugui, per exemple, modificar o esborrar les dades d'un client.
- Filtratges per cerques no operatius. Per a facturar albarans pendents, no funciona cap dels filtres que hi ha, s'han de seleccionar del llistat on apareixen tots els albarans pendents de facturar, de tots els clients, de tots els anys.
- No es pot accedir ràpidament a l'històric de reparacions d'un vehicle, si no està al taller i hi ha una ordre de reparació oberta, no es pot consultar.
- Errors al generar factures: si un vehicle canvia de propietari, no s'emmagatzema el canvi i a partir d'aquest moment, sempre que aquest vehicle vagi al taller, s'haurà de canviar manualment.
- Gestió de còpies de seguretat molt dolenta. Per realitzar una còpia de seguretat s'han de copiar tots els fitxers corresponents a l'aplicació.

1.2. Motivacions personals

La motivació personal d'aquest projecte ve donada per voler plasmar els coneixements adquirits durant la carrera a la Universitat de Girona. L'objectiu és tractar conceptes tant de base de dades, disseny de classes, i d'altres més encaminats al web, com pot ser trobar i conèixer un Framework per fer-lo servir.

Al mateix temps, poder ajudar al negoci familiar oferint una aplicació útil i pràctica. Des de la meua implicació al negoci, després de fer algun curs, parlar amb altres tallers i provar diferents aplicacions, he detectat un problema: la majoria d'aplicacions per a tallers mecànics, són genèriques i amb mancances.

1.3. Propòsit i Objectius

L'objectiu del projecte és desenvolupar l'anàlisi, disseny i implementació d'una aplicació per a la gestió de l'empresa. L'aplicació té com a propòsit gestionar els usuaris de l'aplicació, clients, vehicles, recanvis, ordres de reparació, pressupostos, albarans i factures.

1.4. Estructura del document

Aquesta memòria està composta per a 15 capítols, on s'hi pot trobar tota la documentació i explicacions per a la comprensió del projecte:

1. Introducció, motivacions, propòsit i objectius del projecte. Aquest capítol explica quin és el projecte que es desenvoluparà, el perquè del desenvolupament del projecte, quins són els objectius proposats i la seva organització.

2. Estudi de viabilitat. En aquest capítol es justifiquen quins son els paràmetres que fan possibles el desenvolupament del projecte.
3. Metodologia. Aquest capítol fa referència a la metodologia que s'ha seguit per a realitzar el desenvolupament del projecte.
4. Planificació. En aquesta part es defineix l'estratègia seguida per a arribar als objectius plantejats.
5. Marc de treball i conceptes previs. Aquest apartat és on s'explica el marc de treball per a qui es realitza el projecte, i tots els conceptes previs que s'han de conèixer per tal d'iniciar el desenvolupament del projecte i tenir-ne un millor coneixement per a poder seguir amb els següents capítols.
6. Requisits del sistema. Aquest capítol descriu els requeriments que ha de complir l'aplicació juntament amb les funcionalitats.
7. Estudis i decisions. En aquest apartat s'exposen les raons de les eleccions de maquinari, frameworks, llibreries i programari per a realitzar el projecte.
8. Anàlisi i disseny del sistema. Aquest capítol exposa de manera detallada les diferents necessitats del sistema, i dóna unes possibles solucions a partir de diagrames. Es fa una descripció de pantalles de l'aplicació, el disseny de la base de dades i les classes que s'han d'implementar.
9. Implementació i proves. En aquest capítol es detalla com s'ha implementat l'aplicació, els problemes i les solucions que se'ls hi ha donat a la part d'implementació, juntament amb els diagrames de classes i els mètodes més rellevants per tal d'entendre el funcionament de l'aplicació. En aquest capítol a més, es presentaran d'algunes proves.
10. Implantació i resultats. En aquesta secció es mostra el desenvolupament que s'ha hagut de fer per a la implantació de l'aplicació. Mostrant també el resultat final obtingut, i l'assoliment dels objectius.
11. Conclusions. En aquest apartat es mostraran les conclusions obtingudes un cop assolit el projecte.
12. Treball futur. Aquest capítol mostra les possibles millores, ampliacions o treballs futurs que es podran realitzar de l'aplicació.
13. Bibliografia. En aquesta secció s'hi trobaran les referències necessàries utilitzades per al desenvolupament del projecte.
14. Annexos. Aquest capítol descriurà el procés de migració de dades de l'aplicació antiga a la implementada en aquest projecte.
15. Manual d'usuari i instal·lació. Amb el manual de l'usuari obtenim una especificació del funcionament del projecte.

2. Estudi de viabilitat

2.1. Recursos necessaris per desenvolupar el projecte

Per a realitzar aquest projecte, no és necessària una gran infraestructura.

El dispositiu que es té inicialment pel desenvolupament és un ordinador portàtil amb unitat de procés Intel Core i5, a una freqüència de rellotge de 1.5GHz, amb 4 GB de RAM, i un disc dur de 640GB. S'ha de destacar que aquest projecte es podria haver realitzat amb un ordinador amb unes especificacions inferiors ja que estem desenvolupant una aplicació web que no requereix de gaires recursos en quant a maquinària.

Com a sistema operatiu es treballa amb un Microsoft Windows 7, però cal remarcar que l'aplicació es pot executar sobre qualssevol sistema operatiu que sigui capaç de processar llenguatges web en un entorn de servidor com el PHP.

També seran necessaris varis aplicatius per poder desenvolupar el projecte:

- Notepad ++
- XAMPP
- PhpMyAdmin

Exceptuant el sistema operatiu Windows 7, la resta de programes son de llicència GNU (estan sota la llicència de software lliure, per tant no comporten cap cost addicional).

Tenint en compte la magnitud d'aquest projecte, la seva viabilitat és assolible per una única persona qualificada, un desenvolupador de software capaç de projectar les idees desitjades en codi en un temps raonable.

2.2. Avaluació prèvia de costos i medis

Pel desenvolupament de l'aplicació és necessari fer un estudi de viabilitat tenint en compte la viabilitat econòmica, la viabilitat tècnica i la viabilitat legal. En el cas d'aquest projecte no tindrem problemes legals, així doncs només hem de considerar:

- Estudi de viabilitat tecnològica.
- Estudi de viabilitat econòmica.

2.2.1. Estudi de viabilitat tecnològica

Referent a la viabilitat tecnològica, considerem que amb la tecnologia de la que disposem prèviament ja és suficient per a assolir l'aplicació a realitzar.

Com a material necessitarem programari nou, l'aplicació XAMPP per tal de poder emular un servidor web per a poder verificar el funcionament de l'aplicació i fer proves amb el sistema en local, el phpMyAdmin per gestionar la base de dades i el Notepad++ com a editor de text.

I referent al costos, no ens ha afectat en absolut, ja que aquest programari és lliure i per tant gratuït.

2.2.2. Estudi de viabilitat econòmica

Per a la realització de l'anàlisi econòmic està separat en dos parts corresponents a la despesa de recursos humans i la despesa de maquinària.

2.2.2.1 Despesa de recursos humans

Per tal de calcular les despeses de recursos humans associem un perfil o rol de treballador per a cadascuna de les tasques:

- Cost analista: 15€/h

- Cost programador: 10€/h
- Cost dissenyador: 10€/h

Tasca	Perfil	Hores	Cost
Estudi d'eines bàsiques web	Analista	20	300
Estudi d'eines de programació amb crides a base de dades	Analista	25	375
Estudi de llibreries i altres	Analista	20	300
Disseny d'algorismes	Analista	80	1.200
Implementació de codi	Programador	360	3.600
Disseny web	Dissenyador	75	750
Proves i optimització	Programador	40	400
Memòria	Analista	100	1.500
TOTAL		720 h	8.425 €

2.2.2.2 Despesa de maquinària

En aquest cas no s'ha tingut cap despesa de hardware, ja que des d'un principi es tenien els components necessaris. Tot i així, s'ha comptabilitzat l'amortització de hardware:

$$\text{Amortització} = (\text{Cost hardware} / \text{Temps d'amortització}) \times \text{Temps de projecte}$$

Considerant:

- Cost hardware = 1200€
- Temps d'amortització = 3 anys
- Temps de projecte = 1 any

Obtenim una amortització de 400€.

Al no pagar llicències de software, i considerant l'amortització, el cost total seria de:

$$8.425 + 400 = \mathbf{8.825 \text{ €}}$$

3. Metodologia

Abans de començar a desenvolupar el projecte, vaig consultar diferents metodologies per veure quina era la més adequada al tipus de projecte que estic elaborant. De les que vaig veure la que més s'acostava al que jo volia és la Mètrica.

3.1. Metodologia de Mètrica 3

La Mètrica 3 és una metodologia de planificació, desenvolupament i manteniment de sistemes d'informació promoguda pel Ministeri d'Administracions Públiques del govern espanyol per a la sistematització de les activitats del cicle de vida dels projectes de programari a l'àmbit de les administracions públiques. Aquesta metodologia pròpia, es basa en el model de processos del cicle de vida de desenvolupament ISO/IEC 12207 (*Information Technology - Software Life Cycle Processes*) així com en la norma ISO/IEC 15504 SPICE (*Software Process Improvement And Assurance Standards Capability Determination*).

Els principals objectius de la Mètrica versió 3 són:

- Proporcionar o definir Sistemes d'Informació mitjançant la definició d'un marc estratègic que ajudin a aconseguir els objectius de l'organització.
- Dotar a l'organització de productes software que satisfacin les necessitats dels usuaris, donant més importància a l'anàlisi de requisits.
- Millorar la productivitat dels departaments del Sistema i TIC.
- Facilitar la comunicació i comprensió entre els diferents participants en la producció del software al llarg del cicle de vida del projecte.
- Facilitar l'operació, manteniment i ús dels productes software obtinguts.

La Mètrica està orientada als processos, i en el cas de la Mètrica 3 consta de diferents processos, els principals són:

- Planificació de Sistema d'Informació (PSI).

Consisteix en l'obtenció d'un marc de referència pel desenvolupament del Sistema d'Informació que respongui als objectius estratègics de l'organització.

En aquesta part s'han realitzat diferents reunions amb els responsables de l'empresa per a definir amb la màxima exactitud el funcionament de l'aplicació.

- Desenvolupament dels Sistemes d'Informació (DSI).

Aquest apartat engloba tots els processos que s'han de realitzar per a desenvolupar un sistema, des de l'anàlisi de requisits fins a la instal·lació del software.

Degut a la seva complexitat, està dividit en cinc processos:

- Estudi de Viabilitat del Sistema (EVS).

L'objectiu és analitzar un conjunt concret de necessitats per a proposar una solució a curt termini, tenint en compte els aspectes econòmics, tècniques, legals i operatius.

- Anàlisi del Sistema d'Informació (ASI).

L'objectiu que té és aconseguir una especificació detallada del Sistema d'Informació a partir dels usuaris i que aquesta serveixi com a base per a posteriorment realitzar el disseny del sistema.

En aquest apartat s'han definit els requisits que ha de complir l'aplicació, i a partir de diagrames i fitxes d'ús, mostrar-ne els diferents mòduls i funcionalitats que tindrà l'aplicació.

- Disseny del Sistema d'Informació (DSI).

S'ha de definir l'arquitectura del sistema i de l'entorn tecnològic que li donarà suport, juntament amb l'especificació detallada dels components del sistema d'informació.

En aquesta secció s'han dissenyat els processos que tindran els diferents mòduls, s'ha definit l'arquitectura que s'utilitzarà en el codi, juntament amb el diagrama de classes. També s'ha escollit el programari i els llenguatges que s'utilitzaran per a desenvolupar l'aplicació.

- Construcció del Sistema d'Informació (CSI).

Es genera el codi dels components del Sistema d'Informació. Es desenvolupen els procediments d'operació i seguretat i s'elaboren tots els manuals d'usuari final i d'explotació amb l'objectiu d'assegurar el correcte funcionament del sistema per a la seva posterior implantació.

Tenint en compte els punts anteriors, en aquest apartat s'ha implementat l'aplicació, considerant els requisits i diagrames que s'han realitzat amb anterioritat, i alhora realitzant diferents tipus de proves.

També, s'ha redactat un manual d'usuari i s'ha generat un joc de proves utilitzat per a verificar el funcionament global de l'aplicació.

- Implantació i Acceptació del Sistema (IAS).

Entrega i acceptació del sistema en la seva totalitat i realització de totes les activitats necessàries pel pas a producció.

Aquest pas no s'ha tingut en compte, ja que l'aplicació es troba acceptada però encara no està en funcionament a l'empresa.

- Manteniment dels Sistema d'Informació (MSI).

S'obté una nova versió d'un Sistema d'Informació desenvolupat amb Mètrica 3, a partir de les peticions de manteniment que els usuaris realitzen amb motiu d'un problema detectat en el sistema, o per la necessitat d'una millora del mateix.

Aquest apartat encara no s'ha realitzat, però s'hi posarà en un futur, quan l'aplicació estigui en funcionament a l'empresa, a mesura que vagin sortint peticions de millora d'aquesta.

4. Planificació

Aquest projecte va començar al setembre de 2012, i estava programat finalitzar-lo al juny de 2013.

Els primers mesos van consistir en estudiar els diferents tipus de llenguatge via web que podrien ser d'utilitat per les necessitats de l'aplicació. Un cop escollit el Framework i les llibreries corresponents, es va seguir per aprendre'n el funcionament. Paral·lelament, s'anava treballant amb el disseny de l'aplicació fent reunions amb el propietari de l'empresa per determinar-ne les característiques.

Després d'acabar l'etapa de recerca, estudi i aprenentatge, es va definir l'envergadura del projecte.

A continuació ja es va poder començar el projecte pròpiament dit amb la realització del codi. Això comportava utilitzar tots els coneixements adquirits fins el moment i, en alguns casos, aplicar-los per afegir noves característiques a l'aplicació. L'implementació s'ha fet per fases, a mesura que s'anaven acabant les diferents parts, s'anaven creant les diferents finestres de l'aplicació, de manera que cada apartat, quan s'acabava, quedava a punt de ser verificat.

Per acabar, després de fer proves per verificar el funcionament i arreglar els errors corresponents, es va perfilar el disseny de l'aplicació per a què fos més agradable per l'usuari.

4.1. Pla de treball i tasques planificades

El pla de treball està definit per 3 grans tasques, que les explicarem a continuació:

4.1.1 Planificació del Sistema d'Informació

En aquest apartat va ser quan es va decidir quines tasques s'havien d'anar fent al llarg del projecte i en quin ordre s'anirien realitzant.

Al mateix temps, es varen començar a fer les primeres reunions amb el responsable de l'empresa, per a definir el funcionament de l'aplicació, i l'envergadura del projecte, entre d'altres.

4.1.2 Desenvolupament dels Sistemes d'Informació

En aquest apartat s'engloben els diferents processos que s'han realitzat per a desenvolupar el sistema.

A continuació n'expliquem les seves 5 parts com defineix la Mètrica 3:

4.1.2.1 Estudi de Viabilitat del Sistema

En aquest fi de projecte actual, no hi havia un interès econòmic pel projecte.

Per altre banda, es va analitzar la tecnologia de la que es disposava, comprovant que aquesta era suficient pel què l'aplicació embarcava.

També es varen tenir en compte els conceptes legals per a la realització de l'aplicació.

4.1.2.2. Anàlisi del Sistema d'Informació

En aquest apartat, s'han definit tots els requisits del sistema que ha de complir l'aplicació. Per a tenir-ne una millor visió, s'han realitzat els diagrames de cas d'ús i fitxes de cas d'ús necessàries. D'aquí se n'han extret els diferents mòduls i funcionalitats que seran necessaris a l'aplicació.

4.1.2.3 Disseny del Sistema d'Informació

Es varen dissenyar els processos que han de tenir els diferents mòduls.

Es va definir l'arquitectura del sistema i es va realitzar els diagrames de classes general.

A part, es va estudiar i investigar els diferents llenguatges de programació que serien més adequats per l'aplicació, juntament amb els frameworks i les llibreries. Un cop decidits els llenguatges de programació, es va escollir el programari necessari.

4.1.2.4 Construcció del Sistema d'Informació

Tenint en compte les decisions que s'han pres en els punts anteriors, en aquest punt s'inicia la implementació de l'aplicació.

Mentre es va realitzant la implementació, es van realitzant diferents tipus de proves per a resoldre petits errors.

Un cop finalitzada la implementació, es va crear un joc de proves per a trobar tots els errors que puguin aparèixer a l'aplicació. Quan es troben, es fan les modificacions necessàries per a arreglar-los.

També es van fer millores a l'aplicació, tant d'optimització de codi com d'utilitat pel client.

4.1.2.5 Implantació i Acceptació del Sistema

Una vegada arribats a aquest punt, el projecte es va mostrar al client, i aquest el va acceptar, però encara no es va implantar a l'empresa ni es va fer la formació necessària, tot i que es farà en poc temps.

4.1.3 Manteniment dels Sistema d'Informació

Quan s'arriba a aquest punt el sistema està finalitzat, i s'esperen les peticions de manteniment per part dels usuaris degut a algun tipus de problema detectat o per la necessitat de millorar l'aplicació.

A la realitat, encara no s'ha arribat a aquest punt, ja que el projecte no està implantat a la empresa. Es començarà a realitzar en el moment que el projecte ja estigui funcionant a l'empresa.

4.2 Temps estimat

Inicialment, es va plantejar que el projecte tindria una durada d'uns 9 mesos, per tant, que estaria acabat als voltants de juny del 2013. A continuació mostrem la seva representació gràfica a partir del diagrama de Gantt:

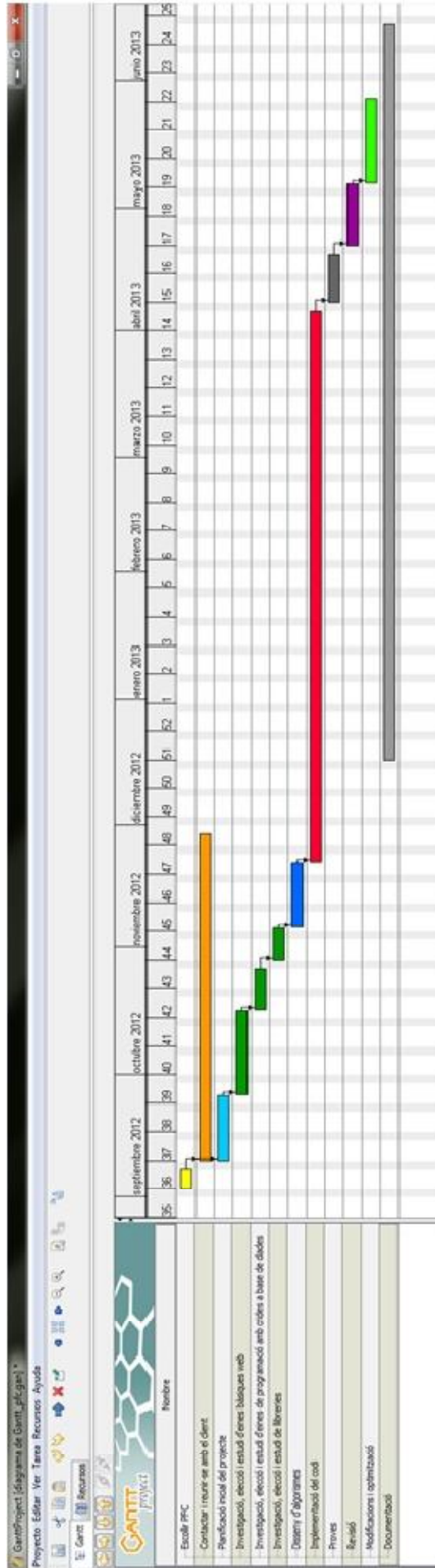


Diagrama de Gantt inicial.

5. Marc de treball i conceptes previs

En aquest apartat es comentaran tots els aplicatius referents al marc de treball, i els conceptes necessaris per a una millor comprensió i seguiment del projecte.

5.1.Marc de treball

L'empresa Collsaplana Motor SL és un taller mecànic petit amb 2 mecànics i una administrativa.

Per entendre més bé el funcionament del taller, a continuació s'explica el procés des que el client arriba al taller amb el vehicle, fins que se l'endu reparat:

S'ha de tenir en compte que tenim 3 tipus de clients:

- els clients particulars,
- les empreses
- i altres tallers que poden portar vehicles o, per exemple, una caixa de canvis desmuntada del vehicle.

Tots ells, dins l'aplicació, es tractaran de la mateixa manera.

Quan un client arriba al taller, s'emplena un *full de recepció* on hi consten: les dades del client, les del vehicle, el problema a solucionar i si vol pressupost o no. Si es vol *pressupost*, un cop els mecànics han detectat l'averia i han calculat el cost de la reparació, es truca el client per informar-lo. Si no en vol o ha acceptat ja el pressupost, els mecànics emplen un full d'*ordre de reparació* (a la pràctica, utilitzen el mateix full que durant la recepció), on s'anota tot el material i les feines realitzades al vehicle, que un cop acabada la feina, s'entrega a l'administració.

A partir d'aquest full es crea la *factura*, o si és una empresa i la seva forma de pagament és una única factura a final de mes, es generarà un *albarà*.

Data: 25 / 10 / 2013.

CLIENT: Anna Vilan Ribas		D.N.I:	403615175
Adreça: C/ Barcelona 59		Telèfon:	
Localitat: Caldes de Malavella		Mòbil:	
VEHICLE amb matrícula: 7983 CHJ		138.625 Km.	
Marca: Peugeot	Model: 206 HDI		
Xassís:			

Anomalies/operacions indicades pel client:

Averia per diagnosticar Diagnosticar i reparar

Fer distribució i manteniment

Full de treball:

Quantitat	Descripció	Import
1	Kit distribució	
1	conetja accesoris	
5l	anti-congelant 50%	
1	Bomba d'aigua.	
4'3l	oli 10W 40	
1	filtre oli	
1	filtre aire	
1	bombeta posició	
3'5h	Distribució + bomba	
1	MO manteniment	

- Autoritzo la reparació descrita sense pressupost previ, i la subcontractació dels treballs necessaris d'altres especialitats.

- Amb la signatura del present full de treball s'autoritza de forma tàcita al taller, a fer les proves de circulació (si s'escau per carretera), per al diagnòstic i reparació del vehicle.

- Desitjo recollir les peces substituïdes al meu vehicle: Sí No



Signatura del responsable de taller

Acceptació del client



Data: ___ / ___ / ___

DATA PREVISTA DE LLIURAMENT DEL VEHICLE: ___ / ___ / ___

Segons les condicions especificades a l'article 14 del decret 298/1993.

Exemple d'un full d'ordre de reparació utilitzat al taller.

Un cop abonat l'import de la factura, el client ve a recollir el vehicle reparat.

Actualment a l'empresa només hi ha un ordinador. En un futur proper, s'incorporarà una pantalla tàctil a la zona del taller per eliminar els formularis en paper.

5.2. Conceptes previs

En aquest apartat es comentarà tota la informació i conceptes previs necessaris per entendre el projecte com poden ser els patrons de disseny utilitzats i la informació legal a tenir en compte.

5.2.1. Patrons de disseny

Existeix un gran ventall de patrons de disseny, que serveixen per a trobar solucions a problemes comuns en el desenvolupament de software. Per a la realització d'aquest projecte, el principal és el patró MVC.

5.2.1.1. MVC (Model Vista Controlador)

L'arquitectura MVC és un patró de disseny per al desenvolupament de programari que separa el model de dades, la interfície d'usuari i la lògica de control.

Molts sistemes informàtics utilitzen un Sistema de Gestió de la Base de Dades per a gestionar les dades: en línies generals del MVC, es corresponen al model. La unió entre la capa de presentació i la capa de negoci, en el paradigma de la Programació per Capes, representa la integració entre la Vista i el corresponent Controlador. El MVC pretén separar la capa visual gràfica de la corresponent programació i accés de dades,

cosa que millora el desenvolupament i el manteniment de la Vista i el Controlador en paral·lel.

Podem trobar 3 conjunts de classes:

- Classes del MODEL

És la representació específica de la informació amb la qual el sistema opera. El model es limita a la relació amb l'objecte vista i l'objecte controlador, que facilita les presentacions visuals complexes. El sistema també pot operar amb més dades no relatives a la presentació, fent ús integrat d'altres lògiques de negoci i de dades afins al sistema modelat.

- Classes de la VISTA

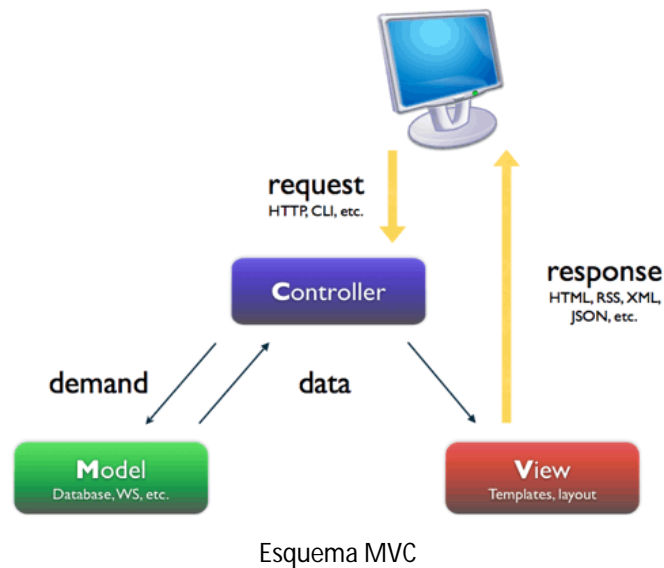
La vista presenta les classes del model en un format adequat per a la interacció, usualment amb la interfície d'usuari, és a dir, la sortida de l'aplicació.

- Classes del CONTROLADOR

El controlador respon als events, usualment accions de l'usuari, i invoca peticions al model i a la vista.

Encara que es puguin trobar diferents implementacions del patró MVC, el flux que segueix el control generalment és el següent:

- L'usuari interactua amb la interfície d'usuari, per exemple polsant un botó.
- El controlador rep per part de la vista la notificació de l'acció sol·licitada per l'usuari. Aquest gestiona l'event que arriba, freqüentment a partir d'un gestor d'events (*handler*) o *callback*.
- El controlador accedeix al model, actualitzant-lo, possiblement modificant-lo de forma corresponent a l'acció sol·licitada per l'usuari. Els controladors complexes estan sovint estructurats utilitzant un patró de comanda que encapsula les accions i simplifica l'extensió.
- El controlador delega als objectes de la vista la tasca de desplegar la interfície d'usuari. La vista obté les dades del model per a generar la interfície adequada per a l'usuari, on es reflecteixen els canvis en el model. El model no ha de tenir coneixement directe sobre la vista. No obstant, es podria utilitzar el patró *Observer* per a proveir d'indirecció entre el model i la vista, permetent al model notificar als interessats qualsevol canvi. Un objecte vista pot registrar-se amb el model i esperar els canvis, però tot i així el model en sí mateix segueix sense saber sobre vista. En general el controlador no passa objectes de domini (el model) a la vista, tot i que pot donar l'ordre a la vista per a que s'actualitzi.
- La interfície d'usuari espera noves interaccions de l'usuari, començant el cicle una altra vegada.



5.2.2.Marc legal

Del marc legal, cal destacar les normatives vigents, aplicades a aquest cas:

- *Real Decreto 1496/2003, de 28 de noviembre, por el que se aprueba el Reglamento por el que se regulan las obligaciones de facturación, y se modifica el Reglamento del Impuesto sobre el Valor Añadido.*
- Decret 298/1993, de 8 d'octubre pel qual es regula l'activitat industrial i de prestació de serveis en els tallers de reparació de vehicles automòbils, dels seus equips i components.
- Llei orgànica 15/1999 de protecció de dades de caràcter personal (LOPD).

A part de les normatives de caràcter general que determinen el mètode de treball, les obligacions de l'empresa i com s'han de generar els diferents documents, s'han de tenir en compte les següents normatives que s'apliquen a productes utilitzats al taller, que definiran les quotes aplicades a les factures, com són la quota de reciclatge d'oli, dels pneumàtics usats i des de l'1 de gener de 2014, la quota aplicada per l'utilització dels gasos fluorats. Aquestes són:

- Reciclatge d'olis usats: *La parte dispositiva del Real Decreto 679/2006, de 2 de junio, por el que se regula la gestión de los aceites industriales usados.*
- Reciclatge de pneumàtics usats: Real Decret 1619/2005 sobre la gestió de pneumàtics fora d'us.
- Normativa utilització gasos fluorats: *Real Decreto 1042/2013, de 27 de diciembre, por el que se aprueba el Reglamento del Impost sobre los Gases Fluorados de Efecto Invernadero.*

5.2.2.1.Requisits del pressupost

Segons el decret 298/1993, al pressupost hi ha de constar:

- El número del taller al Registre Industrial, com també la seva identificació fiscal i l'adreça.
- Nom i domicili del client, i el número de DNI o CIF.
- La identificació del vehicle, esmentant: marca, model, matrícula i el nombre de quilòmetres comptabilitzats.
- Les reparacions que s'han dut a terme, elements que han de reparar-se o substituir-se.
- La data i signatura de qui presta el servei.
- La data prevista de lliurament del vehicle ja reparat, a partir de l'acceptació del pressupost.
- Indicacions del temps de validesa del pressupost (mínim 12 dies).
- La signatura i data de l'acceptació per part del client.

5.2.2.2.Requisits del resguard de dipòsit/recepció

Al resguard de dipòsit hi han de constar les dades del taller i del client com en el pressupost, i també:

- Descripció indicant els aspectes fonamentals pels quals el vehicle ha entrat al taller.
- En el cas que el vehicle sigui lliurat per a la confecció del pressupost: descripció breu però concreta dels treballs realitzats per realitzar el diagnòstic de l'avaría, com també els imports aproximats d'aquests.
- Data prevista del lliurament.
- Data i signatura de qui presta el servei.

5.2.2.3.Requisits de la factura

L'article 15 del decret 298-1993 cita els detalls de les factures:

“Article 15: Factura i despeses d'estada

15.1 Tots els tallers estan obligats a lliurar al client factura escrita, signada i segellada, degudament desglossada i en la qual s'esmenti qualsevol tipus de càrrecs acreditats, les

operacions realitzades, peces o elements que s'hagin emprat i hores de treball utilitzades, i assenyalaran per a cada concepte el seu import, d'acord amb el que disposen els articles 12 i 14 del present Decret.

15.2 Només podran acreditar-se despeses per estada quan, havent-se comunicat a l'usuari que ja s'ha confegit el pressupost o reparat el vehicle, aquest deixi transcórrer més de tres dies feiners sense pronunciar-se sobre l'acceptació del pressupost o sense procedir a la retirada del vehicle.

En tot cas, aquestes despeses d'estada només seran procedents quan el vehicle es trobi dins els locals sota la custòdia del taller i pels dies que excedeixin el termini esmentat i solament en el supòsit que la penalització esmentada i el seu import hagin estat fixats i advertits a l'usuari en el corresponent resguard de dipòsit.

15.3 En l'anvers de les factures haurà de fer-se constar explícitament amb lletra de dimensions no inferiors a 1,5 mil·límetres les condicions en què la reparació resta garantida. Aquesta garantia no pot ser en cap cas inferior a la prevista com mínim a l'article 16 del present Decret.

15.4 En tot cas l'usuari, prèviament al pagament de la factura, té dret de comprovar in situ els treballs, reparacions o instal·lacions que s'hagin dut a terme.”

Al CD adjunt a aquesta memòria es pot trobar el document complet:

Decret 298-1993 taller reparació.pdf

5.2.2.4.Requisits comptables

A part de les obligacions referents a l'activitat de l'empresa, també hi ha les obligacions legals comptables de facturació.

Cal destacar alguns detalls comuns tant en factures, albarans i pressupostos:

- La numeració de tots els documents ha de ser correlativa, igual que les sèries. No hi poden haver dues factures amb el mateix número de factura i sèrie, ni tampoc pot haver-hi un salt en la numeració.
- Les dates d'expedició també han de ser correlatives. No es poden tenir factures anteriors amb una data posterior que les següents.
- Dades tant del client com de l'empresa. És obligatori que en els documents hi consti el número d'identificació fiscal (NIF o DNI) i el nom i cognoms o raó social tant del proveïdor com del client. Si el client és persona física i no és empresari

- o professional no és necessari indicar el domicili.
- Descripció de les operacions. Ha d'incloure el preu unitari sense impost i tots els descomptes que no estiguin inclosos en el preu unitari.
 - El tipus impositiu aplicat a l'operació. Quan hi hagi diferents tipus impositius s'ha d'especificar la base imposable que correspon a cada tipus. S'ha d'especificar el recàrrec d'equivalència en cas d'estar sotmès a aquest.
 - La quota tributària. És la quota que s'obté de multiplicar la base imposable pel tipus i sempre ha d'aparèixer de forma separada.
 - Data de realització de les operacions. S'ha d'indicar la data en què s'han realitzat les operacions o s'ha rebut un pagament anticipat en el cas que sigui diferent a la data d'expedició de la factura.
 - En el cas d'una factura electrònica, perquè tingui la mateixa validesa legal que una emesa en paper, el document electrònic que la representa ha de contenir els camps obligatoris exigibles a tota factura, estar firmat mitjançant una firma electrònica vàlida basada en un certificat reconegut i ser transmesa d'un ordinador a un altre recollint el consentiment d'ambdues parts.

Al CD adjunt a aquesta memòria es pot trobar el document complet:

BOE-A-2003-21845.pdf

5.2.2.5. Protecció de dades dels clients

La llei orgànica 15/1999 de protecció de dades de caràcter personal (LOPD) és una llei orgànica que té com objectiu garantir i protegir les dades personals, les llibertats públiques i els drets fonamentals de les persones físiques, i especialment la seva intimitat i privadesa personal i familiar.

Aquesta llei afecta a totes les dades que fan referència a persones físiques registrades sobre qualsevol suport, informàtic o no.

Les dades de caràcter personal s'estructuren en diferents nivells (bàsic, mitjà i alt) segons el nivell de seguretat. En el nostre cas, les dades emmagatzemades són de nivell bàsic.

Ja que la normativa vigent segons la *Agencia Española de Protección de Datos* és molt extensa, a continuació queda exposat un resum de la normativa de la llei orgànica de protecció de dades:

“En vigor desde 1999, la LOPD 15/1999 (Ley Orgánica de Protección de Datos) impone una serie de obligaciones legales a las empresas y entidades que efectúen tratamientos de ficheros con datos de carácter personal. Es aplicable esta Ley a todos los ficheros que incluyan datos

de cualquier persona física: clientes, proveedores, trabajadores, usuarios de servicios, vendedores, etc. Son por tanto datos de carácter personal: el nombre y apellidos, el NIF, la dirección postal, la dirección de correo electrónico, los de afiliación sindical, los de salud, las imágenes de videocámaras, etc.

El R.D. 994/1999, recientemente derogado y sustituido por el R.D. 1720/2007, establece la obligación a las empresas de implantar diversas medidas de seguridad destinadas a garantizar la protección de dichos datos, afectando al sistema de información general, a los sistemas informáticos, a los soportes de almacenamiento, a los usuarios de dichos sistemas, a la documentación y fichas físicas que contengan dichos datos, etc., etc.

El incumplimiento de estas medidas de seguridad, así como de otras obligaciones que dispone la propia Ley, puede ser sancionado con multas de hasta 600.000 €.”

Al CD adjunt a aquesta memòria es pot trobar el document complet:

lo15_1999_lopdcp.pdf

5.2.2.6. Reciclatge d'olis usats

Segons el que estableix la llei, els fabricants i importadors d'olis industrials han de fer-se càrrec dels olis usats resultats dels seus productes, tant en l'àmbit logístic com en el financer. Així, de forma individual o agrupant-se en un Sistema Integrat de Gestió (SIG), com SIGAUS, financen la recollida selectiva i correcta gestió d'aquest residu.

Per això, la normativa vigent preveu l'establiment d'una quota sobre cada kilogram d'oli nou venut, que es va fixar en 0.06 € o, el que és el mateix, 0.054 € per litre. Aquesta quota s'ha de transmetre des del fabricant al distribuïdor, del distribuïdor al taller, i d'aquest al consumidor final, sense que es pugui modificar al llarg de la cadena.

Aquest concepte, que típicament apareixerà com a "aportació a SIGAUD" o "RD 679/2006" (la norma que regula l'establiment de la quota), s'ha de desglossar expressament a la factura que emet el taller cada vegada que s'efectua un canvi d'oli. En realitat, l'aportació recau sobre l'oli nou que s'utilitza, no sobre l'oli usat.

Així, un turisme, per exemple, aporta aproximadament 24 cèntims en cada canvi d'oli, que es destinen íntegrament, a través de SIGAUS, a cuidar el medi ambient i evitar els danys que podrien utilitzar els olis usats.

SIGAUS, com a entitat sense ànim de lucre, destina tota la seva recaptació al cicle de recollida i gestió de l'oli usat. En aquest sentit, la seva comptabilitat es sotmet a auditories independents i són publicades anualment.

Al CD adjunt a aquesta memòria es pot trobar el document complet:

RD_679_2006_reciclatge_dolis.pdf

Un exemple d'aplicació d'aquesta llei es pot veure al retall de la factura que es mostra a continuació:

...

Codi	Descripció	Quant.	Import	Subtotal
1040c	Oli CASTROL 10w40	4,5	12,64	56,88
ro	Reciclatge d'oli R.D. 679/2006	4,5	0,054	0,243

...

5.2.2.7. Reciclatge de pneumàtics usats

El Real Decret 1619/2005 de 30 de desembre, sobre la gestió de pneumàtics fora d'ús, atribueix la responsabilitat bàsica de la correcta gestió dels PFU (pneumàtics fora d'ús) als responsables de la seva posada en el mercat nacional dels pneumàtics, ja siguin fabricants, importadors o adquirents en un altre estat de la UE. A més prohibeix l'eliminació dels pneumàtics trossejats en dipòsits controlats a partir de juliol del 2006.

Per donar compliment a aquest Real Decret abans esmentat, els productors s'han agrupat en dos sistemes integrats de gestió: SIGNUS ECOVALOR, SL i Tratamiento de Neumáticos Usados, SL (TNU).

El seu àmbit d'aplicació és el mercat nacional de reposició, amb excepció dels de bicicleta i els de més de 1.400 mm.

Estableix que els productors de pneumàtics tenen l'obligació d'elaborar i presentar un pla empresarial de prevenció de pneumàtics fora d'ús per minimitzar les afeccions al medi ambient i que aquests poden ser elaborats a través del sistema integrat de gestió.

A més a més, tot productor està obligat a gestionar una quantitat de pneumàtics fora d'ús fins la quantitat posta en el mercat de reposició. Un dels mecanismes que estableix el Real Decret per complir amb aquestes obligacions és mitjançant la participació en un sistema integrat de gestió com pot ser el que ofereix SIGNUS.

Al CD adjunt a aquesta memòria es pot trobar el document complet:

[rd1619-2005sobrenfu.pdf](#)

Igual que en el cas de la quota de reciclatge d'olis, s'ha establert un ECOVALOR que és el preu que paga el productor a SIGNUS per garantir la correcta gestió mediambiental del pneumàtic usat al ser substituït al vehicle per un de nou. Aquest preu és el mateix que productors i distribuïdors repercuteixen en tota la cadena de valor fins el consumidor final, en aquest cas, el client del taller.

Depenent del tipus de pneumàtic, la quota ECOVALOR canvia, a continuació es pot veure la classificació dels tipus de pneumàtics i el seu import:

Categorías	Descripción	Detalle de los productos	Precio/Ud
(A)	moto, scooter, ciclomotor	todos los productos	0,95 €
(B)	turismo	todos los productos	1,58 €
(C)	camioneta, 4x4, suv	todos los productos	2,75 €
(D)	camión, autobus	todos los productos	13,25 €
		(*)	
(E1)	agricultura, obra pública, industrial, macizos, mantenimiento, aeronaves, quad, kart, otros Nota: los neumáticos cuyo diámetro sea superior a más de 1.400 mm., quedan excluidos (RD1619/2005)	0,01kgs - 5.00kgs	0,41 €
(E2)		5,01kgs - 20.00kgs	2,41 €
(E3)		20,01kgs - 50.00kgs	5,27 €
(E4)		50,01kgs - 100.00kgs	12,88 €
(E5)		100,01kgs - 200.00kgs	25,10 €
(E6)		200,01kgs - 450.00kgs	42,87 €
(E7)		>450kgs	91,54 €

(*) Pesos referidos a neumático nuevo.

Tarifas vigentes a partir del 1 de enero de 2014
I.V.A. o I.G.I.C. no incluido

Taula ECOVALOR 2014

Observant la taula anterior, el taller està obligat a detallar aquesta quota al desgloss de la factura sota el concepte de "ECOVALOR SIGNUS (R.D. 1619/2005), categoria del pneumàtic".

Un exemple d'aplicació d'aquesta llei es pot veure al retall de la factura que es mostra a continuació, en aquest cas, el client ha canviat 4 pneumàtics d'un vehicle 4x4:

...

Codi	Descripció	Quant.	Import	Subtotal
	Goodyear WRANGLER HP: 255/65 R17 110T	4	162,47	669,88
rp	ECOVALOR SIGNUS (R.D. 1619/2005), C	4	2,75	11,00

...

5.2.2.8. Normativa utilització gasos fluorats

El Ministerio de Hacienda y Administraciones Públicas va publicar el passat 30 de desembre el Real Decret 1042/2013 pel que s'aprova el reglament de l'Impost sobre Gasos fluorats d'Efecte Hivernacle, utilitzats comunment als tallers mecànics per la càrrega d'aires acondicionats.

Aquest reglament suposa el desplegament reglamentari de la Llei 16/2013 que, a l'article 5, estableix un nou impost des de l'1 de gener de 2014.

El Real Decret defineix als tallers com a consumidors finals, pel que els converteix en transmissors de l'impost. D'aquesta manera, quan compren el **gas 134a** destinat a la càrrega d'aires acondicionats, les empreses distribuïdores els hi cobraran un impost que hauran de repercutir al client final a la factura. Aquesta serà l'única obligació que tingui el taller.

Al CD adjunt a aquesta memòria es pot trobar el document complet:

[BOE-A-2013-13759-C_gasos_fluorats.pdf](#)

Per tant, la informació que hauran de reflectir a la factura emesa al client final serà la següent: quantitat utilitzada en la càrrega expressada en kilograms, l'epígraf corresponent que segons la normativa al Gas 134^a serà l'epígraf 1.8 i l'import de l'impost suportat, que durant el 2014 serà de 8,58 €/Kg, durant el 2015 de 17,16 €/Kg i el 2016 de 26€/Kg.

Com en els altres casos, un exemple d'aplicació d'aquesta llei es pot veure al retall de la factura que es mostra a continuació, en aquest cas, s'ha carregat l'aire acondicionat d'un vehicle amb 650g de gas 134^a:

...

Codi	Descripció	Quant.	Import	Subtotal
G134a	Càrrega aire acondicionat (en grams)	0,650	87,69	56,99
rdgf	Quota HFC – 134a, epígraf 1.8	0,650	8,58	5,577

...

Per altra banda, els tallers mecànics hauran de realitzar un curs específic per a la manipulació d'aquest gas.

6. Requisits del sistema

Per poder establir els requisits que haurà de complir l'aplicació, es van fer varies reunions amb el responsable de l'empresa. A mesura que ens anàvem reunint, es van anar pactant diferents requeriments, que s'anaven revisant a mesura que s'anava realitzant el projecte.

Els requisits de l'aplicació són:

- Per poder accedir a l'aplicació, es trobarà una primera pantalla d'autenticació. No es podrà accedir a cap pantalla si la identificació no és correcta.

- Gestió dels usuaris. Hi haurà diferents usuaris amb diferents rols, els tipus de rols són: **administrador**, **gerent** i **mecànic**. Segons el rol, es podran modificar diferents atributs. El rol mecànic només podrà modificar la seva contrasenya, el rol gerent podrà accedir a tota l'aplicació sense restriccions, però no podrà donar de baixa o eliminar cap informació, en canvi, l'administrador, si que ho podrà fer. Les tasques possibles amb el permís d'administrador són:
 - Gestió d'ordres de reparació, albarans, factures, pressupostos.
 - Alta
 - Baixa
 - Modificació
 - Consulta
 - Impressió
 - Gestionar les còpies de seguretat.
 - Gestió de clients, vehicles, recanvis, usuaris.
 - Alta
 - Baixa
 - Modificació
 - Consulta
 - Llistats
 - Consulta
 - Impressió

- Quan un usuari entri a l'aplicació, es trobarà l'escriptori, on només podrà accedir a les opcions a les que tingui accés.
- A l'escriptori apareixeran tots els vehicles que estiguin al taller en aquest moment, diferenciant l'estat. També apareixeran els pressupostos que estan pendents d'aprovació fins que hagi arribat la data de caducitat.
- L'escriptori de l'aplicació tindrà els menús:
 - taller, per accedir a la pantalla inicial de recepció d'un vehicle
 - històric, on es podran consultar les reparacions o pressupostos anteriors d'un vehicle determinat
 - caixa, per comptabilitzar el pagament de les factures
 - informes, per imprimir duplicats de factures, albarans, pressupostos i facturar albarans
 - consultes, on es podran obtenir llistats dels clients, vehicles, tots els vehicles d'un client, factures pendents de cobrar, previsions de cobrament, ..
 - base de dades, es podran fer altes, baixes i modificacions de clients, articles, vehicles, preus de ma d'obra, ...
 - configuració, en aquest apartat es configuraran totes les variables de l'aplicació, com poden ser els números de les sèries de les factures, modificar el logotip o les dades de l'empresa, ...
- Un cop un vehicle ha entrat el taller i s'ha emplenat la recepció, tant si el client vol un pressupost o no, omplirem una **ordre de reparació**, aquest document serà la plantilla que s'utilitzarà tant per les factures com pels pressupostos. El contingut d'aquest serà:
 - dades del client
 - dades del vehicle
 - número d'ordre de reparació
 - descripció dels treballs a realitzar
 - llistat dels recanvis
 - llistat de ma d'obra
 - llistats de treballs externs
- Els llistats de recanvis, ma d'obra i treballs externs han de proporcionar les següents opcions:
 - afegir línies a cada llistat

- modificar el nom
 - modificar el descomptes
 - modificar la quantitat
 - modificar l'import unitari
 - eliminar una línia
- Les factures i pressupostos, a part de contenir una ordre de reparació, tindran:
 - número de factura o pressupost
 - dades completes del client
 - dades completes del vehicle
 - data del pressupost
 - base imposable
 - impostos (tant per cent d'iva)
 - import total
 - normativa: en el cas del pressupost, la valides d'aquest, i en el cas de la factura, les dades corresponents a la garantia de la mateixa, l'esment de la llei de protecció de dades i el mètode de pagament si està definit.
- Altres característiques que han de tenir tots 3 documents són:
 - els imports sempre seran calculats automàticament.
 - Els documents podran ser cancel·lats o anul·lats.
 - Els números de cada tipus de document hauran de ser correlatius igual que la data. No podem tenir una factura anterior a una altra amb una data posterior.
- Un cop s'hagi creat una factura, albarà o un pressupost, no es podran modificar, per fer-ho s'haurà de generar un abonament i un cop fet aquest, s'obrirà automàticament l'ordre de reparació que contenia.
- Si es modifiquen les dades d'un client o un vehicle, no es modificarà a tots els documents en què apareixin. Només als documents que es creïn a partir de la modificació. Ens podem trobar, per exemple, que un client canviï de domicili.
- Tancament de l'any comptable. Si es selecciona aquesta opció, no es podran fer més modificacions de l'any que estigui tancant en cap dels àmbits de factura emesa, pressupostos, albarans i ordres de reparació. Però si es podran visualitzar les dades i realitzar-ne impressions en les seccions en que existeixi

l'opció d'imprimir.

- Llistats. L'aplicació ha de ser capaç de generar diferents tipus de llistats demanats pel client. Aquests són:
 - Visualitzar la facturació de l'empresa donat dues dates. Al llistat hi apareixeran, les dates de factura, el número de factura, el client, la base imposable, l'iva i el preu total.
 - Visualitzar els clients de l'empresa.
 - Visualitzar tots els vehicles d'un client. Si es tracta d'una empresa o un client que ens porta diferents vehicles, poder consultar els vehicles que porta al taller.
 - Obtenir una relació dels tipus de ma d'obra que es tenen a l'empresa.
 - Visualitzar els recanvis entrats a l'aplicació.

- L'aplicació ha de ser fàcil d'utilitzar i intuïtiva per l'usuari.

7. Estudis i decisions

7.1. Maquinari

Per a la realització d'aquest projecte s'ha utilitzat un únic ordinador, concretament un portàtil amb les següents característiques:

- Processador: Intel® Core™ i5-2410M
- Memòria RAM: 4 GB
- Disc dur: 640 GB

El projecte s'ha realitzat amb l'ordinador de característiques esmentades anteriorment ja que era del que es disposava. Cal remarcar que aquest projecte té uns requisits de maquinari inferiors per a la seva realització. Per exemple, amb un ordinador amb una unitat de procés d'un sol Core a una freqüència de 1.50 GHz i 1 GB de RAM, ja seria suficient.

7.2. Llenguatges de programació, frameworks i llibreries

A causa de les necessitats del projecte, sabent que en un futur seria necessari un servidor extern i que s'hi pogués accedir des de qualssevol terminal, va decidir-se que l'aplicació es realitzaria via web. Així doncs, a continuació esmentem els possibles llenguatges de programació, els frameworks i les llibreries que es podrien utilitzar per a la creació de l'aplicació.

7.2.1. Llenguatges de programació

Un llenguatge de programació és un idioma artificial dissenyat per a expressar processos. Tenint en compte que no tots els llenguatges de programació estan suportats pels servidors, a continuació n'esmentem la col·lecció que en varen seleccionar per a investigar.

7.2.1.1. PHP

PHP és un acrònim recursiu que significa PHP Hypertext Pre-processor. PHP és un llenguatge de programació interpretat, dissenyat originalment per a la creació de pàgines web dinàmiques amb accés a informació emmagatzemada en una base de dades. S'utilitza principalment per a la interpretació del costat del servidor però actualment pot ser utilitzat des de una interfície de línia de comandes o en la creació d'altres tipus de programes.

Punts a destacar:

- És considerat un llenguatge fàcil d'aprendre, ja que en el desenvolupament es varen simplificar diferents especificacions.
- El codi font que s'utilitza en PHP és invisible al navegador web i al client, ja que és el servidor el que s'encarrega d'executar el codi i enviar el resultat HTML al navegador.
- Té capacitat de connexió amb la majoria dels motors de bases de dades que s'utilitzen en l'actualitat, se'n destaca la seva connectivitat amb MySQL i PostgreSQL.
- Té capacitat per a expandir el seu potencial utilitzant mòduls (anomenats ext's o extensions).
- Posseeix una àmplia documentació en el lloc web oficial, entre la qual se'n destaca que totes les funcions del sistema estan explicades i exemplificades.
- És lliure, pel què es presenta com una alternativa de fàcil accés per a tots.
- Permet aplicar tècniques de programació orientades a objectes.
- No requereix definició de tipus de variables, tot i que les seves variables es poden avaluar també pel tipus que s'estiguin utilitzant en temps d'execució.
- Fa ús d'excepcions.
- No n'obliga a qui l'utilitza a seguir una determinada metodologia a l'hora de programar, tot i fent-ho, el programador pot aplicar en el treball qualsevol

tipus de tècnica de programació o de desenvolupament que li permeti escriure codi ordenat, estructurat i manejable.

7.2.1.2 ASP

ASP (Active Server Pages), és una tecnologia de Microsoft del tipus "costat del servidor" per a pàgines web generades dinàmicament. Aquesta tecnologia està estretament relacionada amb el model tecnològic i de negoci del fabricant. Intenta ser una solució per a un model de programació ràpida, ja que ha estat programada en Visual Basic o C#, amb moltes limitacions i alguns avantatges específics d'entorn web.

L'interessant d'aquest model de tecnologia, es poder utilitzar diversos components ja desenvolupats com alguns controls ActiveX, així com components del costat del servidor, que permetin la interacció d'scripts amb el servidor.

Facilita la programació de llocs web mitjançant nombrosos objectes integrats, com un objecte de sessió basat en cookies, per a mantenir les variables mentre es va passant d'una pàgina a una altre.

7.2.1.3. Perl

Perl (Practical Extraction and Report Language) és un llenguatge de programació. L'estructura deriva del llenguatge C. Perl és un llenguatge imperatiu, amb variables, expressions, assignacions, blocs de codi delimitats per claus, estructures de control i subrutines.

També pren característiques del Shell. Totes les variables son marcades amb un distintiu precedent. Aquestes distintius identifiquen inequívocament els noms de les variables, permetent a Perl tenir una ampla sintaxi. Notablement, els sigils permeten interpolar variables directament dins de les cadenes de caràcters. Com en els shells (terminal de comandes, on es poden introduir instruccions per teclat, que el sistema operatiu entén), Perl té moltes funcions integrades per tasques comunes i per accedir als recursos del sistema.

Perl pren les llistes a l'estil del Lisp, hash del llenguatge AWK i expressions regulars del sed. Tot això simplifica i facilita totes les formes d'anàlisi, usabilitat del text i les tasques de gestió de dades.

En la versió Perl 5, es varen afegir característiques per suportar estructures de dades complexes, funcions de primer ordre, un model de programació orientat a objectes, i el varen preparar per a empaquetar codi reutilitzable com mòduls.

Està limitat a funcionar només amb IIS (Internet Information Services).

7.2.1.4. Ruby

Ruby és un llenguatge de programació interpretat, reflexiu i orientat a objectes. Combina una sintaxis inspirada en Python i Perl amb característiques de programació orientada a objectes similars a Smalltalk. Comparteix també funcionalitat amb altres llenguatges de programació com Lisp, Lua, Dylan i CLU. Ruby és un llenguatge de programació interpretat en una sola passada i la seva implementació oficial està distribuïda sota una llicència de software lliure.

Algunes de les seves característiques:

- Quatre nivells d'àmbit de variables: global, classes, instància i local.
- Usabilitat d'excepcions.
- Iteradors i clausures (passant blocs de codi).
- Expressions regulars natives similars a les de Perl a nivell de llenguatge.
- Possibilitat de redefinir els operadors (sobrecarrega d'operadors).
- Recollida d'escombraries automàtic.
- Altament portable.
- Fils d'execució simultanis en totes les plataformes utilitzant threads.
- Carrega dinàmica de DLL/biblioteques compartides en la majoria de plataformes.
- Àmplia llibreria estàndard.
- Suporta injecció de dependències.
- Suporta alteració d'objectes en temps d'execució.

7.2.1.5. Python

Python és un llenguatge de programació interpretat, amb la filosofia de donar èmfasi a una sintaxi més neta i que afavoreixi un codi llegible. És de programari lliure.

Es tracta d'un llenguatge de programació multiparadigma, ja que permet varis estils: programació orientada a objectes, programació imperativa i programació funcional.

D'altres paradigmes estan suportats mitjançant l'ús d'extensions.

Una característica important de Python és la resolució dinàmica de noms, és a dir, el què enllaça un mètode i un nom de variable durant l'execució del programa.

Un altre objectiu d'aquest llenguatge és la facilitat d'extensió. Es poden escriure nous mòduls fàcilment en C o C++. Python pot incloure-se en aplicacions que necessiten una interfície programable. Sovint es compara amb llenguatges com Java, Tcl, Perl o Scheme.

7.2.1.6. Conclusió

Considerant tots els punts anterior, es va decidir escollir com a llenguatge de programació el **PHP**, ja que és el llenguatge més utilitzat per a realitzar pàgines i aplicacions web. També és el llenguatge que se'n pot trobar més informació, facilitant molt més el tipus d'aprenentatge, i a la vegada és el que rep més suport de la comunitat.

A part, també em vaig acabar decantant més cap al PHP, ja que és ara mateix un dels llenguatges de programació amb més fama i més utilitzat, i també vaig pensar que seria una bona oportunitat per aprendre a treballar-hi.

7.2.2. Frameworks

En el desenvolupament de software, un framework o infraestructura digital, és una estructura conceptual i tecnològica de suport definit, normalment amb mòduls de software concrets, amb base al qual d'altres projectes de software poden ser més fàcilment organitzats i desenvolupats. Típicament, poden incloure suport de programes, llibreries i llenguatge interpretat, entre altres eines, per així ajudar a desenvolupar i unir els diferents components del projecte.

A continuació s'anomenen els frameworks que es van estudiar, ja que es va creure que podien ser els més indicats per el projecte.

7.2.2.1. CakePHP

CakePHP és un framework de desenvolupament d'aplicacions web escrit en PHP, creat sobre els conceptes de Ruby on Rails.

Facilita al programador la interacció amb la base de dades mitjançant el patró ActiveRecord. A més, fa ús del patró Model Vista Controlador. Algunes de les característiques importants són:

- És compatible amb PHP4 i PHP5.
- CRUD (alta, baixa, modificació i consulta) de la base de dades ja integrat.
- URLs amigables.
- Sistema de plantilles ràpid i flexible.
- Helpers per a AJAX, Javascript, HTML, formularis i altres.
- Treballa sobre qualsevol carpeta del lloc.
- Incorpora validació de formularis integrada.
- Disposa de components de seguretat i de sessió.
- Les seves funcionalitats es poden ampliar a través de plugins i components.

7.2.2.2. Smarty

Smarty és un motor de plantilles per a PHP, és a dir, separa el codi PHP, com a lògica de negoci, del codi HTML, com a lògica de presentació, i genera continguts web mitjançant la col·locació d'etiquetes Smarty en un document. Es troba sota la llicència GNU, que és lliure.

És comú en grans projectes on el rol de dissenyador gràfic i el de programador es realitzen per diferents persones. No obstant, la programació PHP té la tendència a combinar aquestes dues feines en una sola persona i dins del mateix codi, i a l'hora de fer modificacions dificulta la tasca del programador, Smarty té com a objectiu solucionar aquest problema.

A més té una neteja basada en etiquetes de sintaxi, i també ofereix un gran nombre d'eines per a gestionar la presentació.

7.2.2.3. CodeIgniter

CodeIgniter està construït per a programadors PHP que necessiten un conjunt d'eines simples i elegants per crear totes les funcions d'aplicacions web. L'objectiu principal és ajudar als desenvolupadors que puguin fer projectes molt més ràpid que creant tota l'estructura des de zero.

Algunes característiques destacables:

- Té una àmplia llibreria ben estructurada i molta documentació.
- És altament extensible.
- Es basa en l'estil de programació MVC.
- Ofereix eines de validació de formularis, proporciona una classe per a validar els formularis de manera senzilla i més segura (prevé atacs XSS o per injecció SQL), i ens permet crear funcions pròpies de validació de dades.
- Possibilitat de paginar dades i utilitzar taules.
- Ofereix acoblament amb tecnologies com Ajax i llibreries JQuery.
- Té una fàcil instal·lació.

7.2.2.4. Symfony

Symfony és un complet framework dissenyat per a optimitzar el desenvolupament d'aplicacions web basades en el MVC. Separa la lògica de negoci, la lògica de servidor i la presentació de l'aplicació web. Facilita varies eines i classes encaminades a reduir temps al desenvolupament d'una aplicació complexa. A més, automatitza les tasques més comunes, permetent al desenvolupador dedicar-se totalment als aspectes específics de cada aplicació.

Alguns dels punts destacables:

- Permet internacionalització per a la traducció del text de la interfície, de dades i de contingut de localització.
- La presentació utilitza templates i layouts que poden ser construïts per dissenyadors d'HTML que no tinguin coneixements de framework.
- Els formularis suporten la validació automàtica, assegurant millor qualitat de dades.
- L'usabilitat de cache redueix la utilització d'amplada de banda i la càrrega del servidor.
- La facilitat de suportar autenticació i credencials facilita la creació d'àrees restringides i d'utilització de seguretat dels usuaris.

- Les llistes són més amigables, permet la paginació, classificació i filtratge automàtic.
- Els plugins provenen un alt nivell d'extensibilitat.
- La interacció amb Ajax és molt senzilla.

7.2.2.5. Conclusió



De les opcions que s'haurien proposat de frameworks, al final es va escollir el **Symfony**, ja que ofereix una documentació molt senzilla i completa, sumat al fet que utilitza un lèxic idèntic al PHP, facilitant-ne el seu aprenentatge. A més, té una fàcil instal·lació que només comporta la inserció de la carpeta descarregada amb totes les llibreries que la componen dins de la carpeta del projecte. La versió actual del framework és la 2.3.

7.2.3. Llibreries

A partir de les decisions preses anteriorment corresponents al llenguatge de programació (apartat 7.2.1.6) i al framework (apartat 7.2.2.4), en aquest punt escollim unes llibreries, que són un conjunt de subprogrames utilitzats per a desenvolupar software, els quals ens ajudaran a realitzar diverses tasques dins del projecte.

A continuació hi ha detallades les diferents llibreries que s'han utilitzat.

7.2.3.1. Doctrine 2

Doctrine 2 és un ORM de PHP. ORM es pot traduir com "mapeig objecte-relacional". En concret és una tècnica o patró arquitectònic que permet comunicar dos sistemes

diferents, normalment una base de dades relacional amb objectes d'un llenguatge orientat a objectes mitjançant un sistema que mapeja/vincula ambdós sistemes.

Existeixen ORMs per la majoria de llenguatges de programació, per exemple Hibernate per Java, ADO.NET Entity Framework per .NET i Doctrine és el més utilitzat per PHP. Està inspirat en Hibernate.

A continuació en destaquem algunes de les característiques:

- Molt senzill de configurar
- Té el seu propi dialecte de SQL anomenat DQL (Doctrine Query Language)
- Pot generar els models (les classes PHP que representen una fila de la base de dades) a partir de la base de dades. Només s'han de definir les dependències entre ells.
- De la mateixa manera, també pot generar la base de dades a partir dels models.

El benefici de Doctrine pel programador és la capacitat de centrar-se exclusivament en la lògica del negoci orientada a objectes i preocupar-se de la persistència només com a una feina secundària.

A continuació es mostra la instrucció shell per crear una entitat:

```
C:\xampp\www\Symfony>php app\console doctrine:generate:entity
```

Un cop executada, s'hauran de contestar totes les preguntes necessàries per generar-la com per exemple: el nom complet de l'entitat, el format amb el que es volen les anotacions, i la descripció de tots els atributs.

Un cop creades les entitats corresponents, podem crear les taules a la base de dades amb la comanda:

```
C:\xampp\www\Symfony>php app\console doctrine:schema:create
```

Si revisem la base de dades amb phpMyAdmin podem veure com les taules han estat creades i podrem apreciar com Doctrine, mitjançant les nostres Entitats, coneix perfectament com crear-les.

Ara que ja tenim les Entitats, s'han de crear les relacions entre elles. Triarem, per exemple, la relació entre Client i Vehicle. Un Client pot tenir molts Vehicles, i un Vehicle només pertany a un Client. Per tant, haurem d'afegir una clau forana apuntant a la taula Client. Per fer-ho afegirem la següent propietat amb els respectius getter i setter a l'entitat Vehicle:

```
/**
 * @ORM\ManyToOne(targetEntity="Client", inversedBy="Vehicle")
 * @ORM\JoinColumn(name="Client_id", referencedColumnName="id")
 * @return integer
 */

private $client;

public function setClient(\Taller\TallerBundle\Entity\Client $client)
{
    $this->client = $client
}

public function getClient()
{
    return $this->client;
}
```

Amb aquest codi estem definint la relació entre les 2 taules.

L'anotació **@ORM\ManyToOne**: defineix que des d'aquesta taula de vehicles existeix una relació de molts a un amb l'entitat Client.

I la **@ORM\JoinColumn**: especifica les columnes que s'utilitzaran per fer el join.

Amb això hem modelat la clau forana des del punt de vista de l'entitat Vehicle, també s'haurà de fer a l'entitat Client amb el següent codi:

```
/**
 * @ORM\OneToMany(targetEntity="Vehicles", mappedBy="client")
 */

private $comments;

public function __construct(){
    $this->comments = new \Doctrine\Common\Collections\ArrayCollection();
}

public function addVehicles(\Taller\TallerBundle\Entity\Vehicle
$vehicles){
    $this->vehicles[] = $vehicles;
}

public function getVehicles(){
```

```
        return $this->vehicles;
    }
```

Afegint la propietat *\$vehicles*, estem creant la referència amb l'altra taula ja que un client pot tenir varis vehicles. Estem utilitzant l'anotació inversa **@ORM\OneToMany**. També es pot veure com s'afegeix un constructor que inicialitza la propietat amb un objecte del tipus *ArrayCollection*, que ens permetrà que un client contingui varis vehicles i poder-los obtenir amb el mètode *getComments()*.

Amb aquestes modificacions realitzades, hem de tornar a generar les taules, però al estar ja creades, primer les haurem d'esborrar amb la comanda:

```
C:\wamp\www\Symfony>php app\console doctrine:schema:drop -force
```

I tornar-les a crear:

```
C:\wamp\www\Symfony>php app\console doctrine:schema:create
```

El framework Doctrine, al tenir les dades de la base de dades i de les taules, ens proporciona un suport molt potent per treballar amb elles.

A continuació s'explica com manipular les dades i d'aquesta manera podrem proporcionar més informació a les entitats per ajudar a validar les dades entrades als nostres camps. L'accés a les entitats es fa mitjançant un objecte de Doctrine anomenat **EntityManager**, que podríem dir que és l'administrador de les entitats i qui interactuarà amb elles. L'invocarem de la següent manera:

```
$em = $this->getDoctrine()->getEntityManager();
```

Dins la variable *\$em* tenim la referència al nostre objecte. Ara, per treballar amb les dades es necessita un repositori. Aquest, ens permetrà sol·licitar i actualitzar dades, i per utilitzar-lo només es cridarà el nom lògic de l'entitat:


```
$em->get Repository('TallerBundle:Client');
```

Per obtenir dades de les taules, tenim els mètodes:

- **findAll()**: Obté tots els registres d'una taula. Retorna un array.
- **find()**: Obté un registre a partir de la clau primària de la taula.
- **findBy()**: Obté els registres trobats poguent passar com argument els valors que anirien dins d'un WHERE. Retorna un array.
- **findOneBy()**: Obté un registre poguent passar com argument els valors que anirien dins un WHERE.

Un exemple d'utilització d'aquests mètodes seria:

```
$em = $this->getDoctrine()->getEntityManager();  
/-- Obtenim tots els Clients de la taula  
$clients = $em->getRepository('TallerBundle:Client')->findAll();
```

O,

```
-- Obtenim tots els Clients que es diguin Josep de Girona  
$clients = $em->getRepository('TallerBundle:Client')->findBy(  
    array(  
        'nom' => 'Josep',  
        'poblacio' => 'Girona'  
    )  
);
```

Un cop tenim totes les eines per treballar, només queda mostrar els resultats obtinguts. Per fer-ho, necessitem els Controladors i les plantilles TWIG. Un fragment del fitxer de routing seria:

```
client_llistar:  
    pattern: /clients/llistar  
    defaults: { _controller: TallerBundle:Clients:llistar }
```

El *controller* que apareix al fitxer de ruta serà el fitxer on hi hauran les accions que necessitem per l'aplicació.

Ara creem una plantilla amb HTML per visualitzar la resposta a les nostres accions del controlador: Els fitxers els crearem a `src\Taller\TallerBundle\Resources\views\Clients\` amb el nom **listar.html.twig**.

Aquesta plantilla, donat una taula de clients, mostra les dades:

```
<h1> Llistat de clients </h1>
<table border="1">
  <tr>
    <th> DNI </th>
    <th> Nom </th>
    <th> Cognoms </th>
  </tr>
  {% for client in clients %}
  <tr>
    <td>{{ client.dni }}</td>
    <td>{{ client.nom }}</td>
    <td>{{ client.cognoms }}</td>
  </tr>
  {% endfor %}
</table>
```

Altres exemples de manipulació de dades són:

- El mètode `crearAction()` del controlador serveix per insertar registres. Cal destacar que s'ha d'afegir al controlador el namespace de l'entitat amb la paraula reservada `use`.

```
$client = new Clients();
$client ->setDNI('40404040D');
$client ->setNom('Josep');
$client ->setCognoms('Garriga Bosch');
[...]
$client ->setProvincia('Girona');
```

Utilitzant els getters i setters insertem les dades.

- Per insertar les dades utilitzarem l'EntityManager:

```
$em = $this->getDoctrine()->getEntityManager();
$em->persist($client);
$em->flush();
```

El mètode *persist()* indica que l'objecte passat per argument serà guardat per insertar-lo i el mètode *flush()* l'inserta. Amb aquesta ordre, Doctrine generarà la sentència INSERT.

Si amb els mètodes explicats no és suficient, Doctrine ens permet treballar amb un llenguatge molt semblant al SQL estàndar, s'anomena DQL (Doctrine Query Language). Una de les poques diferències és que en lloc de fer les consultes sobre registres de les taules, es fan sobre els objectes de tipus Entitat. Veiem-ne un exemple:

```
SELECT * FROM clients
```

En DQL seria:

```
SELECT a FROM TallerBundle:Client c
```

Per a què Doctrine executi aquesta sentència, es fa a través de *EntityManager*:

```
$em = $this->getDoctrine()->getEntityManager();  
$dql = "select a from TallerBundle:Client c";  
$query = $em->createQuery($dql);  
$clients = $query->getResult();
```

7.2.3.2. CRUD de Doctrine

El CRUD no és una llibreria més, però ho hem separat perquè és un mòdul prou important.

Un cop s'han generat totes les classes del formulari, Doctrine ens permet crear un mòdul de Symfony que permeti treballar amb tots els objectes a través del navegador. És a dir, generarà les operacions bàsiques que es realitzen contra la base de dades. Per fer-ho utilitzarem el CRUD (Create, Read, Update and Delete).

L'instrucció shell és:

```
$ php app/console doctrine:generate:crud
```

A continuació, anirà fent preguntes per configurar totes les classes i el fitxer de routing.

Un cop generat, només quedarà personalitzar les plantilles.

7.2.3.3. PdfBundle

Aquesta llibreria de Symfony genera fàcilment arxius i imatges en format PDF. Per fer-ho és necessari afegir l'anotació *@Pdf()* al controlador:

```
use Ps\PdfBundle\Annotation\Pdf;

/**
 * @Pdf()
 */
public function imprimir_facturaAction($f)
{
    $format = $this->get('request')->get('_format');

    return $this->render(
        sprintf('ClientsBundle:DefaultController:factura.%s.twig',
            $format), array('f=> $f)
    );
}
```

A més de crear 2 plantilles, la plantilla HTML, i una altra amb les etiquetes adequades per generar l'arxiu en PDF.

Plantilla HTML:

```
Factura número {{ f.numero }}
Client {{ f.client.nom }}

<table>
    {% for item in f.items %}
        <tr>
```

```
        <td>{{ item.codi }}</td>
        {# ... #}
    </tr>
    {% endfor %}
</table>
```

Plantilla per generar el PDF:

```
<pdf>
  <dynamic-page>
    Factura número {{ f.numero }}
    Client {{ f.client.nom }}

    <table>
    {% for item in f.items %}
      <tr>
        <td>{{ item.codi }}</td>
        {# ... #}
      </tr>
    {% endfor %}
    </table>

    {# ... #}
  </dynamic-page>
</pdf>
```

7.3. Programari

En aquest apartat hi ha explicats els possibles sistemes operatius, juntament amb la meua elecció, i a continuació els diferents entorns de treball.

Seguidament, a partir de decisions preses a partir d'estudis anteriors, s'escolleixen uns entorns de treball compatibles.

7.3.1. Sistemes Operatius

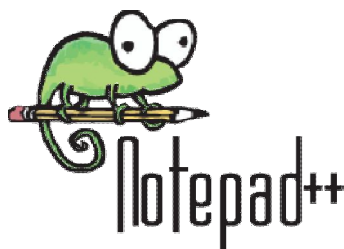
Degut a que l'ordinador utilitzat disposava d'una versió del sistema operatiu Windows 7, s'ha optat per treballar en aquest entorn. L'elecció d'un altre sistema operatiu hagués comportat més temps per a l'instal·lació.

De totes maneres, és completament viable utilitzar l'aplicació en un altre sistema operatiu com Mac OSX, Ubuntu, etc.

7.3.2. Entorns de treball

A partir dels llenguatges de programació que s'han escollit anteriorment (apartat 7.2.1.6), el framework (apartat 7.2.2.4) i les llibreries escollides (apartat 7.2.3.4), i juntament amb el sistema operatiu seleccionat (apartat 7.3.1), a continuació s'expliquen els diferents entorns de treball que hem decidit utilitzar ja que són compatibles amb les decisions preses anteriorment.

7.3.2.1. Notepad ++



Notepad ++ és un editor de text i de codi font lliure amb suport per diferents llenguatges de programació. Té molta semblança amb el "Bloc de notes" de Windows en quant al fet de que pot editar text sense format de manera senzilla. Inclou opcions més avançades que poden ser útils per usuaris avançats com desenvolupadors i programadors com el fet de ser capaç de reconèixer el llenguatge de programació i ressaltar les expressions pròpies de la sintaxi per facilitar-ne la lectura, poder obrir diferents documents amb pestanyes per poder treballar millor, o fins i tot, el programador pot definir el seu propi llenguatge.

També pot utilitzar-se en GNU/Linux mitjançant Wine. Es distribueix sota els termes de llicència GNU.

7.3.2.2. XAMPP



XAMPP de l'acrònim X (per a qualsevol dels diferents sistemes operatius), A (Apache), M (MySQL), P (PHP) i P(Perl), és un paquet de programari lliure que conté el servidor HTTP Apache, base de dades MySQL i eines necessàries per utilitzar el PHP i el llenguatge de programació Perl. El programa es llença sota el GNU i un servidor web, de fàcil ús, capaç de servir pàgines dinàmiques.

El programa només requereix descarregar i executar un arxiu *.zip*, *.tar* o *.exe*, realitzar unes petites configuracions en alguns dels components que el servidor Web necessitarà. També inclou altres mòduls com OpenSSL i phpMyAdmin.

7.3.2.3. phpMyAdmin



PhpMyAdmin és una eina escrita en PHP amb la intenció de manejar l'administració de MySQL a través de pàgines web, utilitzant un navegador. Actualment pot crear i eliminar Bases de dades, crear, eliminar i alterar taules, esborrar, editar i afegir camps, executar qualsevol sentència SQL, administrar claus en camps, administrar privilegis, exportar dades en diversos formats i està disponible en 50 idiomes. Es troba disponible sota la llicència GPL. Aquest projecte es troba vigent des de l'any 1998, i va ser avaluat com el millor en la comunitat de descàrregues de SourceForge.net com la descàrrega del mes de Desembre del 2002. Donat que aquesta eina corre en màquines amb

Servidors Web i Suport de PHP, i MySQL, la tecnologia utilitzada ha anat variant durant el desenvolupament.

8. Anàlisi i disseny del sistema

8.1. Anàlisi

En aquest apartat s'explicaran les diferents parts que té l'aplicació, els diagrames de casos d'ús generals pels diferents usuaris de l'aplicació, i els diagrames i fitxes de casos d'ús concrets.

8.1.1. Parts de l'aplicació

Un cop decidits els requisits i escollides les eines que s'utilitzaran, analitzarem les diferents parts que tindrà l'aplicació.

L'aplicació contindrà un menú principal visible a cadascuna de les pàgines per a tenir més facilitat d'accés a qualssevol de les parts. Aquestes són:

- Taller
- Històric
- Caixa
- Informes
- Consultes
- Base de dades
- Configuració

Per detallar una mica més els termes anteriors, a continuació fem una definició de cada part:

- Taller: és la pantalla principal de l'aplicació encarregada de la gestió del taller. Aquí es podrà crear una nova recepció quan un vehicle entra al taller, veure els pressupostos que encara no han estat acceptats i els vehicles que estan en reparació.

- Històric: en aquest apartat es podrà consultar l'historial d'un vehicle (totes les reparacions fetes) i els vehicles d'un client.
- Caixa: aquesta pestanya permetrà gestionar els cobraments de factures, es marcarà com a pagada una factura a partir del número de factura.
- Informes: es podran imprimir duplicats de factura, albarans, pressupostos. I facturar els albarans si el client només vol una única factura a final de mes, per exemple.
- Consultes: avarca tots els tipus de llistats que es poden crear, com per exemple: llistats de clients, de vehicles, llistar les factures pendents de cobrar.
- Base de dades: conté les opcions d'alta, modificació i baixa de client, vehicle, article, definir el preu de ma d'obra, les quotes de reciclatge. I assignar un client a un vehicle.
- Configuració: en aquest mòdul es troba tot el necessari per configurar l'aplicació per les necessitats del taller. Definir els percentatges d'IVA, els logos que s'utilitzaran a les factures, els tipus de formes de pagament, les dades de l'empresa, modificar les anotacions de les factures i pressupostos, canviar l'any contable. També es podran donar d'alta usuaris i canviar la contrasenya.

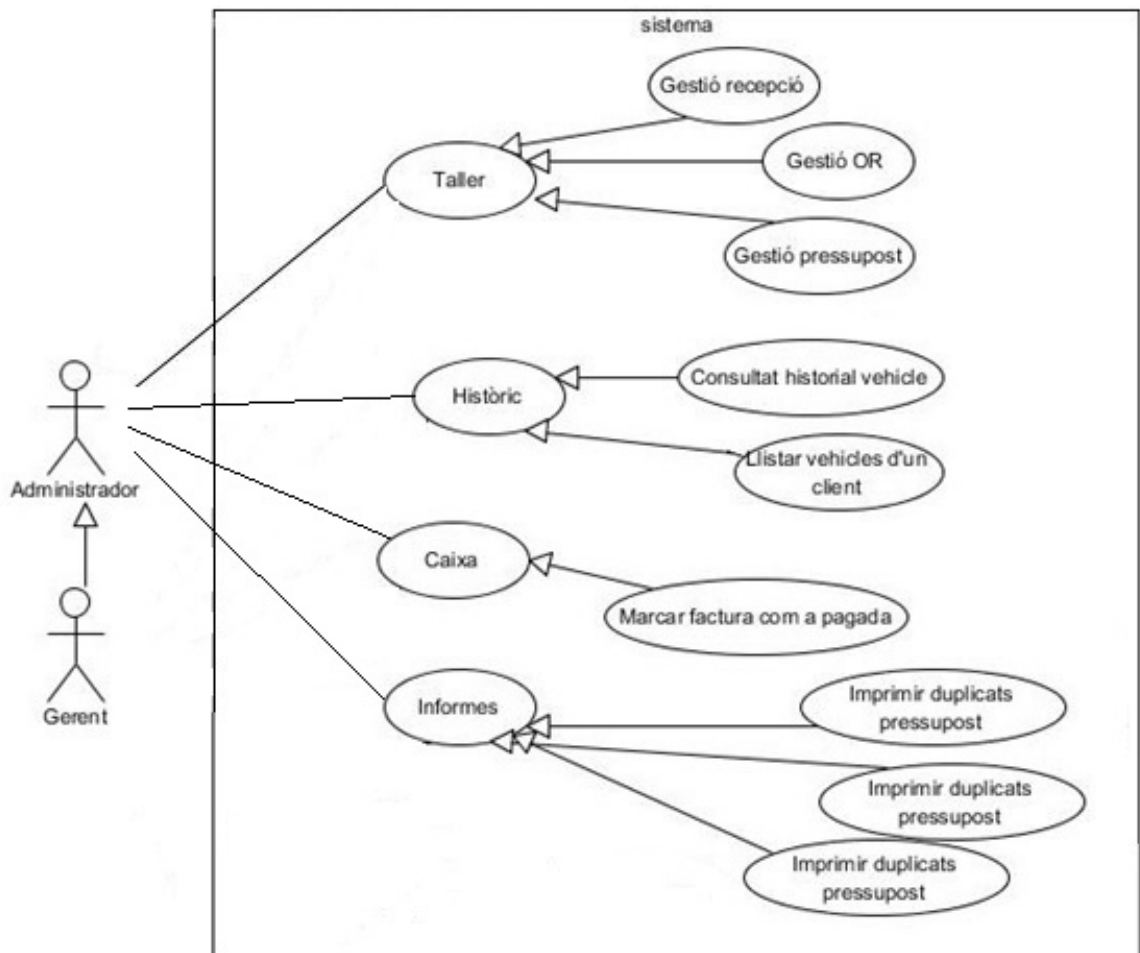
Un cop definides quines parts ha de tenir l'aplicació, a continuació es veuran especificades les diferents funcionalitats que té el programa, és a dir, què podrà fer cada usuari a cadascuna de les pantalles de l'aplicació.

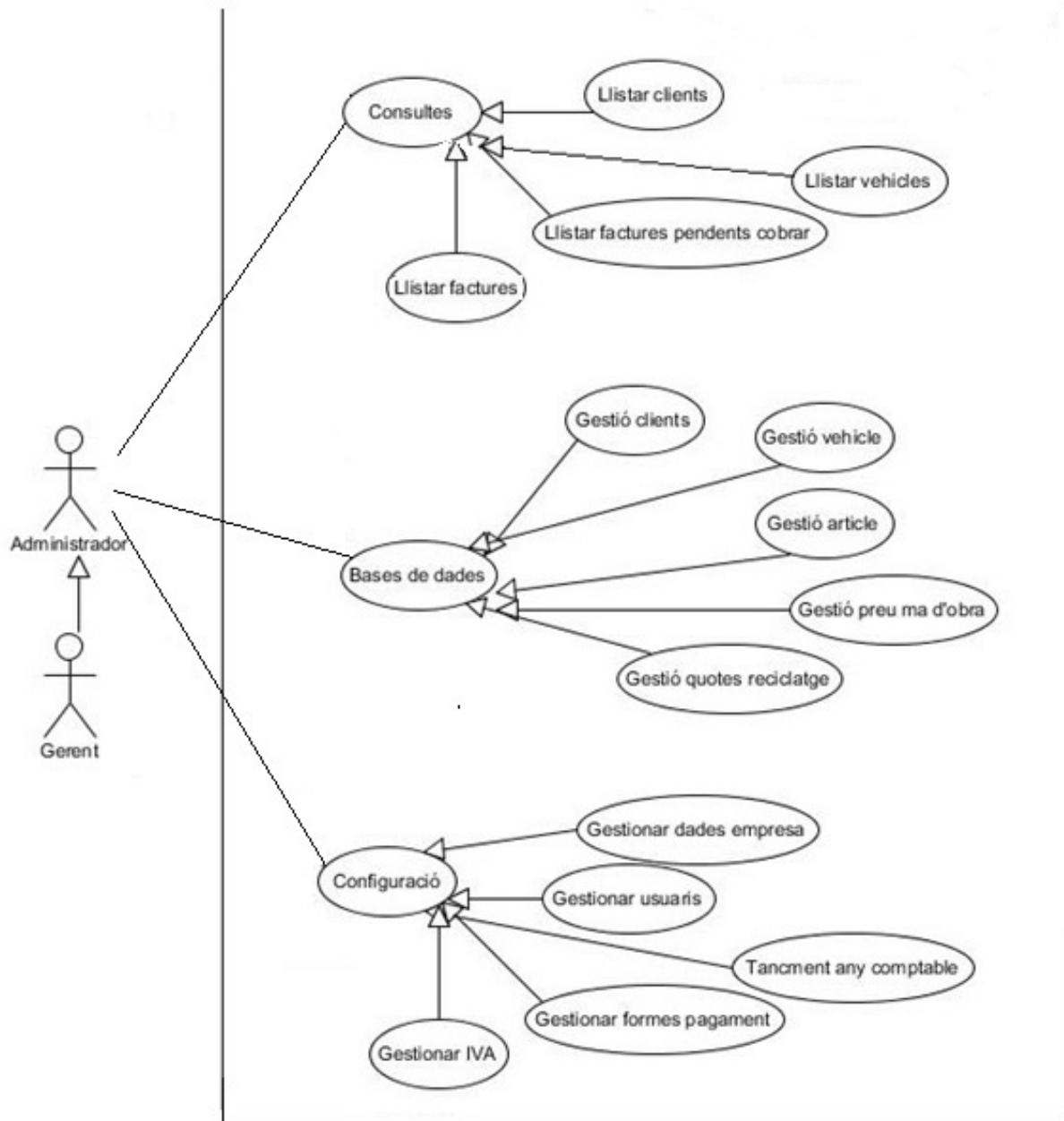
A continuació es mostren els diagrames de cas d'ús de forma molt general per a mostrar les funcionalitats del sistema, separats per cada mòdul dels esmentats anteriorment.

8.1.2. Diagrama de cas d'ús general

L'aplicació consta de diferents usuaris amb diferents permisos. Per aquesta raó, molts dels diagrames que es mostrin a continuació estaran representats amb dues parts, o bé introduint els diferents actors dins del mateix diagrama, segons s'hagi cregut més entenedor.

A continuació es mostra el diagrama de cas d'ús general, fa referència a l'usuari administrador i gerent. Per fer-lo més entenedor, s'ha separat en dues parts:

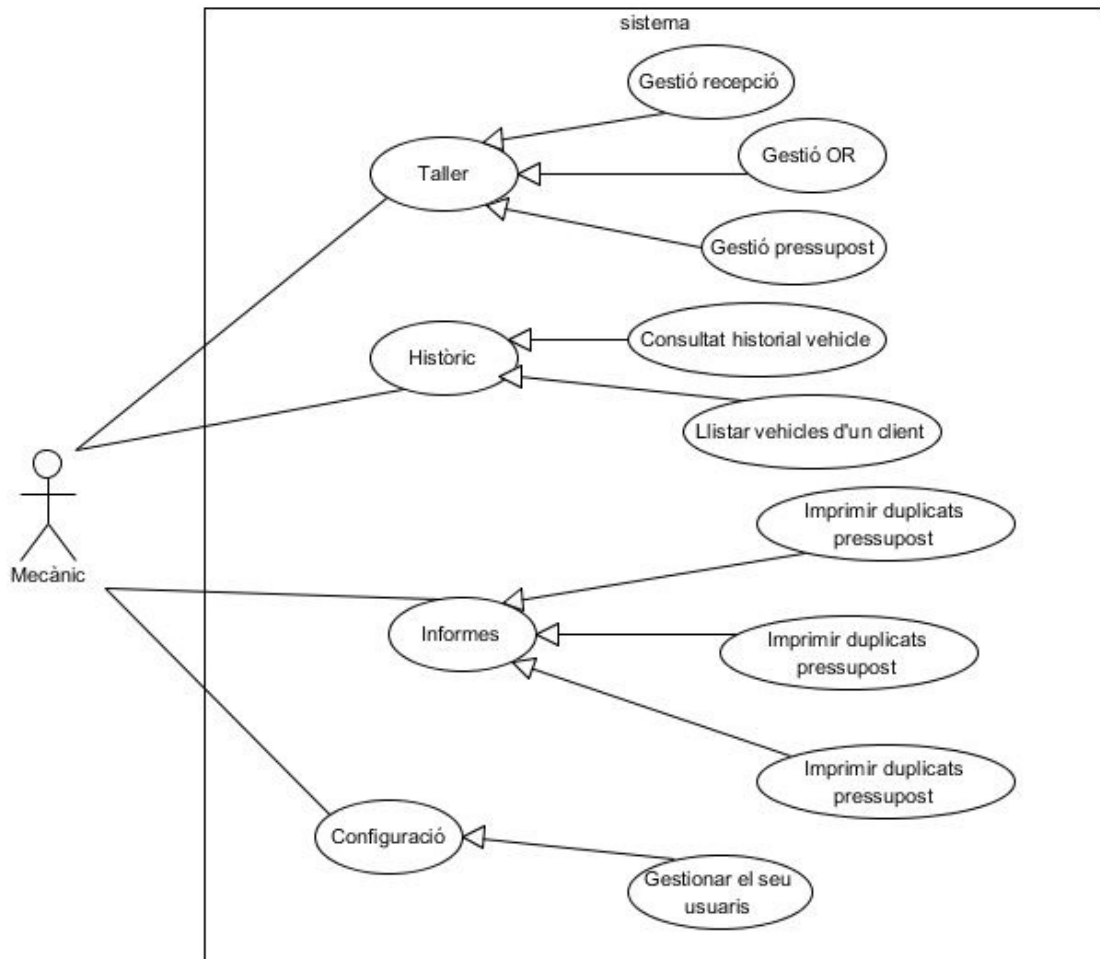




Tal com es pot apreciar, el sistema queda obert perquè s'ha separat el diagrama en dues pàgines. S'han repetit els usuaris per facilitar-ne la comprensió.

En aquest diagrama tant general no podem apreciar la diferència de privilegis entre administrador i gerent. L'única diferència és que el gerent no pot donar de baixa cap element. Més endavant, als diagrames concrets quedarà reflectit més clarament.

A continuació es mostra el segon diagrama de cas d'ús general amb l'actor corresponent al mecànic o operari, que té menys privilegis que l'administrador o gerent:



Als diagrames anteriors es pot apreciar de forma molt general el funcionament de l'aplicació. A continuació es fa una breu descripció de cada mòdul, per a poder entendre-ho una mica millor.

Primerament cal destacar que qualsevol dels usuaris que vol accedir a l'aplicació primer ha de passar per una pantalla de seguretat, on haurà d'introduir el nom d'usuari i la contrasenya. En cas contrari, no tindrà accés a l'aplicació.

Taller

En aquest mòdul es podran realitzar les accions referents al funcionament del taller, tots els usuaris podran accedir a totes les pantalles respectives. Aquestes són:

- Gestionar recepcions
- Gestionar ordres de reparació
- Gestionar pressupostos

Depenent del tipus de permís que tingui l'usuari, podrà realitzar més o menys operacions:

- Mecànic: A totes les pantalles tindrà les mateixes opcions a realitzar, que seran visualitzar-ne les dades, escollir-ne una i modificar-ne les dades. També en podran crear de noves.
- Gerent o administrador: Tenen les mateixes opcions que l'usuari mecànic però a més, podran donar de baixa qualssevol dels 3 elements (recepció, pressupost o ordre de reparació). Realment, mai s'eliminarà una entitat al donar-la de baixa, només es marcarà com a anul·lada.

Històric

En aquest apartat es podrà consultar:

- L'historial de reparacions d'un vehicle
- Els vehicles del que és propietari un client

En aquest cas, tots els usuaris tenen els mateixos permisos d'accés.

Caixa

A l'apartat caixa, podrem marcar com a pagada una factura a partir del número de factura.

L'usuari, en aquest cas amb permís de gerent o administrador, és l'únic que hi podrà accedir ja que els operaris no cobraran mai les factures als clients.

Informes

En aquest apartat tot els usuaris podran accedir a les pantalles que es corresponen amb:

- Imprimir duplicats de factura
- Imprimir duplicats d'albarans
- Imprimir duplicats de pressupostos
- Facturar albarans

Segons el tipus de permís l'usuari, podrà fer diferents opcions:

- Mecànic: Aquests usuaris podran imprimir duplicats d'albarans i de pressupostos, que són els que hauran creats ells mateixos i hauran d'entregar als clients.
- Gerent o administrador: Els usuaris pertanyents a aquest rol podran accedir a les 4 opcions.

Consultes

El mòdul de consultes avarca tots els tipus de llistats que es poden crear:

- Llistats de clients
- Llistar factures emeses
- Llistats de vehicles
- Llistar les factures pendents de cobrar

Els únics usuaris que hi tindran accés seran amb rol gerent o administrador.

Base de dades

Conté les opcions de:

- Gestió de client
- Gestió de vehicle (quan es dona d'alta un vehicle nou, s'assigna el client al vehicle per definir-ne el propietari)
- Gestió d'article
- Gestió del preu de ma d'obra
- Gestió de quotes de reciclatge

Segons el tipus de permís que tingui l'usuari, podrà fer diferents opcions:

- Mecànic: a partir de la pantalla d'alta d'una recepció, podrà donar d'alta un client o un vehicle, per fer-ne la seva reparació, però des de l'apartat base de dades, no podrà accedir a cap de les opcions.
- Gerent: aquests usuaris podran donar d'alta i modificar qualssevol de les opcions.
- Administrador: amb permís d'administrador es podran accedir a totes les opcions i a més, donar de baixa les entitats corresponents. S'ha de tenir en compte que quan es dona de baixa un element, no s'esborra de la base de dades, es marca com a anul·lat.

Configuració

En aquest mòdul es troba tot el necessari per configurar l'aplicació per les necessitats de l'empresa:

- Gestionar l'IVA
- Gestionar les formes de pagament
- Fer el tancament comptable de l'any
- Gestionar usuaris
- Gestionar empresa. Contindrà els logos que s'utilitzaran a les factures, les dades de l'empresa, les anotacions que apareixen a les factures i pressupostos,...

Depenent del tipus de permís, es podrà accedir o no a les pantalles que s'ofereixen:

- Mecànic: els usuaris amb aquest rol podran accedir a gestionar el seu usuari. Podran canviar-ne les dades i la contrasenya per accedir a l'aplicació.
- Gerent: podran visualitzar i modificar totes les pestanyes d'aquest mòdul, però no donar-ho de baixa.
- Administrador: tindran les mateixes característiques que els usuaris anteriors i a més, podran donar de baixa les dades de l'empresa, els usuaris i les formes de pagament. Com s'ha dit anteriorment, no es dona de baixa sinó que es marca com a anul·lat.

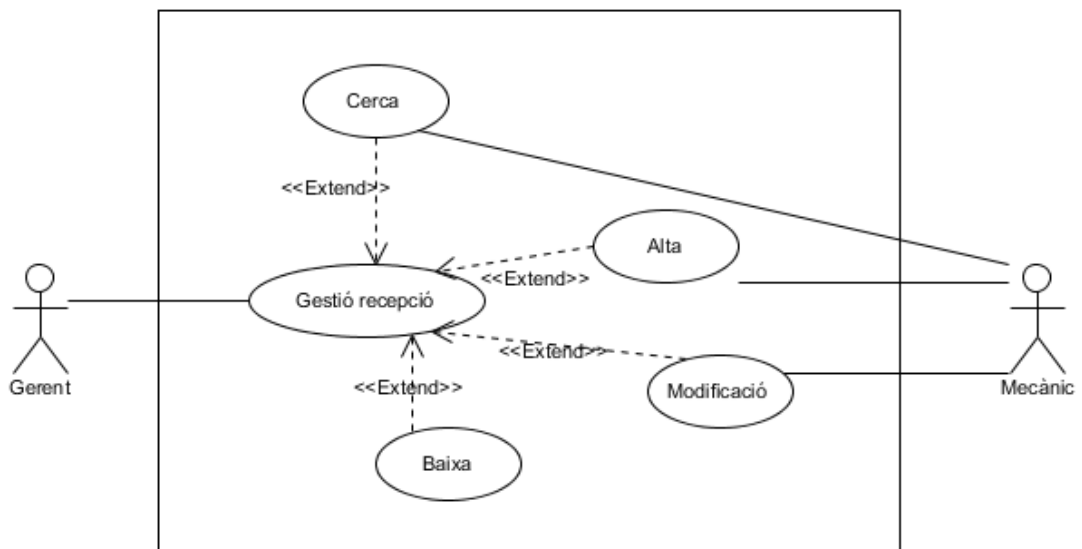
8.1.3. Diagrames i fitxes de cas d'ús de cada mòdul

A continuació apareixen tots els diagrames i fitxes de casos d'ús de l'aplicació separats pels mòduls explicats anteriorment. En els diagrames que no es mostra l'administrador, és perquè té els mateixos permisos que l'usuari amb rol gerent. L'administrador es mostra només en algun cas per veure com s'executen les baixes.

A les fitxes s'explicarà el cas tant dels gerents com dels mecànics. Si fos complicat d'entendre en un únic diagrama, se'n faran els necessaris.

8.1.3.1. Mòdul taller

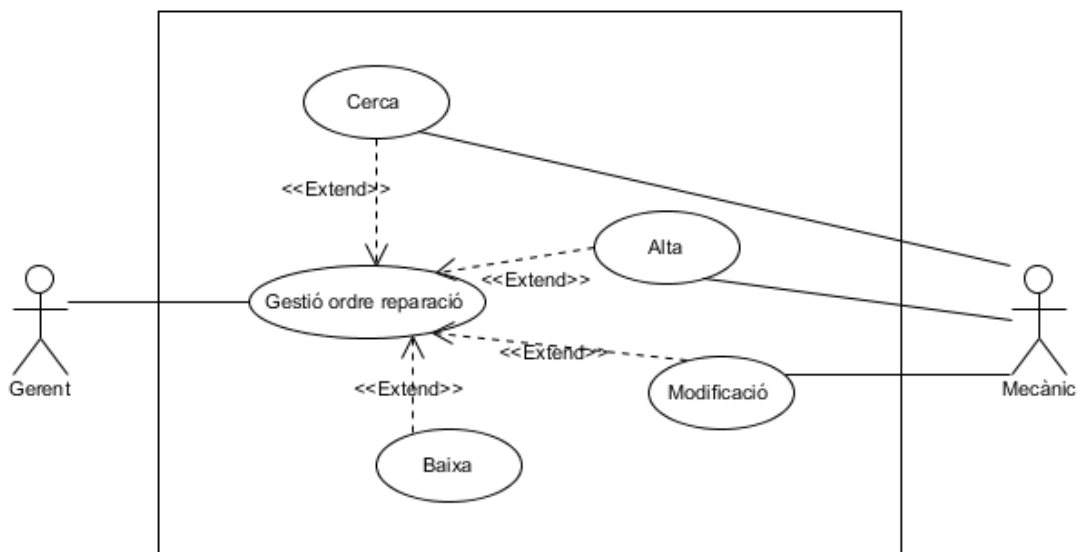
8.1.3.1.1. Gestió recepció



Gestió recepció	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació o pestanya taller 2. Obrir pantalla d'alta <ol style="list-style-type: none"> 2.1. Entrar dades client i vehicle 2.2. Guardar 3. Obrir pantalla de cerca <ol style="list-style-type: none"> 3.1. Introdur matricula o nom client per trobar recepció 3.2. Buscar 4. Obrir pantalla de modificació <ol style="list-style-type: none"> 4.1. Entrar dades a modificar 4.2. Guardar

	5. Obrir pantalla baixa Si l'usuari és Administrador o Gerent, 5.1. Escollir opció eliminar 5.2. Eliminar
Flux alternatiu	2.1, 3.1, 4.1 i 5.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

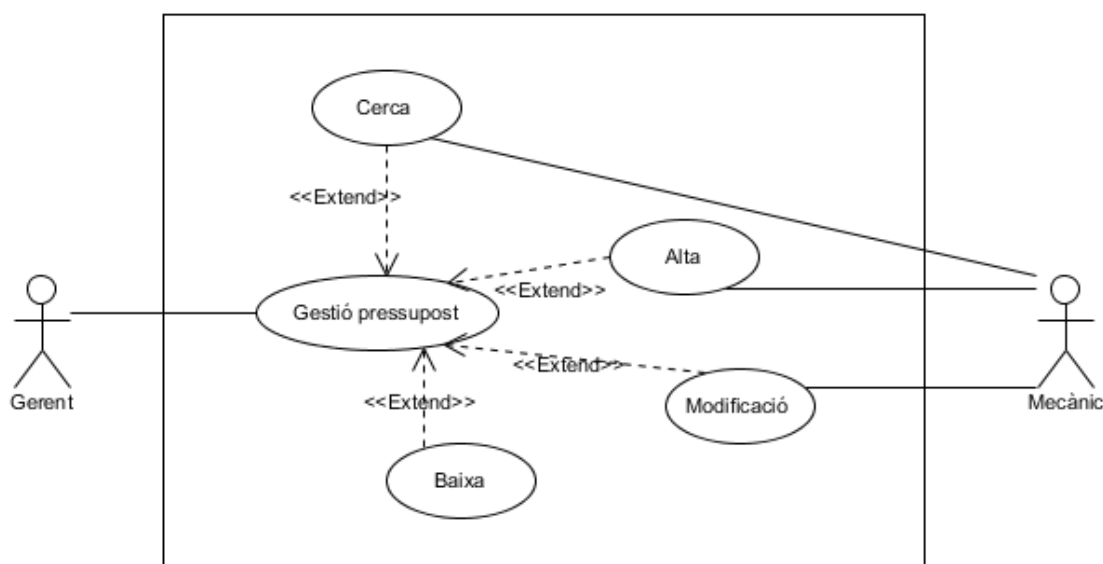
8.1.3.1.2. Gestió ordre de reparació (O.R.)



Gestió ordre de reparació	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació o pestanya taller <ol style="list-style-type: none"> 1.1. Escollir recepció o ordre de reparació del llistat de vehicles que estan en reparació al taller 2. Obrir pantalla d'alta (si és una recepció) <ol style="list-style-type: none"> 2.1. Entrar dades client i vehicle 2.2. Guardar 3. Obrir pantalla de cerca <ol style="list-style-type: none"> 3.1. Introduir matrícula o nom client per trobar recepció 3.2. Buscar 4. Obrir pantalla de modificació (si és una ordre de reparació) <ol style="list-style-type: none"> 4.1. Entrar dades a modificar o afegir 4.2. Guardar 5. Obrir pantalla baixa <p>Si l'usuari és Administrador o Gerent,</p> <ol style="list-style-type: none"> 5.1. Escollir opció eliminar 5.2. Eliminar

Flux alternatiu	2.1, 3.1, 4.1 i 5.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.
-----------------	--

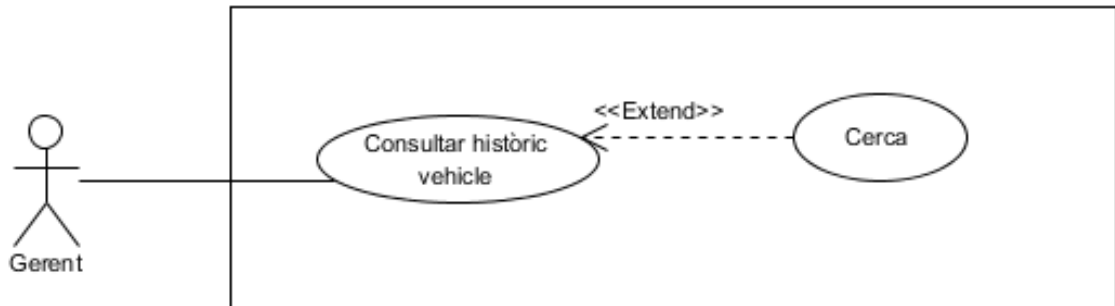
8.1.3.1.3. Gestió pressupost



Gestió pressupost	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació o pestanya taller <ol style="list-style-type: none"> 1.1. Escollir recepció del llistat de vehicles que estan en reparació al taller 2. Obrir pantalla d'alta (si és una recepció) <ol style="list-style-type: none"> 2.1. Entrar dades client i vehicle 2.2. Guardar 3. Obrir pantalla de cerca <ol style="list-style-type: none"> 3.1. Introduir matrícula o nom client per trobar recepció 3.2. Buscar 4. Obrir pantalla de modificació (si és un pressupost) <ol style="list-style-type: none"> 4.1. Entrar dades a modificar 4.2. Guardar 5. Obrir pantalla baixa <p>Si l'usuari és Administrador o Gerent,</p> <ol style="list-style-type: none"> 5.1. Escollir opció eliminar 5.2. Eliminar
Flux alternatiu	2.1, 3.1, 4.1 i 5.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.2. Mòdul històric

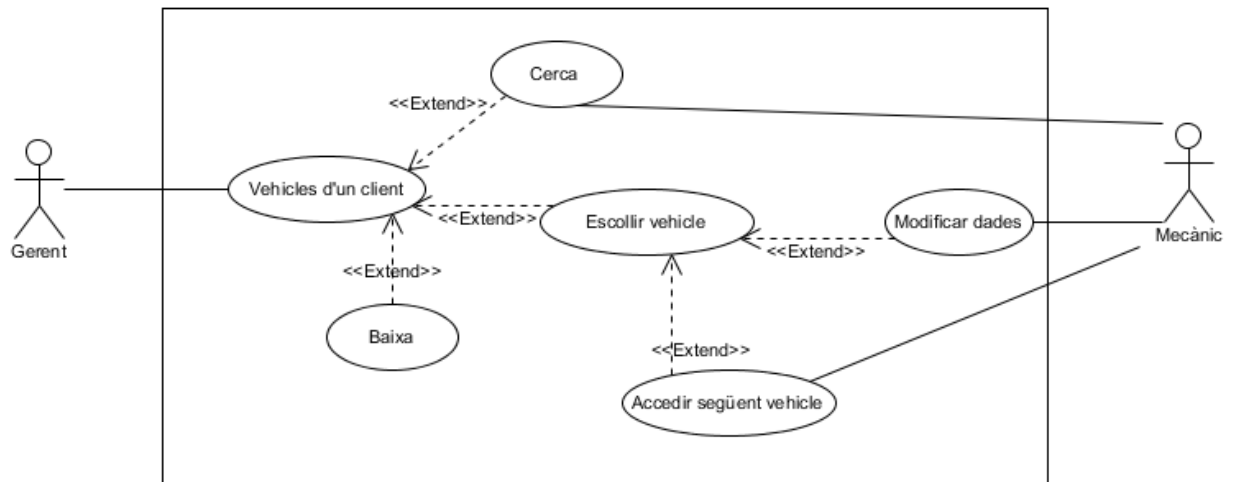
8.1.3.2.1. Consultar històric vehicle



Aquesta fitxa de cas d'ús seria igual tant si l'usuari té rol d'administrador o mecànic.

Consultar històric vehicle	
Versió	Usuari
Actors	Gerent o administrador o mecànic
Flux principal	<ol style="list-style-type: none">1. S'obre la pantalla inicial de l'aplicació2. Escollir pestanya històrics3. Obrir opció cerca<ol style="list-style-type: none">3.1. Entrar matrícula vehicle3.2. Buscar
Flux alternatiu	<ol style="list-style-type: none">3.1. Si les dades són incorrectes o no s'introdueix cap matrícula, mostra un missatge d'error detallat

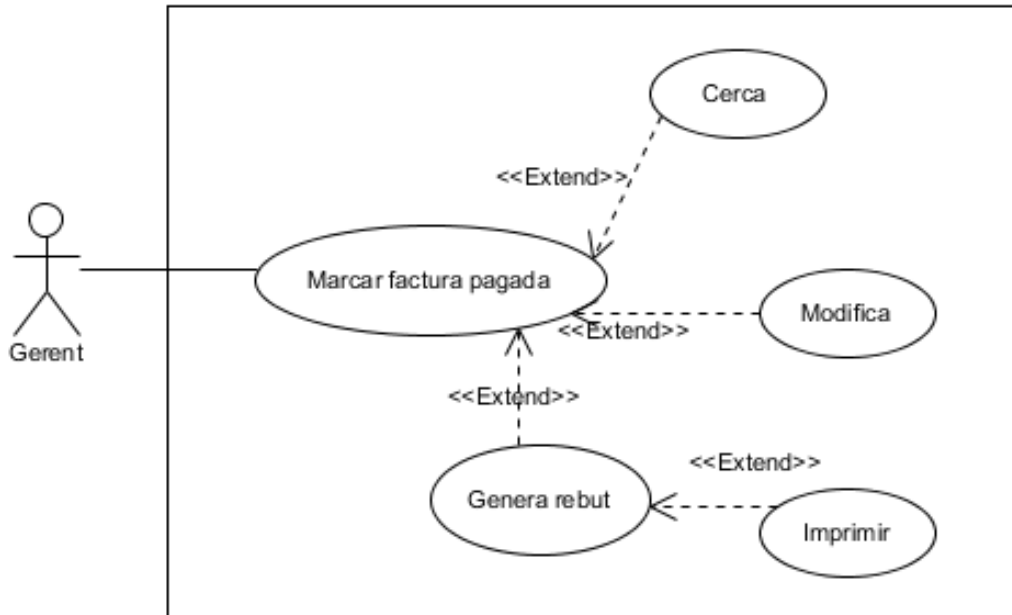
8.1.3.2.2. Vehicles d'un client



Vehicles d'un client	
Versió	Usuari
Actors	Gerent o mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Escollir pestanya històric 3. Obrir opció cerca <ol style="list-style-type: none"> 3.1. Introduir dades client 3.2. Buscar 4. Obrir opció escollir vehicle <ol style="list-style-type: none"> 4.1. Introduir dades client 4.2. Seleccionar vehicle a consultar 4.3. Si es vol veure un altre vehicle del client <ol style="list-style-type: none"> 4.3.1. Seleccionar veure següent 4.4. Si modificar dades <ol style="list-style-type: none"> 4.4.1. Entrar dades 4.4.2. Guardar 5. Si l'usuari és gerent <ol style="list-style-type: none"> 5.1. Escollir l'opció eliminar 5.2. Introduir matrícula a eliminar 5.3. Eliminar
Flux alternatiu	3.1, 4.1 i 4.3.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.3. Mòdul caixa

8.1.3.3.1. Marcar factura com a pagada

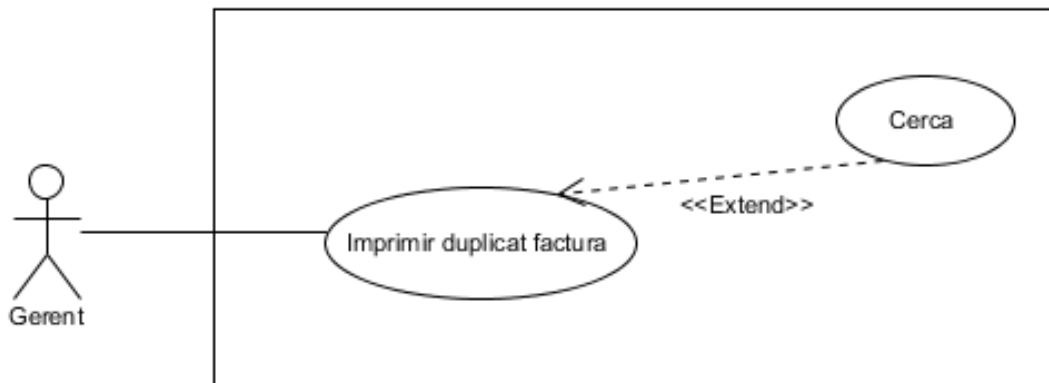


Aquesta fitxa de cas d'ús seria igual tant si l'usuari té rol d'administrador o mecànic.

Marcar factura com a pagada	
Versió	Usuari
Actors	Gerent
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar la pestanya caixa 3. Obrir pantalla de cerca <ol style="list-style-type: none"> 3.1. Entrar dades 3.2. Buscar 4. Obrir pantalla de modificació <ol style="list-style-type: none"> 4.1. Entrar dades 4.2. Guardar 5. Si factura pagada <ol style="list-style-type: none"> 5.1. Generar rebut 5.2. Si el client requereix rebut, imprimir rebut
Flux alternatiu	3.1 i 4.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.4. Mòdul informes

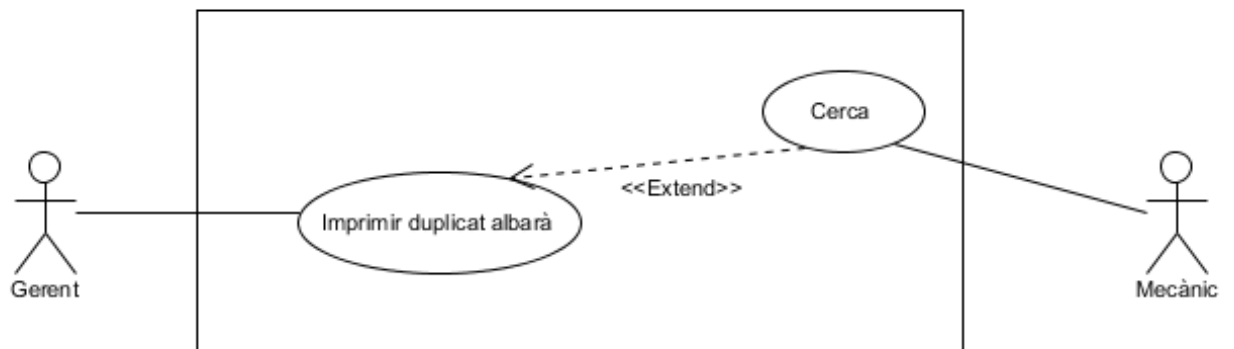
8.1.3.4.1. Imprimir duplicat factura



Aquesta fitxa de cas d'ús és tant pel rol gerent com per l'administrador. L'usuari amb rol mecànic no hi té accés.

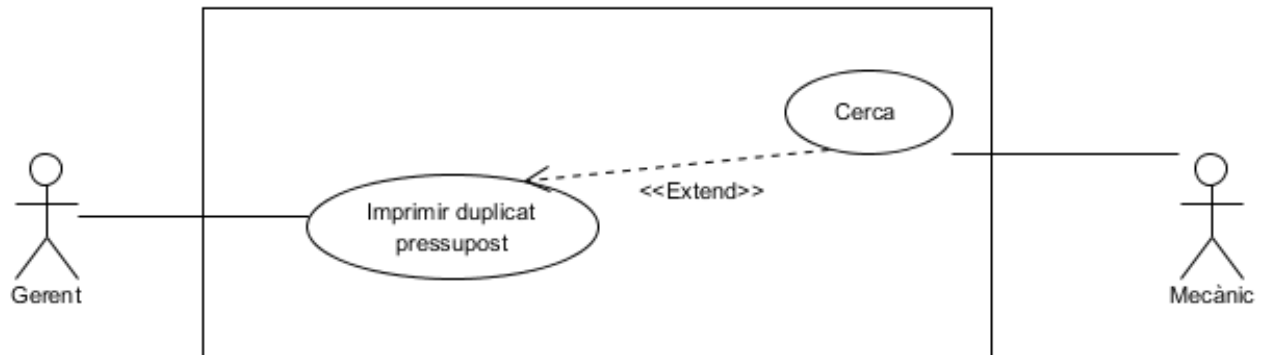
Imprimir duplicat factura	
Versió	Usuari
Actors	Gerent o administrador
Flux principal	<ol style="list-style-type: none">1. S'obre la pantalla inicial de l'aplicació2. Obrir pestanya informes3. Obrir opció imprimir duplicat factura4. Seleccionar opció cercar<ol style="list-style-type: none">4.1. Entrar dades4.2. Buscar4.3. Imprimir
Flux alternatiu	4.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.4.2. Imprimir duplicat albarà



Imprimir duplicat albarà	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inici de l'aplicació 2. Obrir pestanya informes 3. Obrir opció imprimir duplicat albarà 4. Seleccionar opció cerca <ol style="list-style-type: none"> 4.1. Entrar dades 4.2. Buscar 4.3. Imprimir
Flux alternatiu	4.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.4.3. Imprimir duplicat pressupost

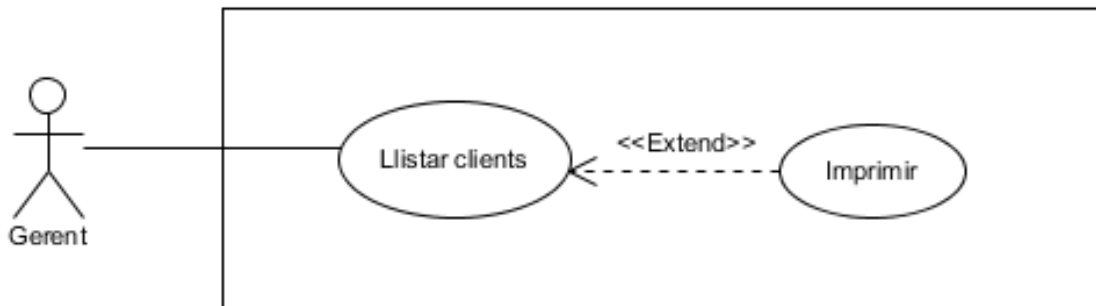


Imprimir duplicat pressupost	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inici de l'aplicació 2. Obrir pestanya informes 3. Obrir opció imprimir duplicat pressupost 4. Seleccionar opció cerca <ol style="list-style-type: none"> 4.1. Entrar dades 4.2. Buscar 4.3. Imprimir
Flux alternatiu	4.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.5. Mòdul consultes

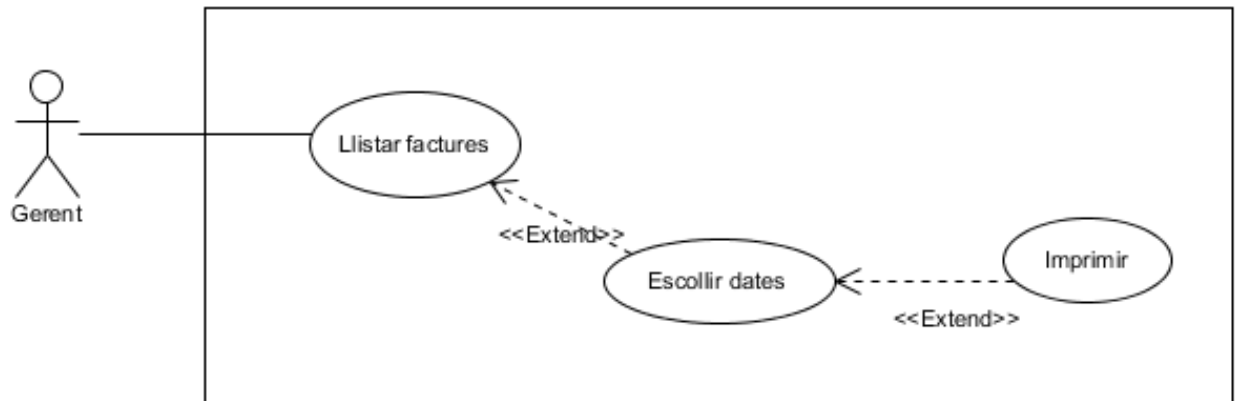
En totes les fitxes de cas d'ús d'aquest mòdul, els usuaris amb rol mecànic no hi tenen accés, només els administradors o gerents.

8.1.3.5.1. Llistar clients



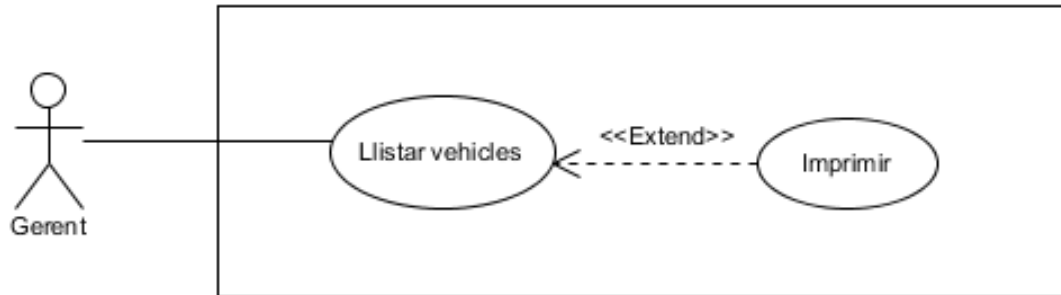
Llistar clients	
Versió	Usuari
Actors	Gerent o administrador
Flux principal	<ol style="list-style-type: none">1. S'obre la pantalla inicial de l'aplicació2. Seleccionar pestanya consultes3. Obrir opció llistar clients<ol style="list-style-type: none">3.2. Es mostren les dades per pantalla3.3. Si imprimir3.4. Imprimir llistat
Flux alternatiu	-

8.1.3.5.2. Llistar factures



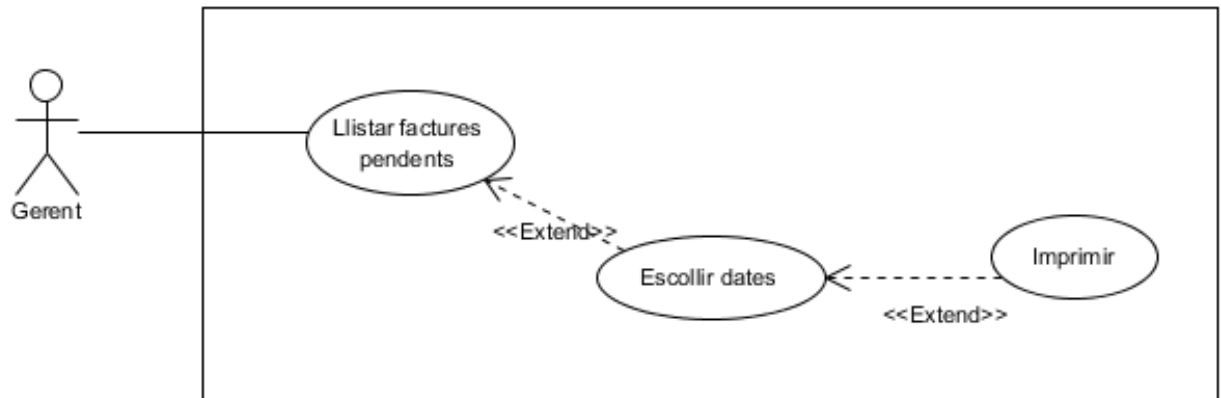
Gestió Llistar factures	
Versió	Usuari
Actors	Gerent o administrador
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya consultes 3. Obrir opció escollir dades <ol style="list-style-type: none"> 3.1. Entrar dades 3.2. Escollir camps de cerca 3.3. Buscar 3.4. Si imprimir 3.5. Imprimir llistat
Flux alternatiu	3.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.5.3. Llistar vehicles



Llistar vehicles	
Versió	Usuari
Actors	Gerent o administrador
Flux principal	<ol style="list-style-type: none">1. S'obre la pantalla inicial de l'aplicació2. Seleccionar pestanya consultes3. Obrir opció llistar vehicles<ol style="list-style-type: none">3.2. Es mostren les dades per pantalla3.3. Si imprimir3.4. Imprimir llistat
Flux alternatiu	-

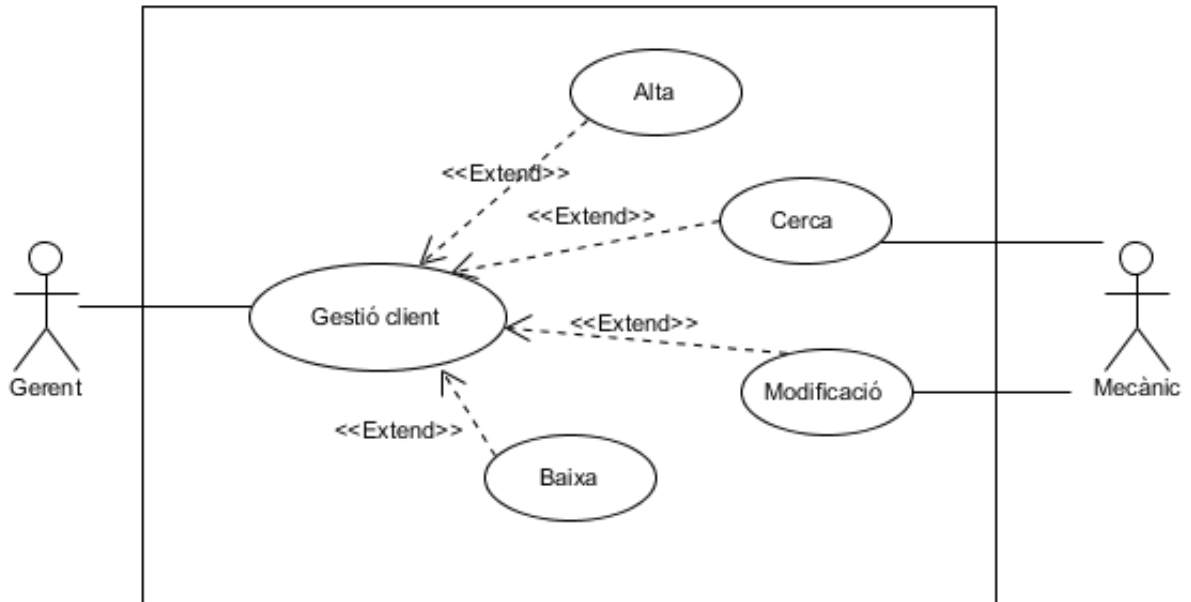
8.1.3.5.4. Llistar factures pendents



Llistar factures pendents	
Versió	Usuari
Actors	Gerent o administrador
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya consultes 3. Obrir opció escollir dades <ol style="list-style-type: none"> 3.1. Entrar dades 3.2. Escollir camps de cerca 3.3. Buscar 3.4. Si imprimir 3.5. Imprimir llistat
Flux alternatiu	3.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

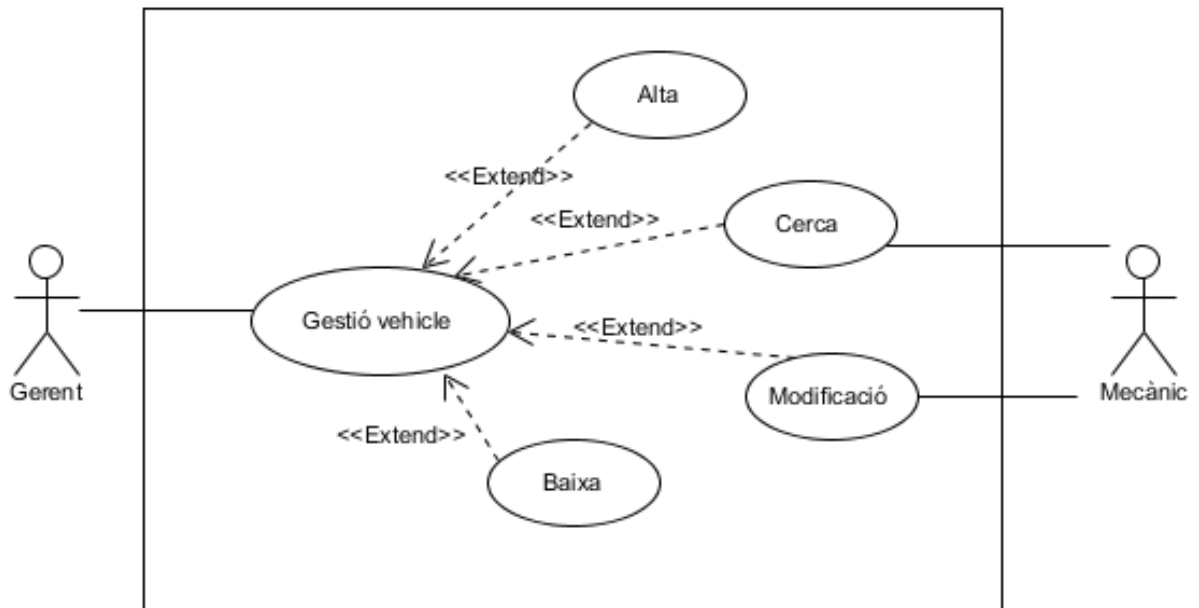
8.1.3.6. Mòdul base de dades

8.1.3.6.1. Gestió client



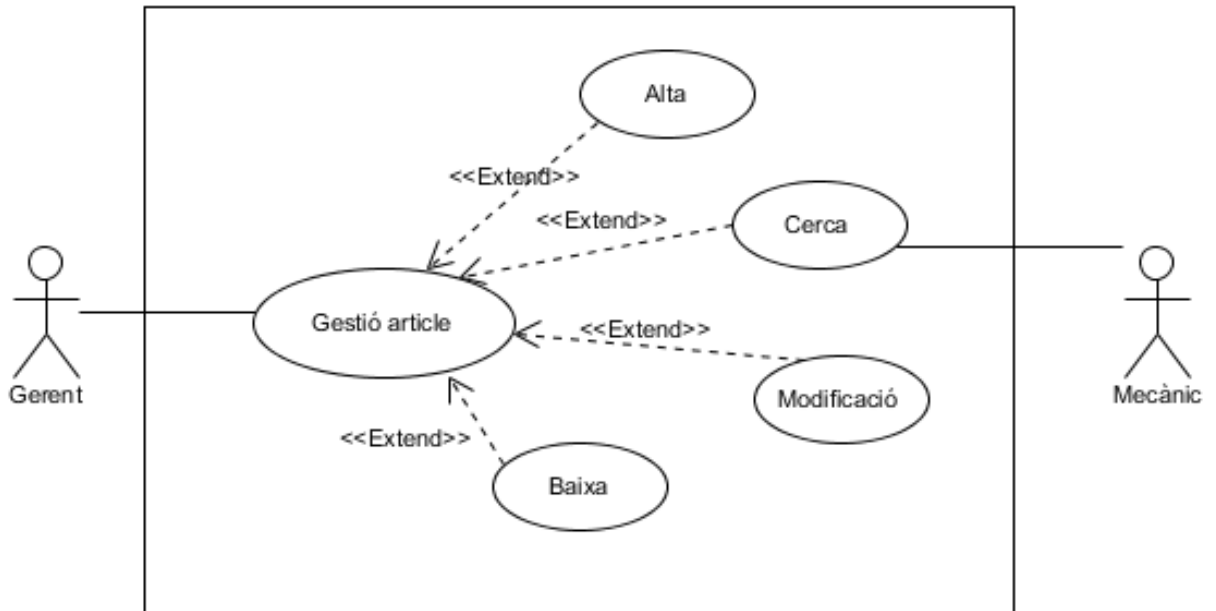
Gestió client	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya base de dades 3. Seleccionar gestió client 4. Obrir pantalla d'alta <ol style="list-style-type: none"> 4.1. Entrar dades client i vehicle 4.2. Guardar 5. Obrir pantalla de cerca <ol style="list-style-type: none"> 5.1. Introduir dades client 5.2. Buscar 6. Obrir pantalla de modificació <ol style="list-style-type: none"> 6.1. Entrar dades a modificar 6.2. Guardar 7. Obrir pantalla baixa <p>Si l'usuari és Administrador o Gerent,</p> <ol style="list-style-type: none"> 7.1. Escollir opció eliminar 7.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.6.2. Gestió vehicle



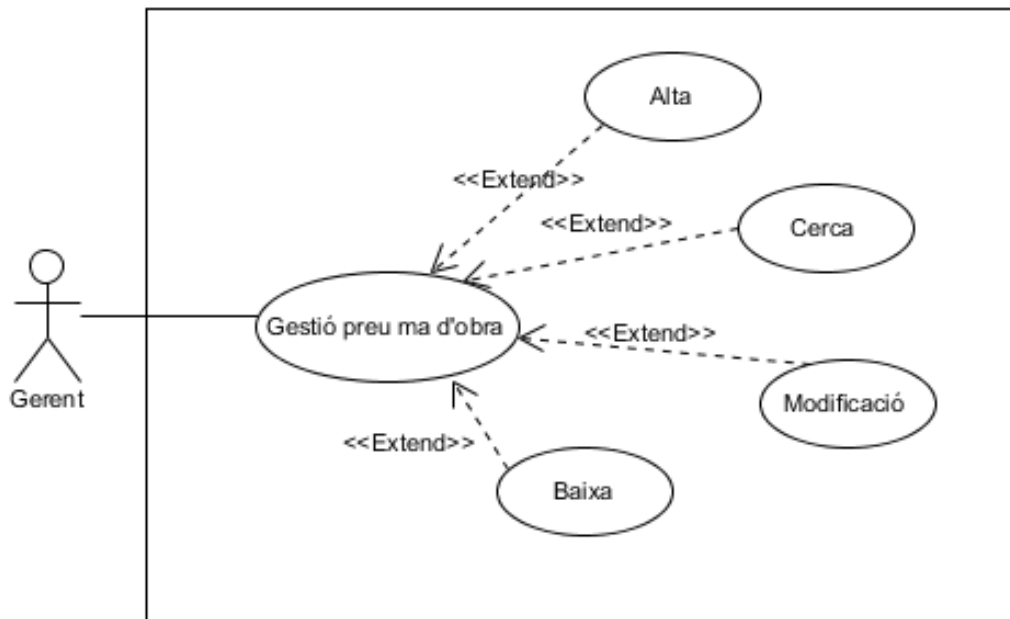
Gestió vehicle	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya base de dades 3. Seleccionar gestió vehicle 4. Obrir pantalla d'alta <ol style="list-style-type: none"> 4.1. Entrar dades client i vehicle 4.2. Verificar matrícula 4.3. Guardar 5. Obrir pantalla de cerca <ol style="list-style-type: none"> 5.1. Introduir matrícula 5.2. Verificar matrícula 5.3. Buscar 6. Obrir pantalla de modificació <ol style="list-style-type: none"> 6.1. Entrar dades a modificar 6.2. Guardar 7. Obrir pantalla baixa <p>Si l'usuari és Administrador o Gerent,</p> <ol style="list-style-type: none"> 7.1. Escollir opció eliminar 7.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.6.3. Gestió article



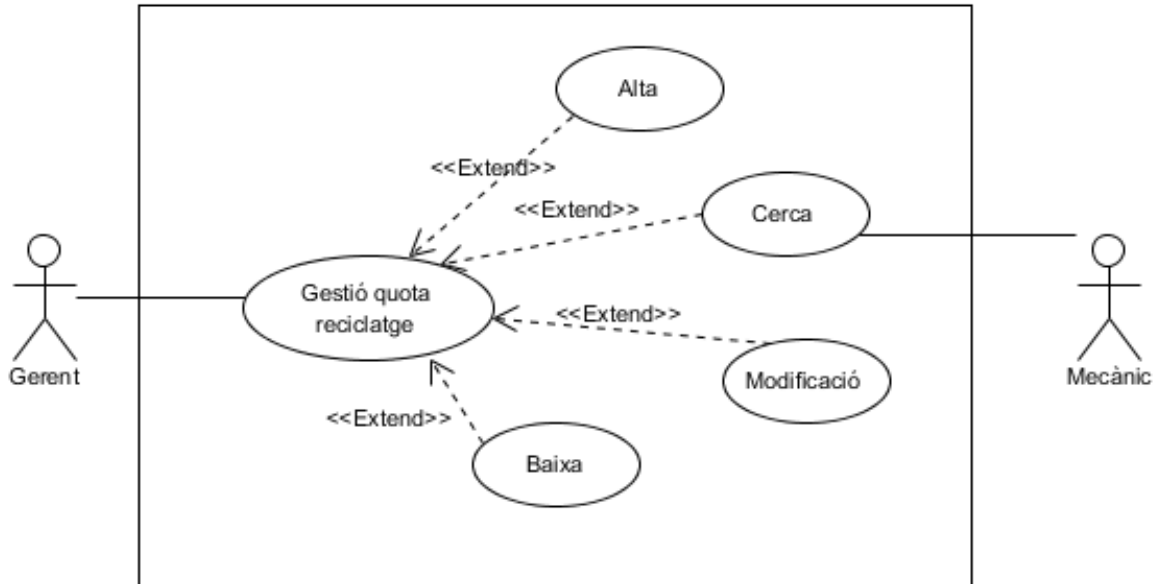
Gestió article	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya base de dades 3. Seleccionar gestió article 4. Obrir pantalla d'alta <ol style="list-style-type: none"> 4.1. Entrar dades client i vehicle 4.2. Guardar 5. Obrir pantalla de cerca <ol style="list-style-type: none"> 5.1. Introduir matrícula 5.2. Buscar 6. Obrir pantalla de modificació <ol style="list-style-type: none"> 6.1. Entrar dades a modificar 6.2. Guardar 7. Obrir pantalla baixa <p>Si l'usuari és Administrador o Gerent,</p> <ol style="list-style-type: none"> 7.1. Escollir opció eliminar 7.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.6.4. Gestió preu ma d'obra



Gestió preu ma d'obra	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya base de dades 3. Seleccionar gestió preu ma d'obra 4. Obrir pantalla d'alta <ol style="list-style-type: none"> 4.1. Entrar dades client i vehicle 4.2. Guardar 5. Obrir pantalla de cerca <ol style="list-style-type: none"> 5.1. Introduir matrícula 5.2. Buscar 6. Obrir pantalla de modificació <ol style="list-style-type: none"> 6.1. Entrar dades a modificar 6.2. Guardar 7. Obrir pantalla baixa <p>Si l'usuari és Administrador o Gerent,</p> <ol style="list-style-type: none"> 7.1. Escollir opció eliminar 7.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

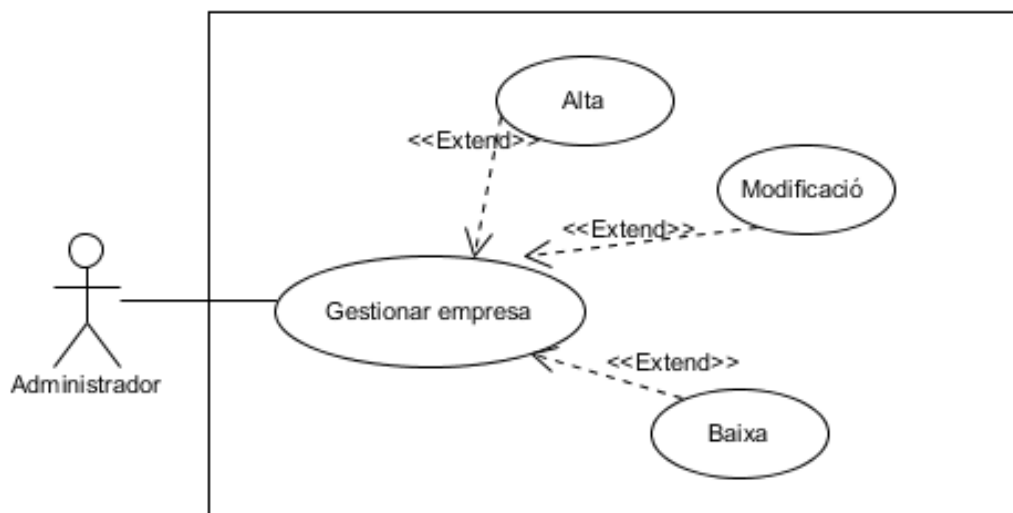
8.1.3.6.5. Gestió quota de reciclatge



Gestió quota de reciclatge	
Versió	Usuari
Actors	Gerent, mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya base de dades 3. Seleccionar gestió quota reciclatge 4. Obrir pantalla d'alta <ol style="list-style-type: none"> 4.1. Entrar dades client i vehicle 4.2. Guardar 5. Obrir pantalla de cerca <ol style="list-style-type: none"> 5.1. Introduir matrícula 5.2. Buscar 6. Obrir pantalla de modificació <ol style="list-style-type: none"> 6.1. Entrar dades a modificar 6.2. Guardar 7. Obrir pantalla baixa <p>Si l'usuari és Administrador o Gerent,</p> <ol style="list-style-type: none"> 7.1. Escollir opció eliminar 7.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

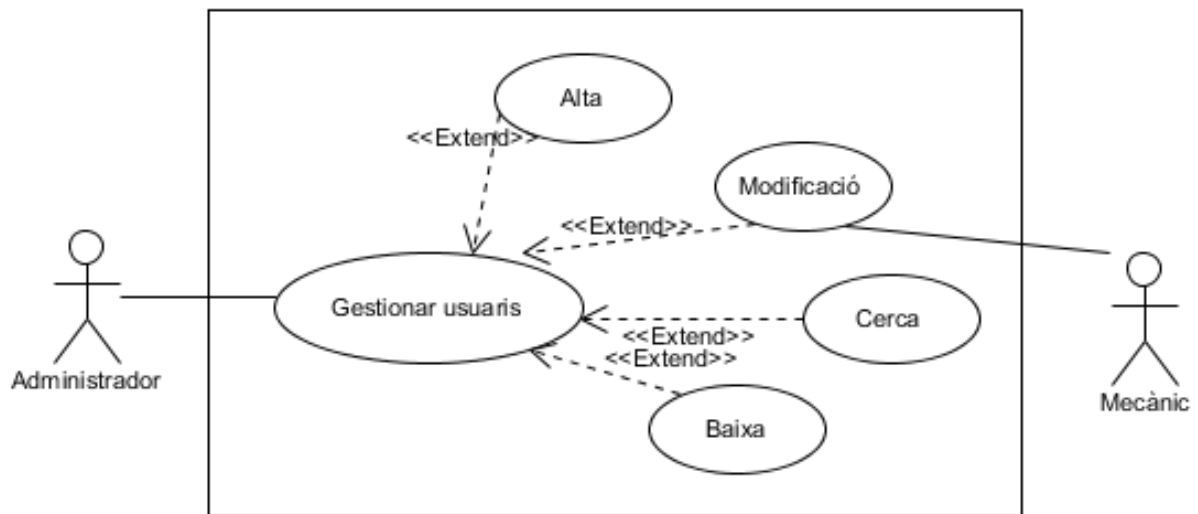
8.1.3.7. Mòdul configuració

8.1.3.7.1. Gestionar empresa



Gestionar empresa	
Versió	Usuari
Actors	Administrador
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya configuració 3. Seleccionar gestionar empresa 4. Obrir pantalla d'alta <ol style="list-style-type: none"> 4.1. Entrar dades 4.2. Guardar 5. Obrir pantalla de modificació <ol style="list-style-type: none"> 5.1. Entrar dades 5.2. Guardar 6. Obrir pantalla baixa <p>Si l'usuari és Administrador,</p> <ol style="list-style-type: none"> 6.1. Escollir opció eliminar 6.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.7.2. Gestionar usuaris



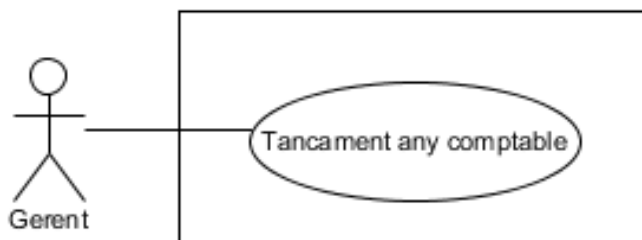
En aquest diagrama de cas d'ús els usuaris amb rol mecànic només tenen accés a la modificació de les dades pròpies d'accés a l'aplicació. L'administrador és l'encarregat del manteniment d'aquest mòdul.

Gestionar usuaris	
Versió	Usuari
Actors	Administrador
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya configuració 3. Seleccionar gestionar usuaris 4. Obrir pantalla d'alta <ol style="list-style-type: none"> 4.1. Entrar dades 4.2. Guardar 5. Obrir pantalla de cerca <ol style="list-style-type: none"> 5.1. Introduir matrícula 5.2. Buscar 6. Obrir pantalla de modificació <ol style="list-style-type: none"> 6.1. Entrar dades a modificar 6.2. Guardar 7. Obrir pantalla baixa <p>Si l'usuari és Administrador,</p> <ol style="list-style-type: none"> 7.1. Escollir opció eliminar 7.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

Per veure les diferències existents, farem la fitxa de cas d'ús del mecànic a part:

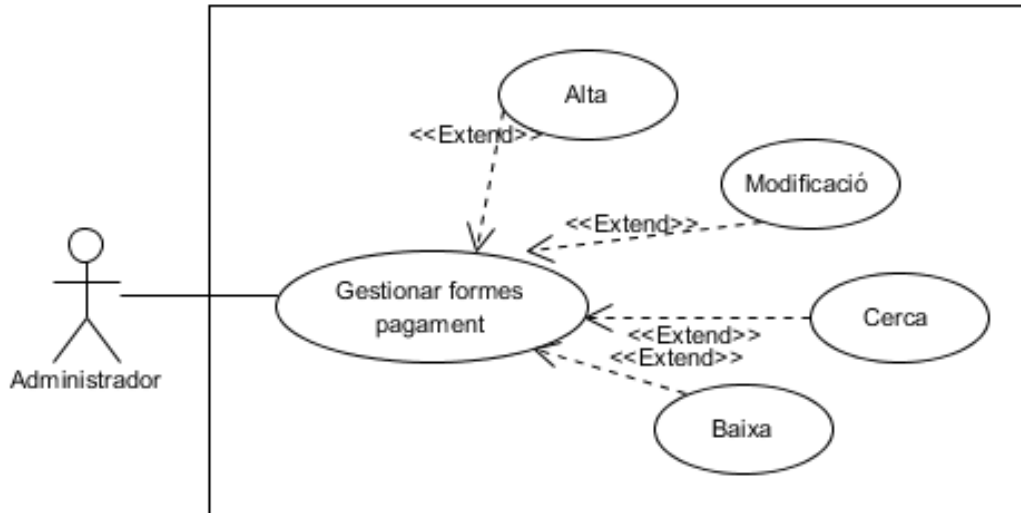
Gestionar usuaris	
Versió	Usuari
Actors	Mecànic
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya configuració 3. Seleccionar gestionar usuaris 4. Obrir pantalla de modificació <ol style="list-style-type: none"> 6.1. Entrar dades de l'usuari, login 6.2. Entrar dades a modificar 6.3. Guardar
Flux alternatiu	6.2 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.7.3. Tancament any comptable



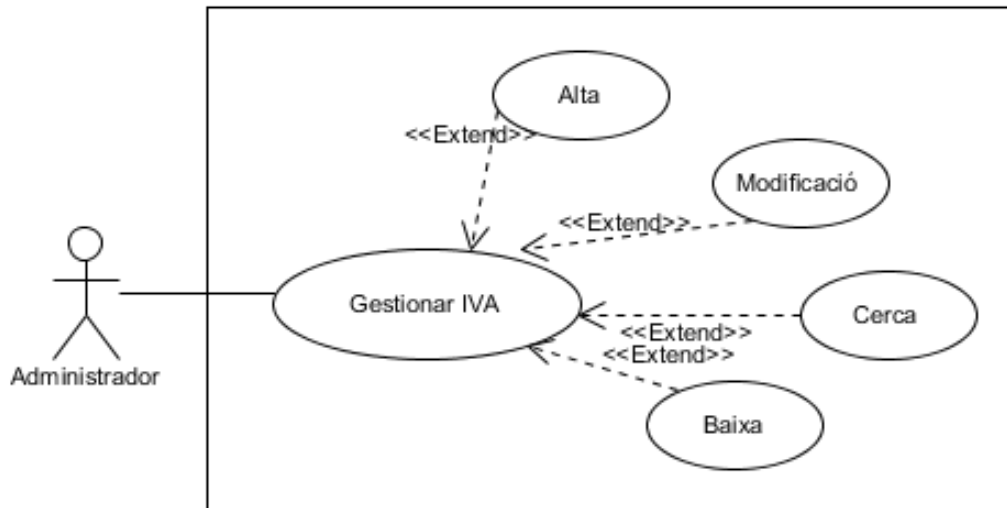
Tancament any comptable	
Versió	Usuari
Actors	Administrador o gerent
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla principal de l'aplicació 2. Seleccionar pestanya configuració 3. Clic al botó tancament any comptable
Flux alternatiu	-

8.1.3.7.4. Gestionar formes de pagament



Gestionar formes de pagament	
Versió	Usuari
Actors	Administrador
Flux principal	<ol style="list-style-type: none"> 1. S'obre la pantalla inicial de l'aplicació 2. Seleccionar pestanya configuració 3. Seleccionar gestionar formes pagament 4. Obrir pantalla d'alta <ol style="list-style-type: none"> 4.1. Entrar dades 4.2. Guardar 5. Obrir pantalla de cerca <ol style="list-style-type: none"> 5.1. Entrar dades 5.2. Buscar 6. Obrir pantalla de modificació <ol style="list-style-type: none"> 6.1. Entrar dades a modificar 6.2. Guardar 7. Obrir pantalla baixa <ol style="list-style-type: none"> 7.1. Escollir opció eliminar 7.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.1.3.7.5. Gestionar IVA



Gestionar IVA	
Versió	Usuari
Actors	Administrador
Flux principal	8. S'obre la pantalla inicial de l'aplicació 9. Seleccionar pestanya configuració 10. Seleccionar gestionar iva 11. Obrir pantalla d'alta 4.1. Entrar dades 4.2. Guardar 12. Obrir pantalla de cerca 5.1. Introduir dades 5.2. Buscar 13. Obrir pantalla de modificació 6.1. Entrar dades a modificar 6.2. Guardar 14. Obrir pantalla baixa 7.1. Escollir opció eliminar 7.2. Eliminar
Flux alternatiu	4.1, 5.1, 6.1 Si les dades són incorrectes o hi han camps obligatoris buits o incorrectes, mostra un missatge d'error detallat i marca les caselles incorrectes.

8.2. Disseny del sistema

En aquest apartat s'explicaran els models físics de dades i de processos.

8.2.1 Estructura de la base de dades

Abans de començar a programar, es va optar per realitzar el model de la base de dades i així definir l'estructura de les taules que tindriem. Inicialment vàrem realitzar el model conceptual a partir del model Entitat-Relació.

8.2.1.1. Model Entitat-Relació

A la pàgina següent es mostra el Model Entitat-Relació. Per qüestió d'espai s'han omès els noms de les relacions del diagrama, però es podran veure a les explicacions següents.

8.2.1.2. Explicació de les entitats de la base de dades

Un cop explicats els camps i les relacions entre les taules, a continuació fem una breu explicació de cada entitat de l'aplicació:

CONDICIÓ PAGAMENT: Aquesta taula conté les diferents condicions de pagament que l'empresa ofereix: Pagament a 30 dies, al comptat, dia 15 del mes següent, etc.

CLIENT: Aquesta taula guarda totes les dades de contacte i facturació dels clients relacionats amb l'empresa.

VEHICLE: A la taula hi consten totes les dades dels vehicles que han passat pel taller, tots els vehicles reparats o que s'hi ha pressupostat una feina.

RECEPCIÓ: En aquesta taula es guarden totes les recepcions del taller, és a dir, tots els vehicles o reparacions que entren.

TREBALLADOR: Aquesta taula guarda els usuaris que poden interactuar amb l'aplicació i treballadors de l'empresa. Cada treballador tindrà un rol assignat i una contrasenya d'accés.

ROL: Aquesta taula conté els diferents rols que poden tenir els usuaris a l'aplicació. Aquests rols són: administrador, gerent o mecànic.

ORDRE REPARACIÓ: En aquesta taula es guarden totes les dades d'una ordre de reparació, és a dir, les dades de qui ha reparat el vehicle i els km. que portava el vehicle quan ha sortit del taller.

LINIA ORDRE REPARACIÓ: Aquesta taula té el contingut de les línies de l'ordre de reparació, corresponents als diferents treballs realitzats al vehicle, els recanvis utilitzats, etc.

TAXES: La taula conté les taxes que s'apliquen a certs recanvis.

RECANVI: En aquesta taula es troben els diferents recanvis que l'empresa utilitza per a la reparació dels vehicles.

MA D'OBRA: En aquesta taula es troben els treballs realitzats als vehicles per part dels mecànics. Conté una breu descripció de l'operació i les hores dedicades.

TREBALL EXTERN: Aquesta taula emmagatzema els treballs externs que es realitzen en una reparació, com podria ser feina d'algun tècnic extern, treballs de planxa i pintura, etc.

MÈTODE PAGAMENT: Aquesta taula té les diferents formes de pagament que treballa l'empresa, com poden ser en efectiu, transferència bancària, xec, etc.

FACTURA: En aquesta taula es troben totes les dades de la capçalera de les factures emeses als clients, juntament amb els sumatoris finals.

L'entitat factura és **pare** de les entitats factura única mensual i factura immediata. Per tant, aquestes tindran les mateixes característiques que l'entitat factura i les específiques de cada una d'elles.

FACTURA ÚNICA MENSUAL: En aquesta taula es troben totes les dades de la capçalera de les factures emeses als clients, juntament amb els sumatoris finals. Al ser una única factura mensual, contindrà els treballs realitzats de diferents reparacions. Per tant, contindrà les dades de diferents ordres de reparació.

FACTURA IMMEDIATA: En aquesta taula es troben totes les dades de la capçalera de les factures emeses als clients, juntament amb els sumatoris finals. En aquest cas només contindrà les dades d'una ordre de reparació.

PRESSUPOST: Aquesta taula es troben totes les dades corresponents a la capçalera de tots els pressupostos dels clients. També s'hi guarden els sumatoris totals.

ALBARÀ: Aquesta taula conté totes les dades de la capçalera dels albarans emesos als clients, juntament amb l'import base. Cada albarà correspon a una reparació.

MÈTODE PAGAMENT_FACTURA: Taula que guarda amb quin mètode de pagament es paga una factura.

IMPOSTOS_PRESSUPOST: En aquesta taula es guarda la relació d'impostos que s'apliquen a un pressupost.

IMPOSTOS_FACTURA: La taula guarda la relació d'impostos que s'apliquen a una factura determinada.

8.2.1.3. Model Relacional

Després de realitzar el model Entitat-Relació, s'ha creat un model relacional per tal de visualitzar el disseny lògic.

A continuació queden detallades totes les taules obtingudes per l'aplicació. Abans però, cal detallar la nomenclatura utilitzada per a expressar el model relacional:

- Els **camp principals** apareixen en negreta.
- Els camps de tipus *clau forana*, apareixeran en cursiva.

- Els camps claus principals compostes, apareixen subratllats amb una línia contínua que engloba els diferents camps. A més, si també és *clau forana*, la part corresponent apareixerà en cursiva.

El model relacional és:

CONDICIÓ PAGAMENT (**Id**, tipus, dia, dies)

CLIENT (**Id**, DNI, nom, cognoms, telèfon, mòbil, adreça, població, codi_postal, província, e-mail, *Id_CondicióPagament*)

VEHICLE (**Id**, matrícula, bastidor, km_arribada, tipus_motor, cilindrada, potència_fiscal, any, marca, model, *Id_Client*)

RECEPCIÓ (**Id**, data, renúncia_pressupost, descripció_operacions, data_prevista_lliurament, recollir_peces, reparació_en_garantia, *Id_Client*, *Id_Vehicle*)

TREBALLADOR (**Id**, DNI, nom, cognoms, categoria, especialitat, contrasenya, *Id_Rol*)

ROL (**Id**, nom)

ORDRE REPARACIÓ (**Id**, km_sortida, tancat, *Id_Mecànic*, *Id_Recepció*, *Id_FacturaÚnicaMensual*)

LÍNIA ORDRE REPARACIÓ (**Id**, *Id_OrdreReparació*, codi, preu, quantitat, descompte, *Id_Recanvi*, *Id_MaObra*, *Id_TreballExtern*)

TAXES (**Id**, descripció, preu_unitari)

RECANVI (**Id**, descripció, preu_unitari, *Id_Taxes*)

MA D'OBRA (**Id**, descripció, preu_unitari)

TREBALL EXTERN (**Id**, descripció, preu_unitari)

MÈTODE PAGAMENT (**Id**, tipus)

FACTURA ÚNICA MENSUAL (**Id**, preu_base, total_factura, data, pagat, *Id_impost*)

FACTURA IMMEDIATA (**Id**, preu_base, total_factura, data, pagat, *Id_OrdreReparació*, *Id_impost*)

PRESSUPOST (**Id**, descripció, data, preu_base, total_pressupost, acceptat, *Id_OrdreReparació*, *Id_impost*)

ALBARÀ (**Id**, preu_base, data, *Id_FacturaÚnicaMensual*, *Id_OrdreReparació*)

IMPOST (**Id**, descripció, quantitat)

MÈTODE PAGAMENT_FACTURA (Id MètodePagament, Id Factura)

8.2.2 Diagrama de classes

A l'esquema següent es mostra el diagrama de classes de l'aplicació amb els atributs de cada classe:



Diagrama de classes amb atributs .

8.2.3 Explicació de les classes i relacions

En aquest apartat s'explicaran totes les classes del projecte representades anteriorment al diagrama de classes. Per cada classe es comentaran els atributs, relacions que puguin tenir amb altres classes i els mètodes més representatius. S'ometen mètodes com els *setters* i *getters* utilitzats per obtenir o canviar el valor de les variables de cada classe.

Els atributs clau estaran marcats amb negreta i les claus foranes amb cursiva. Al costat de cada atribut se n'indicarà el tipus.

8.2.3.1 Condició pagament

CONDICIÓ PAGAMENT
id : int tipus : String dia : int dies : int
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) addClient(\$client) removeClient(\$client) buildForm(FormBuilderInterface \$builder, array \$options) searchbuildForm(FormBuilderInterface \$builder, array \$options)

Totes les dades anteriors són referents a les dades de les condicions de pagament. A continuació n'expliquem el significat:

- **Id**: és el número identificatiu de la condició de pagament.
- **Tipus**: indica el tipus de venciment de pagament, és a dir, si es pagarà al comptat, a 30 dies, fraccionat, etc.

- **Dia:** indica el dia que es realitzarà el pagament, per exemple, el dia 30 de cada mes, o el 15.
- **Dies:** indica els dies del venciment, com pot ser 30 dies de la data de factura.

Els mètodes principals de la classe condició de pagament són:

- Controlador:
 - _construct():** és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula de clients. Els següents mètodes són generats automàticament pel CRUD:
 - **newAction()**
 - **createAction(Request \$request)**
 - **editAction(\$id)**
 - **updateAction(Request \$request, \$id)**
 - **searchAction()**
 - **showAction(\$id)**
 - **deleteAction(Request \$request, \$id):** aquests mètodes creen, editen, actualitzen, esborren, mostren l'objecte d'aquesta classe i el busquen a partir de l'*id*.

Tots aquests mètodes són creats a cada classe. Per tant, per no repetir les explicacions corresponents, no s'explicaran a les següents classes.

- Model:
 - buildForm(FormBuilderInterface \$builder, array \$options):** genera el formulari de creació de l'objecte.
 - searchbuildForm(FormBuilderInterface \$builder, array \$options):** genera el formulari de cerca de l'objecte.

Igual que amb els mètodes del CRUD, aquests també són generats a cada classe, per tant, no s'explicaran a les següents classes.

8.2.3.2 Client

CLIENT
id : int dni : String nom : String cognoms : String telèfon : int mòbil : int adreça : String població : String codi_postal : int provincia : String e-mail : String <i>id_CondicióPagament</i> : int
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) addRecepcio(\$recepcio) removeRecepcio(\$recepcio) addVehicle (\$vehicle) removeVehicle(\$vehicle) buscar_vehiclesAction() obtenirCodicioPagament() obtenirClient(\$cognoms) obtenirAlbarans(\$di, \$df)

Les dades anteriors majoritàriament són dades personals del client, excepte alguna que s'utilitza pel funcionament de l'aplicació:

- **Id**: és el número identificatiu de l'usuari.
- **Dni, nom, cognoms, telèfon, mòbil, adreça, població, codi_postal, provincia i e-mail**: són les dades personals del client.
- **Id_CondicióPagament**: és l'atribut que defineix la relació 1 a N que existeix amb la classe Condició de pagament. Serveix per indicar quina condició de pagament té assignada el client.

Els mètodes principals de la classe client són:

- Controlador:

_construct(): és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula de vehicles i una de recepcions.

buscar_vehiclesAction(): genera un llistat amb tots els vehicles donats d'alta a la base de dades i que són propietat del client actual.

obtenirCondicioPagament(): retorna la informació corresponent a les condicions de pagament pactades amb aquest client.

obtenirClient(\$cognoms): retorna el client que té com a cognoms els entrats per paràmetre.

obtenirAlbarans(\$di, \$df): retorna la llista d'albarans entre les dates inici i final entrades per paràmetre.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.3 Vehicle

VEHICLE
id : int matrícula : String bastidor : String km_arribada : long tipus_motor : String cilindrada : int potència_fiscal : float any : int marca : String model : String <i>id_Client</i> : int
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) addRecepcio (\$recepcio) removeRecepcio(\$recepcio) obtenirClient()

```
generarHistoric()
obtenirVehicle()
obtenirORs()
mostra($lors)
assignar_clientAction()
buscar_reparacionsAction()
```

Les dades anteriors són referents a les dades del vehicle.

- **Id**: és l'identificador del vehicle per l'aplicació.
- **Matrícula, bastidor, tipus_motor, cilindrada, potència_fiscal, any, marca, model**: són les dades del vehicle.
- **Km_arribada**: indica els quilòmetres que porta el vehicle al moment d'entrar al taller.
- **Id_Client**: és l'atribut que indica que el vehicle té una relació N a 1 amb la classe Client. Guarda l'identificador del client a les dades del vehicle. D'aquesta manera s'indica que un vehicle és d'un únic client, però que un client pot tenir diferents vehicles en propietat.

Els mètodes principals de la classe vehicle són:

- Controlador:
 - _construct()**: és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula de recepcions.
 - obtenirClient()**: retorna el client que és propietari del vehicle actual.
 - generarHistoric()**: retorna un llistat de totes les reparacions que s'han realitzat al vehicle.
 - obtenirVehicle()**: retorna l'objecte vehicle.
 - obtenirORs()**: retorna una llista d'ordres de reparacions del vehicle actual.
 - mostra(\$lors)**: mostra per pantalla la llista amb totes les ordres de reparació del vehicle actual.
 - assignar_clientAction()**: assigna un client al vehicle actual a través del formulari.
 - buscar_reparacionsAction()**: a partir d'una matrícula d'un vehicle, retorna les reparacions efectuades.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.4 Recepció

RECEPCIÓ
id : int data : String renúncia_pressupost : boolean descripció_operacions : String data_prevista_lliurament : String recollir_peces : boolean reparació_en_garantia : boolean <i>id_Client</i> : int <i>id_Vehicle</i> : int
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) obtenirClient() obtenirVehicle() validarData() esVolPressupost() esGarantia() obtenirRecepcio() crearOR() imprimir_pdfAction()

Els atributs d'aquesta classe recullen totes les característiques d'una Recepció i les relacions corresponents:

- **Id**: és l'identificador de la recepció.
- **Data**: és la data en què el vehicle arriba al taller i es genera la recepció.
- **Renúncia_pressupost**: indica si el client desitja pressupost o es pot passar directament a reparar el vehicle.
- **Descripció_operacions**: descriu les operacions a realitzar al vehicle.
- **Data_prevista_lliurament**: és la data en que es preveu lliurar el vehicle al client.
- **Recollir_peces**: indica si el client desitja recollir les peces o no al moment de recollir el vehicle.
- **Reparació_en_garantia**: si la reparació és en garantia, es selecciona aquest atribut per indicar que el cost de la factura serà nul.

- **Id_Client**: és l'atribut que defineix la relació N a 1 que hi ha amb la classe Client.
- **Id_Vehicle**: aquest atribut defineix la relació N a 1 que hi ha amb la classe Vehicle.

Les 2 relacions N a 1 que hi ha amb les classes Client i Vehicle indiquen que una Recepció està formada per un Client i el Vehicle. I aquestes, poden tenir moltes Recepcions.

Els mètodes principals de la classe recepció són:

- Controlador:
 - _construct()**: és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents.
 - obtenirClient()**: retorna el client assignat a aquesta recepció.
 - obtenirVehicle()**: retorna el vehicle assignat a aquesta recepció.
 - validarData()**: comprova si la data seleccionada és vàlida.
 - esVolPressupost()**: retorna cert si el client vol pressupost i fals en cas contrari.
 - esGarantia()**: retorna cert si la reparació és en garantia i fals altrament.
 - crearOR()**: a partir de les dades de la recepció, es genera automàticament una ordre de reparació nova.
 - obtenirRecepcio()**: retorna l'objecte recepció actual.
 - imprimir_pdfAction()**: genera la plantilla corresponent per imprimir a pdf la recepció.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.5 Treballador

TREBALLADOR
id : int dni : String nom : String cognoms : String categoria : String especialitat : String contrasenya : String

<i>id_Rol</i> : int
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) addOR (\$or) removeOR(\$or) validarTreballador() validarContrasenya()

A continuació es fa una breu explicació dels atributs d'aquesta classe:

- **Id**: identifica el mecànic
- **Dni, nom, cognoms, categoria, especialitat**: són les dades del mecànic.
- **Contrasenya**: guarda la clau d'accés del mecànic a l'aplicació.
- **Id_Rol**: és l'atribut que defineix la relació N a 1 que hi ha amb la classe Rol. Indica que cada mecànic té un rol assignat per definir quins permisos té a l'hora d'utilitzar l'aplicació.

Els mètodes principals de la classe treballador són:

- Controlador:
 - _construct()**: és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula d'ordres de reparació.
 - validarTreballador()**: aquest mètode comprova que el treballador existeixi a la base de dades i la contrasenya introduïda sigui correcte.
 - validarContrasenya()**: comprova que la contrasenya introduïda per l'usuari tingui un format correcte.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.6 Rol

ROL
id : int nom : String
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) addTreballador(\$treballador) removeTreballador(\$treballador)

Els seus atributs són:

- **Id**: és l'identificador de cada permís.
- **Nom**: indica el nom del rol que té assignat: administrador, gerent o mecànic.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.7 Ordre de reparació (O.R.)

ORDRE REPARACIÓ
id : int km_sortida : long <i>id_Treballador</i> : int <i>id_Recepció</i> : int <i>id_FacturaÚnicaMensual</i> : int tancat : boolean
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id)

```
addLOR($lor)
removeLOR($lor)
obtenirVehicle()
obtenirClient()
obtenirRecepcio()
obtenirMecanic()
completarAction($id)
afegir_recanvi_orAction($idor, $idr)
afegir_textern_orAction($idor, $idr)
afegir_mo_orAction($idor, $idr)
imprimir_factura_pdfAction($id)
assignarFacturaUM()
obtenirLiniesOR()
imprimir($id)
imprimir_pdfAction()
generarPressupost()
tePressupost()
tancarOR()
afegirLOR()
crearLOR($r)
crearAlbara()
crearFacturalmmediata()
```

Els atributs de l'ordre de reparació són els següents:

- **Id:** és l'identificador que utilitza l'aplicació.
- **Km_sortida:** correspon als quilòmetres que porta el vehicle al moment de finalitzar-se la reparació. Aquest atribut s'indicarà a la factura.
- **Id_Mecànic:** és l'atribut que indica la relació N a 1 que hi ha amb la classe Mecànic. Es considera que cada mecànic repara molts vehicles, però que cada reparació la realitza un mecànic, per tant, cada ordre de reparació la fa un mecànic.
- **Id_Recepció:** indica la relació i a j que hi ha amb la classe Recepció. Al crear una recepció, es crea automàticament una OR.
- **Id_FacturaÚnicaMensual:** és l'atribut que descriu la relació N a 1 que existeix amb la classe FacturaÚnicaMensual. Cada factura única mensual pot contenir diferents ordres de reparació corresponents a les reparacions que s'hagin realitzar durant aquest mes.
- **Tancat:** indica si l'ordre de reparació està tancada o no.

Els mètodes principals de la classe ordre de reparació són:

- Controlador:

_construct(): és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquets cas, el constructor conté una taula de línies d'ordre de reparació.

obtenirClient(): el mètode retorna el client assignat a aquesta ordre de reparació.

obtenirVehicle(): el mètode retorna el vehicle assignat a aquesta ordre de reparació.

obtenirRecepcio(): retorna la recepció a partir de la qual s'ha generat l'ordre de reparació actual.

obtenirMecanic(): retorna el treballador que ha realitzar la reparació del vehicle.

assignarFacturaUM(): el mètode assigna una factura única mensual a l'ordre de reparació al moment que es realitza l'acció de facturar.

obtenirLiniesOR(): obté les reparacions i recanvis utilitzats durant la reparació del vehicle.

imprimir(\$id): genera una plantilla d'impressió amb les dades de l'ordre de reparació amb id *\$id*.

imprimir_pdfAction(): genera la plantilla corresponent per imprimir a pdf l'o.r. objecte actual.

generarPressupost(): un cop s'ha emplenat l'ordre de reparació, genera el pressupost corresponent al client del vehicle.

tePressupost(): indica si l'ordre de reparació té assignada o no un pressupost.

tancarOR(): aquest mètode, un cop s'ha realitzat la reparació del vehicle, genera la factura immediata o albarà corresponent segons el mètode de pagament del client.

afegirLOR(): afegeix a la taula de línies d'ordre de reparació de l'OR.

crearLOR(\$r): crea una línia d'OR a la taula de línies d'OR amb el recanvi r passat per paràmetre.

crearAlbara(): aquest mètode genera un albarà a partir de l'OR actual.

crearFacturaImmediata(): genera automàticament una factura a partir de l'OR actual.

completarAction(\$id): afegeix a l'ordre de reparació amb l'id passat per paràmetre els atributs necessaris per poder tancar l'o.r.

afegir_recanvi_orAction(\$idor, \$idr): afegeix a l'o.r. una línia d'ordre de reparació amb el recanvi amb id passat per paràmetre.

afegir_textern_orAction(\$idor, \$idr): afegeix a l'o.r. una línia d'ordre de reparació amb el treball extern amb id passat per paràmetre.

afegir_mo_orAction(\$idor, \$idr): afegeix a l'o.r. una línia d'ordre de reparació amb la ma d'obra amb id passat per paràmetre.

imprimir_factura_pdfAction(\$id): envia a pantalla la plantilla corresponent a imprimir les dades de la factura amb pdf.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.8 Línia ordre de reparació

LINIA ORDRE REPARACIÓ
id : int id_OrdreReparació : int codi : int preu : int quantitat : int descompte : float <i>id_Recanvi</i> : int <i>id_MaObra</i> : int <i>id_TreballExtern</i> : int
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) esRecanvi() esMaObra() esTreballExtern() afegir(\$x)

Els atributs fan referència a:

- **Id:** identificador de la classe a la base de dades.
- **Id_OrdreReparació:** aquest atribut defineix la relació N a 1 que hi ha amb una ordre de reparació. Cada ordre de reparació estarà formada per diverses línies d'ordre de reparació (OR). I cada línia serà d'una única OR.
- **Codi, preu, quantitat i descompte:** són les dades de la línia d'ordre de reparació, contindran el codi, el preu i la quantitat de recanvis, ma d'obra o treball extern.

- **Id_Recanvi**: aquest atribut descriu la relació N a 1 que hi ha amb la classe Recanvi. Cada línia d'ordre de reparació farà referència a un recanvi diferent.
- **Id_MaObra**: l'atribut fa referència a la relació N a 1 que té amb la classe Ma d'obra. Cada línia d'ordre de reparació contindrà un tipus de ma d'obra diferent.
- **Id_TreballExtern**: l'atribut correspon a la relació N a 1 que hi ha amb la classe Treball Extern, igual que en les relacions anteriors, cada línia d'OR correspon a un únic treball extern.

Els mètodes principals de la classe línia d'ordre de reparació són:

- Controlador:
 - _construct()**: és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents.
 - esRecanvi()**: el mètode indica si la línia d'ordre de reparació correspon a un recanvi.
 - esMaObra()**: el mètode indica si la línia d'ordre de reparació correspon a una operació dels treballadors.
 - esTreballExtern()**: indica si la línia d'ordre de reparació correspon a un treball extern.
 - afegir(\$x)**: aquest mètode afegeix a l'objecte actual el recanvi passat per paràmetre.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.9 Taxa

TAXA
id : int descripció : String preu_unitari : float
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction()

```

showAction($id)
deleteAction(Request $request, $id)
imprimir_llistat_taxes_pdfAction()

```

- **Id:** és l'identificador de la taxa.
- **Descripció, preu_unitari:** aquests atributs descriuen el tipus de taxa i el preu unitari d'aquesta.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.10 Recanvi

RECANVI
id : int descripció : String preu_unitari : float <i>id_Taxes</i> : int
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) addLOR (\$lor) removeLOR(\$lor) obtenirTaxes() buscar_per_codiAction(\$id) crear_recanvi_orAction(\$id) imprimir_pdfAction()

- **Id:** és l'identificador que utilitza l'aplicació.
- **Descripció, preu_unitari:** són els atributs que descriuen el tipus de taxa i el preu unitari d'aquesta.
- **Id_Taxes:** és l'atribut que defineix la relació N a 1 que hi ha amb la classe Taxes. Cada recanvi pot tenir una taxa determinada, i aquesta pot ser aplicada a nombrosos recanvis diferents.

Els mètodes principals de la classe recanvi són:

- Controlador:
 - _construct()**: és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula de línies d'ordre de reparació.
 - obtenirTaxes()**: obté les taxes a aplicar a aquest recanvi.
 - buscar_per_codiAction(\$id)**: retorna l'objecte recanvi corresponent a l'*id* passat per paràmetre.
 - crear_recanvi_orAction(\$id)**: genera el formulari corresponent a la creació d'un recanvi mentre s'emplena l'ordre de reparació.
 - imprimir_pdfAction()**: genera la plantilla corresponent per imprimir a pdf el llistat de recanvis de la base de dades.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.11 Ma d'obra

MA D'OBRA
id : int descripció : String preu_unitari : float
<code>_construct()</code> <code>newAction()</code> <code>createAction(Request \$request)</code> <code>editAction(\$id)</code> <code>updateAction(Request \$request, \$id)</code> <code>searchAction()</code> <code>imprimir_pdfAction()</code> <code>showAction(\$id)</code> <code>deleteAction(Request \$request, \$id)</code> <code>addLOR (\$lor)</code> <code>removeLOR(\$lor)</code> <code>buscar_per_codiAction(\$id)</code> <code>crear_mo_orAction(\$id)</code>

- **Id**: és l'identificador de la ma d'obra que utilitza l'aplicació.

- **Descripció, preu_unitari:** aquests atributs descriuen el tipus de ma d'obra i el preu unitari d'aquesta.

Els mètodes principals de la classe ma d'obra són:

- Controlador:
 - _construct():** és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula de línies d'ordre de reparació.
 - buscar_per_codiAction(\$id):** retorna l'objecte ma d'obra corresponent a l'*id* passat per paràmetre.
 - crear_mo_orAction(\$id):** genera el formulari corresponent a la creació d'una operació de ma d'obra mentre s'emplena l'ordre de reparació.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.12 Treball extern

TREBALL EXTERN
id : int descripció : String preu_unitari : float
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) imprimir_pdfAction() deleteAction(Request \$request, \$id) addLOR (\$lor) removeLOR(\$lor) buscar_per_codiAction(\$id) crear_textern_orAction(\$id)

- **Id:** és l'identificador que utilitza l'aplicació.

- **Descripció, preu_unitari:** aquests atributs descriuen el tipus de treball extern i el preu unitari d'aquesta.

Els mètodes principals de la classe treball extern són:

- Controlador:
 - _construct():** és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula de línies d'ordre de reparació.
 - buscar_per_codiAction(\$id):** retorna l'objecte treball extern corresponent a l'*id* passat per paràmetre.
 - crear_textern_orAction(\$id):** genera el formulari corresponent a la creació d'un treball extern mentre s'emplena l'ordre de reparació.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

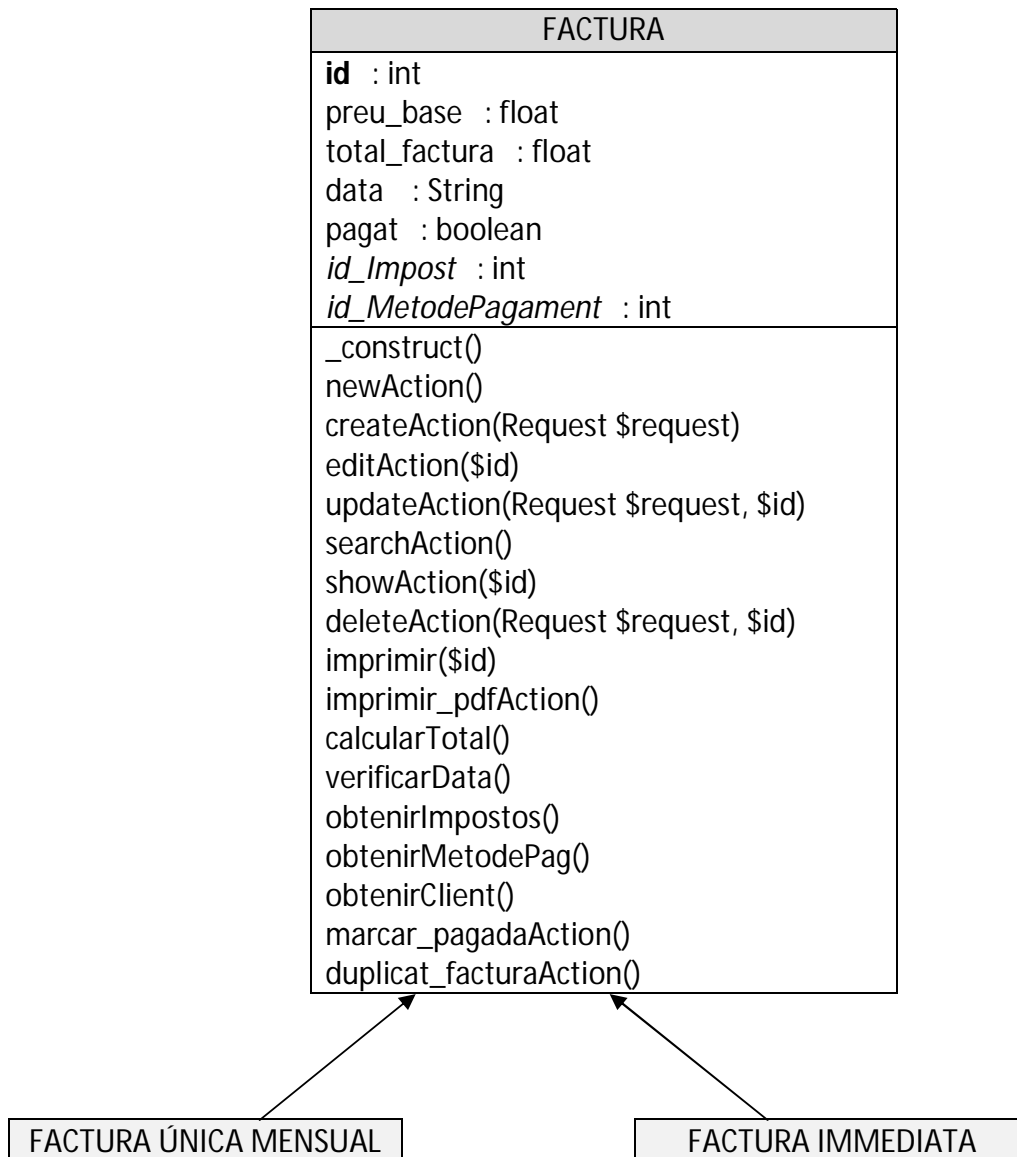
8.2.3.13 Mètode pagament

MÈTODE PAGAMENT
id : int tipus : String
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id)

- **Id:** és l'identificador del mètode de pagament que utilitza l'aplicació.
- **Tipus:** l'atribut descriu el tipus de mètode de pagament.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.14 Factura



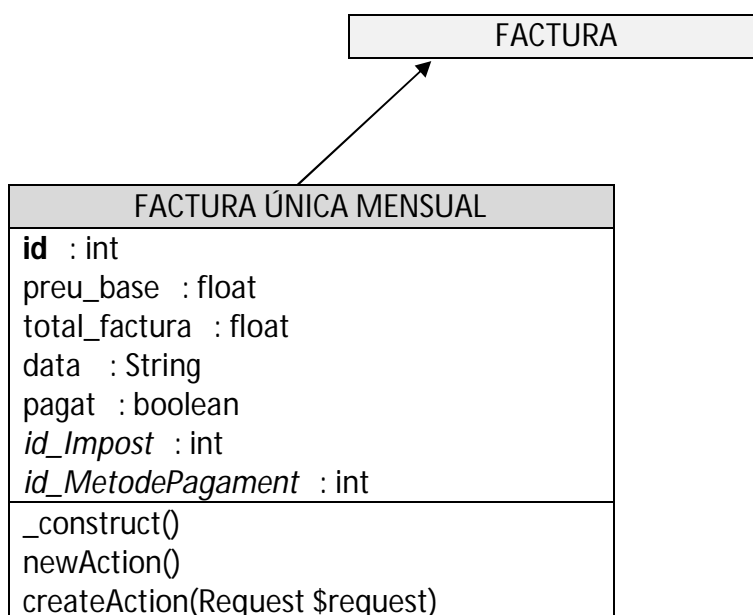
- **Id:** és l'identificador que utilitza l'aplicació.
- **Preu_base:** aquest atribut descriu el preu base de la factura, és a dir, la suma dels recanvis utilitzats amb les taxes corresponents, la ma d'obra i els treballs externs realitzats durant la reparació del vehicle.
- **Total_factura:** és el resultat d'aplicar els impostos corresponents al preu base.
- **Data:** és la data d'emissió de la factura.
- **Pagat:** indica si una factura ha sigut pagada o no.
- **Id_Impost:** aquest atribut descriu la relació 1 a n que hi ha entre aquesta classe i la classe Impost. Cada factura tindrà un tipus d'impost assignat.
- **Id_MetodePagament:** aquest atribut descriu la relació n a m que hi ha entre aquesta classe i la classe Mètode de pagament. Cada factura té assignat un mètode de pagament.

Els mètodes principals de la classe factura són:

- Controlador:
 - _construct()**: és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula d'impostos.
 - imprimir(\$id)**: genera una plantilla d'impressió amb les dades de la factura amb id \$id. També es pot utilitzar per imprimir duplicats.
 - imprimir_pdfAction()**: genera la plantilla corresponent per imprimir a pdf la factura.
 - calcularTotal()**: calcula el total de la factura a partir del preu base i l'aplicació de l'impost corresponent.
 - verificarData()**: comprova si la data de la factura és correcta.
 - obtenirImpostos()**: obté els impostos que s'han d'aplicar a aquesta factura.
 - obtenirMetodePag()**: obté el mètode de pagament assignat a aquesta factura.
 - obtenirClient()**: obté el client de la factura a partir de l'ordre de reparació.
 - marcar_pagadaAction()**: modifica l'atribut *pagat* de l'objecte actual a *true*.
 - duplicat_facturaAction()**: a partir de l'*id* entrat pel formulari, mostra la factura per pantalla.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.15 Factura única mensual



```
editAction($id)
updateAction(Request $request, $id)
searchAction()
show($id)
deleteAction(Request $request, $id)
obtenirORs($client, $inici, $fi)
obtenirAlbarans($client, $inici, $fi)
addOR ($or)
removeOR($or)
addAlbara ($albara)
removeAlbara($albara)
crear($lla)
facturarAlbarans($lla)
```

La classe Factura única mensual tindrà els mateixos atributs i mètodes que la classe pare Factura.

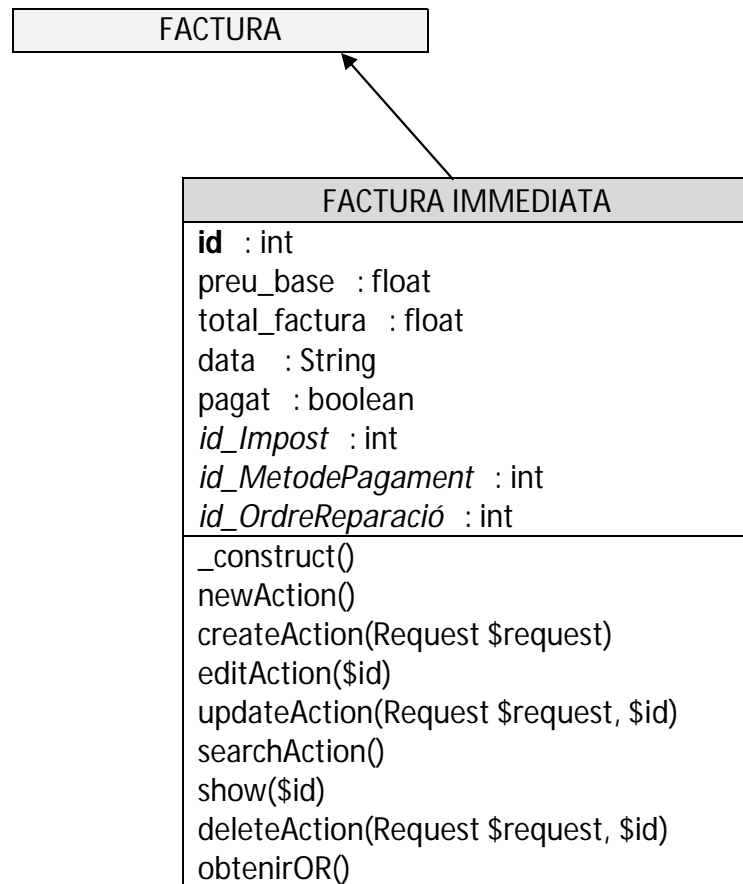
Aquesta classe té una relació 1 a N amb la classe Ordre de Reparació, l'atribut de relació està a la classe OR. La factura única mensual contindrà totes les ordres de reparació generades durant el mes en curs d'un mateix client.

Els mètodes principals de la factura única mensual són:

- Controlador:
 - _construct()**: és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula d'ordres de reparació i una d'albarans.
 - obtenirORs(\$client, \$inici, \$fi)**: obté les dades de les ordres de reparació del client passat com a paràmetre entre les dates *\$inici* i *\$fi* per generar la factura única mensual.
 - obtenirAlbarans(\$client, \$inici, \$fi)**: obté les dades dels albarans del client passat com a paràmetre entre les dates *\$inici* i *\$fi* per generar la factura única mensual.
 - crear(\$lla)/facturarAlbarans(\$lla)**: aquest mètode crea una factura única mensual amb la llista d'albarans passats per paràmetre.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.16 Factura immediata



La classe Factura immediata tindrà els mateixos atributs i mètodes que la classe pare Factura i l'atribut:

- **Id_OrdreReparació:** l'atribut descriu la relació 1 a 1 que existeix entre la classe Factura immediata i Ordre de reparació. Cada OR correspondrà a una factura immediata, a menys que el client ho hagi sol·licitat. En aquest cas, es farà una única factura mensual, explicat en l'apartat anterior.

Els mètodes principals de la classe factura immediata són:

- **Controlador:**
 - _construct():** és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents.
 - obtenirOR():** obté les dades l'ordre de reparació a partir de la qual s'ha generat l'albarà.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.17 Pressupost

PRESSUPOST
id : int descripció : String data : String preu_base : float total_pressupost : float id_OrdreReparació : int id_Impost: int acceptat: boolean
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id) obtenirOR() obtenirImpostos() imprimirPressupost(\$id) imprimir_pdfAction() calcularTotal() obtenirClient() modificarPressupostPerFactura() llistarpressupostosPendents()

- **Id**: és l'identificador que utilitza l'aplicació.
- **Descripció, data**: corresponen a la descripció i la data del pressupost realitzat.
- **Preu_base**: aquest atribut descriu el preu base del pressupost, és a dir, la suma dels recanvis utilitzats amb les taxes corresponents, la ma d'obra i els treballs externs realitzats durant la reparació del vehicle.
- **Total_pressupost**: és el resultat d'aplicar els impostos corresponents al preu base del pressupost.
- **Id_OrdreReparació**: l'atribut descriu la relació 1 a 1 que hi ha entre aquesta classe i la classe Ordre de reparació. Cada pressupost correspon a una única OR.

- **Id_Impost:** aquest atribut descriu la relació 1 a n que hi ha entre la classe Pressupost i la classe Impost. Cada pressupost tindrà un tipus d'impost assignat.
- **Acceptat:** indica si el pressupost ha sigut acceptat o no pel client.

Els mètodes principals de la classe pressupost són:

- Controlador:
 - _construct():** és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents. En aquest cas, el constructor conté una taula d'impostos.
 - obtenirOR():** obté les dades l'ordre de reparació a partir de la qual s'ha generat el pressupost.
 - obtenirImpostos():** obté els impostos que s'han d'aplicar a aquest pressupost.
 - imprimirPressupost(\$id):** genera una plantilla d'impressió amb les dades del pressupost amb id *\$id*.
 - imprimir_pdfAction():** genera la plantilla corresponent per imprimir a pdf el pressupost.
 - calcularTotal():** calcula el total del pressupost a partir del preu base i l'aplicació de l'impost corresponent.
 - obtenirClient():** obté el client de la factura a partir de l'ordre de reparació.
 - modificarPressupostPerFactura():** si el client accepta el pressupost, aquest mètode permet modificar l'ordre de reparació per a generar-ne la factura corresponent.
 - listarpressupostosPendents():** genera una consulta SQL per obtenir els pressupostos pendents d'acceptar i els mostra a l'escriptori de l'aplicació.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.18 Albarà

ALBARÀ
id : int preu_base : float data : String <i>id_FacturaÚnicaMensual</i> : int <i>id_OrdreReparació</i> : int

```
_construct()
newAction()
createAction(Request $request)
editAction($id)
updateAction(Request $request, $id)
searchAction()
showAction($id)
deleteAction(Request $request, $id)
obtenirOR()
imprimirAlbara($id)
imprimir_pdfAction()
obtenirClient()
```

- **Id**: és l'identificador que utilitza l'aplicació.
- **Preu_base**: aquest atribut descriu el preu base de l'albarà, és a dir, la suma dels recanvis utilitzats amb les taxes corresponents, la ma d'obra i els treballs externs realitzats durant la reparació del vehicle.
- **Data**: correspon a la data de creació de l'albarà.
- **Id_FacturaÚnicaMensual**: descriu la relació N a 1 que existeix entre aquesta classe i la classe Factura Única Mensual. Si el client, al recollir el vehicle, requereix el lliurament d'algun document, se li lliurarà un albarà, que un cop al mes seran facturats i agrupats amb una única factura mensual.
- **Id_OrdreReparació**: l'atribut descriu la relació 1 a 1 que hi ha entre aquesta classe i la classe Ordre de reparació. Cada albarà correspon a una única OR.

Els mètodes principals de la classe albarà són:

- Controlador:
 - _construct()**: és el constructor de la classe. Un cop definides les relacions entre les classes, el CRUD de Symfony el genera automàticament amb les taules de dades corresponents.
 - obtenirOR()**: obté les dades l'ordre de reparació a partir de la qual s'ha generat l'albarà.
 - imprimirAlbara(\$id)**: genera una plantilla d'impressió amb les dades de l'albarà amb id *\$id*.
 - imprimir_pdfAction()**: genera la plantilla corresponent per imprimir a pdf l'albarà.
 - obtenirClient()**: obté el client de la factura a partir de l'ordre de reparació.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.19 Impost

IMPOST
id : int descripció : String quantitat : int
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id)

- **Id**: és l'identificador de la taxa.
- **Descripció, quantitat**: aquests atributs descriuen el tipus d'impost i el preu d'aquest.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

8.2.3.20 Configuració

CONFIGURACIÓ
nif : String nom : String nº_registre_industrial : String codi_postal : int població : String província : String telèfon : int mòbil : int e-mail : String
_construct() newAction() createAction(Request \$request) editAction(\$id) updateAction(Request \$request, \$id) searchAction() showAction(\$id) deleteAction(Request \$request, \$id)

Els atributs d'aquesta classe són les dades de l'empresa. Un usuari amb rol d'administrador les podrà modificar si n'és necessària l'actualització.

S'han omès els mètodes comuns a cada classe, per veure'n la descripció, consultar l'apartat 8.2.3.1.

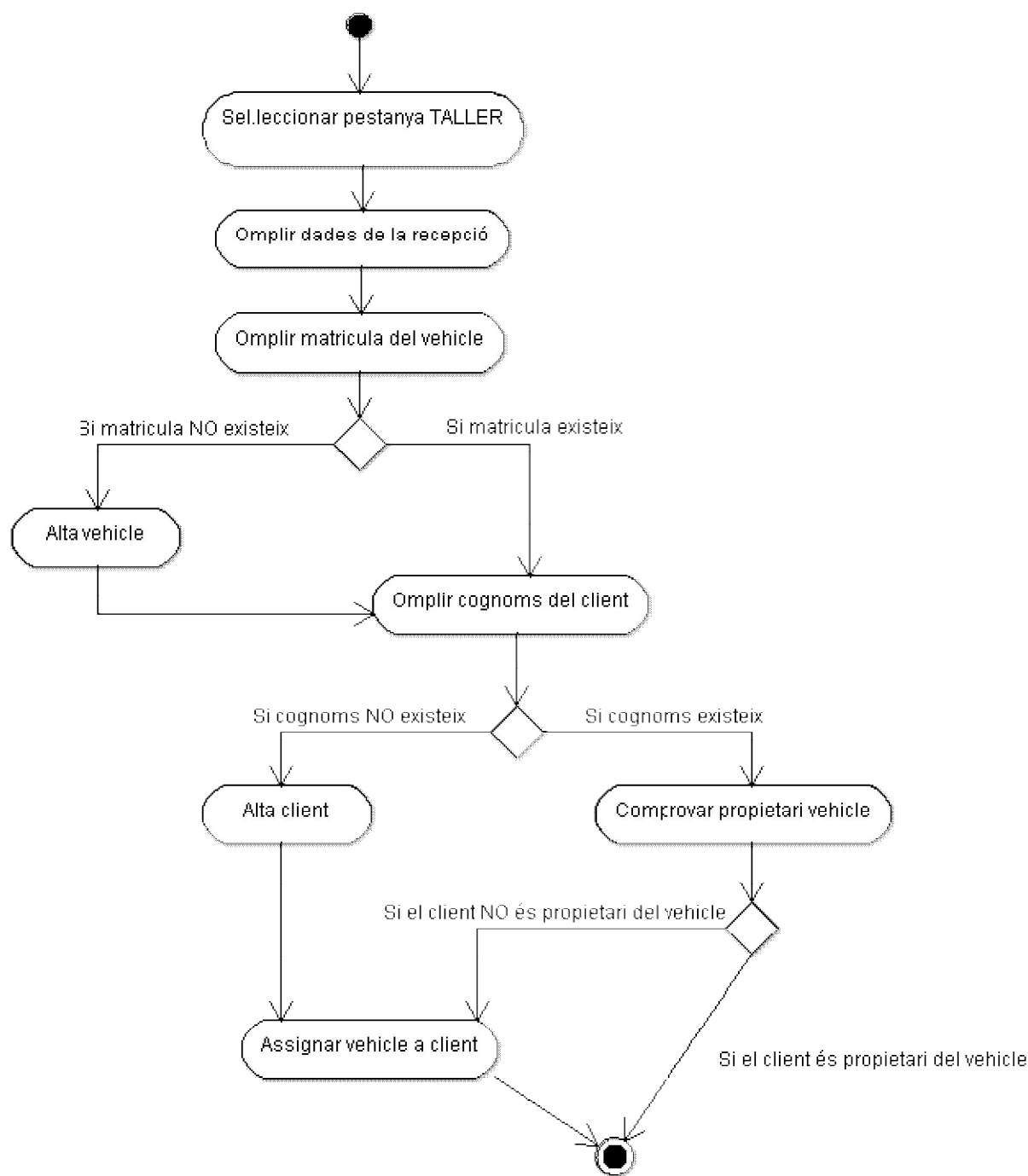
8.2.4 Diagrames d'activitat

En aquest apartat es mostraran els diagrames d'activitat que serveixen per destacar alguna part de l'aplicació.

8.2.4.1. Alta d'una recepció

Aquest diagrama explica el procés d'alta d'una recepció, des del moment en què un client porta el vehicle al taller i aquest, queda enregistrat al sistema per procedir a la reparació o confecció d'un pressupost.

S'ha considerat oportú fer aquest diagrama ja que no és un alta trivial perquè s'ha d'assignar el client al vehicle recepcionat.



Tal com s'ha dit anteriorment, aquest diagrama d'activitat mostra el procés d'alta d'una recepció.

Quan un usuari rep a un client amb el vehicle, selecciona la pestanya TALLER de l'aplicació, i automàticament es genera l'alta de la recepció. A aquesta pantalla, l'aplicació demana a l'usuari:

- la data d'entrada,
- si el client desitja o no fer un pressupost abans de reparar el vehicle,
- si recollirà les peces substituïdes,
- si la reparació és en garantia,
- la data aproximada de recollida,
- la descripció del problema a reparar
- la matrícula del vehicle
- i els cognoms del client

L'aplicació en aquest punt, comprova si la matrícula introduïda existeix a la base de dades i en cas afirmatiu, comprova si els cognoms entrats corresponen al propietari del vehicle entrat amb anterioritat. Si és així, s'ha finalitzat el procés d'alta. En cas que la matrícula no existeixi a la base de dades, és procedirà a donar d'alta el vehicle. Un cop fet, es consultarà a la base de dades si el client introduït existeix a la base de dades. En cas afirmatiu, assignarem el client al vehicle i en cas negatiu, es donarà d'alta el client i s'assignarà el client al vehicle.

A partir d'aquest moment, el vehicle queda en dipòsit al taller fins que un operari realitzi el pressupost o la reparació. Per fer-ho més visual, a la pantalla ESCRIPTORI de l'aplicació, apareixerà en la taula a la dreta de la pantalla indicant l'estat.

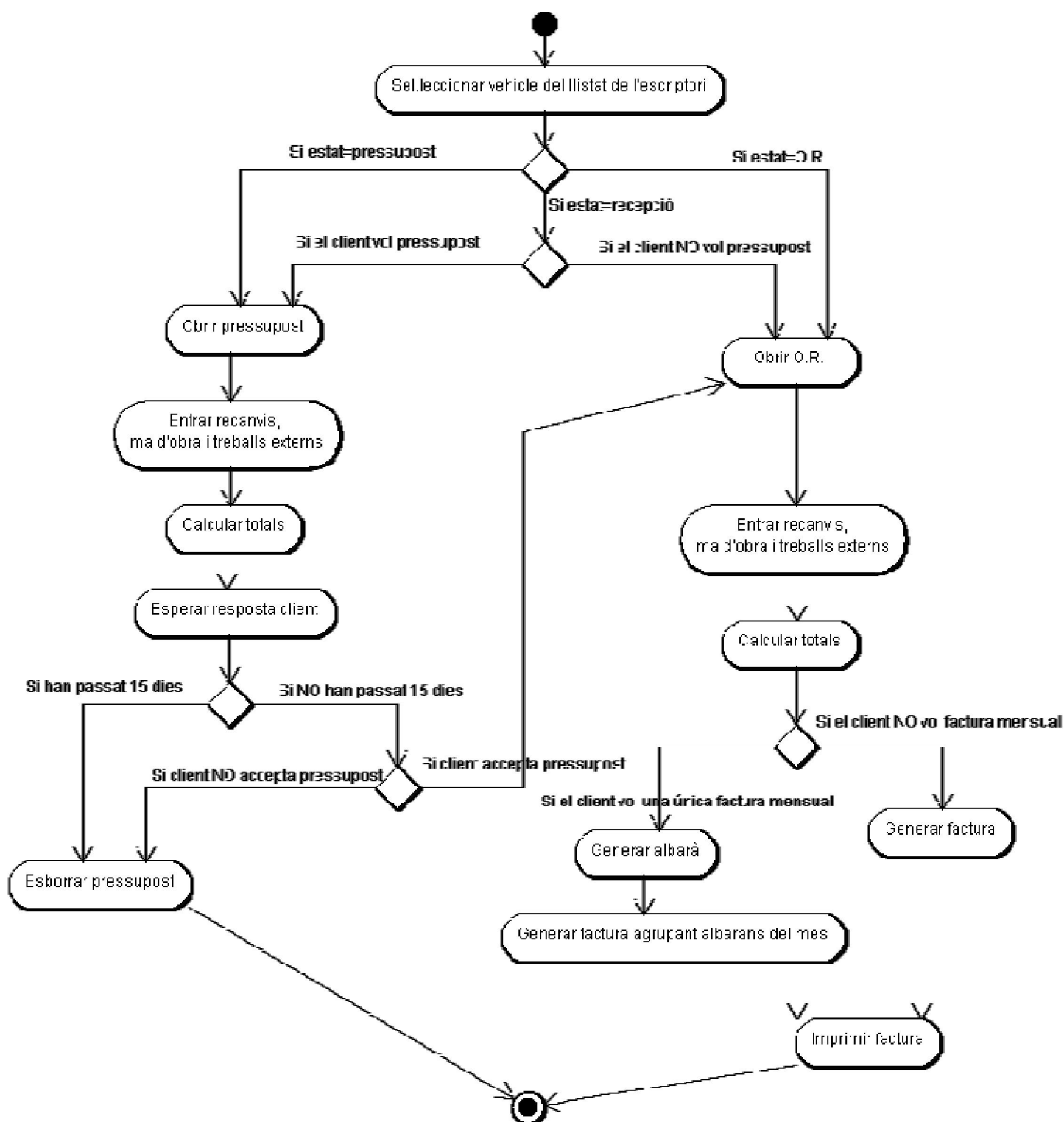
8.2.4.2. Gestió de l'escriptori

Aquest diagrama ajuda a entendre el procés que segueix una reparació, des de l'entrada del vehicle al taller, fins que en surt amb la reparació realitzada.

Cal destacar que la pantalla principal de l'aplicació és l'escriptori, on es pot trobar el menú de l'aplicació amb les diferents opcions i l'element principal que en són les 2 taules que indiquen els vehicles que hi ha actualment al taller.

La taula de l'esquerra, mostra els pressupostos vigents. És a dir, els pressupostos fets que estan pendents d'acceptació del client. Si passats 15 dies no s'ha obtingut cap resposta per part del client, són esborrats automàticament del llistat.

La taula de la dreta mostra els vehicles recepcionats i en reparació. El vehicles en estat de recepció estan pendents de realitzar un pressupost o de reparar, i els vehicles amb estat OR (ordre de reparació) s'estan reparant.



Tal com es pot veure en aquest diagrama, quan un usuari selecciona un vehicle del llistat que apareix a la pantalla escriptori, té diverses opcions segons l'estat:

- recepció
- pressupost

- ordre de reparació

Si es selecciona una recepció, cal consultar la recepció per saber si el client vol o no fer un pressupost de la reparació, en cas afirmatiu, s'obre un pressupost. En cas negatiu, s'obre automàticament una ordre de reparació.

En el cas d'escollir un pressupost, l'operari mirarà i farà les proves pertinents al vehicle per decidir quins recanvis, la ma d'obra i els treballs externs que cal efectuar en el vehicle. Ho introduirà a l'aplicació, i en aquest moment, quedarà pendent d'acceptació del client. Si el client accepta, es generarà una ordre de reparació. Si no, el pressupost serà esborrat, de la mateixa manera que si passats 15 dies no s'obté cap resposta.

Si l'usuari selecciona una ordre de reparació, significarà que aquest vehicle ja ha passat per l'estat de recepció, i en cas d'haver-hi pressupost, aquest ha estat acceptat. Els mecànics, al reparar el vehicle, poden obrir l'ordre de reparació tantes vegades com faci falta per afegir-hi els treballs realitzats, els recanvis, etc. Un cop finalitzada la reparació, es tancarà l'OR calculant-ne el total i generant la factura. Depenent de les condicions de facturació del client, es generarà un albarà si es confecciona una única factura a final de mes, o una factura en cas contrari. L'operació s'acabarà imprimint la factura.

Cal destacar que no és obligatori que en un pressupost o ordre de reparació hi hagi recanvis, ma d'obra i treball externs. Ens podem trobar algun cas que amb ma d'obra n'hi hagi prou per reparar el vehicle, però no s'ha especificat al diagrama per fer-lo més entenedor.

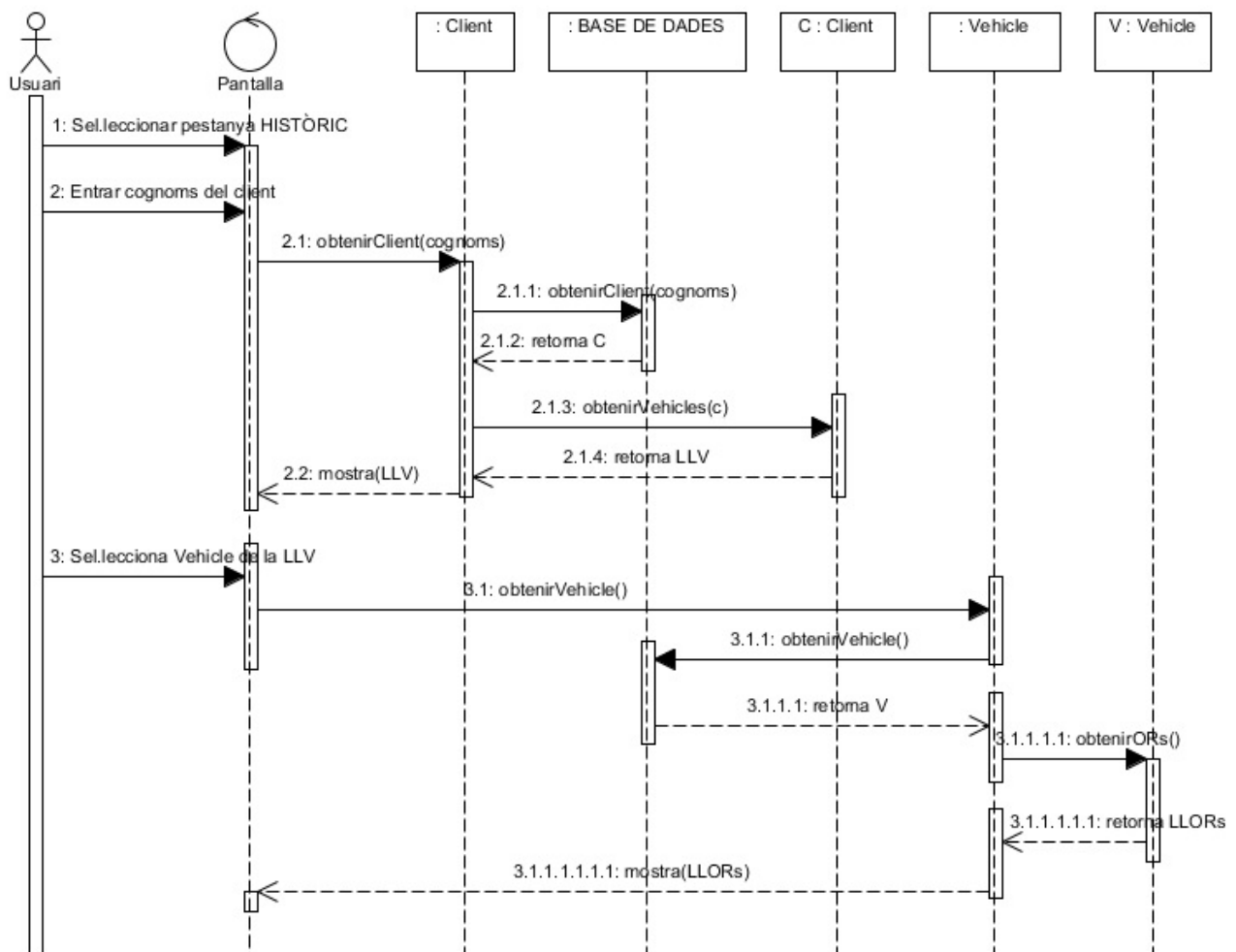
8.2.5 Diagrames de seqüència

En aquest apartat es mostren alguns dels diagrames de seqüència que s'han cregut més complerts o que tenen més interacció amb altres classes.

8.2.5.1. Consultar històric d'un vehicle

Aquest diagrama de seqüència mostra el procés de consultar l'històric d'un vehicle a partir dels cognoms del client. Aquesta consulta és molt útil pels operaris de l'empresa ja que en alguns casos, per exemple, cal controlar què s'ha fet a l'anterior manteniment o si s'ha substituït la corretja de distribució.

A l'aplicació també es pot consultar a partir de la matrícula del vehicle. El diagrama seria similar a aquest, senzillament s'accediria a la base de dades per buscar un vehicle per matrícula en lloc de facilitar els cognoms del client.



L'usuari que necessiti consultar l'històric del vehicle, haurà de seleccionar la pestanya de l'aplicació amb nom Històric. La següent pantalla li demanarà els cognoms del client i al confirmar-ho, li apareixerà la llista de vehicles que han visitat alguna vegada al taller i se li han assignat. Per arribar a aquest resultat, l'aplicació cercarà a la base de

dades el client amb els cognoms entrats i mostrarà per pantalla el llistat dels vehicles guardats a la corresponent taula de vehicles.

Un cop el llistat apareix per pantalla, l'usuari triarà el vehicle a consultar. S'accedirà a la taula vehicle de la base de dades per consultar les ordres de reparació que té assignades el vehicle i es mostraran per pantalla.

En aquest moment, l'usuari podrà veure cada ordre de reparació seleccionant-la del llistat que apareix per pantalla, tornar endarrere per consultar-ne una altra o tornar a l'escriptori.

Els mètodes utilitzats per realitzar aquesta consulta, estan explicats a les seves classes corresponents de l'apartat 8.2.3.

8.2.5.2. Gestionar OR

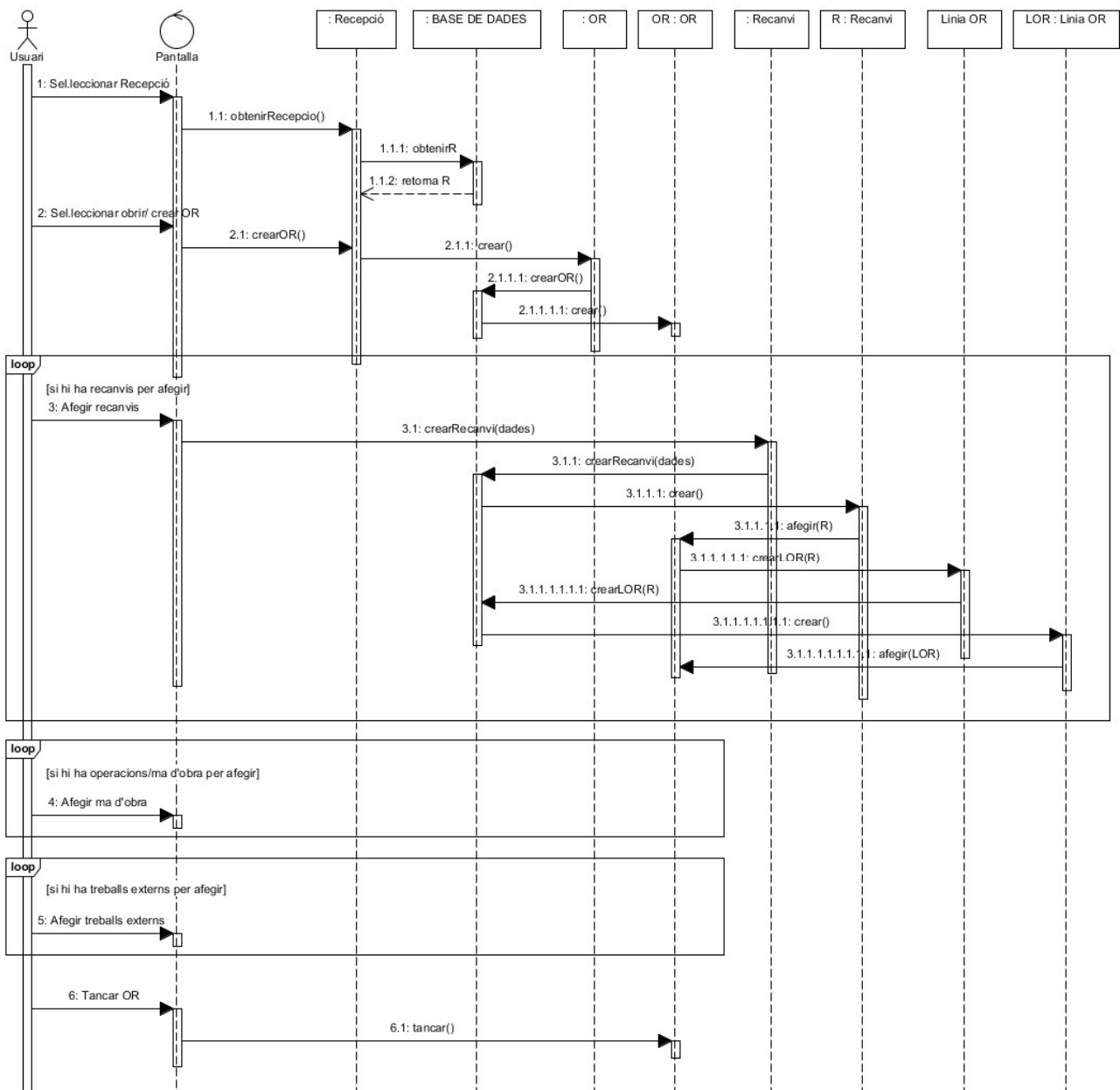
El diagrama de seqüència que es troba a continuació descriu el procés de crear una ordre de reparació a partir de la recepció, mostra com s'emplena l'O.R. i fins el moment del seu tancament. A l'apartat 8.2.5.3. es descriu el procés de facturació de l'ordre de reparació un cop tancada.

Quan un client porta un vehicle al taller, s'obre una recepció on apareix:

- el motiu de la visita,
- les dades del client i
- les dades del vehicle.

L'usuari que l'atén, li pregunta si vol realitzar un pressupost o no abans de fer la reparació, en cas afirmatiu s'obre un pressupost, i en cas contrari, l'ordre de reparació.

En aquest cas, s'ha decidit fer el diagrama de l'ordre de reparació, però el del pressupost seria similar tot i que al final del pressupost, un cop acceptat, s'obriria una ordre de reparació amb tots els recanvis, ma d'obra i treballs externs que s'havien pressupostat. L'usuari hauria de modificar l'ordre de reparació si és necessari i a continuació es tancaria, tal com es veu en aquest diagrama.



Tal com es pot veure al diagrama, l'usuari seleccionarà una recepció que apareixerà a l'escriptori de l'aplicació. A continuació escollirà l'opció d'obrir una ordre de reparació, (en aquest cas no es fa pressupost, es comença a reparar el vehicle automàticament). Un cop s'ha creat l'objecte OR a la base de dades, és el moment de reparar el vehicle i paral·lelament, l'operari introduirà a l'aplicació els recanvis que s'utilitzen durant la reparació. Per no fer feixuc el diagrama, només s'ha explicat el procés d'alta d'un recanvi, però els de mà d'obra i treball externs serien molt semblants.

Mentre no es tanqui l'OR, l'usuari podrà anar entrant recanvis. Per fer-ho li apareixerà una finestra on haurà d'introduir la quantitat, descripció, preu base del recanvi, es donarà d'alta el recanvi i es crearà una línia d'ordre de reparació on s'afegirà. Per cada

recanvi diferent, es crearà una línia d'OR que s'afegirà automàticament a l'OR que s'està emplenant.

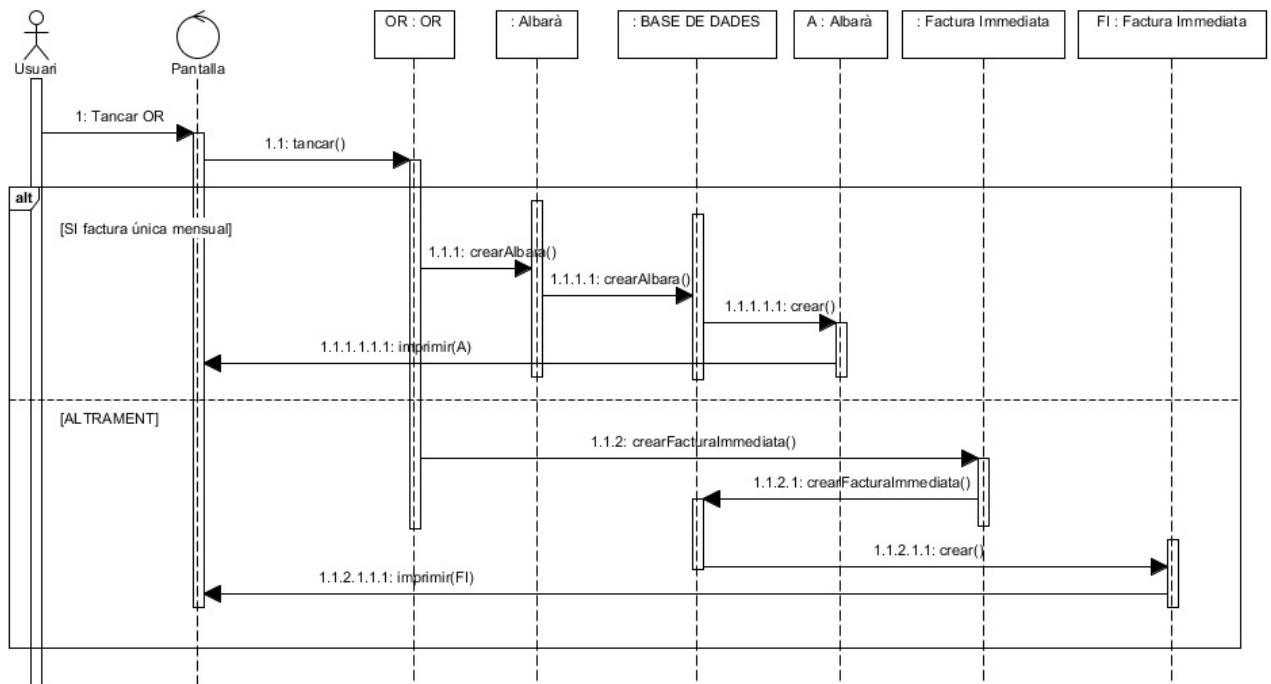
Un cop s'ha finalitzat la reparació i l'OR conté totes les línies d'OR amb els recanvis i treballs realitzats, l'usuari tancarà l'ordre de reparació seleccionant el botó corresponent. Un cop l'OR és tancada, queda emmagatzemada a la base de dades com una reparació més del vehicle. A l'usuari li apareixerà una pantalla on podrà facturar-la directament o deixar-ho pendent.

Al següent apartat, es descriu el procés de facturació de l'ordre de reparació un cop tancada.

Els mètodes utilitzats en aquest diagrama, estan explicats a les classes corresponents de l'apartat 8.2.3.

8.2.5.3. Facturar OR

Aquest diagrama de seqüència mostra el procés de facturació d'una ordre de reparació un cop s'ha finalitzat la reparació del vehicle.



Quan l'usuari tanca l'ordre de reparació, hi ha dues opcions:

- generar un albarà corresponent a aquesta OR (a final de mes es generarà una única factura mensual que contindrà tots els albarans del client),
- o generar una factura immediata

Aquest procés és automàtic ja que al donar d'alta un client ja se li assigna la forma de pagament.

En el cas de generar una factura única mensual, quan s'executa el mètode `tancar()`, es crea un albarà que conté l'OR actual, un cop emmagatzemat a la base de dades, s'imprimeix per pantalla donant l'opció a l'usuari d'imprimir-la en paper.

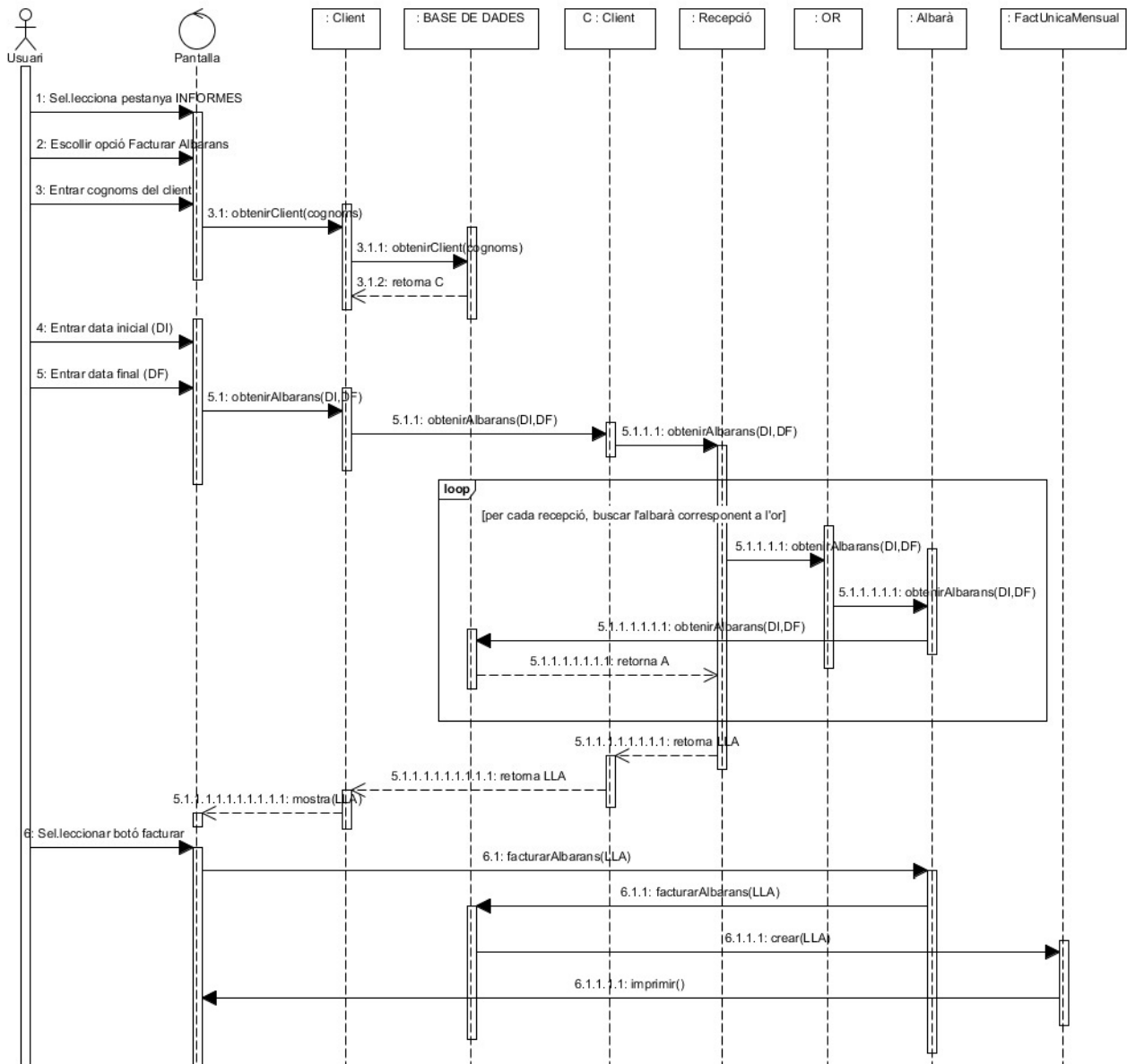
En cas contrari, quan s'executa el mètode `tancar()`, es crea una factura immediata amb el contingut de l'OR, i es donen les mateixes opcions que en l'altre cas, imprimir la factura per pantalla i en paper.

Al següent apartat es descriu el procés de facturació d'albarans.

Els mètodes utilitzats en aquest diagrama, estan explicats a les classes corresponents de l'apartat 8.2.3.

8.2.5.4. Facturar albarans

El diagrama de seqüència següent mostra el procés de facturació d'albarans. Un cop reparat el vehicle, si la forma de pagament assignada al client és generar una única factura mensual, s'agruparan tots els albarans del mes i es generarà una factura amb totes les reparacions del client.



Quan un usuari ha de facturar els albarans d'un client, haurà de seleccionar la pestanya Informes de l'aplicació. En aquesta pestanya es triarà l'opció Facturar albarans i a la nova finestra, introduir els cognoms del client.

Es buscarà a la base de dades el client corresponent als cognoms introduïts i en cas de que la cerca sigui correcta, apareixerà una nova finestra demanant la data d'inici i final. Un cop escollides les dates, el mètode obtenirAlbarans(DI,DF) buscarà a partir de les recepcions del client, tots els albarans corresponents entre les dates entrades per teclat.

El resultat es mostrarà per pantalla i triant el botó corresponent, es crearà una única factura amb la llista d'albarans seleccionats utilitzant el mètode facturar(llistaAlbarans)

de la classe Albarà. Com sempre, tindrem l'opció d'imprimir la factura per pantalla i en paper.

Els mètodes utilitzats en aquest diagrama, estan explicats a les classes corresponents de l'apartat 8.2.3.

8.2.6 Descripció de l'interfície de l'aplicació

En aquest apartat s'explica el disseny de l'interfície.

Aquest disseny s'ha desenvolupat a partir de les pautes que ha donat el client, les més destacables són:

- es vol una aplicació senzilla, entenedora i pràctica,
- amb una pantalla de login i logout,
- una pantalla que serà l'escriptori de l'aplicació on hi haurà dues taules:
 - o una amb els vehicles que estan físicament al taller,
 - o i una altra amb els pressupostos pendents d'acceptar.
- Una barra amb diferents pestanyes a la part de dalt de la finestra. Aquestes pestanyes són:
 - o Taller
 - o Històric
 - o Caixa
 - o Informes
 - o Consultes
 - o Base de dades
 - o Configuració
- Cada finestra tindrà a dalt la barra amb les diferents pestanyes per poder-se moure amb facilitat per l'aplicació.
- Des de cada finestra es podrà accedir a la pantalla principal de l'aplicació, en aquest cas, la de l'escriptori.

8.2.6.1. Pantalla de login i logout

A la pantalla següent es pot veure com ha de ser el login i logout de l'aplicació. A dalt a l'esquerra apareixerà el logotip de l'empresa o el seu nom, i com a fons s'utilitzarà una fotografia del taller.

A la part central hi apareixerà un quadrat on l'usuari introduirà l'identificador i la contrasenya.

Nom empresa o logotip

Login o Logout

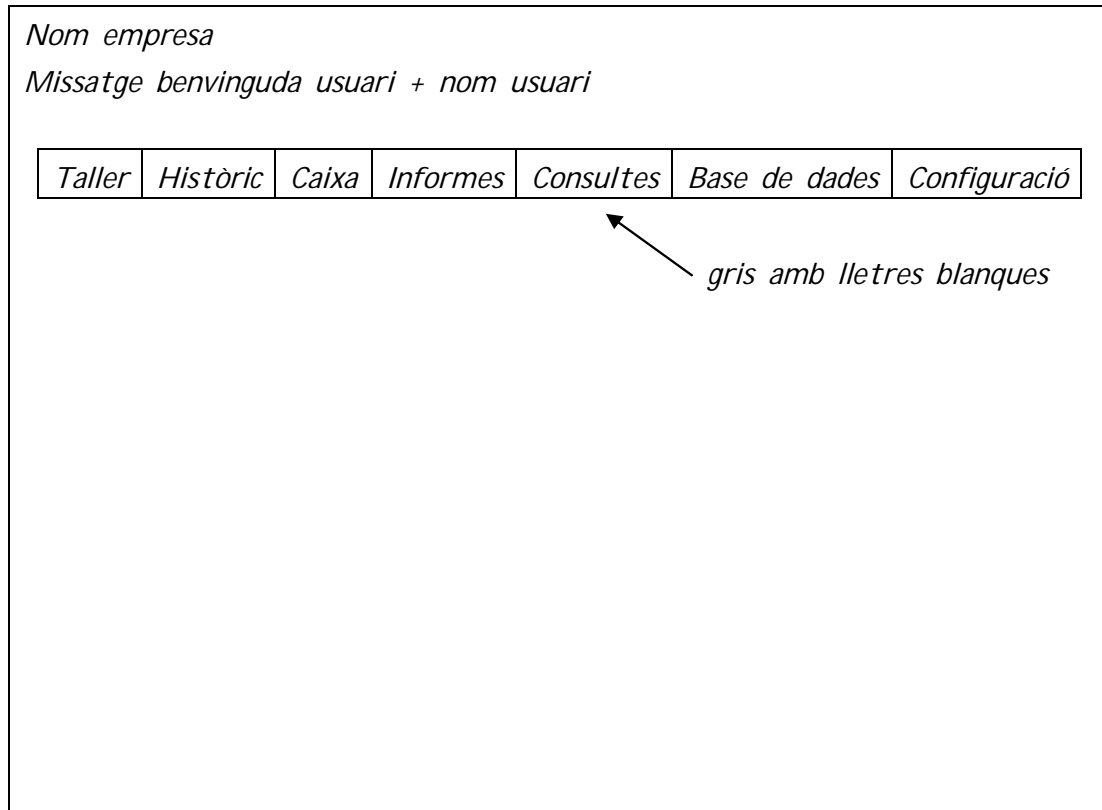
Usuari:
Password:

posar una foto de fons

8.2.6.2. Pantalla amb la barra d'opcions per tota l'aplicació

Tal com s'ha dit anteriorment, el client ha definit una barra amb les diferents opcions que vol que tingui l'aplicació.

A part de la barra, a cada pantalla apareixerà el nom de l'empresa i el nom de l'usuari que esta identificat actualment. Clicant sobre el nom, podrà modificar les seves opcions de configuració.



8.2.6.3. Pantalla escriptori

L'escriptori es considera la pantalla principal de l'aplicació. Hi apareixeran les mateixes dades que s'han descrit a l'apartat anterior, i les relacionades amb els vehicles que hi ha actualment al taller o treballs pendents.

A l'esquerra es pot trobar un llistat amb els pressupostos pendents d'acceptar. Es mostraran les matricules dels pressupostos pendents i es podran seleccionar per veure'n el contingut. Transcorreguts 15 dies, si el client no ha acceptat el pressupost, aquest s'esborrarà automàticament del llistat.

A la dreta hi apareixerà un llistat amb els vehicles que estan físicament al taller:

- les recepcions pendents de pressupostar o reparar

- les ordres de reparacions obertes, és a dir, els vehicles que s'estan reparant actualment.
- I les ordres de reparacions pendents de facturar.

Tots els vehicles que apareixen en aquest llistat, al seleccionar-ne la seva fila, s'obrirà automàticament l'ordre de reparació o la recepció corresponent.

Nom empresa

Missatge benvinguda usuari + nom usuari

<i>Taller</i>	<i>Històric</i>	<i>Caixa</i>	<i>Informes</i>	<i>Consultes</i>	<i>Base de dades</i>	<i>Configuració</i>
---------------	-----------------	--------------	-----------------	------------------	----------------------	---------------------

<i>Pressupostos</i>	<i>Vehicles al taller</i>			
<i>matrícula 1</i>	<i>Matrícula</i>	<i>Client</i>	<i>?</i>	<i>?</i>
<i>matrícula 2</i>				
<i>matrícula 3</i>				
<i>...</i>	<i>...</i>		<i>...</i>	
<i>...</i>	<i>...</i>		<i>...</i>	

(als 15 dies s'esborren)

Diferenciar l'estat amb colors o alguna icona

8.2.6.4. Pantalla de la pestanya Consultes

La resta de pestanyes de l'aplicació contindrà de manera molt clara el que s'ha de realitzar en cada una d'elles. En aquest cas s'ha escollit la pestanya de Consultes per posar-ne un exemple.

Aquesta finestra contindrà el mateix menú que s'ha descrit anteriorment i 2 taules. A la taula de l'esquerra podrem veure els llistats automàtics, és a dir, les consultes generals de la base de dades, i a la de la dreta les consultes definides pel client.

Nom empresa
Missatge benvinguda usuari + nom usuari

<i>Taller</i>	<i>Històric</i>	<i>Caixa</i>	<i>Informes</i>	<i>Consultes</i>	<i>Base de dades</i>	<i>Configuració</i>
---------------	-----------------	--------------	-----------------	------------------	----------------------	---------------------

<i>Llistats</i>	<i>Consultes</i>
- <i>de clients</i>	- <i>buscar vehicle per matrícula</i>
- <i>de vehicles</i>	- <i>factures pendents de cobrar</i>
...	...
...	...
...	...

torna a l'escriptori

9. Implementació i proves

Aquest capítol mostrarà la implementació dels algorismes i processos del projecte. Es detallarà la implementació de classes i frameworks, se'n realitzarà una descripció del funcionament, s'informarà de quines són les finalitats i millores obtingudes gràcies a tot el conjunt d'eines desenvolupades o utilitzades, ja sigui en quant a millora de rendiment, obtenció de noves funcionalitats o temps de desenvolupament. Aquestes implementacions s'agrupen en diferents apartats, ja que part d'aquests algorismes es reutilitzen per nombroses parts de la web.

En alguns casos, es detallaran petites proves d'execució i els seus respectius resultats de manera descriptiva.

9.1. Autenticació d'usuaris

En aquest apartat es detallarà la implementació dels mètodes de login i logout.

El framework Symfony ofereix diferents opcions per crear el formulari d'accés a l'aplicació: especificant els usuaris únicament al fitxer *security.yml* o carregant els usuaris des de la base de dades. En aquest cas s'utilitzaran les dues opcions. Els usuaris de l'aplicació es desaran a la base de dades, però per fer les proves, s'especificarà al mateix fitxer un usuari administrador.

Per començar, s'ha de modificar el fitxer *Symfony/app/config/security.yml* per activar el firewall que utilitzarà el framework per realitzar les autenticacions. Symfony serà l'encarregat de fer totes les gestions per nosaltres amb unes línies al fitxer de configuració.

El contingut del fitxer *security.yml* és:


```

security:
  encoders:
    Symfony\Component\Security\Core\User\User: plaintext
    Taller\TallerBundle\Entity\Treballador: plaintext
  role_hierarchy:
    ROLE_ADMIN: ROLE_GERENT
    ROLE_SUPER_ADMIN: [ROLE_MECANIC, ROLE_ALLOWED_TO_SWITCH]
  providers:
    users_db:
      entity: { class: Taller\TallerBundle\Entity\Treballador,
        property: username }
    in_memory:
      memory:
        users:
          admin: { password: admin, roles: ['ROLE_ADMIN'] }
  firewalls:
    users:
      pattern: ^/
      anonymous: ~
      form_login:
        login_path: login
        login_check: login_check
        logout_path: logout

```

Tal com es pot veure al fitxer, s'han definit els 3 rols de l'aplicació: ROLE_ADMIN, ROLE_GERENT i ROLE_MECANIC. L'últim representa els usuaris de la base de dades que són del tipus Treballador, i l'emmagatzema a memòria que és l'usuari *admin*.

La pestanya de *security/encoders* permet codificar les claus d'accés en aquest cas s'ha decidit mantenir-les sense fins al moment de l' implantació.

A continuació, cal modificar les classes Treballador i Rol ja que Symfony ho necessita per a poder gestionar les autenticacions. Aquestes entitats han d'implementar les interfícies `Symfony\Component\Security\Core\User\UserInterface` i `Symfony\Component\Security\Core\Role\RoleInterface` respectivament.

Entitat Treballador:

```

<?php
namespace Taller\TallerBundle\Entity;
use Symfony\Component\Security\Core\User\UserInterface;
use Doctrine\ORM\Mapping as ORM;
/**
 * @ORM\Entity
 * @ORM\Table(name="treballador")
 */
class Treballador implements UserInterface {
    [...]

```

l'entitat Rol:

```
[...]  
class Rol implements RoleInterface {  
    [...]
```

Aquestes dues entitats tenen una relació un a molts ja que un treballador pertany a un rol i cada rol pot tenir diferents treballadors.

A continuació queda definir la plantilla Twig per poder fer el login i afegir la ruta al fitxer d'enrutament per a poder-la utilitzar. La plantilla de *login.html.twig* és:

```
{% if error %}  
    <div>{{ error.message }}</div>  
{% endif %}  
<form action="{{ path('login_check') }}" method="post">  
  
    <label for="username">Username:</label>  
<input id="username" type="text" name="_username"  
value="{{ last_username }}" />  
  
    <label for="password">Password:</label>  
<input id="password" type="password" name="_password" />  
  
    <input type="submit" name="login" />  
  
</form>
```

Fitxer *routing.yml*:

```
login:  
    pattern: /login  
    defaults: { _controller: TallerBundle:Security:login }  
login_check:  
    pattern: /login_check  
logout:  
    pattern: /logout
```

Una ruta és l'associació entre un patró d'URL i un controlador. El fitxer d'enrutament *routing.yml* defineix les rutes de totes les pantalles de l'aplicació.

La pantalla del login és la ruta de l'aplicació seguida de */login*, tal com defineix l'opció *pattern*, i està associada amb el controlador *TallerBundle:Security:login*. Symfony utilitza el valor de l'opció *_controller* per determinar la funció PHP que s'executa per respondre a la petició.

Per controlar el contingut de l'aplicació segons el rol assignat a cada usuari, es pot fer a nivell de codi, o personalitzant les plantilles. En la majoria de casos personalitzaré les plantilles ja que d'aquesta manera els controladors queden lliures i es poden reutilitzar sense condicions. La condició a utilitzar a les plantilles segons el rol és:

```
{% if is_granted('ROLE_ADMIN') %}
    <a href="..."> Esborrar </a>
{% endif %}
```

9.2. Pantalla principal, l'escriptori

Un cop l'usuari s'ha identificat, accedeix a la pantalla principal de l'aplicació. A la part superior de la pantalla, sota el nom de l'empresa, apareix un missatge de benvinguda seguit del nom.



Com es pot veure a continuació, el fragment de codi mostra com la plantilla Twig obté l'usuari que s'acaba d'identificar. La variable *app* és global i s'hi pot accedir des de qualsevol plantilla:

```
<div id="header">
    <h1><a href="logout">Collsaplana Motor, SL</a></h1>
    <h2>Benvingut usuari: {{ app.user }}</h2>
</div>
```

També es pot observar que, seleccionant el nom de l'empresa, l'usuari tancarà la sessió a l'aplicació.

La plantilla de l'escriptori és on es defineix el menú principal de l'aplicació, el codi del qual és :

```
[...]
<div id="menu">
    <ul>
        <li><a href="recepcio_create">Taller</a></li>
        <li><a href="historic">Històric</a></li>
        <li><a href="caixa">Caixa</a></li>
        <li><a href="informes">Informes</a></li>
        <li><a href="consultes">Consultes</a></li>
        <li><a href="basededades">Base de dades</a></li>
        <li><a href="configuracio">Configuració</a></li>
    </ul>
</div>
[...]
```

En aquesta mateixa plantilla, hi apareixeran els pressupostos pendents d'acceptar i els vehicles que hi ha actualment al taller.

9.2.1. Pressupostos pendents d'acceptar

El mètode pressupostos pendents d'acceptar està creat dins el controlador de l'entitat Pressupost. El mètode accedeix a la base de dades per obtenir els pressupostos que no s'han acceptat i en retorna un llistat amb la matrícula i el número de pressupost.

Symfony i Doctrine permeten modificar les classes o entitats sense problemes en tot moment, ja que aplicant l' instrucció:

```
$ php app/console doctrine :schema :update --force
```

Al shell de XAMPP, s'actualitza automàticament la base de dades. Doctrine genera les declaracions SQL necessàries per actualitzar-la.

El mètode *PressupostController* és :

```
public function llistarPressupostosPendentsAction(){
    $em = $this->getDoctrine()->getManager();
    $dql = "SELECT p
          FROM TallerBundle:Pressupost p
          WHERE p.acceptat=false
          ";
    $query = $em->createQuery($dql);
    $entities = $query->getResult();
    return $this->
    render('TallerBundle:Default:escriptori.html.twig',
        array('pressupostos' => $entities,
        ));
}
```

Symfony permet declarar comandes SQL dins els mètodes de la classe, utilitzant un llenguatge propi anomenat DQL.

Com es pot veure al mètode anterior, es busquen tots els pressupostos de la base de dades amb l'atribut acceptat a fals. Aquí no es controla la data de caducitat dels pressupostos ja que la plantilla ho realitzarà cada vegada que s'utilitzi.

A continuació es mostra el fragment de codi de la vista del llistat de pressupostos pendents d'acceptar que apareix a la plantilla *escriptori.html.twig*:

```
[...]
<div class="left">
  <h2>Pressupostos pendents d'acceptar :</h2>
  {% if pressupostos is empty %}
    <p>No existeixen pressupostos pendents</p>
  {% else %}
    <ul>
      {% for pressupost in pressupostos %}
        {% if pressupostos.or.recepcio.data+15 <= "now" %}
          <li>{{ pressupost.or.recepcio.vehicle.matricula }}</li>
        {% endif %}
      {% endfor %}
    </ul>
  {% endif %}
  <br/><br/>
  (Passats 15 dies desapareixeran de la llista)
</div>
[...]
```




L'etiqueta `<div class="left">` indica que tot el contingut serà mostrat a la part esquerra de la pantalla. En aquest fragment de codi es pot apreciar el funcionament de les plantilles amb llenguatge Twig. Construir condicionals és molt fàcil i entenedor ja que, sense definir cap variable, la plantilla obté les dades del controlador i la base de dades.

Tal com s'ha dit anteriorment, a la plantilla és a on es controla si la data del pressupost és superior a quinze dies. D'aquesta manera, el mètode del controlador és reutilitzable.

9.2.2. Mostrar vehicles al taller

El mètode mostra tots els vehicles que hi ha al taller, ja sigui recepcionats o en reparació. Des d'aquest llistat es podrà accedir a l'ordre de reparació per afegir més material o ma d'obra, o a la recepció per generar l'ordre de reparació.

El problema trobat per fer aquest mètode és la dificultat per diferenciar l'estat en què es troba un vehicle al taller. Depenent de l'estat, es visualitzarà per pantalla amb un color o un altre. A la següent taula es fa un resum del codi de colors utilitzat i les condicions a complir per diferenciar un estat d'un altre :

ICONA	ESTAT	CONDICIONS
	Pressupost pendent d'acceptar	<code>or.pressupost.acceptat == false & or.recepcio.data+15dies >= NOW</code>
	OR d'un vehicle que esta esperant per ser reparat	<code>or.recepcio.renuncia_pressupost== true & or.liniaor == NULL</code>
	OR d'un vehicle que s'està reparant	<code>or.recepcio.renuncia_pressupost == true & or.liniaor != NULL & or.tancat=false</code>

Tal com es pot veure, a la taula també apareix el codi de color dels pressupostos pendents d'acceptar de l'apartat anterior.

El mètode que s'utilitza per llistar els vehicles que hi ha al taller, és de la classe *DefaultController.php* :

```
public function llistarVehiclesTallerAction(){
```

```

$em = $this->getDoctrine()->getManager();

$dql = "SELECT r
      FROM TallerBundle:OrdreReparacio r
      WHERE r.tancat=false
      AND r.pressupost IS NULL
      ";

$query = $em->createQuery($dql);
$entities = $query->getResult();

return $this->render(
    'TallerBundle:Default:escriptori.html.twig', array(
    'ors' => $entities,
    ));
}

```

I per diferenciar si és una ordre de reparació o una recepció, utilitzem les condicions descrites a la plantilla *escriptori.html.twig* :

```

[...]
<h1>Vehicles al taller :</h1>
<br />
{# OR FETES: #}
{% if ors is empty %}
    <p>No tenim o.r.'s al taller</p>
{% else %}
<table cellpadding="10" cellspacing="10">
    {% set c = 0 %}
    <tr>
        {% for o in ors %}

            {% if c == 5 %}
            </tr> <tr>
                {% set c = 0 %}
            {% endif %}
            {% if o.liniaor is empty %}
                {# si no te linies or, cotxe taronja #}
                <td><a href="{{ path('ordrereparacio_completar',
                    {'id': o.id}) }}">
                <br> Recepció={{ o.id }} <br>
                Matrícula={{ o.recepcio.vehicle.matricula }}
                </a></td>
            {% else %}
                {# si or pero esperant, el cotxe es de color verd #}
                <td><a href="{{ path('ordrereparacio_completar',
                    {'id': o.id}) }}"> 
                <br> OR={{ o.id }} <br> Matrícula=
                {{ o.recepcio.vehicle.matricula }}</a></td>
            {% endif %}

            {% set c=c+1 %}

```

```

    {% endfor %}

</tr>
</table>
{% endif %}
<br /><br />
[...]
```

La plantilla recorre totes les ordres de reparació *ors* passades per paràmetre *i*, en cas que n'hi hagi, les va mostrant amb files de 5 en 5. Si l'ordre de reparació té línies d'o.r. (línies d'ordre de reparació) significarà que ja s'ha començat a treballar amb el vehicle. Per tant, el vehicle es mostrarà com una ordre de reparació amb l'ícona verda.

A les plantilles *twig*, per accedir a la matrícula d'un vehicle, és tan senzill com utilitzar les relacions que existeixen entre les taules :

```
{{ o.recepcio.vehicle.matricula }}
```

En aquest cas, la variable *o* és l'objecte que s'utilitza per iterar sobre la llista d'ordres de reparació *ors*. Accedint a *o.recepcio* s'obté la recepció realitzada per a crear l'ordre de reparació *o*, i a través de la recepció es rep l'objecte *vehicle* de la recepció. Per consultar l'atribut només queda cridar-lo.

Es mostra un exemple de la vista de l'escriptori :

The screenshot shows the Collsaplana Motor, SL web application interface. At the top, it says "Collsaplana Motor, SL" and "Benvingut usuari: Anna". Below this is a navigation menu with items: TALLER, HISTÒRIC, CAIXA, INFORMES, CONSULTES, BASE DE DADES, and CONFIGURACIÓ. The main content area is divided into two sections. On the left, under "Pressupostos pendents d'acceptar :", there are two red car icons representing pending orders: "Pressupost=1 Matricula=1000BHT" and "Pressupost=2 Matricula=1234DHD". A note below states "(Passats 15 dies desapareixeran de la llista)". On the right, under "Vehicles al taller :", there are three car icons representing vehicles in the workshop: "Recepció=1 Matricula=5896GTH" (orange icon), "Recepció=5 Matricula=2204CSN" (orange icon), and "OR=8 Matricula=2037DFH" (green icon).

9.3. Pantalles de tipus CRUD

Les pantalles de tipus CRUD (*Create, Read, Update and Delete*) engloben les operacions bàsiques que es realitzen sobre una entitat com són: l'alta, baixa, modificació i cerca. Ja que moltes entitats, per no dir totes, comparteixen aquestes pantalles i les diferències venen aportades per incloure o excloure petites funcionalitats, s'ha decidit explicar-ne només una de les més utilitzades, la classe Client.

La creació d'aquestes operacions bàsiques és automàtica utilitzant les opcions que ofereix Doctrine, tal com s'ha explicat a l'apartat 7.2.3.2 CRUD. Un cop generades, s'hauran de revisar i personalitzar segons les necessitats de l'aplicació.

Per veure'n algun exemple, a continuació es mostra el mètode de creació d'un Client:

```
public function createAction(Request $request) {

    $entity = new Client();
    $form = $this->createForm(new ClientType(), $entity);
    $form->bind($request);
    $em = $this->getDoctrine()->getManager();
    $client=($em->getRepository('TallerBundle:Client')->
        findOneByDni($form[dni]->getData()));
    if ($form->isValid() && !$client) {
        $em = $this->getDoctrine()->getManager();
        $em->persist($entity);
        $em->flush();
        return $this->redirect($this->
            generateUrl('taller_homepage'));
    }
    if ($client) {
        // si el client ja existeix, mostrar missatge avís
        $this->get('session')->getFlashBag()->add(
            'notice',
            'AVÍS: El client ja existeix a la base de dades.'
        );
    }
    return $this->render('TallerBundle:Client:new.html.twig', array(
        'entity' => $entity,
        'form' => $form->createView(),
    ));
}
```

El mètode *createAction* del Controlador crea un objecte de tipus Client. Per crear el formulari que obtindrà les dades, cridarà la classe ClientType. Un cop s'ha obtingut l'informació dels atributs, es crea l'enllaç cap a la base de dades i és desat. Per no tenir clients duplicats a la base de dades, es controla si existeix un client amb el mateix *dni*.

A continuació es mostra la creació del formulari de la classe *ClientType* :

```
public function buildForm(FormBuilderInterface $builder,
array $options) {
    $builder
        ->add('dni')
        ->add('nom')
        ->add('cognoms')
        ->add('telefon')
        ->add('mobil')
        ->add('poblacio')
        ->add('codi_postal')
        ->add('provincia')
        ->add('email')
        ->add('condiciopagament')
    ;
}
```

La plantilla *Client/new.html.twig* que el Controlador crida perquè l'usuari pugui introduir les dades al formulari:

```
[...]
{% block body %}

<div id="content">
<h1>Alta client</h1>
<form action="{{ path('client_new') }}" method="post"
    {{ form_enctype(form) }}>
    {{ form_rest(form) }}
    {{ form_widget(form) }}
    <p> <button type="submit">Crea</button> </p>
</form>

[...]
```

Aquest fragment que es troba a continuació apareix pràcticament a totes les plantilles de l'aplicació per a poder mostrar per pantalla un missatge d'avís amb els errors trobats:

```
[...]
<div id="header">
    {% for flashMessage in app.session.flashbag.get('notice') %}
        <div class="flash-notice">
            <FONT COLOR="red"> {{ flashMessage }} </FONT>
        </div>
    {% endfor %}
</div>

[...]
```

S'adjunta un exemple de la consulta buscar client per veure l'aparició del missatge:

Buscar client:

Introdueix els cognoms del client a buscar:

CognomsPuj

Buscar client:

Introdueix els cognoms del client a buscar:

CognomsPujol

AVIS: El client NO existeix a la base de dades.

Un cop s'ha introduït el *cognom* i s'ha seleccionat el botó enviar consulta, apareix el missatge.

En molts llocs de l'aplicació podem trobar l'ús d'aquestes funcionalitats, com per exemple, totes les opcions de la pestanya Base de Dades.

9.4. Assignar un vehicle a un client

Aquest mètode anomenat *assignar_clientAction* del Controlador de l'entitat Vehicle, mostra com s'assigna a un vehicle un client. Per fer-ho seguirem els següents passos:

Obtenim el *Request* que contindrà les dades:

```
$request = $this->getRequest();
```

Creem l'objecte Vehicle i Client que s'utilitzarà.

```
$vehicle = new Vehicle();
$client = new Client();
```

Es defineix el formulari que es cridarà a la plantilla.

```
$form = $this->createFormBuilder()
->add('matricula', 'text')
->add('dni', 'text')
->getForm();
```

Es passa el *Request* al mètode *bindRequest()* de l'objecte del formulari el qual obté les dades del formulari i els carrega a l'objecte.

```
if($request->getMethod() == 'POST'){
    $form->bindRequest($request);
}
```

Si les dades del formulari són vàlides, ja es poden processar les dades. En aquest cas, amb la *matrícula* i el *dni* entrats, es busca un objecte Vehicle i un Client respectivament a la base de dades que continguin aquestes dades. Si els objectes no existeixen, apareixerà un missatge d'avís que ho indicarà.

```
if($form->isValid()){

    $em = $this->getDoctrine()->getManager();
    $matricula=$form['matricula']->getData();
    $vehicle=$em->getRepository('TallerBundle:Vehicle')->
        findOneByMatricula($matricula);

    if (!$vehicle) {
        $this->get('session')->getFlashBag()->add('notice',
            'AVIS: El vehicle NO existeix a la base de dades.');
```

S'assigna el client al vehicle.

```
$vehicle->setClient($client);
```

S'actualitza l'objecte vehicle a la base de dades.

```
$em = $this->getDoctrine()->getEntityManager();  
$em->persist($vehicle); // guarda l'objecte per ser insertat  
$em->flush();           // insertar a la bd
```

Es defineix el retorn un cop s'ha realitzat l'assignació.

```
return $this->redirect($this-> generateURL('taller_homepage'));
```

En aquest cas, quan s'ha assignat el client al vehicle, l'aplicació ens retorna a la pantalla principal, és a dir, l'escriptori.

Es defineix la plantilla a utilitzar per el formulari d'assignació :

```
return $this->render(  
    'TallerBundle:Vehicle:assignar_v_c.html.twig',  
    array('form' => $form->createView())  
);
```

9.5. Llistar treballadors

El mètode llistar mecànics, es considera una operació bàsica del Crud, però cal remarcar el funcionament de les plantilles i el full d'estils utilitzat. La Classe TreballadorType defineix el formulari a utilitzar, però és la plantilla l'encarregada de definir com es mostren les dades. A continuació es pot veure l'exemple de la plantilla *Treballador/index.html.twig* :

```
{% extends "TallerBundle:Default:escriptori.html.twig" %}  
{% block title %} Base de dades {% endblock %}  
{% block stylesheet %} {{ parent() }} {% endblock %}
```


Totes les instruccions dins l'etiqueta *block body* corresponen al codi de la plantilla en si. En aquest cas, es defineix una taula amb HTML on les capçaleres seran les dades dels treballadors : *id, dni, nom, cognoms* i les accions que es poden fer sobre cada treballador, al tenir el rol d'Administrador, es podran mostrar i editar.

Symfony inclou el llenguatge de plantilles Twig que simplifica molt l'escriptura i és més potent que el PHP. La sintaxi es basa en dues etiquetes especials : `{{ ... }}` i `{% ... %}`. La primera serveix per mostrar el contingut d'una variable i la segona per definir la lògica de la plantilla. Al codi es pot veure un exemple de bucle *for* on sense definir les *entities*, la plantilla obté els objectes Treballador que li envia el mètode `listar(indexAction())` del controlador.

La vista d'aquesta plantilla és:

Collsaplana Motor, SL
Benvingut usuari: Anna Administrador

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Llistat mecànics

Id	Dni	Nom	Cognoms	Accions
1	40361517J	Anna	Vilar Ribas	Mostra Editar
2	40365896d	Pere	Vilar Vivolas	Mostra Editar
3	40404040d	Adria	Garcia Arco	Mostra Editar
4	48596580b	Albert	Vilar Ribas	Mostra Editar

Torna a l'escriptori Imprimeix

Si s'executa la mateixa plantilla amb un usuari amb rol Gerent o Mecànic, s'obté la següent vista:

Collsapiana Motor, SL
 Benvingut usuari: Albert Mecànic

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Llistat treballadors

Id	Dni	Nom	Cognoms
1	40361517J	Anna	Vilar Ribas
2	40365896d	Pere	Vilar Vivolas
3	40404040d	Adria	Garcia Arco
4	48596580b	Albert	Vilar Ribas

No apareixen accions a realitzar

Torna a l'escriptori Imprimeix

A l'imatge es pot apreciar que s'ha definit un estil per diferenciar les línies de la taula amb els treballadors de l'empresa. L'estil utilitzat es defineix al full d'estil *C:\xampp\htdocs\Symfony\web\style.css*:

```
[...]
.datagrid table { border-collapse: collapse; text-align: left; width:
100%; }
.datagrid {font: normal 12px/150% Arial, Helvetica, sans-serif;
background: #fff; overflow: hidden; }
.datagrid table td, .datagrid table th { padding: 5px 10px; }
.datagrid table thead th {background:-webkit-gradient( linear, left
top, left bottom, color-stop(0.05, #8C8C8C), color-stop(1, #7D7D7D)
);background:-moz-linear-gradient( center top, #8C8C8C 5%, #7D7D7D
100%);filter:progid:DXImageTransform.Microsoft.gradient(startColorstr=
'#8C8C8C', endColorstr='#7D7D7D');background-color:#8C8C8C;
color:#FFFFFF; font-size: 15px; font-weight: bold; border-left: 1px
solid #A3A3A3; }
.datagrid table thead th:first-child { border: none; }
.datagrid table tbody td { color: #7D7D7D; border-left: 1px solid
#DBDBDB;font-size: 12px;border-bottom: 1px solid #E1EEF4;font-weight:
normal; }
.datagrid table tbody tr:nth-child(odd) { background: #EBEBEB; color:
#7D7D7D; }
.datagrid table tbody td:first-child { border-left: none; }
.datagrid table tbody tr:last-child td { border-bottom: none; }
[...]
```


A la plantilla *index.html.twig* es crida mitjançant l'instrucció :

```
<div class="datagrid"> <table> ... </table>
```

I a la plantilla pare de l'aplicació, definim l'utilització del fitxer *style.css* d'aquesta manera:

```
{% block stylesheet %}  
<link rel="stylesheet" type="text/css"  
      href="{{ asset('style.css') }}" media="screen" />  
{% endblock %}
```

9.6. Marcar factura com a pagada

El mètode per marcar factura com a pagada del *FacturaController*, l'hereten també els controladors fills *Factura* immediata i *Factura* única mensual.

A partir d'un número de factura, és a dir, l'*id*, es busca la factura corresponent a la base de dades i es marca com a pagada. Per fer-ho, s'utilitza el següent mètode :

```
public function marcar_pagadaAction(Request $request){
```

Cal destacar el mètode *findBy()* que busca a la base de dades, a la taula *Factura*, l'objecte factura que té per identificador l'entrat per teclat.

```
$factura=$em->getRepository('TallerBundle:Factura')->  
  findBy($form['id']->getData());
```

El framework *Symfony* ofereix diferents mètodes de cerca. L'utilitzat en aquest mètode obté els registres trobats poguent passar com a paràmetre els valors que es

posarien dins la clàusula *WHERE* de la consulta *Sql* i retorna un *array*. La comanda clau d'aquest mètode que és:

```
$factura->getPagat(true);
```

Després d'obtenir la factura, es posa l'atribut *pagat* a cert.

9.7. Imprimir llistats

L'opció d'imprimir llistats apareix a l'aplicació en varies ocasions, des del fet d'imprimir un albarà fins als llistats obtinguts de les consultes.

Per imprimir en format Pdf, s'utilitza el PdfBundle que s'ha instal·lat al Symfony. Aquesta llibreria és molt senzilla d'utilitzar. Un cop s'ha configurat el sistema, només cal definir la plantilla i cridar-la en funció del format escollit.

Com a exemple es mostra el mètode *listarTaxes* :

El mètode de *listarTaxes* de *TaxaController.php* s'anomena *indexAction* :

```
public function indexAction(){
    $em = $this->getDoctrine()->getManager();
    $entities = $em->getRepository(
        'TallerBundle:Taxa')->findAll();

    return $this->render(
        'TallerBundle:Taxa:index.html.twig', array(
            'entities' => $entities,
        ));
}
```

Aquest, mostra per pantalla el llistat corresponent i si l'usuari escull l'opció d'imprimir, s'executa el mètode :

```
[...]
use Ps\PdfBundle\Annotation\Pdf;
[...]
```

/**
 * @Pdf()

```

*/
public function imprimir_llistat_taxes_pdfAction(){
    $em = $this->getDoctrine()->getManager();

    $entities = $em->getRepository(
        'TallerBundle:Taxa')->findAll();

    $format = $this->get('request')->get('_format');

    return $this->render(sprintf(
        'TallerBundle:Taxa:index.%s.twig', $format
        array('entities' => $entities)
    ));
}

```

Si l'usuari tria l'opció d'imprimir per pantalla, es mostrarà la plantilla *index.html.twig*, i si tria l'opció d'imprimir amb format Pdf, la plantilla *index.pdf.twig*.

Existeixen poques diferències entre una plantilla i l'altre, se'n mostra un fragment :

Plantilla *index.html.twig*:

```

{% extends "TallerBundle:Default:escriptori.html.twig" %}

{% block title %}
    Base de dades
{% endblock %}

{% block stylesheet %}
    {{ parent() }}
{% endblock %}

{% block body %}
<div id="content">
    <br><br>
    <h1>Llistat taxes</h1>
    <br>
    <div class="datagrid"><table>
    <thead>
    <tr>
        <th>Id</th>
        <th>Descripcio</th>
        <th>Preu_unitat</th>
        <th>Actions</th>
    </tr>
    </thead>
    <tbody>

    {% for entity in entities %}
        <tr>
            <td><a href="{{ path('taxes_show',
                { 'id': entity.id }) }}">{{ entity.id }}</a></td>
            <td>{{ entity.descripcio }}</td>
            <td>{{ entity.preuunitat }}</td>
            <td>

```

```
[...]  
{% endblock %}
```

A la plantilla *index.pdf.twig*, no seran necessaris ni els links a modificar les taxes, ni la ruta de dependència amb la pàgina principal de l'aplicació. La majoria d'etiquetes utilitzades al llenguatge html també seran vàlides pel format pdf.

Plantilla *index.pdf.twig*:

```
<pdf>  
<dynamic-page>  
  
<div id="content">  
  <br><br>  
  <h1>Llistat taxes</h1>  
  <br>  
  <div class="datagrid"><table>  
    <thead>  
      <tr>  
        <th>Id</th>  
        <th>Descripcio</th>  
        <th>Preu_unitat</th>  
      </tr>  
    </thead>  
    <tbody>  
  
    {% for entity in entities %}  
      <tr>  
        <td>{{ entity.id }}</td>  
        <td>{{ entity.descripcio }}</td>  
        <td>{{ entity.preuunitat }}</td>  
      </tr>  
    [...]  
  </tbody>  
</div>  
</dynamic-page>  
</pdf>
```

Obtindrem el resultat :

Llistat taxes

Id	Descripcio	Preu_unitat
1	Reciclatge d'oli R.D. 679/2006	0.054
2	ECOVALOR SIGNUS (R.D. 1619/2005), A	0.95
3	ECOVALOR SIGNUS (R.D. 1619/2005), B	1.58
4	ECOVALOR SIGNUS (R.D. 1619/2005), C	2.75
5	ECOVALOR SIGNUS (R.D. 1619/2005), D	13.25

9.8. Relacionar 2 entitats mitjançant el formulari d'alta

En diferents pantalles d'alta de l'aplicació, cal relacionar una entitat amb una altra, com per exemple assignar vehicle a un client o assignar taxa a un recanvi. El mètode utilitzat per assignar un vehicle a un client s'ha explicat a l'apartat 9.4. En aquest cas, l'assignació es realitza mitjançant codi: l'usuari introdueix la matrícula i el dni del client, i es crida el mètode corresponent. L'opció que s'explica a continuació és fer-ho a partir d'un desplegable mostrat per pantalla.

Quan l'usuari dona d'alta un recanvi nou, li apareix la següent pantalla :



The screenshot shows a web form titled "Alta recanvi vehicle". It contains three input fields: "Descripció", "Preu unitari", and "Taxa". The "Taxa" field is a dropdown menu with a list of options. The first option is "Tria una opció, si el recanvi te taxa", which is currently selected. The other options are "ECOVALOR SIGNUS (R.D. 1619/2005). A", "ECOVALOR SIGNUS (R.D. 1619/2005). B", "ECOVALOR SIGNUS (R.D. 1619/2005). C", "ECOVALOR SIGNUS (R.D. 1619/2005). D", and "Reciclatge d'oli R.D. 679/2006". There are two buttons: "Crea" and "Torna a l'escriptori".

Tal com es pot veure a l'imatge, l'usuari introduirà la *descripció* i el *preu unitari* del recanvi, i, al tenir una relació 1 a 1 amb l'entitat Taxa, si el recanvi té una *taxa* assignada, n'escollirà una del llistat, en cas contrari, deixarà l'opció per defecte : *Tria una opció, si el recanvi té taxa*.

Per generar el formulari d'alta del recanvi, s'utilitza el mètode *buildForm* de la classe *RecanviType.php*:

```
public function buildForm(FormBuilderInterface $builder,
    array $options){

    $builder
        ->add('descripcio')
        ->add('preu_unitari')
        ->add('taxa', 'entity', array(
            'class' => 'TallerBundle:Taxa',
            'query_builder' => function(EntityRepository $er) {
                return $er->createQueryBuilder('t')
                    ->orderBy('t.descripcio', 'ASC');
            },
            'empty_value' => 'Tria una opció, si el recanvi te taxa',
```

```

        'required' => false,
        'property' => 'descripcio',
    ));
}

```

La funció *buildForm* defineix els paràmetres del formulari i crea una consulta DQL on *EntityRepository* busca totes les entitats *Taxa* a la base de dades i les ordena alfabèticament per *descripció*. Es defineix el valor per defecte quan no hi ha cap taxa seleccionada amb l'opció *empty_value* i amb l'opció *required* es descriu que un recanvi pot tenir, o no, una taxa assignada.

Aquest mètode és cridat per la classe *RecanviController.php*:

```

public function createAction(Request $request){
    $entity = new Recanvi();
    $form = $this->createForm(new RecanviType(), $entity);
    $form->bind($request);
    $em = $this->getDoctrine()->getManager();
    $recanvi=($em->getRepository('TallerBundle:Recanvi')
        ->findOneByDescripcio($form['descripcio']->getData()));

    if ($form->isValid() && !$recanvi) {
        $em->persist($entity);
        $em->flush();
        return $this->redirect(
            $this->generateUrl('taller_homepage'));
    }

    if ($recanvi) {
        // mostrar missatge flash
        [...]
    }

    return $this->render('TallerBundle:Recanvi:new.html.twig',
        array('entity' => $entity,
            'form' => $form->createView(),
        ));
}

```

Quan s'executa el mètode *crearAction()*, es crea un formulari nou del tipus *RecanviType()*. El contingut d'aquest formulari també es podria especificar al controller, però d'aquesta manera el formulari pot ser reutilitzat.

Si al crear un recanvi nou no existeix la taxa corresponent, s'haurà de donar d'alta manualment a la pestanya Base de dades/Taxes/alta.

9.9. Afegir recanvis, ma d'obra o treballs externs a una ordre de reparació

Al taller, a mesura que es va treballant amb un vehicle, el mecànic pot introduir els recanvis o feines realitzats. Per fer-ho, l'usuari clica sobre l'ícona corresponent a la Recepció del vehicle, i un cop s'ha afegit un recanvi o ma d'obra, es convertirà en ordre de reparació. Ja que el procediment és el mateix per les 3 opcions, s'explicarà afegir ma d'obra a una ordre de reparació :

Quan l'usuari decideix afegir ma d'obra a l'O.R., s'executa el mètode *buscar_per_codiMOAction()* de *MaObraController.php*.

L'*EntityManager* busca la *ma d'obra* corresponent a l'*id* entrat al formulari :

```
public function buscar_per_codimoAction(Request $request, $id){  
  
    $em = $this->getDoctrine()->getManager();  
    $entity = $em->getRepository(  
        'TallerBundle:OrdreReparacio')->find($id);  
  
    if (!$entity) {  
        throw $this->createNotFoundException(  
            'Unable to find OrdreReparacio entity.');    }  
  
    $form = $this->createFormBuilder()  
        ->add('id', 'text')  
        ->getForm();
```

Si l'*id* existeix, es procedeix a afegir l'objecte a l'ordre de reparació. Per fer-ho es segueix la ruta *afegir_mo_or* definida al fitxer *routing.yml* que mostrarà a quina classe està definit.

```
$r = new MaObra();  
  
if($request->getMethod() == 'POST'){  
  
    $form->bind($request);  
    if($form->isValid()){  
  
        $r=$em->getRepository('TallerBundle:MaObra')  
            ->findOneById($form['id']->getData());  
        if($r){  
            return $this->redirect($this  
                ->generateURL('afegir_mo_or', array(  
                    'idr' => $r->getId(),  
                    'idor' => $entity->getId()  
                )));  
        }  
    }  
}
```

Fitxer *routing.yml*:

```
afegir_mo_or:
  pattern:  /{idor}/{idr}/afegir_mo_or
  defaults: { _controller:TallerBundle:OrdreReparacio:afegir_mo_or }
```

Si no correspon a cap *ma d'obra* existent a la base de dades, es crea un objecte nou Ma d'Obra i s'afegeix a la base de dades :

```
        } else {
            return $this->redirect($this
                ->generateURL('crear_mo_or', array(
                    'id'      => $entity->getId(),
                    'form' => $form->createView(),
                )));
        }
    }
}
```

Fitxer *routing.yml*:

```
crear_mo_or:
  pattern:  /{id}/crear_mo_or
  defaults: { _controller: TallerBundle:MaObra:crear_mo_or }
```

Per acabar, trobem definida la plantilla des d'on es crida el mètode explicat en aquest apartat. Cal destacar que a la plantilla es passa per paràmetre l'entitat *ordre de reparació* i l'objecte *recanvi* trobat.

```
return $this->render(
    'TallerBundle:MaObra:buscar_per_codi.html.twig', array(
        'entity' => $entity,
        'recanvi' => $r,
        'form' => $form->createView(),
    ));
}
```

A continuació s'explicaran els 2 mètodes necessaris per completar el cicle, és a dir, tornar a l'ordre de reparació per continuar afegint material o feina.

9.9.1. Afegir ma d'obra a l'ordre de reparació

Si la ma d'obra entrada pel treballador existeix a la base de dades, només cal indicar la quantitat utilitzada i el descompte aplicat. Per fer-ho, s'utilitza el mètode *afegir_mo_or* de la classe *OrdreReparacioController.php*:

```
public function afegir_mo_orAction(Request $request, $idor, $idr){

    $em = $this->getDoctrine()->getManager();
    $entity = $em->getRepository('TallerBundle:OrdreReparacio')
        ->find($idor);
    $r = $em->getRepository('TallerBundle:MaObra')->find($idr);
    $form = $this->createFormBuilder()
        ->add('quantitat', 'text')
        ->add('descompte', 'text')
        ->getForm();
```

Després de buscar a la base de dades l'ordre de reparació i el recanvi corresponents als *id* passats per paràmetre, es crea un formulari per indicar *quantitat* i *descompte* del recanvi.

Si les dades introduïdes al formulari són vàlides, es crea una *línia d'ordre de reparació*, se li assignen els valors del formulari i es calcula el *preu total* de la línia :

```
if($request->getMethod() == 'POST'){
    $form->bind($request);

    if ($form->isValid()) {

        $liniaor = new LiniaOR();
        $liniaor->setCodi($idr);
        $liniaor->setQuantitat($form['quantitat']->getData());
        $liniaor->setDescompte($form['descompte']->getData());

        // calcular preu total de la línia
        $preu = $form['quantitat']
            ->getData()* $em->getRepository('TallerBundle:MaObra')
            ->find($idr)->getPreuUnitari();
        $liniaor->setPreu($preu - ($preu/100 * $liniaor
            ->getDescompte()));
```

Al tractar-se del mètode d'afegir ma d'obra, assignarem l'*id* de l'operació a la línia d'o.r., els atributs *recanvi_id* i *treballExtern_id* seran nuls.

Pels mètodes *afegir_textern_or()* seran nuls *recanvi_id* i *maobra_id*, i per *afegir_recanvi_or()*, *treballExtern_id* i *maobra_id* respectivament.

```

$liniaor->setRecanvi(null);
$liniaor->setMaobra(
    $em->getRepository('TallerBundle:MaObra')
    ->find($idr));
$liniaor->setTreballeextern(null);
$liniaor->setOrdreReparacio($entity);

```

Un cop afegida la *liniaor* a la taula de línies de l'entitat *ordre de reparació*, es desen els canvis a la base de dades i es redirecciona la sortida a la plantilla *edit.html.twig*. Aquesta plantilla mostrarà l'ordre de reparació actualitzada amb l'operació de ma d'obra afegida i es passaran per paràmetre l'entitat, la *línia d'ordre de reparació* per poder visualitzar les dades per pantalla i permetre a l'usuari seguir completant l'O.R.

```

$entity->addLiniaOR($liniaor);

$em->persist($liniaor);
$em->persist($entity);
$em->flush();

return $this->render(
    'TallerBundle:OrdreReparacio:edit.html.twig',
    array(
        'entity' => $entity,
        'lors' => $entity->getLiniaor(),
        'form' => $form->createView(),
    ));
}

return $this->render(
    'TallerBundle:OrdreReparacio:seguir_edit_mo.html.twig',
    array('entity' => $entity,
        'recanvi' => $r,
        'lors' => $entity->getLiniaor(),
        'form' => $form->createView(),
    ));
}

```

9.9.2. Crear ma d'obra a l'ordre de reparació

Quan l'identificador de l'operació de ma d'obra introduïda al mètode *buscar_per_codiMOAction()* no correspon a cap objecte de la base de dades, es

redireccionarà la sortida de la funció `cap` al mètode `crear_mo_orAction()` de la classe `MaObraController.php`:

```
public function crear_mo_orAction(Request $request, $id){

    $em = $this->getDoctrine()->getManager();
    $entity = $em->getRepository('TallerBundle:OrdreReparacio')
        ->find($id);

    if (!$entity) {
        throw $this->createNotFoundException(
            'Unable to find OrdreReparacio entity.');
    }

    $request = $this->getRequest();

    $r = new MaObra();

    $form = $this->createFormBuilder()
        ->add('descripcio', 'text')
        ->add('preu_unitari', 'text')
        ->getForm();

    if($request->getMethod() == 'POST'){
        $form->bindRequest($request);

        if($form->isValid()){
            // emplenar la m.o. amb les dades del formulari
            $em = $this->getDoctrine()->getManager();

            $r->setDescripcio($form['descripcio']->getData());
            $r->setPreuUnitari($form['preu_unitari']->getData());

            $em->persist($r);
            $em->flush();
            return $this->redirect($this->generateURL('afegir_mo_or',
                array('idr' => $r->getId(),
                    'idor' => $entity->getId()
                )));
        }
    }

    return $this->render('TallerBundle:MaObra:crear_mo_or.html.twig',
        array('entity' => $entity,
            'form' => $form->createView()
        ));
}
```

Aquest mètode és un altre exemple de *create* de tipus CRUD. Cal destacar-ne la redirecció al mètode `afegir_mo_or()` de l'apartat anterior.

Un cop afegida l'operació, fins que no es seleccioni l'opció Tancar O.R. de la plantilla `edit.html.twig`, es podran anar afegint operacions o recanvis a l'ordre de reparació.

9.10. Tancar ordre de reparació

Quan una ordre de reparació conté tots els recanvis i operacions utilitzats durant la reparació del vehicle, es procedeix al tancament de l'ordre de reparació i generació de la factura o albarà, depenent de les condicions de pagament del client.

El fet que defineix el moment de tancar l'ordre de reparació és la selecció per part de l'usuari del botó Tancar O.R. de la plantilla *edit.html.twig*. L'ubicació del mètode necessari pel tancament es defineix a l'etiqueta *href* que conté el botó :

```
<a href="{{ path('tancar_or', {'id': entity.id}) }}">
    <button type="submit"> Tanca l'O.R.</button>
</a>
```

Seguint el fitxer *routing.yml* s'observa que el mètode *tancar_or* pertany a la classe *OrdreReparació* i conté el paràmetre *id* de l'O.R.

```
tancar_or:
    pattern:  /{id}/tancar_or
    defaults: { _controller: TallerBundle:OrdreReparacio:tancar_or }
```

El mètode *tancar_orAction* es troba al fitxer *OrdreReparacióController.php* :

```
public function tancar_orAction(Request $request, $id){

    $em = $this->getDoctrine()->getManager();
    $entity = $em->getRepository('TallerBundle:OrdreReparacio')
        ->find($id);
    if (!$entity) {
        throw $this->createNotFoundException(
            'Unable to find OrdreReparacio entity.');
    }
}
```

Abans de crear l'objecte Factura, cal saber de quin tipus ha de ser. Per fer-ho, un cop obtinguda l'ordre de reparació amb l'*id* passat per paràmetre, s'accedeix a la condició de pagament del client de l'ordre de reparació amb el codi :

```
$cp = $entity->getRecepcio()->getClient()->getCondicioPagament() ;
```

Si la condició de pagament del client és *mensual*, es crearà un *Albarà*. Altrament una entitat de tipus *FacturalInmediata*.

```
if($scp->getDescripcio()=='Mensual'){
    // crear albara
    $factura = new Albara();
}else{
    $factura = new FInmediata();
}
```

En aquest mètode s'ha decidit crear el formulari que es mostrarà per pantalla al mateix Controlador, ja que el mètode és específic d'aquesta classe i no es reutilitzarà en cap altre mètode o funció.

Es defineixen els quilòmetres que porta el vehicle al finalitzar la reparació, el treballador que ha realitzat la reparació i la data de tancament de l'O.R. Si les dades introduïdes al formulari són vàlides, s'assignen les dades obtingudes del formulari a l'ordre de reparació.

```
$form = $this->createFormBuilder()
->add('km_sortida', 'text')
->add('treballador', 'entity', array(
    'class' => 'TallerBundle:Treballador',
    'query_builder' => function(EntityRepository $er) {
        return $er->createQueryBuilder('t')
            ->orderBy('t.id', 'ASC');
    },
    'empty_value' => 'Tria el codi de treballador',
    'property' => 'id',
))
->add('data', 'date')
->getForm();

if($request->getMethod() == 'POST'){
    $form->bind($request);

    if ($form->isValid()) {
        $km_sortida= $form['km_sortida']->getData();
        $treballador=$form['treballador']->getData();
        $data=$form[data]->getData();

        $entity->setKmSortida($km_sortida);
        $entity->setTreballador($treballador);
        $entity->setData($data);
    }
}
```

Per tancar l'O.R. només cal assignar a l'atribut *tancat* el valor *true*. Un cop tancada, es calcula el *\$preu_base* de la factura o albarà. Per fer-ho es sumaran tots els preus totals de cada línia d'ordre de reparació i, si el recanvi té una taxa assignada, es calcularà al

moment el preu total d'aquesta multiplicant la *\$quantitat* del recanvi de la línia d'o.r. pel *\$preu_unitat* de la taxa corresponent al recanvi.

S'assigna el resultat del càlcul obtingut a la variable *\$preu_base* de l'entitat Factura, la data del tancament i s'assigna a false l'atribut que determina si la factura ha estat pagada o no. Posteriorment s'haurà de marcar com a pagada.

```
// tancar or
$entity ->setTancat(true);

// calcular preu base total
$lors=$entity->getLiniaor();
$pb=0;
foreach($lors as $lo){
    $pb=$pb+$lo->getPreu();
    if($lo->getRecanvi()!=null){
        if($lo->getRecanvi()->getTaxa()!=null){
            $pb=$pb+$lo->getQuantitat()*
            ($lo->getRecanvi()->getTaxa()->
            getPreuUnitat());
        }
    }
}

$factura->setPreuBase($pb);
$factura->setTotalFactura($pb+($pb/100*
    $factura->getImpost()));
$factura->setData(new \DateTime("now"));
$factura->setPagada(false);
```

Depenent del tipus de l'atribut *\$factura*, s'afegirà la factura a l'ordre de reparació com a Factura Immediata o com a Albarà.

Un cop s'ha assignat valor a totes les variables, cal desar els canvis a la base de dades i definir la redirecció del mètode a la plantilla *imprimir.html.twig*, que mostrarà per pantalla el contingut de la Factura.

```
if($factura is Albara){
    $entity->setAlbara($factura);
}else{
    $entity->setFImmediata($factura);
}

$em->persist($entity);
$em->persist($factura);
$em->flush();

return $this->render(
    'TallerBundle:OrdreReparacio:imprimir.html.twig',
```

```

        array('entity'      => $entity,
              'recepccio'   => $entity->getRecepccio(),
              'lors'        => $entity->getLiniaor(),
              'factura'     => $factura,
              'form'        => $form->createView(),
        ));
    }
}
return $this->render('TallerBundle:OrdreReparacio:tancar.html.twig',
    array('entity'      => $entity,
          'lors'        => $entity->getLiniaor(),
          'factura'     => $factura,
          'form'        => $form->createView(),
    ));
}

```

En aquest punt, l'usuari podrà imprimir la factura o tornar a l'escriptori.

10. Implantació i resultats

En aquest apartat es mostraran els resultats obtinguts de l'aplicació a partir de diferents exemples. No es mostrarà la implantació de l'aplicació a l'empresa, ja que s'ha decidit instal·lar-la un cop presentat el projecte.

10.1. Implantació

L'implantació de l'aplicació a l'empresa es farà un cop finalitzat i entregat el projecte final de carrera, tal com s'ha comentat anteriorment, però s'explicarà breument l'idea de muntatge que es té després d'analitzar les necessitats del client i les instal·lacions de l'empresa.

10.1.1. Muntatge/instal·lació al taller

Al ser un local amb 2 nivells, on a baix es troba el taller i a dalt les oficines, s'ha cregut convenient instal·lar una pantalla tàctil des d'on els treballadors podran realitzar les recepcions a l'entrada d'un vehicle, i serà més pràctic a l'hora d'introduir les tasques realitzades i els materials utilitzats. D'aquesta manera s'eliminaran els fulls de treball en paper que s'utilitzen actualment. En podem veure un exemple a l'apartat 5.1. Marc de treball.

Un cop el mecànics hagin finalitzat la reparació, l'administrador o gerent podrà veure que l'ordre de reparació s'ha tancat, i podrà procedir a la facturació d'aquesta.

El material necessari per aquesta instal·lació és:

- cable de xarxa suficientment llarg per anar de les oficines al taller
- ordinador tot en un, s'ha escollit el model Asus AIO P1801-T Nvidia Tegra 3/2GB/32GB/18.4" Tàctil



Característiques:

- Processador NVIDIA® Tegra® 3 Quad-core CPU
- Memòria RAM 2GB DDR3 SODIMM a 1600MHz
- Disc dur 32GB eMMC Flash
- Display 18.4"(46.7cm), 16:9, Panoràmica, Full HD 1920x1080, retroiluminació LED, Angle de visió de 178°, IPS
- Connectivitat (802.11 a/b/g/n i Bluetooth V3.0 High Speed ED)
- Sensor G-Sensor
- Dimensions (Amplada x Profunditat x Altura) 466 x 18 x 294 mm (WxDxH)
- Pes 2.4 kg

Finalment ens hem decantat cap a un ordinador "tot en un" ja que amb una pantalla tàctil l'aplicació no seria multi usuari.

Fent aquesta inversió, l'empresa tindrà nombrosos avantatges:

- rapidesa a l'hora de recepcionar un vehicle
- sense necessitat de pujar a l'oficina podran consultar:
 - o històrics dels vehicles
 - o bases de dades externes com el programa Autodata, peritatges, anuals de taller, etc.
- eliminació dels blocs de fulls de treball, ja sigui en el cas de realitzar la recepció com en el cas d'anar completant els treballs realitzats

El principal inconvenient és la llargada del cable de xarxa des de l'oficina fins a la zona de taller.

10.1.2. Migració de dades

Per a poder implantar l'aplicació a l'empresa és necessari fer la migració de dades del sistema utilitzat fins el moment.

L'aplicació actual utilitza una base de dades de Microsoft Access i després d'estudiar-la i veure les diferències que hi ha amb la dissenyada en aquest projecte, s'ha decidit migrar les taules principals com són les de les factures, clients, treballadors, recanvis i ma d'obra. Per fer la migració serà necessari passar les taules a un fitxer de text, que serà llegit per un programa implementat en Java i executarà les insercions a la base de dades nova.

Per veure el procés de migració de dades pas a pas, consultar l'Annex 14.1. Migració de dades.

10.1.3. Conceptes per al desenvolupament de l'aplicació

Cal destacar que per a realitzar aquest projecte s'ha hagut d'adquirir uns coneixements per tal de seguir les normatives vigents que segueix l'empresa i conèixer el seu funcionament. Entre elles cal destacar la Llei de Protecció de Dades i les lleis de regulació de serveis, en aquest cas les específiques per a un taller mecànic.

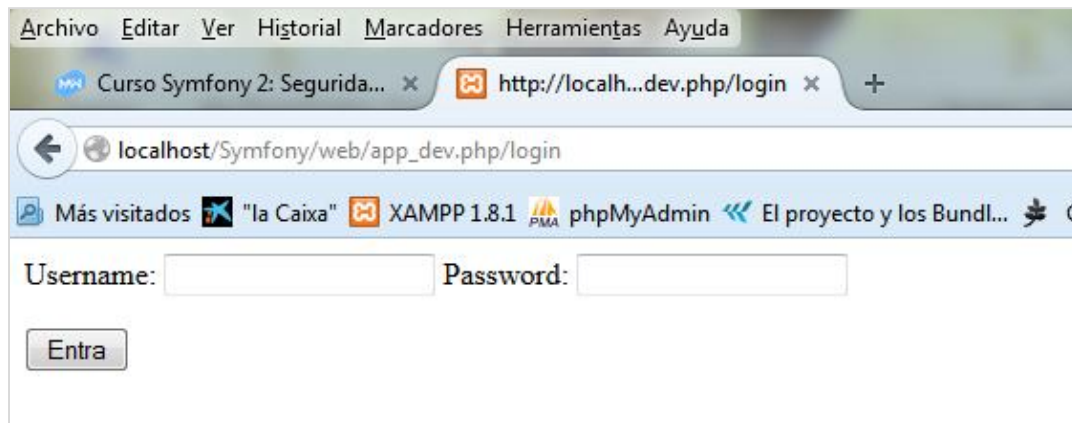
Tots aquests requeriments i conceptes previs es poden veure ampliat a l'apartat 5.2.2. Marc legal d'aquest mateix projecte.

10.2. Resultats

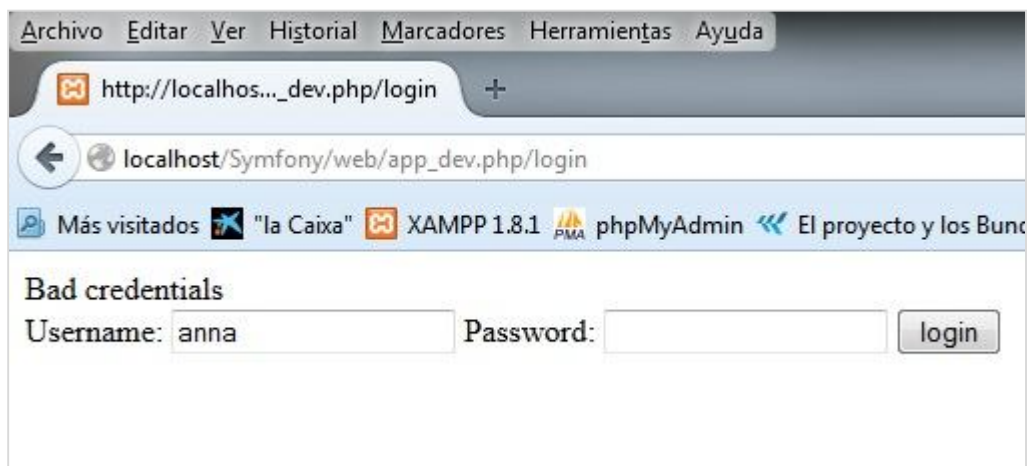
Per mostrar els resultats obtinguts s'aniran recorrent les diferents pestanyes de l'aplicació, però abans cal aturar-nos a la pantalla principal, l'escriptori.

10.2.1. Login/logout

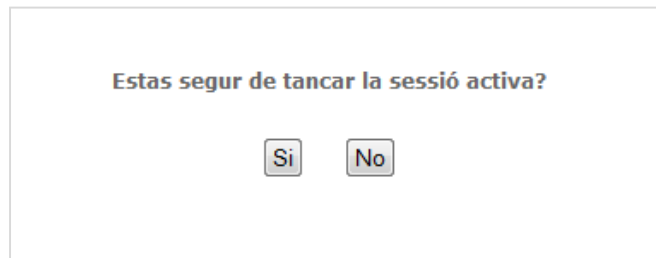
Perquè no pugui accedir qualssevol persona a l'aplicació, i més pensant en instal·lar una pantalla tàctil al taller, es va decidir bloquejar l'aplicació perquè només hi poguessin accedir els usuaris registrats. Per fer-ho, s'ha creat una finestra de login com la següent:



Tal com es pot veure a l'imatge, per a què un usuari pugui accedir-hi, haurà d'introduir l'usuari i password. L'aplicació comprovarà que siguin correctes, si és així, accedirà a l'escriptori. En cas contrari, apareix un missatge d'error:



Després d'utilitzar l'aplicació, quan l'usuari clica sobre el nom, apareix la següent pantalla per fer el logout:



Si selecciona "Si", es torna a la pantalla de login i si selecciona "No", es recupera la pantalla de l'escriptori de l'aplicació.

10.2.2. Escriptori

Quan l'usuari accedeix a la pantalla inicial de l'aplicació, pot veure com sota el nom de l'empresa, apareix un missatge de benvinguda seguit del nom de l'usuari identificat. La pantalla de l'escriptori és:

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Pressupostos pendents d'acceptar :

- Pressupost=1
Matricula=1000BHT
- Pressupost=2
Matricula=1234DHD

(Passats 15 dies desapareixeran de la llista)

Vehicles al taller :

- Recepció=1
Matricula=5896GTH
- Recepció=5
Matricula=2204CSN

Un dels requisits de l'aplicació era poder gestionar els vehicles del taller. Per fer-ho, des de la pantalla principal, l'escriptori, es pot veure clarament el llistat de pressupostos vigents (a l'esquerra de la pantalla), és a dir, els pressupostos pendents d'acceptar abans que arribi la seva data de caducitat, i també les ordres de reparació dels vehicles en reparació (a la dreta).

Perquè sigui més visual i pràctic pels operaris, s'ha decidit utilitzar icones amb diferents colors:

- Vermell: pels pressupostos pendents d'acceptar.
- Taronja: per les recepcions, vehicles esperant per ser reparats.
- Verd: ordres de reparació obertes, vehicles que s'estan reparant.

Per veure'n el funcionament, a continuació es pot veure un exemple del procés que segueix un vehicle al taller, des de la seva recepció a la facturació de la reparació realitzada.

Accedint a la pestanya Taller es crea una nova recepció:

Nova recepció

Data	2014 ▾ - Aug ▾ - 28 ▾
Renuncia pressupost	<input checked="" type="checkbox"/>
Descripció operacions	manteniment
Data prev lliurament	2014 ▾ - Aug ▾ - 28 ▾
Recollir peces	<input type="checkbox"/>
Reparació en garantia	<input type="checkbox"/>
Matrícula	2037DFH
Dni	40281750B

Si el client o el vehicle no existissin a la base de dades, l'aplicació redireccionaria l'usuari a la pantalla d'alta corresponent.

Selecció del botó "Crea", es torna a l'escriptori. Aquí es pot veure com ha aparegut una nova recepció, la del vehicle amb matrícula 2037-DFH.

Vehicles al taller :

 Recepció=1 Matrícula=5896GTH	 Recepció=5 Matrícula=2204CSM	 Recepció=8 Matrícula=2037DFH
--	--	--

Recepció nova

Quan és el torn del vehicle, l'operari procedeix a la reparació. Per obrir l'ordre de reparació, només cal seleccionar el vehicle de l'escriptori. Tal com es pot veure a

l'imatge que hi ha a continuació, s'acaba d'obrir l'ordre de reparació número 8 i és buida, és a dir, no conté cap línia d'ordre de reparació amb recanvis, ma d'obra o treballs externs.

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Ordre de reparació : 8

Afegeix recanvi Afegeix treballs externs Afegeix ma d'obra

Atenció: Ordre de reparació buida

Tanca l'O.R. Torna a l'escriptori

A mesura que va reparant el vehicle, l'usuari completa l'ordre de reparació. En aquest cas, s'hi afegirà un recanvi:

Ordre de reparació : 8 : Buscar recanvi:

Id

Cerca

Torna a l'escriptori

Si es coneix el *codi* del recanvi, al clicar "Cerca" ens mostrarà la següent pantalla per poder afegir la *quantitat* i el *descompte* a aplicar a aquesta línia d'o.r., en cas contrari, es crearà un recanvi nou.

Afegir recanvi amb *id* = 1:

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Omplir ordre de reparació: Oli Castrol 10w40 Recanvi amb id=1

Quantitat 4.5

Descompte

Fet

Afegir recanvi nou:

Ordre de reparació: 8 : Crear recanvi:

Descripció

Preu unitari

Fet

Si en algun moment l'operari intenta realitzar algun procés sense completar tots els camps obligatoris, l'aplicació dona un missatge d'avís:

Omplir ordre de reparació: Oli Castrol 10w40

Quantitat 4.5

Descompte

Fet

Rellene este campo.

Quan es selecciona el botó "Fet", el recanvi queda desat a la línia d'ordre de reparació i la línia d'o.r. a l'ordre de reparació número 8.

Ordre de reparació : 8

Recanvi afegit

RECANVIS :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
1	Oli Castrol 10w40	4.5	10.5	0	47.25

MA D'OBRA :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
------	------------	-----------	--------------	-----------	------------

TREBALLS EXTERNES :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
------	------------	-----------	--------------	-----------	------------

Un cop s'ha afegit algun material o operació a l'ordre de reparació, si s'accedeix a l'escriptori de l'aplicació, es pot veure com la recepció número 8 del vehicle amb matrícula 2037-DFH, ha passat a ser una ordre de reparació:

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Pressupostos pendents d'acceptar :

- Pressupost=1
Matrícula=1000BHT
- Pressupost=2
Matrícula=1234DHD

(Passats 15 dies desapareixeran de la llista)

Vehicles al taller :

Recepció=1
Matrícula=5896GTH

Recepció=5
Matrícula=2204CSN

OR=8
Matrícula=2037DFH

A l'imatge anterior es mostra com apareixen les diferents icones a l'escriptori juntament amb una petita informació del vehicle i del document associat a ell.

Després d'afegir una operació de ma d'obra, que es realitza exactament igual que afegir un recanvi, es mostra l'ordre de reparació:

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Ordre de reparació : 8

RECANVIS :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
1	Oli Castrol 10w40	4.5	10.5	0	47.25
10	Filtre oli	1	9	10	8.1

MA D'OBRA :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
1	Ma d'obra	1	35	0	35

TREBALLS EXTERNES :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
------	------------	-----------	--------------	-----------	------------

Quan l'operari ha finalitzat la reparació del vehicle, selecciona el botó "Tancar l'O.R.". Al fer-ho apareixerà una nova finestra amb un formulari per identificar el treballador que ha realitzat la reparació i els quilòmetres que té el vehicle al final d'aquesta.

També hi constaran els totals desglossats en recanvis, ma d'obra i treballs externs. Cal destacar que si el recanvi té alguna taxa associada, també apareixeran desglossades.

Factura :

Km sortida

Treballador

TOTALS :

Total Recanvis: 55.35 Euros

Total Taxes (recanvis): 0.243 Euros

Total Ma d'obra: 35 Euros

Total Treball Extern: 0 Euros

(preus sense IVA)

Al generar la factura, l'operari visualitzarà la factura per pantalla i tindrà l'opció d'imprimir-la en format pdf. Actualment, i fins que s'implementin les millores descrites a l'apartat 12, Treball futur; s'imprimeix sense logotip ni dades de l'empresa ja que s'utilitzen fulls impresos.

Collsaplana Motor, SL

Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Factura num.: 56

CLIENT: 16

DNI: 40281750B

Cognoms: Martí Pou Nom: Santi

Adreça: C/Nou,9

Codi postal: 17005 Població: Girona

Província: Girona

VEHICLE: 33

MATRICULA: 2037DFH

MARCA: Seat MODEL: Leon

16 Km.

RECANVIS :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
1	Oli Castrol 10w40	4.5	10.5	0	47.25
10	Filtre oli	1	9	10	8.1

Taxes recanvis :

Codi	Descripció	Quantitat	Preu unitari	Preu total
1	Reciclatge d'oli R.D. 679/2006	4.5	0.054	0.243

MA D'OBRA :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
1	Ma d'obra	1	35	0	35

TREBALLS EXTERNS :

Codi	Descripció	Quantitat	Preu unitari	Descompte	Preu total
------	------------	-----------	--------------	-----------	------------

TOTALS :

Preu Base	Iva	Total
90.59	IVA: 21 %	109.62

[Imprimir](#)

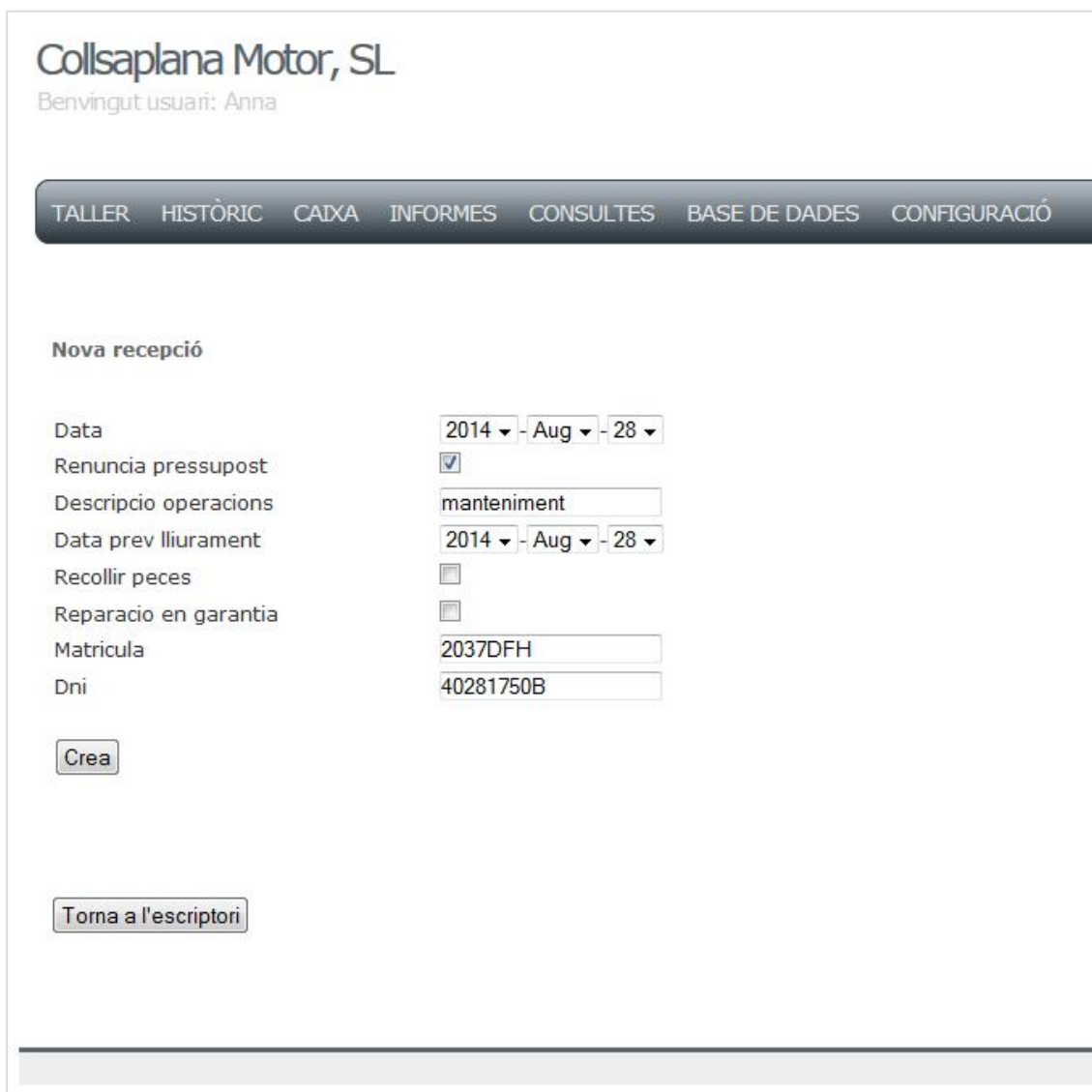
[Torna a l'escriptori](#)

Cal destacar que a cada pantalla de l'aplicació, l'usuari té l'opció de tornar a l'escriptori i, d'aquesta manera, cancel·lar el procés que estigui realitzant en aquest moment.

Per gestionar les recepcions de vehicles nous, s'accedirà a la pestanya Taller.

10.2.3. Taller

Escollint aquesta pestanya del menú principal, s'accedeix automàticament a fer una nova recepció, tal com mostra l'exemple:



The screenshot shows the 'Collsaplana Motor, SL' web application interface. At the top, the user is logged in as 'Anna'. A navigation menu includes 'TALLER', 'HISTÒRIC', 'CAIXA', 'INFORMES', 'CONSULTES', 'BASE DE DADES', and 'CONFIGURACIÓ'. The 'TALLER' tab is active, displaying the 'Nova recepció' form. The form contains the following fields and options:

Data	2014 ▾ - Aug ▾ - 28 ▾
Renuncia pressupost	<input checked="" type="checkbox"/>
Descripció operacions	manteniment
Data prev lliurament	2014 ▾ - Aug ▾ - 28 ▾
Recollir peces	<input type="checkbox"/>
Reparacio en garantia	<input type="checkbox"/>
Matricula	2037DFH
Dni	40281750B

Buttons: 'Crea' and 'Torna a l'escriptori'.

Al acabar qualssevol tasca o consulta, sempre es redirigeix l'usuari a la finestra principal per tal de poder tenir controlats els moviments dels vehicles al taller.

En aquest cas, a l'escriptori apareixerà una nova icona amb les dades d'aquesta recepció d'exemple:

The screenshot shows a web application interface. At the top, there is a dark navigation bar with the following menu items: HISTÒRIC, CAIXA, INFORMES, CONSULTES, BASE DE DADES, and CONFIGURACIÓ. Below the navigation bar, on the left side, there is a sidebar with the text 'ostos', 's', 'tar :', 'supost=1 =1000BHT', and 'supost=2 =1234DHD'. The main content area is titled 'Vehicles al taller :'. It displays three vehicle cards, each with an orange car icon, 'Recepció' number, and 'Matrícula' (license plate) number. The first card has 'Recepció=1' and 'Matrícula=5896GTH'. The second card has 'Recepció=5' and 'Matrícula=2204CSN'. The third card has 'Recepció=8' and 'Matrícula=2037DFH'. This third card is highlighted with a red rectangular border. Below the cards, the text 'Recepció nova' is displayed in red.

10.2.4. Històric

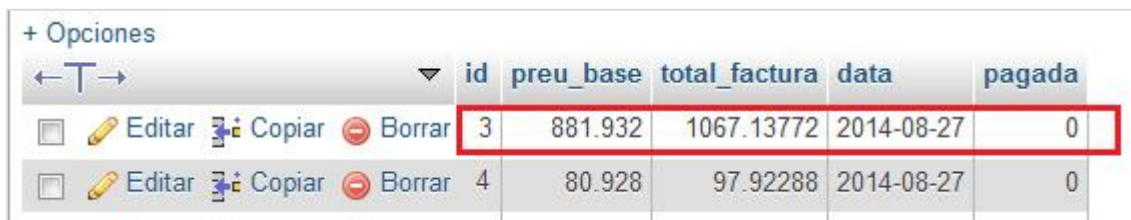
Un dels principals requisits de l'aplicació era poder accedir de manera fàcil a la relació de reparacions efectuades sobre un vehicle, a partir de la matrícula. Aquest requisit és molt important pels mecànics del taller per poder consultar les reparacions anteriors de cares a planificar-ne de noves o per aconsellar al client. Per fer-ho, accedirem a la pestanya Històric:

The screenshot shows the 'Històric' page of the web application. At the top, the company name 'Collsaplana Motor, SL' is displayed, followed by the user greeting 'Benvingut usuari: Anna'. Below this is a dark navigation bar with the following menu items: TALLER, HISTÒRIC, CAIXA, INFORMES, CONSULTES, BASE DE DADES, and CONFIGURACIÓ. The main content area contains two instructions: 'Donat un CLIENT, llistar els seus vehicles' and 'Donat un VEHICLE, llistar les seves reparacions'. At the bottom of the page, there is a button labeled 'Torna a l'escriptori'.

Tal com es pot veure a l'imatge, es podrà consultar la relació de vehicles d'un client i l'històric d'un vehicle per poder veure totes les reparacions efectuades a partir de la matrícula.

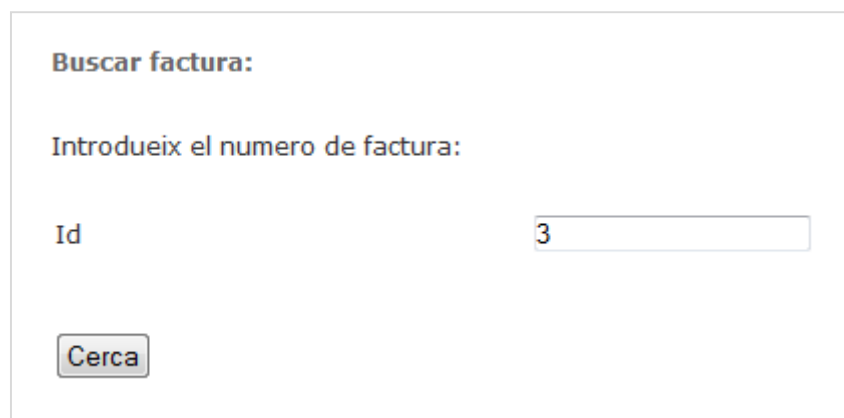
10.2.5. Caixa

Per l'administrador i el gerent de l'empresa era molt important poder controlar el cobrament de les factures. Per fer-ho, s'accedeix a la pestanya Caixa, i a partir del número de la factura, es marca com a pagada. Veiem-ne un exemple: A la base de dades es pot veure com la factura nº3 no està pagada:



+ Opciones		id	preu_base	total_factura	data	pagada
<input type="checkbox"/>	Editar	3	881.932	1067.13772	2014-08-27	0
<input type="checkbox"/>	Editar	4	80.928	97.92288	2014-08-27	0

Per procedir a marcar-la com a pagada, només cal indicar-ne el codi o identificador.



Buscar factura:

Introdueix el numero de factura:

Id

Ara, a la base de dades, apareix com a pagada:

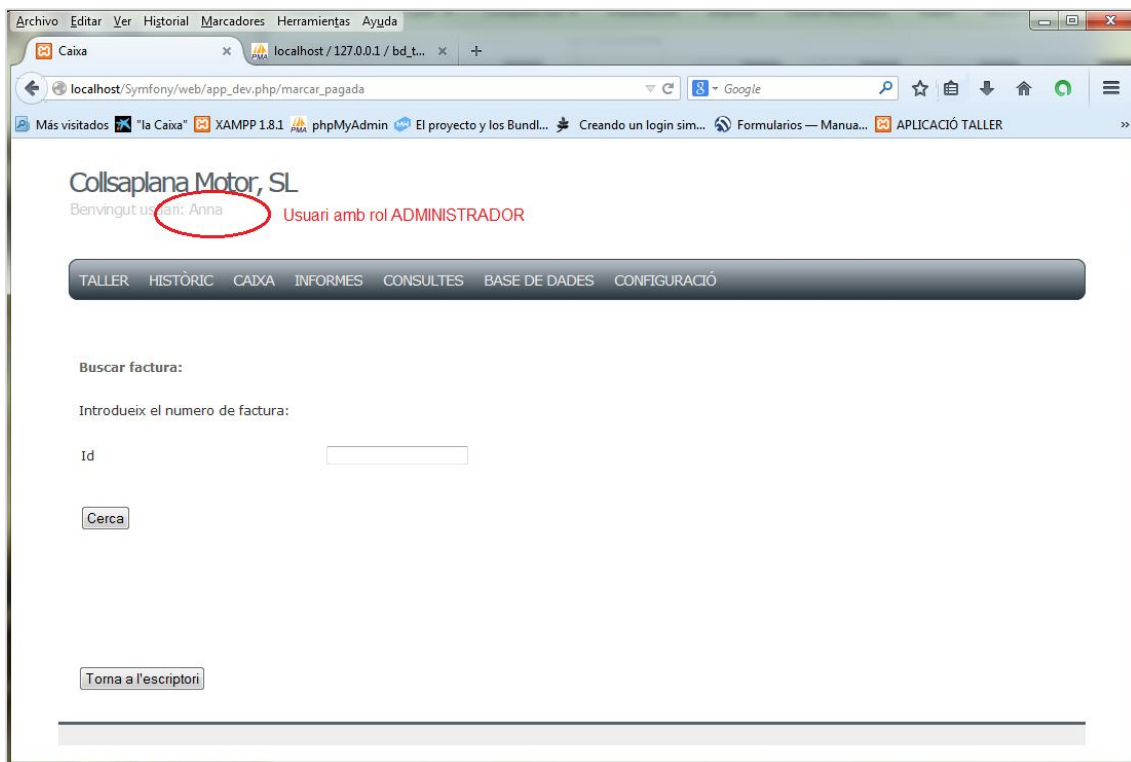


+ Opciones		id	preu_base	total_factura	data	pagada
<input type="checkbox"/>	Editar	3	881.932	1067.13772	2014-08-27	1
<input type="checkbox"/>	Editar	4	80.928	97.92288	2014-08-27	0

pagada = true

Els usuaris de l'aplicació amb rol mecànic no han de tenir accés a l'aplicació ja que els operaris de l'empresa no tenen contacte amb la caixa. Si ho fan, veuran un missatge informatiu conforme no poden realitzar cap acció.

Vista de la pestanya Caixa d'un usuari amb rol Administrador o Gerent:



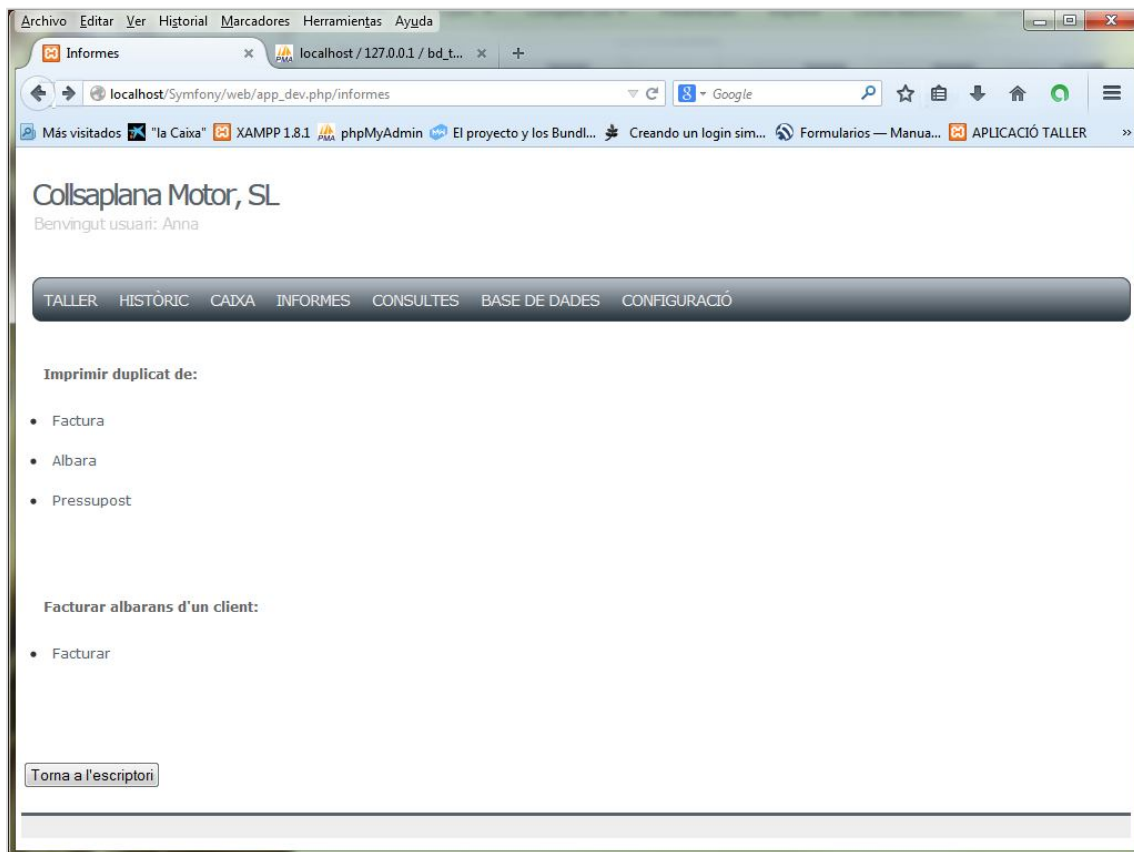
Vista de la pestanya Caixa d'un usuari amb rol Mecànic:



A l'apartat de consultes es pot veure la consulta de la relació de factures pendents de cobrament.

10.2.6. Informes

Ja que l'aplicació és per una empresa, és molt important poder imprimir duplicats dels documents generats, com són els albarans, els pressupostos i les factures.

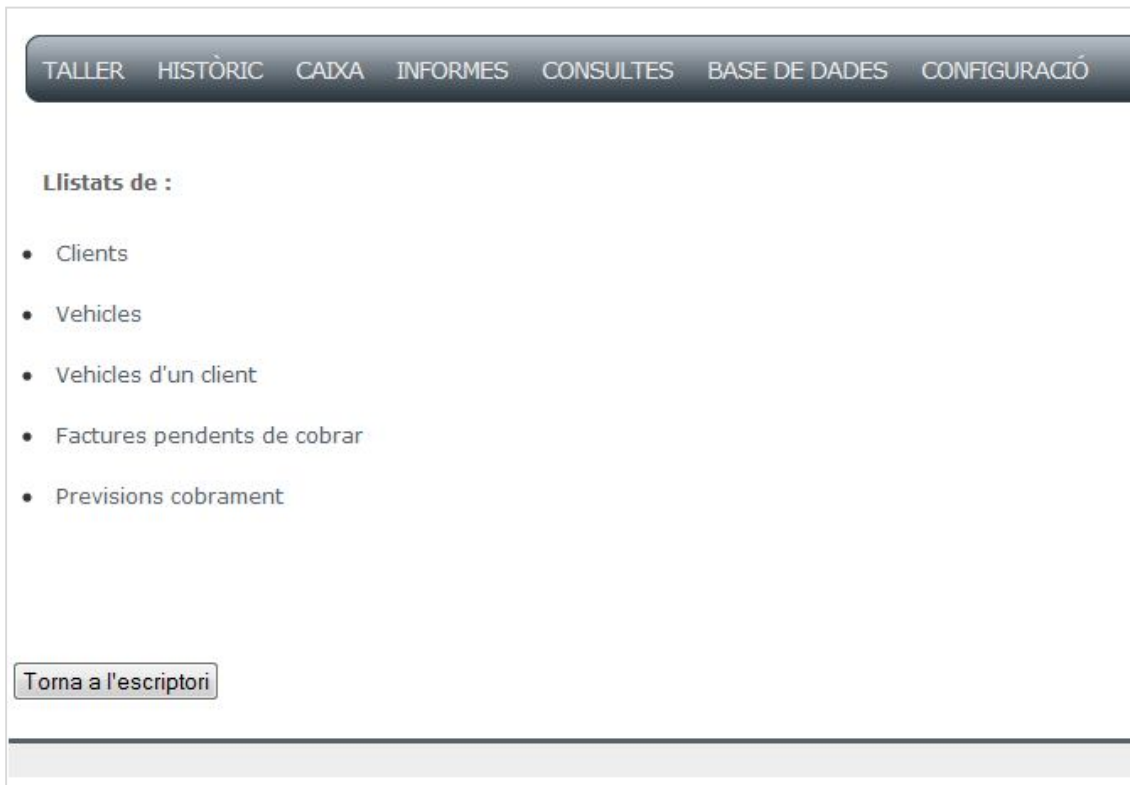


A partir del número del document, es podrà accedir a l'informació sol·licitada.

En aquest mateix apartat també es troba l'opció de facturar els albarans d'un client.

10.2.7. Consultes

L'aplicació també es capaç de confeccionar llistats de clients, factures i vehicles, a part de llistar les factures pendents de cobrar com ja s'ha comentat en l'apartat Caixa.



Més endavant, quan l'aplicació s'hagi implantat al taller i els usuaris vegin les relacions que els hi manquen, s'ampliaran les possibilitats.

10.2.8. Base de dades

A part dels llistats comentats a l'apartat anterior, el gerent de l'empresa va trobar necessari poder realitzar altes, modificacions, cerques i llistats de: clients, vehicles, recanvis, ma d'obra i treballs externs. És a dir, les coses més necessàries per l'empresa, quedant les altres condicionades a que l'administrador, mitjançant la base de dades, les entrés. Però, finalment, després de diferents reunions, es va decidir incorporar-les a aquesta pantalla ja que es poden necessitar en qualssevol moment.

Collsaplana Motor, SL
 Benvingut usuari: Adria **Usuari amb rol mecànic**

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Vehicle:
Alta Consulta Llista

Client:
Alta Consulta Llista

Treballador:
Alta Consulta Llista

Impostos:
Alta Consulta Llista

Condicció de pagament:
Alta Consulta Llista

Mètode de pagament:
Alta Consulta Llista

Recanvis:
Alta Consulta Llista

Ma d'obra:
Alta Consulta Llista

Treballs externs:
Alta Consulta Llista

Taxes:
Alta Consulta Llista

Assignar client a vehicle:
Assigna

Dins d'aquest mateix apartat es troba l'opció d'assignar un client a un vehicle. Aquesta opció és necessària per si un client es ven un vehicle o en un moment determinat s'ha de realitzar una factura a un client diferent al propietari del vehicle.

Com es pot apreciar a l'imatge, aquesta vista correspon a un usuari amb rol mecànic.

Per veure'n més detalls, es mostra la pantalla de creació d'un client:

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Alta client

Dni

Nom

Cognoms

Telefon

Mobil

Poblacio

Codi postal

Provincia

Email

Un cop s'ha clicat sobre l'opció Crea, accedint a la base de dades de phpMyAdmin, s'observa com l'usuari s'ha creat correctament :

Copiar	Borrar											
Copiar	Borrar	8	NL	40361517J	Anna	Vilar Ribas	972 47 00 00	650 814 404	Caldes de Malavella	17455	Girona	annavillarribas@gmail.com
Copiar	Borrar	9	NL	40424587f	Ramon	Mas Selva	856 965 547	588 888 888	Sta Coloma de Farners	17450	Girona	ramon@masselva.com
Copiar	Borrar	10	NL	40404040d	Marti	Pla Roure	972 25 86 96	658 965 444	Girona	17005	Girona	marti@email.com

os / Desmarcar todos Para los elementos que están marcados: Cambiar Borrar Exportar

Per modificar el client, accedirem a l'opció Consulta d'aquest mateix apartat:

Collsaplana Motor, SL

Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Buscar client:

Introdueix els cognoms del client a buscar:

Cognoms

S'introdueixen els cognoms. Seleccionant Enviar consulta, apareix la finestra:

Collsaplana Motor, SL

Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Client

Id	Dni	Nom	Cognoms	Telefon	Mobil	Poblacio	Codi_postal	Provincia	Email
10	40404040d	Marti	Pla Roure	972 25 86 96	658 965 444	Girona	17005	Girona	marti@email.com

A l'imatge es pot veure com apareix l'informació del client Martí Pla Roure. Tenim l'opció de tornar a l'escriptori, editar l'informació del client o eliminar-lo (només en el cas que l'usuari sigui l'administrador).

Si seleccionem editar i canviem el telèfon i el domicili, veiem com a la base de dades s'han actualitzat les dades correctament:

rar	9	NL	404243071	Ramon	Mas Seiva	972 56 87 45	972 56 87 45	Sta Coloma de Ramons	17430	Girona	ramon@masseiva.com
rar	10	NL	40404040d	Marti	Pla Roure	972 56 87 45	658 965 444	Salt	17190	Girona	marti@email.com

10.2.9. Configuració

La pestanya de configuració permet gestionar les dades de l'empresa, gestionar els usuaris, realitzar el tancament comptable de l'any, gestionar els impostos i els mètodes de pagament.

Collsaplana Motor, SL

Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Dades empresa:
Consulta

Anotacions factures:
Edita logo Editar anotacions

Impostos:
Alta Consulta Llista

Mètode de pagament:
Alta Consulta Llista

Canvi any comptable:
Tancar

Alta usuaris aplicació:
Alta Consulta Llista

Aquest apartat serà visible només pels usuaris amb rol administrador i gerent, tal com es va pactar amb l'empresa.

Independentment del rol de l'usuari identificat, cada usuari només podrà modificar les seves dades.

A continuació es pot veure el detall de l'opció Editar logotip de l'empresa:

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CADXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

El logo actual és:



Edita logo:
Introdueix la ruta de l'imatge del logo de l'empresa:

[Torna a l'escriptori](#)

El llistat dels usuaris que estan registrats, vist des d'un usuari amb rol Administrador:

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CADXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Llistat treballadors

Id	Dni	Nom	Cognoms	Accions
1	40361517J	Anna	Vilar Ribas	Mostra Edita
2	40365896d	Pere	Vilar Vivolas	Mostra Edita
3	40404040d	Adria	Garcia Arco	Mostra Edita
4	48596580b	Albert	Vilar Ribas	Mostra Edita

[Torna a l'escriptori](#) [Imprimeix](#)

A partir de diferents reunions amb el gerent, es va decidir no realitzar la part de creació de còpies de seguretat de manera que el propi gerent de l'empresa el pogués realitzar. Fins a l'implementació d'aquest apartat, l'administrador del sistema les generarà manualment.

11. Conclusions

Un cop acabat el projecte, puc dir que estic satisfeta dels coneixements adquirits i la feina realitzada.

El principal objectiu de realitzar aquest projecte era poder plasmar els coneixements adquirits al llarg de la carrera creant una aplicació des de zero, desenvolupant-ne el disseny, la implementació i que pogués ser utilitzada. En aquest cas, això ha significat fer una aplicació a mida pel negoci familiar, ja que les aplicacions genèriques existents pels tallers mecànics tenen moltes mancances.

Personalment el projecte m'ha aportat confiança i seguretat en veure que podia assolir l'objectiu marcat. També m'ha aportat experiència en un projecte de certa magnitud, buscar llenguatges per a solucionar els requeriments del projecte i anar solventant els problemes a mesura que anaven sorgint.

Dels objectius inicials del projecte, s'han complert tots: s'ha dissenyat i implementat una aplicació molt més pràctica i fàcil d'utilitzar que l'existent. L'aplicació permet gestionar els usuaris de l'aplicació, els clients, vehicles, recanvis, les ordres de reparació, els pressupostos, albarans i les factures.

Respecte la planificació del temps a realitzar el projecte, no s'ha pogut complir ja que em vaig introduir al món laboral i no tenia gaire temps per dedicar-hi. El projecte s'ha allargat un any més del temps previst inicialment tal com es pot veure al diagrama de Gantt següent:

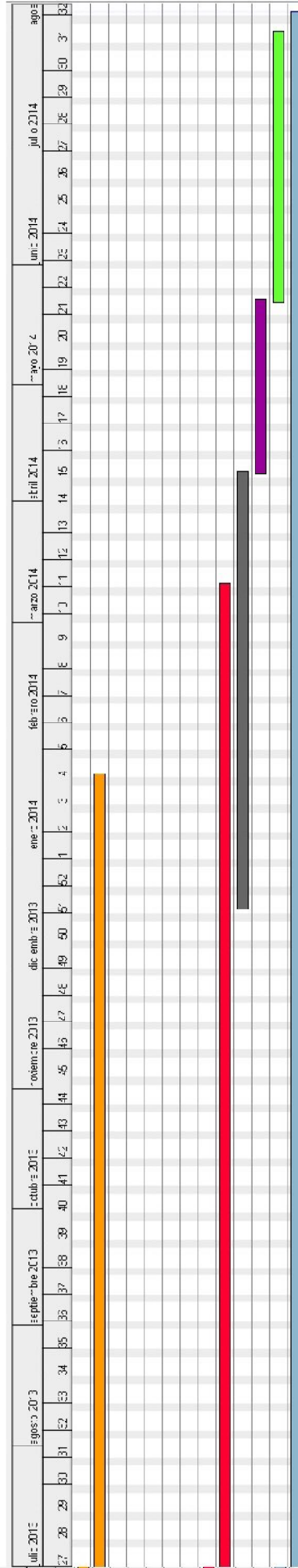
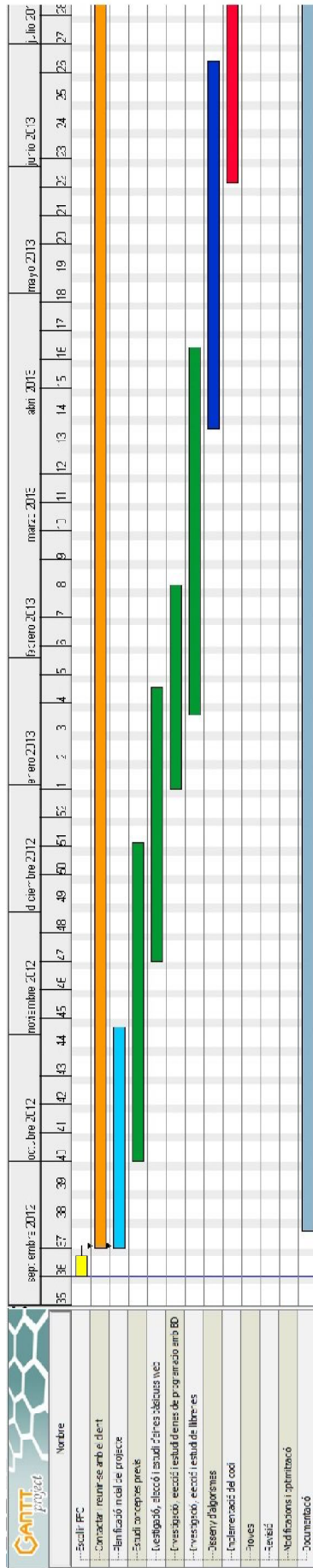


Diagrama de Gantt. S'ha dividit en dues parts per la seva llargada.

12. Treball futur

Un cop finalitzat el projecte i després de diferents reunions amb el gerent de l'empresa, han sorgit noves propostes i possibles ampliacions, aquestes són :

- Ampliar l'aplicació amb un calendari per planificar les visites dels clients al taller.
- Pogueu realitzar còpies de seguretat de l'aplicació sense la necessitat de l'administrador. Que un usuari, des de la mateixa aplicació, pugui realitzar-les.
- Afegir l'opció de generar rebuts per controlar els pagaments a compte dels clients.
- Modificar el sistema d'impressió de factures per eliminar l'utilització de fulls impresos, és a dir, que l'aplicació a part d'imprimir el contingut de la factura, imprimeixi amb certa qualitat el logo i el disseny dels fulls de l'empresa.

Més propostes, encara que més a llarg termini podrien ser :

- Afegir a l'aplicació el control d'estoc del material que hi ha al taller.
- Generar automàticament el llistat corresponent al Model 347.
- Ampliar l'aplicació amb gestió de Proveïdors.
- Estudiar l'integració de programes com Autodata o TecDoc utilitzats als tallers mecànics com a consulta de referències, temporització de les reparacions, informació tècnica del vehicle, etc.

13. Bibliografía

W3schools. 1999-2012. REFSNES DATA AS. 2012-2014. <http://www.w3schools.com>.

Symfony.es. 2013. *Symfony.es*. 2012-2013. <http://symfony.es/bundles/pqliwa/pdfbundle/>.

Twig. 2010-2012. Sensio Labs. 2012-2014. <http://twig.sensiolabs.org/>.

Php. 2001-2014. The PHP Group. 2012-2014. <https://php.net/manual/es/>.

Composer. 2001-2014. The PHP Group. 2012-2014. <http://getcomposer.org/>.

GitHub. 2008-2012. GitHub Inc. 2012-2014. <https://github.com/>.

phpMyAdmin. 2003-2014. *phpMyAdmin contributors*. 2012-2014. http://www.phpmyadmin.net/home_page/index.php.

XAMPP. 2002-2012. Apache Friends. 2012. <https://www.apachefriends.org/>.

Aceite Usado, Recogida y Reciclaje de Aceites Y Lubricantes Usados. 2013. SIGAUS. <http://www.siga.us/>.

Sistema Integrado de Gestión de Neumáticos Usados. 2012. SIGNUS Ecovalor. <http://www.signus.es/>.

Agència Espanyola de Protecció de Dades. 2010. *Agencia Española de Protección de Datos*. 21 de desembre de 2013. <https://www.agpd.es>.

Wikipedia, la enciclopedia libre. 2001-2012. Fundacion Wikimedia. 2012-2014. <http://es.wikipedia.org>.

DesarrolloWeb.com. 1999-2014. *DesarrolloWeb.com*. 2012-2014. <http://www.desarrolloweb.com>.

StackOverflow. 2008-2014. *Stack Exchange Network*. 2012-2014. <http://stackoverflow.com>.

Desarrollando y emprendiendo. 2011. Roberto Costumero Moreno. <http://roberto.costumero.es/2011/08/15/creando-un-login-simple-con-symfony2/>.

Librosweb. 2014. LibrosWeb.es. 2012-2014. <http://librosweb.es/>.

Google Groups. 1998-2014. Google Inc. 2012-2014. <https://groups.google.com>.

ShowMeTheCode. 1999-2014. Creative Commons Reconocimiento-Compartir Igual 3.0 Unported License. Google+. 2012-2014. <http://showmethocode.es/>.

14. Annexos

En aquest apartat s'explicarà com s'ha fet la migració de dades de l'aplicació que utilitza actualment l'empresa a la implementada en aquest projecte.

14.1 Migració de dades

L'aplicació actual utilitza una base de dades de Microsoft Access. Per fer la migració s'haurà de llegir la base de dades actual i bolcar les dades a la nova base de dades de phpMyAdmin. Les dues bases de dades utilitzen taules diferents, a continuació s'explica el contingut de les taules de la base de dades del programa utilitzat fins el moment :

- Alb : conté els albarans generats fins el moment.
- Albb : albarans amb les bases calculades.
- Albl : línies de l'albarà.
- Alm : magatzem (taula no utilitzada).
- Ant : emmagatzema els pagaments a compte dels clients (taula no utilitzada).
- Ban : dades dels bancs.
- Bar : temps / barems (taula no utilitzada).
- Caj : caixa (taula no utilitzada).
- Cli : desa les dades dels clients.
- Cmv : enregistra els moviments de caixa.
- Cnf : configuració de l'aplicació.
- Com : comandes de compres.
- Coml : conté una línia de comanda de compra.
- Efe : efectes.
- Ext : desa els treballs externs donats d'alta a l'aplicació.
- Fac : conté les factures generades.
- Facb : factura amb les bases calculades.
- Facl : línia de factura.
- Fch : s'emmagatzema les reparacions que fa cada mecànic (taula no utilitzada).

- Fco : factura de compra.
- Fcob : factura compra amb les bases calculades.
- Fpg : conté les diferents formes de pagament utilitzades.
- Gam : gamma vehicle.
- Kit : descripció del kit (podria ser un conjunt de peces).
- Kitl : línia del kit.
- Knt : contador.
- Mar : marca vehicle.
- Mdl : model vehicle.
- Obs : observacions documents.
- Omi : omisions (taula no utilitzada).
- Ope : operació ma d'obra.
- Orc : desa les ordres de reparacions.
- Orl : línies d'ordres de reparació.
- Pob : poblacions.
- Pro : províncies.
- Prv : proveïdors.
- R347 : informe 347 – operacions amb tercers.
- Rec : recanvi.
- Stk : estoc (taula no utilitzada).
- Stkm : moviment de estoc(taula no utilitzada).
- Tec : tècnic (taula no utilitzada).
- Tef : tipus de efecte.
- Teh : tècnic / hores mes, per realitzar un control (taula no utilitzada).
- Tie : temps (taula no utilitzada).
- Tip : tipus de vehicle.
- Tti : tipus temps (taula no utilitzada).
- Veh : emmagatzema els vehicles donats d'alta.
- Ver : control versió de l'aplicació.

Un cop s'han descrit totes les taules de la base de dades de l'aplicació actual, cal buscar l'equivalència entre les taules de la base de dades actual i la nova que s'utilitzarà.

Base de dades nova	Base de dades actual (M. Access)
Condicció pagament	
Client	Cli, (pob, pro)
Vehicle	Veh, (gam, mar, mdl, tip)
Recepció	
Ordre de reparació	Orc, (alb, fact)

línia ordre de reparació	Orl, (albl, fact)
Pressupost	
Factura (fact. Immediata)	factb
Mètode pagament	fpg
Impostos	
Albarà	albb
Treballador	tec
Rol	
Treball extern	ext
Ma d'obra	ope
Recanvi	rec
Taxes	
Configuració	cnf

En algun dels casos, l'equivalència entre taules no és automàtica, tal com es pot veure en el cas de la taula *Client*. Aquesta taula equival a la taula *Cli* de la base de dades existent i les taules relacionades amb aquesta, que apareixen entre parèntesis: la taula *Pro* i *Pob*. A la base de dades de Microsoft Access existeix una relació entre aquestes 3 taules. Cada client és d'una població (*Pob*) i d'una província (*Pro*).

Amb la taula *Vehicle* passa el mateix, la base de dades actual utilitza taules diferents per emmagatzemar el tipus (taula *Tip*), la gamma (*Gam*), el model (*Mdl*) i la marca (*Mar*) del vehicle.

El cas de la taula *Ordre de reparació* és més complicat. A l'aplicació realitzada, es considera que una ordre de reparació és l'unitat d'emmagatzemament de totes les operacions que es realitzen al taller, i que aquesta, depenent de si és un pressupost, un albarà o una factura, va canviant d'estat i al moment de tancar el procés, es genera l'objecte pressupost, factura o albarà corresponent amb els seus totals calculats. A l'aplicació utilitzada fins ara, aquest funcionament només s'aplica amb els pressupostos, ja que com es pot veure al llistat de taules, no existeix cap taula pressupost. Els albarans i factures són objectes independents tinguent la relació amb la taula línia albarà (taula *Alb*) i línia factura (taula *Fac*) corresponent.

Al veure que les aplicacions utilitzen les taules de diferent manera, s'ha decidit migrar les taules amb l'informació bàsica com poden ser :

- els clients,
- vehicles,
- les factures,
- treballadors,

- mètode de pagament,
- els recanvis,
- ma d'obra i
- treball extern.

No es migraran taules com per exemple els albarans, ja que si no s'han facturat durant l'any en curs, ja no es facturaran.

No s'ha anomenat la taula factura única mensual ja que és una novetat de la nova aplicació.

Un cop decidides les taules a utilitzar, es realitza l'exportació de dades. Per fer-ho, el programa Microsoft Access permet fer l'exportació de les dades amb un fitxer de text, per tant, es definiran els separadors de columnes que en el nostre cas seran punts i coma (;), i es generaran els fitxers de text de totes les taules. A continuació se'n mostra un exemple :

Retall de la taula *veh* al programa Microsoft Access :

matric	cli	mar	mdl	gam	bas	uvisita	ukm	uobs	ifranq	aseg	tip	itv	codeVivid	codr
0123DDX	16	RENAULT	TRAFIC	1	VF1FLACA6540	20100906	106528		0	0	dci 100			
0166FVT	469	SEAT	LEON	1		20101006	72810		0	0				
0170DSJ	505	MERCEDES BEN	VITO	2		20100204	0		0	0				
0240DNH	447	KIA	SPORTAGE	1		20100414	42782		0	0	CRDi 16v 4WC			
027	8	Màquina 027		1		20100226	0		0	0				
0286CFG	168	MITSUBISHI	L 200	1		20100217	106966		0	0	2.5 TD 4x4			
0355CNG	474	VOLVO	XC70 CROSS CC	1		20100920	172492		0	0	2.5 D5 XC AWD			
0560DZD	437	MITSUBISHI		1		20100312	0		0	0	Evo 9			
0562FJP	397	BMW	X3	1	WBAPD11030V	20100917	0		0	0	2.0 d			
0659DZY	121	FIAT	DUCATO	1		20100823	61507		0	0				

A la taula *veh* es pot veure com la primera dada correspon a la matrícula del vehicle, la segona al codi del client, la marca del vehicle, el model, la gamma, el bastidor del vehicle, la data de l'última visita, els quilòmetres, observacions, si té franquícia, l'asseguradora, el tipus de vehicle, quan li toca passar l'itv, i la resta d'atributs només són útils en cas de tenir l'aplicació lligada a una aplicació de recanvis i informació per als tallers.

Exemple del fitxer *veh.txt*:

```
[...]

"3996DBZ";200;"TOYOTA";"AVENSIS";"1";"";"20100722";143410;"
";0,00;0;"2.0 D-4D";"";"";"";"";""
"4029DGR";62;"VOLKSWAGEN";"POLO";"1";"WVWZZZ9NZ5Y146578";"20100209";16
4321;"";0,00;0;"";"";"";"";"";""
"4105BYF";428;"PEUGEOT";"307";"1";"VF33CRHYB82542457";"20100323";12498
1;"";0,00;0;"";"";"";"";"";""
```


Com es pot veure al pseudocodi anterior, les crides per insertar les dades de la base de dades actual a la nova moltes vegades no seran completes ja que els atributs varien d'una base de dades a l'altre. Per completar les dades, quan el vehicle torni al taller, l'usuari les haurà d'entrar manualment.

15. Manual d'usuari i/o instal·lació

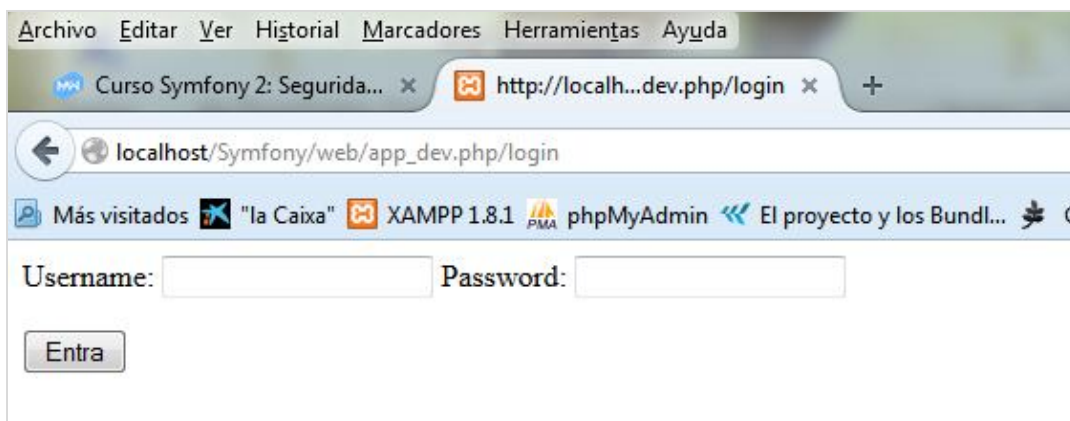
En aquest apartat es descriurà el funcionament de l'aplicació web implementada i també els passos a seguir per a l'instal·lació de la mateixa per un usuari expert.

15.1. Manual d'usuari

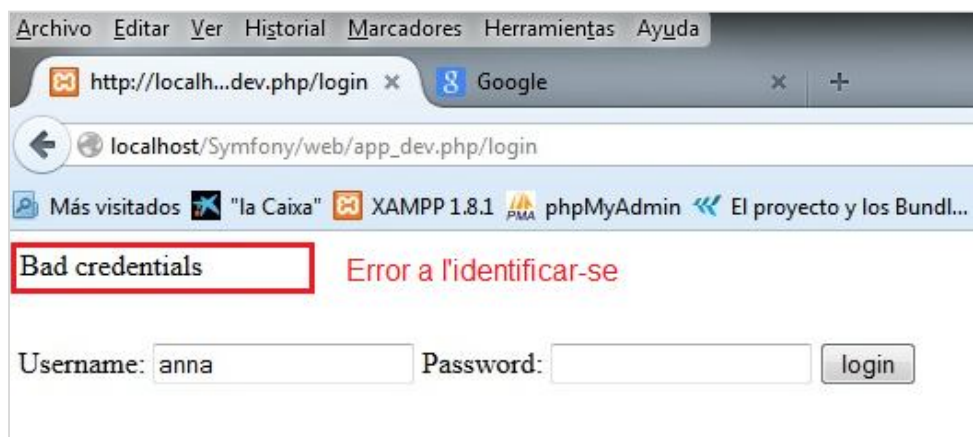
En aquest apartat es farà una explicació per a l'usuari del funcionament de les diferents pantalles realitzades a l'aplicació.

15.1.1. Pantalla d'identificació

És la primera pantalla de l'aplicació. En aquesta pantalla l'usuari ha d'introduir el nom d'usuari i la contrasenya.



Si l'identificació introduïda no és correcta, apareixerà un missatge d'error:






Un cop l'usuari s'identifica, s'obre la pantalla de l'escriptori.

15.1.2. Pantalla escriptori

Aquesta pantalla és la pantalla principal de l'aplicació, des d'aquesta, l'usuari accedirà a totes les opcions que ofereix l'aplicació i tindrà una visió general dels vehicles que hi ha actualment al taller, ja siguin físicament dins el taller o se'n tinguin pressupostos pendents d'acceptar mentre siguin vàlids.

Perquè sigui més visual, s'utilitza el següent codi de colors per identificar l'estat dels vehicles al taller:

	PRESSUPOST pendent d'acceptar (vigent)
	RECEPCIÓ d'un vehicle (esperant per ser reparat)
	ORDRE DE REPARACIÓ d'un vehicle que s'està reparant

Com es pot veure a la taula anterior, el codi de colors utilitzat és el mateix que un semàfor.

Cada vegada que l'usuari vulgui modificar algun vehicle que estigui al taller, o acceptar el pressupost i començar la reparació del mateix, només haurà de clicar sobre l'icona del vehicle corresponent.

L'escriptori, igual que totes les pestanyes de l'aplicació, contindrà el següent menú principal:

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Pressupostos pendents

Vehicles al taller :

Cal remarcar que a totes les pantalles menys les d'impressió en format Pdf es veurà l'usuari que esta identificat actualment.

15.1.3. Pantalla taller

Des d'aquesta pantalla l'usuari podrà donar d'alta una recepció d'un vehicle i un client.

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

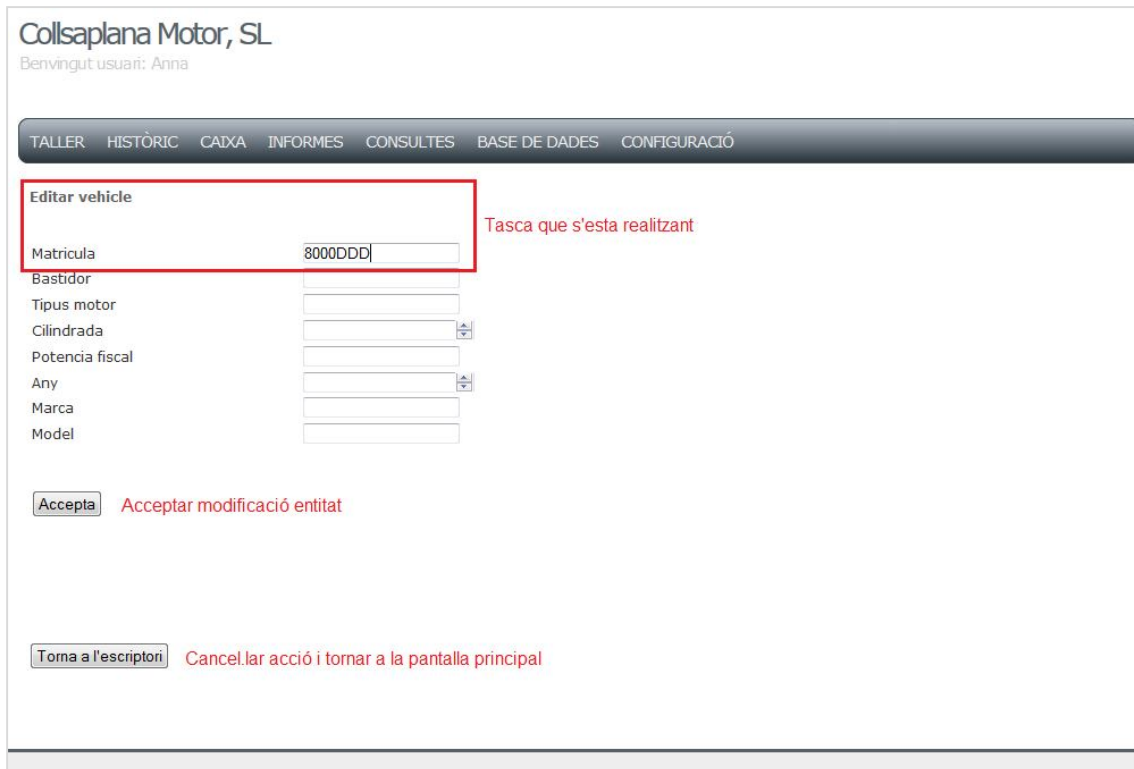
Nova recepció Tasca que s'esta realitzant

Data 2009 ▾ - Jan ▾ - 1 ▾
Renuncia pressupost
Descripcio operacions
Data prev lliurament 2009 ▾ - Jan ▾ - 1 ▾
Recollir peces
Reparacio en garantia
Matricula
Dni

Crea Confirmar tasca

Torna a l'escriptori Cancel·lar tasca i tornar a la pantalla principal

Si el client o el vehicle no existeixen, automàticament es generarà un formulari d'alta perquè es pugui acabar la recepció.



Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Editar vehicle

Matricula Tasca que s'esta realitzant

Bastidor

Tipus motor

Cilindrada

Potencia fiscal

Any

Marca

Model

Acceptar modificació entitat

Cancel·lar acció i tornar a la pantalla principal

Un cop finalitzada la recepció, es tornarà a l'escriptori, on ara apareixerà una nova icona amb la recepció creada.

15.1.4. Pantalla històric

Aquesta pantalla conté les consultes necessàries per a obtenir l'informació dels vehicles dels clients, necessària principalment pels mecànics, donat que en moltes ocasions és necessari saber quines reparacions o quan s'han dut a terme alguns treballs.

La pantalla històric és:

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CADXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Donat un CLIENT, llistar els seus vehicles

Donat un VEHICLE, llistar les seves reparacions

Torna a l'escriptori Torna a la pantalla principal

En totes les consultes realitzades en aquesta aplicació, es demana a l'usuari un valor, ja sigui l'identificador o alguna propietat de l'objecte cercat.

Si per alguna raó l'usuari accepta l'acció que esta realitzant sense emplenar el formulari, apareixerà un missatge d'avís.

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CADXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Donat un CLIENT, llistar els seus vehicles: Acció que s'esta realitzant

Introdueix el dni del client:

Dni

Cerca

Rellene este campo. Missatge d'avís, en aquest cas, no es pot deixar el camp Dni null

Torna a l'escriptori

15.1.5. Pantalla caixa

A la pantalla de caixa, només hi podrà accedir l'usuari amb rol administrador o Gerent.

Collsaplana Motor, SL
Benvingut usuari: Anna **Usuari amb rol ADMINISTRADOR**

TALLER HISTÒRIC **CAIXA** INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Buscar factura: Acció que es realitza

Introdueix el numero de factura:

Id

Cerca

Torna a l'escriptori Opcions possibles

Si un usuari amb rol Mecànic hi intenta accedir, obtindrà la següent resposta:

Collsaplana Motor, SL
Benvingut usuari: Albert **Usuari amb rol MECÀNIC**

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

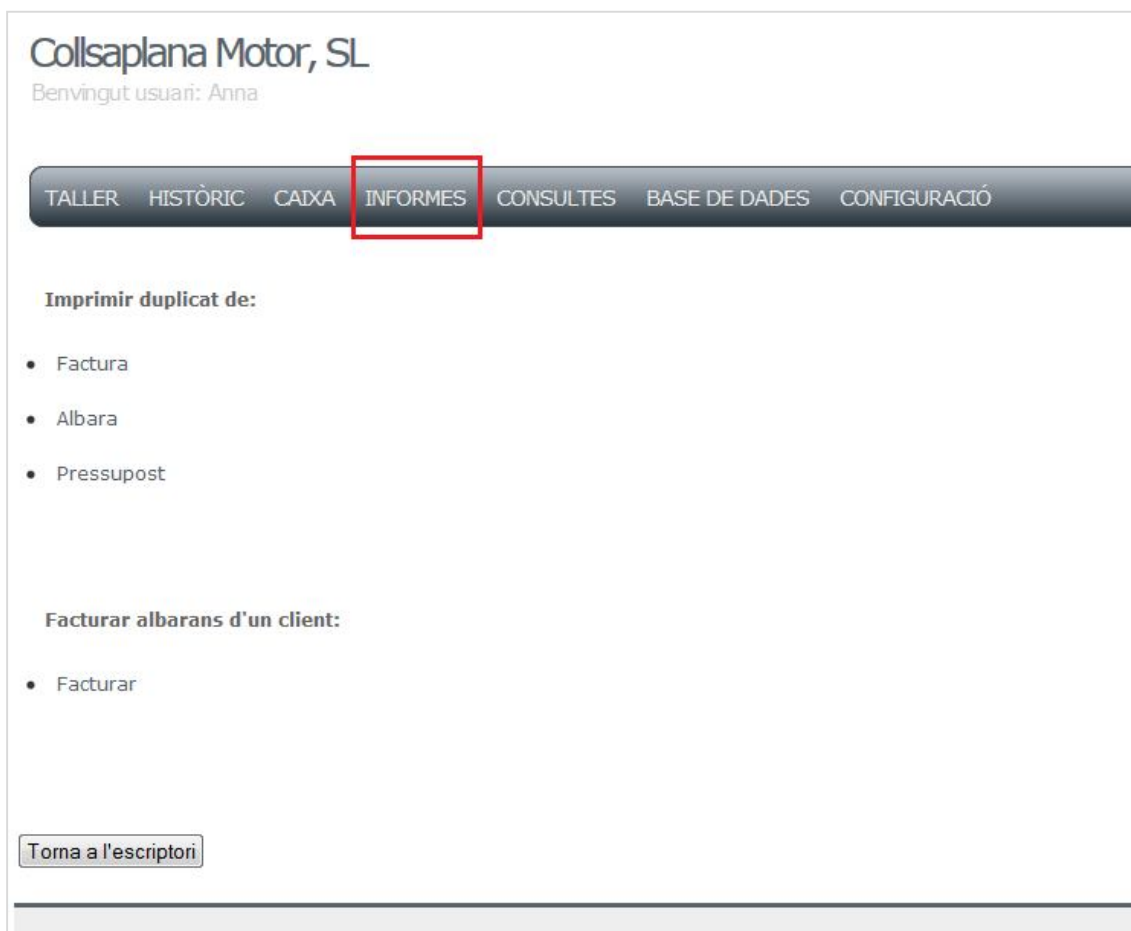
ATENCIÓ: Per accedir a aquesta pestanya es necessiten permisos d'ADMINISTRADOR O GERENT !

Missatge de control d'accès

Torna a l'escriptori

15.1.6. Pantalla informes

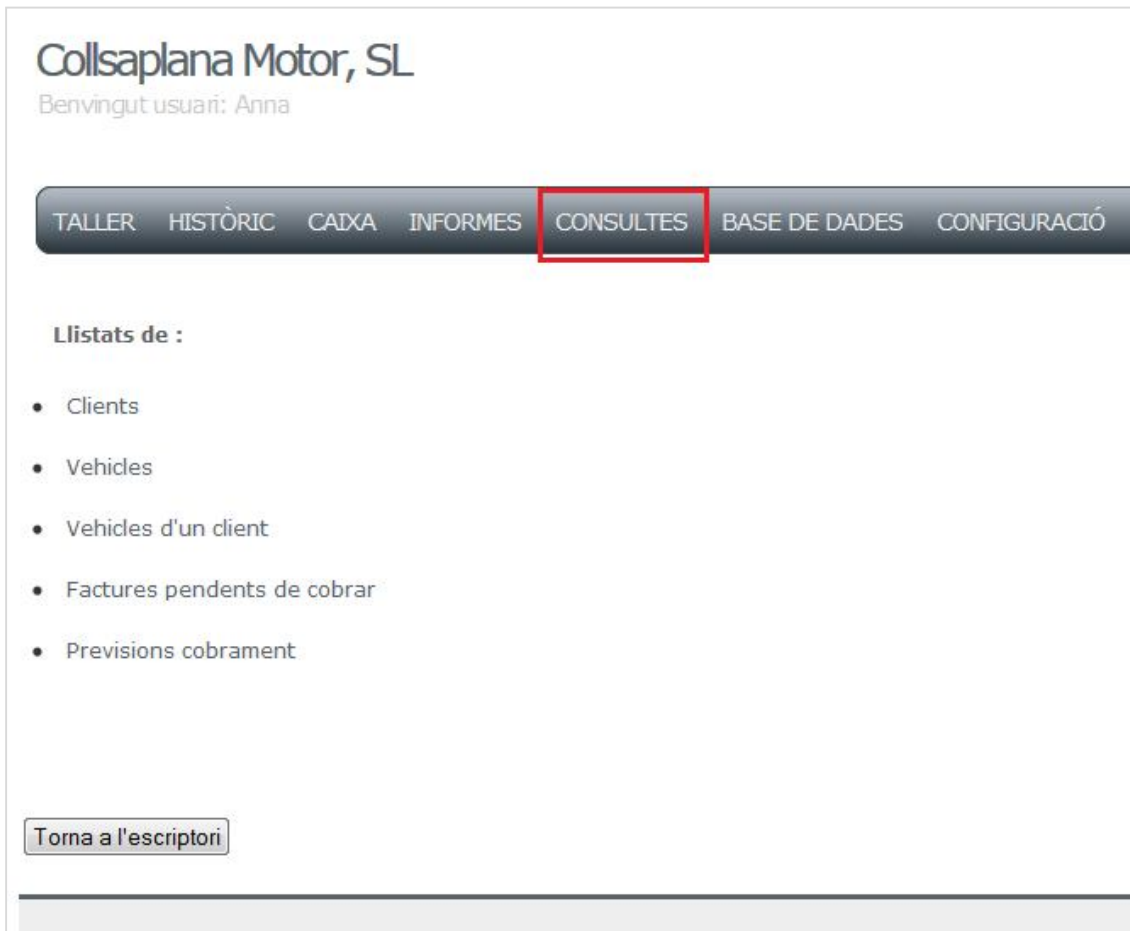
Aquesta pantalla dona la possibilitat d'imprimir duplicats de factures, albarans, pressupostos i facturar albarans d'un client.



Les vistes de les pantalles a les que s'accedeix des d'aquesta pantalla informes són per a fer operacions de cerca, com moltes d'aquesta aplicació.

15.1.7. Pantalla consultes

A la pantalla consultes, l'usuari pot obtenir diferents llistats. Tota l'aplicació manté el mateix format, per tant, la pantalla consultes és:



Si es selecciona una opció en aquesta pantalla, es poden obtenir 2 tipus de pantalla: o de cerca com altres que hem vist, o de llistar com pot ser l'opció de llistar vehicles.

Sobre cada element del llistat es poden realitzar dues opcions:

- Mostrar l'element seleccionat sol (en aquesta pantalla, si l'usuari és de tipus administrador podrà esborrar l'element de l'aplicació)
- Editar-lo

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES **CONSULTES** BASE DE DADES CONFIGURACIÓ

Llistat vehicles Acció que s'esta realitzant

Id	Matricula	Bastidor	Tipus_motor	Cilindrada	Potencia_fiscal	Any	Marca	Model	Actions
19	22	22	22	22	22	22	22	22	Mostra Edita
20	1000BHT	dfdfdf4ffrgg5g	gth-d	1995	85	2005	bmw	serie 1	Mostra Edita
21	100	dfdsf5d8fds2fd5fs5	5896	558	55	1996	seat	600	Mostra Edita
22	66	fff5855f88f56f6f9f5f5f5f	hyj	1998	105	2007	renault	laguna	Mostra Edita
23	2204CSN	fd85f9d6fd8f5dfd6	frt	5896	66	5966	toyota	yaris	Mostra Edita
24	105	8574f8596f5844d	d-5896d	569	258	2007	Mazda	5	Mostra Edita
25	106	fd85df9sf6	8596	6	6	296	Honda	Accord	Mostra Edita
26	107	885fd9fdfsfd8s5	fd85	58	5	2008	Jeep	Cherokee	Mostra Edita

Accions que es poden realitzar sobre cada element de la llista

Toma a l'escriptori Imprimeix Opcions disponibles: tornar a la pantalla principal o imprimir amb format pdf

Per no fer l'imatge tant gran, s'ha retallat el contingut d'aquesta.

15.1.8. Pantalla base de dades

La pantalla base de dades permet a l'usuari consultar diferents elements de la base de dades com són: Vehicles, Clients, Recanvis, Ma d'obra, Mètodes de pagament, etc. I realitzar l'acció d'assignar un client a un vehicle.

Aquestes operacions de consulta es veuran alterades pels usuaris de tipus mecànic ja que podran consultar, però no modificar ni eliminar cap element. Podran fer sense cap restricció d'operació d'assignar un client a un vehicle ja que és necessària per crear una recepció, tal com s'ha vist a l'apartat 15.1.3.Taller.

Totes les vistes relacionades amb aquesta pantalla són similars, en seleccionarem un exemple:

Collsaplana Motor, SL
 Benvingut usuari: Anna **usuari de tipus administrador**

TALLER HISTÒRIC CADXA INFORMES CONSULTES **BASE DE DADES** CONFIGURACIÓ

Vehicle: Alta Consulta Llista **Client:** Alta Consulta Llista **Treballador:** Alta Consulta Llista

Impostos: Alta Consulta Llista **Condicció de pagament:** Alta Consulta Llista **Mètode de pagament:** Alta Consulta Llista

Recanvis: Alta Consulta Llista **Ma d'obra:** Alta Consulta Llista **Treballs externs:** Alta Consulta Llista

Taxes: Alta Consulta Llista **EXEMPLE ESCOLLIT**

Assignar client a vehicle:
 Assigna

Existeixen tres tipus de vista diferent en aquesta plantilla:

- Vista d'alta

TALLER HISTÒRIC CADXA INFORMES CONSULTES **BASE DE DADES** CONFIGURACIÓ

Alta taxes Operació que s'esta realitzant

Descripció
 Preu unitat

Crea

Torna a l'escriptori Opcions disponibles

- Vista de consulta

Com altres exemples que s'han vist en aquest manual, a partir de l'identificador o algun element descriptiu, en aquest cas la descripció de la taxa, es busca l'objecte de la base de dades que contingui aquesta descripció.

The screenshot shows a navigation bar with the following items: TALLER, HISTÒRIC, CAIXA, INFORMES, CONSULTES, **BASE DE DADES**, and CONFIGURACIÓ. Below the navigation bar, there is a search section. A red box highlights the text 'Buscar taxa:' and another red box highlights the text 'Acció que s'esta realitzant'. Below this, there is a text input field labeled 'Introdueix la descripció de la taxa a buscar:' and a label 'Descripcio' next to it. At the bottom of the search section, there is a button labeled 'Enviar consulta' and another red box highlights the text 'Opció a realitzar'.

- Vista de llista

En aquest cas es mostren tots els elements existents a la base de dades de tipus taxa.

The screenshot shows the same navigation bar as the previous image. Below it, there is a list view. A red box highlights the text 'Llistat taxes' and another red box highlights the text 'Acció que s'esta realitzant'. To the right, there is a red box highlighting the text 'Accions sobre cada element de la llista'. Below this, there is a table with the following data:

Id	Descripcio	Preu_unitat	Accions
1	Reciclatge d'oli R.D. 679/2006	0.054	Mostra Edita
2	ECOVALOR SIGNUS (R.D. 1619/2005), A	0.95	Mostra Edita
3	ECOVALOR SIGNUS (R.D. 1619/2005), B	1.58	Mostra Edita
4	ECOVALOR SIGNUS (R.D. 1619/2005), C	2.75	Mostra Edita
5	ECOVALOR SIGNUS (R.D. 1619/2005), D	13.25	Mostra Edita

At the bottom of the list view, there is a red box highlighting two buttons: 'Torna a l'escriptori' and 'Imprimir'. To the right of these buttons, there is a red box highlighting the text 'Opcions: tornar a la pantalla principal o imprimir amb format pdf'.

Si es selecciona un objecte de la taula i se li aplica l'acció *Mostra*, obtenim la següent pantalla:

Taxes

Id	Descripció	Preu unitat
1	Reciclatge d'oli R.D. 679/2006	0.054

Toma a l'escriptori Edita Elimina Opcions disponibles

15.1.9. Pantalla configuració

Aquesta pantalla dona la possibilitat de configurar l'aplicació. Només hi poden accedir els usuaris de tipus administrador o gerent.

Collsaplana Motor, SL
Benvingut usuari: Anna

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES **CONFIGURACIÓ**

Dades empresa:
Consulta

Anotacions factures:
Edita logo Edita anotacions

Impostos:
Alta Consulta Llista

Mètode de pagament:
Alta Consulta Llista

Canvi any comptable:
Tancar

Alta usuaris aplicació:
Alta Consulta Llista

Aquí es podran modificar les dades de l'empresa, els impostos aplicables als serveis oferts, donar d'alta usuaris a l'aplicació, tancar l'any comptable i modificar les anotacions que apareixeran a les factures.

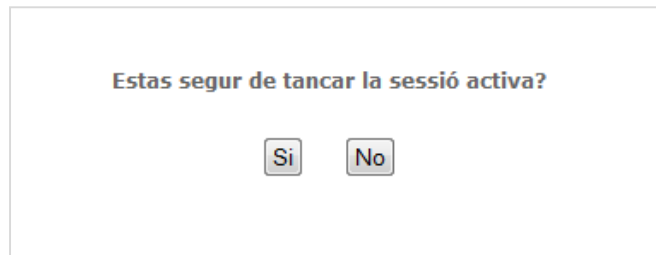
15.1.10. Pantalla tancar sessió

Després d'utilitzar l'aplicació, quan l'usuari decideix tancar la sessió, només ha de clicar sobre el nom que apareix a la pantalla escriptori:

Collsaplana Motor, SL
Benvingut usuari: Anna **Seleccionar per tancar la sessió de l'usuari Anna**

TALLER HISTÒRIC CAIXA INFORMES CONSULTES BASE DE DADES CONFIGURACIÓ

Apareix la següent pantalla:



Si selecciona "Si", es torna a la pantalla de login i si selecciona "No", es recupera la pantalla de l'escriptori de l'aplicació.

15.2. Manual d'instal·lació

Aquest manual explica breument els requisits per a l'instal·lació de l'aplicació web implementada i els passos a seguir per a posar-la en funcionament.

L'únic requisit existent és disposar d'un PC amb Windows, en el cas d'aquest projecte, s'ha realitzat amb Windows 7, i les aplicacions necessàries:

- Apache XAMPP
- Composer
- Git
- Framework Symfony 2. (la versió més actual existent)
- Base de dades phpMyAdmin
- PdfBundle
- Notepad++

Aquestes aplicacions es poden trobar i descarregar a la corresponent pàgina web oficial.

A continuació s'explicarà pas a pas el procés d'instal·lació de XAMPP. D'aquesta manera aconseguirem tenir un *Apache* i *php* per poder començar a instal·lar els altres components necessaris per treballar amb el Framework escollit, el Symfony.

1r PAS: Instal·lar XAMPP.

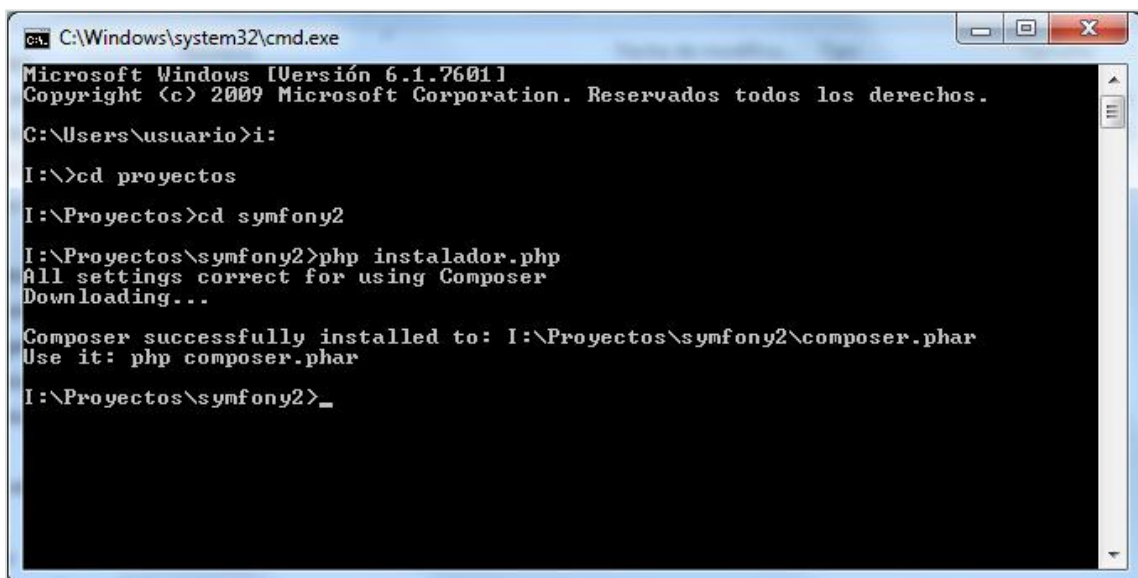
Seguir les instruccions de l'executable.

2n PAS: Instal·lar Composer.

Una vegada instal·lat XAMPP, es pot instal·lar Composer. S'accedeix a la pàgina web i es copia el contingut en un fitxer anomenat *instalador.php*, que guardarem al directori on crearem el projecte de Symfony.

S'obre una finestra del sistema de comandes de Windows (CMD) i s'executa la comanda:

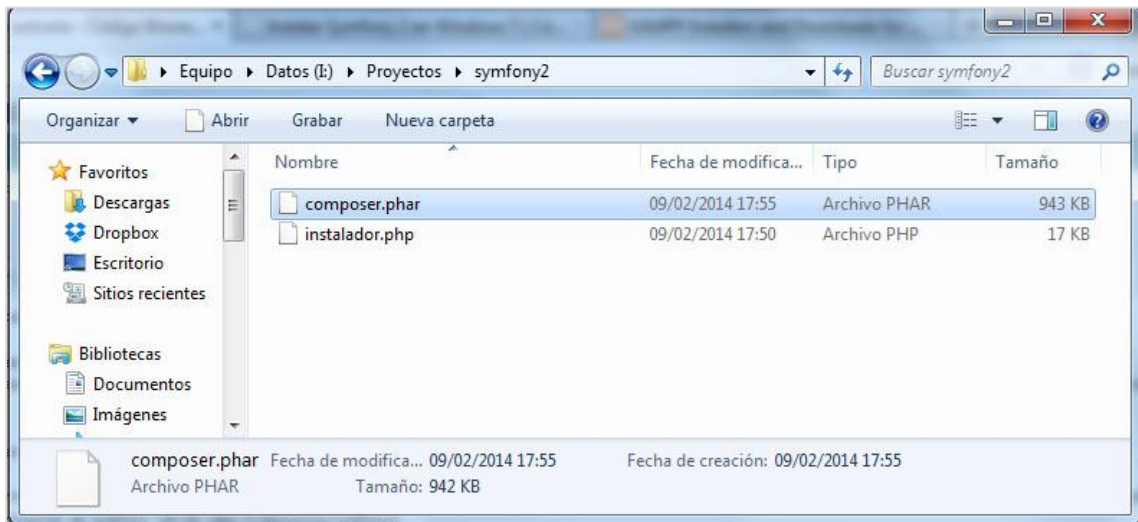
```
php instalador.php
```



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.
C:\Users\usuario>i:
I:\>cd proyectos
I:\Proyectos>cd symfony2
I:\Proyectos\symfony2>php instalador.php
All settings correct for using Composer
Downloading...

Composer successfully installed to: I:\Proyectos\symfony2\composer.phar
Use it: php composer.phar
I:\Proyectos\symfony2>_
```

Amb això s'aconsegueix instal·lar el Composer i per tant, al PC es tindrà un nou fitxer anomenat *composer.phar*.



Perquè aquest fitxer es pugui executar des de diferents llocs, s'ha de moure al directori on s'ha instal·lat el *xamp/php*. També s'ha de crear un fitxer *composer.bat* amb el següent contingut:

```
@ECHO OFF
php "%~dp0composer.phar" %*
```

D'aquesta manera es podrà executar Composer des de qualssevol lloc i no caldrà tornar-lo a instal·lar.

Per actualitzar-lo només cal executar la comanda:

```
C:\> composer self-update
```

3r PAS: Instal·lar Git.

Seguir les instruccions de l'executable.

4t PAS: Instal·lar Symfony.

Una vegada es té el Git instal·lat, s'obre una finestra de comandes de Windows i s'executa la comanda:

```
php composer.phar create-project symfony/framework-standard-edition  
i:/proyectos/symfony2/proyecto1 2.1.x-dev
```

Ja s'haurà instal·lat el Framework Symfony al PC.

Per comprovar que no hi hagin errors, es pot executar la següent comanda des de la ruta del projecte.

```
php app/check.php
```

5é PAS: Configurar *php.ini*.

Per poder treballar amb el Framework, s'ha d'afegir aquesta extensió al fitxer *php.ini*. Es pot trobar dins l'instal·lació de XAMPP a *C:/xampp/php*.

```
extension=php_openssl.dll
```

6é PAS: Realitzar canvis a l'Apache i comprovar la pantalla d'inici de Symfony.

Amb tot el software instal·lat, només falta fer que l'Apache apunti al codi de l'aplicació. S'han de realitzar els següents canvis al fitxer: *C:\xampp\apache\conf\httpd.conf*.

Canviar el següent:

```
DocumentRoot "I:/Proyectos/Symfony2/Proyecto1/web"
```

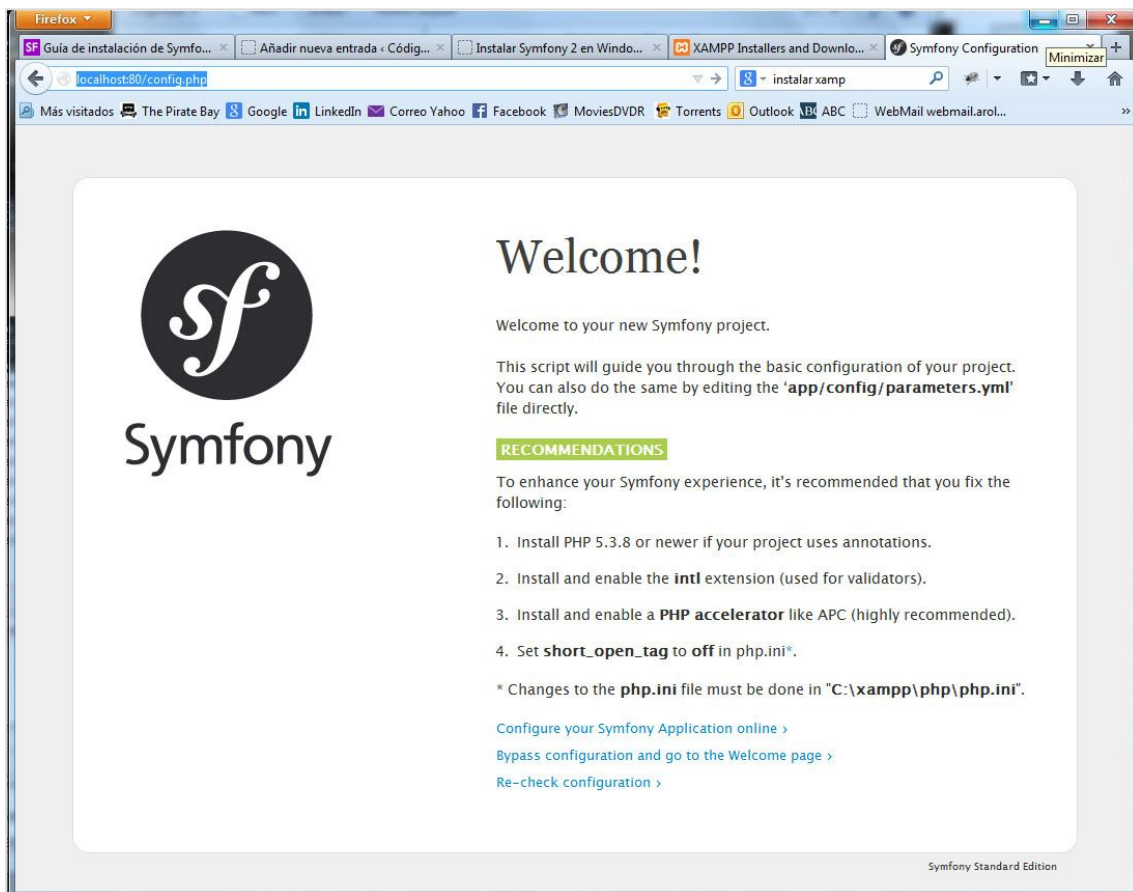
i,

```
Directory "I:/Proyectos/Symfony2/Proyecto1/web"
```

Iniciant l'Apache al panell de control de XAMPP, i posant al navegador la següent adreça:

```
localhost:80/config.php
```

Es pot veure la pàgina web:



Si es pot veure aquesta imatge, indica que s'ha instal·lat correctament Symfony.

7é PAS: Carregar l'aplicació web.

L'últim pas que queda és posar dins el projecte Symfony creat, l'aplicació web realitzada en aquest Projecte.