



Treball Final de Màster

Estudi: Màster en Enginyeria Informàtica

Títol: Recopilació i anàlisi de les dades obtingudes d'una xarxa de sensors per analitzar la qualitat de l'aire d'una ciutat

Document: Memòria

Alumne: Robert Garcia Ventura

Tutor: Dr. Santiago Thió Fernández de Henestrosa

Departament: Informàtica, matemàtica aplicada i estadística

Àrea: Estadística i investigació operativa

Convocatòria (mes/any): Setembre/2019

Agraïments

Aquest treball no s'hagués pogut realitzar sense la col·laboració i suport de les persones i institucions que, de forma desinteressada, m'han ajudat.

En primer lloc vull donar les gràcies al meu tutor, el Dr. Santiago Thió Fernández de Henestrosa, per tot el temps que ha dedicat a la supervisió del treball, així com els consells i els coneixements que m'ha transmès.

Agrair al professor Dr. Nicolas Boccard Marie Maurice la confiança que va depositar en el projecte, ja que gràcies a ell, a través d'un conveni de col·laboració amb la Universitat de Girona, es va poder ampliar el projecte RoMi.

També vull donar les gràcies a la Universitat de Girona, per destinar una part del pressupost a fer col·laboracions educatives amb estudiants, i a la Generalitat de Catalunya, que a través de concursos, com SmartCAT Challenge, fa possible el finançament de projectes de sostenibilitat com aquest.

La meva família mereix un punt i apart, en especial pels meus pares Dolors i Robert, la meva germana Anna i la meva parella Marina.

Concloc aquesta secció dedicant el treball als meus avis.



Índex

1.	Introducció, motivacions, propòsit i objectius del projecte	6
1.1	Introducció	6
1.2	Motivacions.....	7
1.3	Propòsits.....	8
1.4	Objectius	9
2.	Estudi de viabilitat.....	10
2.1	Recursos necessaris	10
2.2	Viabilitat econòmica.....	10
2.2.1	Cost de AWS API Gateway	11
2.2.2	Cost de AWS Lambda.....	11
2.2.3	Cost de Amazon RDS per a MySQL	11
2.2.4	Cost de Amazon EC2.....	12
2.2.5	Cost de Amazon Cognito	12
2.2.6	Cost dels dominis web	12
2.2.7	Cost del servidor virtual a IONOS 1&1	13
2.2.8	Resum de la viabilitat econòmica	13
2.3	Viabilitat tecnològica.....	13
2.4	Conclusió	13
3.	Metodologia	14
4.	Planificació.....	16

5.	Marc de treball i conceptes previs	18
5.1	Amazon Web Services (AWS)	18
5.1.1	Amazon API Gateway	19
5.1.2	AWS Lambda	20
5.1.3	Amazon RDS.....	21
5.1.4	Amazon EC2	21
5.1.5	Amazon Cognito.....	22
5.2	Qualitat de l'aire.....	23
5.2.1	Zones de Qualitat de l'aire (ZQA).....	23
5.2.2	Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica (XVPCA)	24
6.	Requisits del sistema	27
6.1	Requisits bàsics.....	27
6.2	Requisits específics.....	27
7.	Estudis i decisions	28
7.1	Estudi dels serveis de AWS.....	28
7.2	Estudi de l'escalabilitat de la base de dades.....	28
7.3	Origen de les dades de la qualitat de l'aire	31
7.4	Origen de les dades meteorològiques	33
7.5	Anàlisi de les dades de la qualitat de l'aire.....	35
8.	Anàlisi i disseny del sistema	36
8.1	Disseny del sistema de recopilació de dades (SRD)	36

8.1.1	Primera etapa del disseny del sistema de recopilació de dades.....	36
8.1.2	Segona etapa del disseny del sistema de recopilació de dades.....	37
8.1.3	Tercera etapa del disseny del sistema de recopilació de dades.....	39
8.1.4	Quarta etapa del disseny del sistema de recopilació de dades.....	40
8.2	Disseny de la base de dades.....	42
8.3	Disseny de la API REST.....	43
9.	Implementació.....	45
9.1	Configuració de la base de dades.....	45
9.1.1	Configuració de la rèplica de la base de dades.....	49
9.2	Configuració de la API REST amb Amazon API Gateway.....	51
9.2.1	Programació dels endpoint de la API amb AWS Lambda.....	55
9.2.2	Configuració del control de seguretat amb AWS Cognito.....	60
9.3	Configuració dels servidors Shiny.....	63
9.3.1	Creació de la instància EC2 a AWS.....	63
9.3.2	Instal·lació de Nginx.....	68
9.3.3	Creació del certificat SSL/TLS amb Let's Encrypt.....	70
9.3.4	Instal·lació de R i Shiny server.....	74
9.3.5	Actualització de R i els paquets instal·lats.....	77
9.3.6	Configuració del proxy invers amb Nginx.....	79
9.3.7	Programar l'actualització automàtica de l'aplicació Shiny.....	83
9.4	Anàlisi de les dades.....	85

10. Resultats	86
10.1 Pàgina web RoMi Box	86
10.2 Pàgina web Respira .CAT	87
11. Conclusions	90
12. Treball futur.....	91
13. Bibliografia	92

1. Introducció, motivacions, propòsit i objectius del projecte

1.1 Introducció

Actualment cada vegada es pren més atenció a la salut de les persones. En concret, la contaminació de l'aire ha esdevingut un problema molt important, especialment a les grans ciutats, on la gran quantitat de vehicles i indústries provoca una gran contaminació de l'aire.

Segons l'Agència Europea del Medi Ambient (AEMA) a Europa, les emissions de molts contaminants s'han reduït notablement durant les últimes dècades, amb la consegüent millora de la qualitat de l'aire a tota la regió. No obstant això, les concentracions de contaminants atmosfèrics segueixen sent molt elevades i persisteixen els problemes de qualitat de l'aire. Bona part de la població europea viu en zones urbanes, on es sobrepassen els nivells màxims de contaminants en l'aire, comportant greus riscos per a la salut. En especial els contaminants d'ozó, diòxid de nitrogen i el nombre de partícules (PM 10 i PM 2.5) tenen un impacte molt negatiu per a la salut de les persones.

Segons la Unió Europea diversos països van superar un o diversos dels seus límits d'emissió relatius a quatre importants contaminants atmosfèrics al 2010. Per tant, segueix sent important reduir la contaminació atmosfèrica (AEMA, 2017).

Uns d'aquests països va ser Espanya, i a l'actualitat aquest problema encara persisteix. A finals de juliol del 2019, la Comissió Europea va denunciar a Espanya perquè des del 2010 a les ciutats de Barcelona i Madrid s'havien superat els límits de qualitat de l'aire establerts per la Comissió Europea (de Miguel & Planelles, 2019).

1.2 Motivacions

Inicialment, la motivació d'aquest projecte es remunta a dos projectes realitzats durant el primer any del Màster en Enginyeria Informàtica a la Universitat de Girona (UdG) amb el company de classe Miquel Farreras. Per una part, en el primer projecte es va crear un primer prototip d'estació de mesura de qualitat de l'aire amb un sensor i una Raspberry Pi. Posteriorment, en el segon projecte, es va configurar un servidor amb una base de dades i es va programar una primera versió de l'enviament de dades entre l'estació i el servidor.

El projecte que havíem fet li va agradar molt al professor Dr. Nicolas Bocard Marie Maurice, el qual ens va fer un conveni de col·laboració educatiu entre nosaltres i la UdG per ampliar el projecte anomenat RoMi.

Posteriorment, vam participar en la 3^a edició del concurs SmartCAT Challenge 2018 on vam presentar el projecte RoMi i vam obtenir el 3^r premi (SmartCAT Challenge 2018, 2018).



Imatge 1 Entrega del 3r premi SmartCAT Challenge 2018 amb l'Eduard Martin Lineros (Degà de l'Il·lustre Col·legi d'Enginyers en Informàtica de Catalunya)

El premi van ser 2.000 € i l'oportunitat de tornar a fer la presentació a Barcelona a l'exposició Smart City Expo World Congress.



Imatge 2 Imatge dels equips guanyadors al Smart City Expo World Congress

1.3 Propòsits

El propòsit d'aquest projecte per una banda és la creació de un sistema altament escalable i molt econòmic, per rebre i processar totes les dades provinents d'una xarxa d'estacions que mesuren la qualitat de l'aire.

Per un altre part, també s'analitzaran les dades rebudes i s'aplicaran tècniques de mineria de dades per poder fer una predicció de la contaminació de l'aire a curt termini. A més, es crearan eines per poder visualitzar les dades en temps real, i l'històric de les dades agregades, a diferents nivells territorials.

L'estudiant Miquel Farreras s'encarregarà del disseny i el muntatge de les estacions que enviaran les dades en al sistema de recopilació de dades que es dissenyarà i crearà en aquest projecte.

1.4 Objectius

El primer objectiu és la creació de la base de dades i l'API, optimitzant al màxim els recursos econòmics, i oferint al mateix temps un sistema d'alt rendiment i escalabilitat.

El segon objectiu és la visualització de les dades obtingudes en temps real a través d'una pàgina web, representades en mapes amb les estacions geolocalitzades i la qualitat de l'aire de cada una d'elles.

El tercer objectiu és l'anàlisi de les dades, utilitzant les dades provinents de la xarxa de sensors del projecte RoMi.

2. Estudi de viabilitat

2.1 Recursos necessaris

Per a la realització del treball s'utilitzaran diferents serveis de l'empresa Amazon Web Services (AWS) per a muntar tota la infraestructura per rebre i emmagatzemar les dades provinents de la xarxa de sensors del projecte RoMi.

A part de les dades de RoMi, també s'automatitzaran tasques per a obtenir dades de qualitat de l'aire de la xarxa d'estacions de la Generalitat de Catalunya amb una freqüència horària.

Per obtenir les dades meteorològiques s'automatitzaran tasques per a obtenir dades generades per Meteocat amb una freqüència diària.

Per a la creació de l'anàlisi i les dues pàgines webs s'utilitzarà el IDE RStudio i la versió online www.rstudio.cloud. Totes les llicències d'ús són gratuïtes, i en el cas de www.rstudio.cloud és degut a que es troben en fase inicial.

Per a la creació de les dos pàgines webs s'utilitzarà el paquet Shiny d'R que també és totalment gratuït. Tot i que s'ofereix una versió PRO de pagament, per aquest projecte no és necessària i s'utilitzarà la versió estàndard.

Finalment, per controlar les diferents versions de codi de l'anàlisi de dades i de les dues pàgines webs, s'utilitzarà GitHub, que no és de pagament.

2.2 Viabilitat econòmica

El disseny del sistema de recopilació de dades d'aquest treball està pensat i dissenyat perquè amb el mínim pressupost possible ofereixi un alt rendiment i escalabilitat. Això significa que només es pagarà en funció de l'ús que es doni als diferents serveis del sistema.

AWS ofereix durant el primer any als nous clients uns quants serveis de forma gratuïta perquè els clients puguin testejar els serveis. La majoria dels serveis que

s'utilitzaran en el projecte estan dins el període gratuït de AWS. Però hi ha alguns serveis que no ho estan, i s'hauran de pagar a part.

Els serveis de AWS utilitzats en aquest projecte són els següents: AWS API Gateway, AWS Lambda, Amazon RDS per a MySQL, Amazon EC2 i Amazon Cognito. L'explicació del funcionament de cada un d'ells es realitzarà a l'apartat 5.

2.2.1 Cost de AWS API Gateway

Dins el període de prova el primer milió de crides a la API són gratuïtes. Posteriorment, s'ha de pagar 3,5 € per a cada milió de crides. Aquest preu serà per a les primeres 333 milions de crides, que després es redueix a mesura que augmenta el nombre de crides (AWS, 2019a).

Durant la realització del projecte no es té previst superar el milió de crides, motiu pel qual no suposarà cap cost .

2.2.2 Cost de AWS Lambda

Dins el període de prova el primer milió d'execucions o els primers 400.000 segons de temps d'execució són gratuïts. Posteriorment, el primer milió d'execucions continuarà essent gratuït i després per cada milió d'execucions es pagarà a 0,2 € (AWS, 2019b).

Durant la realització del projecte no es té previst superar el milió de crides. Per tant, no suposarà cap cost addicional aquest servei.

2.2.3 Cost de Amazon RDS per a MySQL

El cost d'aquest servei varia en funció de la potència que s'hagi escollit de la base de dades. Per a la realització del projecte es configuraran dues bases de dades MySQL, una que actuarà com a mestre, i l'altre com a esclau. Dins el període de prova, AWS ofereix de forma gratuïta una instància del tipus db.t2.micro, la qual té un cost de 0,018 €/h. El preu pot variar molt en funció del tipus d'instància i el tipus de contracte escollit (preu per hora o per mes) (AWS, 2019c).

En aquest treball s'utilitzaran 2 bases de dades, de les quals una ja estarà inclosa en el període de proves de AWS. Per tant, només s'haurà de pagar una base de dades. El cost de la base de dades extra serà d'aproximadament 10 € al mes.

2.2.4 Cost de Amazon EC2

El cost d'aquest servei també varia en funció de la potència i memòria escollida. Dins el període de proves de AWS ofereixen 750 hores al mes gratuïtes per a una instància del tipus t2.micro. Aquesta instància només té 1 vCPU i 1 GB de memòria RAM (Amazon Elastic Container Service pricing, 2019).

Per aquest projecte, s'utilitzaran dos instàncies EC2. Una del tipus t2.micro i una altre del tipus t3a.small que té 2vCPU i 2 GB de memòria RAM. La primera instància no s'ha de pagar ja que està inclosa dins el període de prova, però la segona instància s'haurà de pagar uns 12 € al mes.

2.2.5 Cost de Amazon Cognito

Aquest servei no està inclòs dins el període de prova. Els primers 50.000 usuaris s'ofereixen de forma gratuïta i indefinida cada mes. A partir dels 50.000 usuaris, la resta valen 0,0055 €/usuari, i en cas que es generin molt més, el preu disminueix progressivament en funció de la quantitat d'usuaris actius mensualment (AWS, 2019e).

Per aquest projecte, no s'espera arribat al límit de 50.000 usuaris en la fase inicial, per tant, no suposarà cap despesa addicional.

2.2.6 Cost dels dominis web

Per aquest projecte es compraran 4 dominis. Un domini .gq, un domini .com i dos dominis .cat.

El preu del domini .gq és gratuït, i el preu de renovació també és gratuït. Aquest domini ha estat registrar a l'empresa Freenom.

El preu dels dominis .com i .cat el primer any és de 1,90 € sense IVA, utilitzant una promoció de l'empresa Nominalia. A partir del segon any, el preu de

renovació del domini .com és de 9,90 € amb IVA i del domini .cat és de 37,40 € amb IVA.

2.2.7 Cost del servidor virtual a IONOS 1&1

A la primera i segona fase del disseny del sistema de recopilació de dades s'utilitzarà un servidor virtual privat contractat a l'empresa IONOS 1&1. Aquest servidor té un cost de 12 € al mes.

2.2.8 Resum de la viabilitat econòmica

Com a resum de la viabilitat econòmica, podem dir que el cost mensual del sistema durant la realització del treball (sense comptar els dominis) ha estat aproximadament de 35 € mensuals. De cares al futur, sense tenir en compte els descomptes del període de prova de AWS, el cost base mensual serà d'aproximadament 45 € mensuals, amb la renovació dels dominis apart. Aquest cost es podria incrementar en funció dels recursos que s'utilitzin.

2.3 Viabilitat tecnològica

Amb les tecnologies actuals aquest treball és completament viable. Per a la realització del treball només serà necessari un ordinador per poder configurar els serveis i programar els reports i les pàgines webs.

2.4 Conclusió

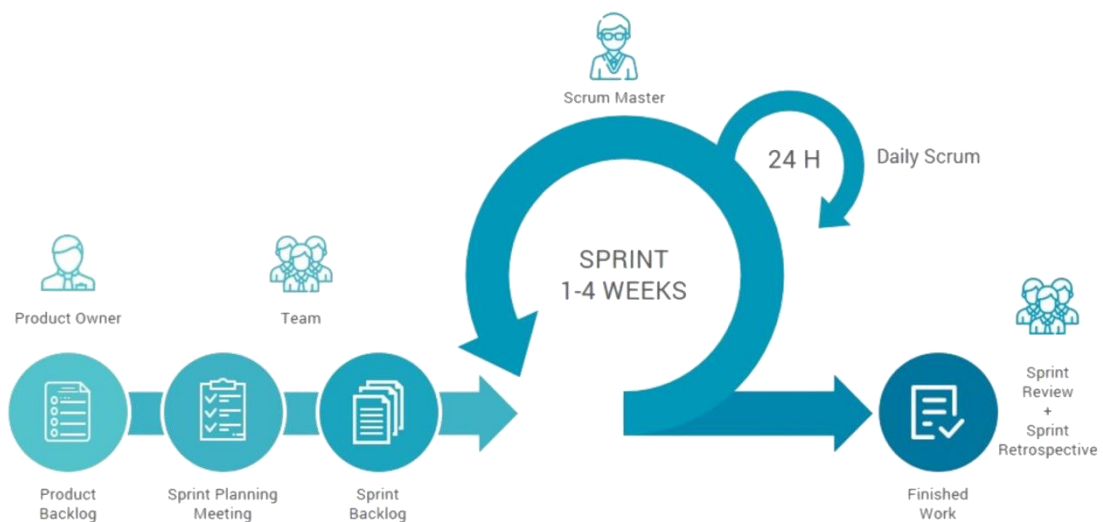
Com a conclusió de l'estudi de viabilitat podem afirmar que és un projecte viable, tant econòmicament com tecnològicament. Al tractar-se de un sistema altament escalable i utilitzar els serveis de AWS, ens permet pagar només per els recursos que es necessitin en cada moment. Per tant, estan correlacionats amb l'ús que es dona al sistema. Com més usuaris (estacions de qualitat d'aire) hi hagi, més car serà el manteniment del sistema.

3. Metodologia

Per a aquest projecte s'ha utilitzat la metodologia àgil anomenada Scrum (Imatge 3). L'objectiu d'aquesta metodologia és desenvolupar i crear un producte en un període determinat, on un equip format per diferents persones treballen conjuntament per arribar a un objectiu comú (Wikipedia, 2019a).

Aquesta metodologia està pensada per adaptar-se als canvis a mesura que es realitza el projecte. D'aquesta forma es pot actuar i reaccionar eficientment a als imprevistos que puguin sorgir al llarg del projecte.

La clau de Scrum es basa en maximitzar la capacitat de l'equip per entregar els productes ràpidament dins dels terminis establerts, i respondre a les necessitats d'última hora.



Imatge 3 Descripció de la metodologia Scrum

La forma de funcionar de Scrum és a base d'sprints. Un sprint (també anomenat iteració) és un període de temps determinat per l'equip al iniciar el projecte, normalment comprés entre 1 setmana i 1 mes.

A l'inici de cada sprint es realitza una reunió per planificar els pròxims objectius a assolir, identificar el treballs que s'ha de realitzar i el temps que suposarà. Al final de cada sprint, s'avaluaran els objectius assolits i es proposaran els nous.

Respecte a aquest projecte, durant algunes etapes (especialment a l'inici) va ser molt necessària la col·laboració i comunicació dels diferents membres de l'equip RoMi per a determinar les tasques a realitzar.

Per a la gestió i assignació de les tasques s'ha utilitzat una eina online anomenada Trello. Aquesta eina permet crear tasques i subtàsques, i assignar-les a una o més persones. Per a cada tasca també es pot especificar en quin estat està, com per exemple: pendent de fer, fent-se o acabada.

Per a la comunicació entre els diferents membres del grup es va crear un grup de Telegram per així fer la comunicació el màxim de fluida i ràpida.

Finalment, per a la gestió del codi de les dos pàgines webs i l'anàlisi de les dades s'ha utilitzat el control de versions Git. Els servidors escollits per guardar el codi del projecte són de GitHub, que permet crear repositoris públics i privats de forma totalment gratuïta. Per a aquest projecte s'han creat dos repositoris privats. Amb l'ús de Git s'aconsegueix tenir un històric de les diferents versions creades, i de cares a un futur pròxim, facilitar la col·laboració amb altres desenvolupadors.

4. Planificació

L'inici del treball va començar amb la idea de fer el projecte i la redacció de la proposta a finals d'octubre del 2018. La data d'entrega del projecte és el 3 de setembre del 2019. Això significa que el projecte s'ha realitzat en uns 10 mesos.

Un cop es va acceptar la proposta, es van analitzar i discutir els passos a seguir amb l'equip de RoMi.

Durant els primers mesos es va acabar de dissenyar la nova estructura, fer cursos de formació de AWS, buscar informació de sistemes altament escalables, solucionar errors en la recepció de les dades provinents de la xarxa d'estacions de RoMi, etc.

També es van crear els primers gràfics per visualitzar les dades rebudes de les diferents estacions i mapes amb les estacions geolocalitzades.

Posteriorment, es va iniciar el disseny de la API i la nova base de dades. Aquest punt era molt important per així poder rebre les dades a través de la API i optimitzar la base de dades. El primer disseny inicial de la base de dades era poc eficient, i hi havia molta informació duplicada, suposant un problema pel rendiment. Al tenir una instància amb pocs recursos, la base de dades es quedava saturada i deixava de funcionar correctament.

L'API i la nova base de dades creada, es va iniciar amb la configuració del servidor Shiny i la creació de la pàgina web Shiny. Inicialment, només es mostraven les dades provinents de la xarxa de les estacions de RoMi. Posteriorment, quan es va programar la recopilació de les dades de les estacions de qualitat d'aire de la Generalitat de Catalunya, també es van afegir aquestes dades en al mapa de la pàgina web Shiny.

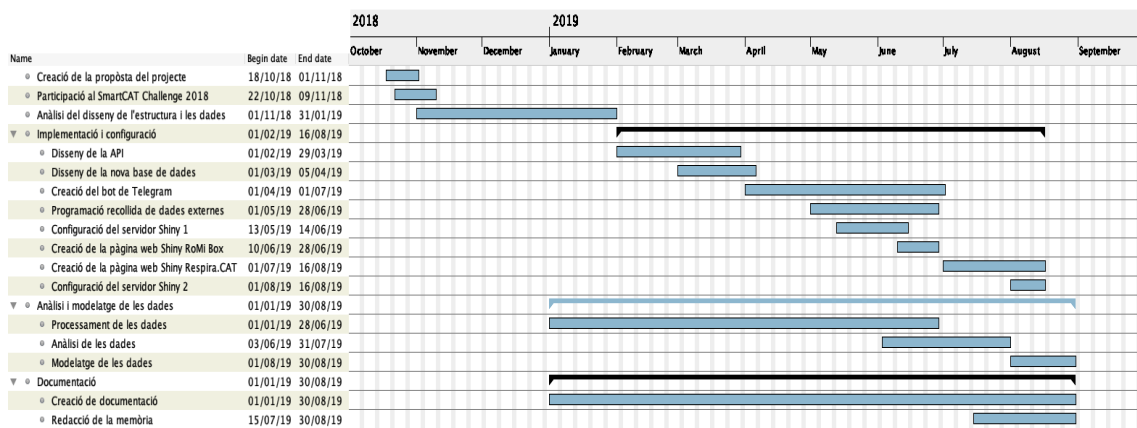
Al mateix temps, es va iniciar el processament de les dades per poder fer l'anàlisi a posteriori.

En els mesos següents es va anar millorant la pàgina web Shiny i ampliant el report de l'anàlisi. A més a més, es van realitzar cursos de formació de AWS, mineria de dades i aprenentatge automàtic.

Durant l'últim mes es va configurar el segon servidor i es va dividir el codi de la pàgina web entre les dades provinents de la xarxa de sensors de RoMi i de la Generalitat de Catalunya. També es va finalitzar l'anàlisi de les dades i la redacció de la memòria.

El temps dedicat a la realització d'aquest projecte s'estima en més de 500 hores, tenint en compte el temps dedicat a cursos de formació per aprendre les noves tecnologies utilitzades en aquest treball.

En el següent diagrama de Gantt, Imatge 4, es pot veure representada visualment la planificació d'aquest projecte.



Imatge 4 Diagrama de Gantt del projecte

5. Marc de treball i conceptes previs

El marc de treball del projecte consisteix en:

- La creació d'un sistema de recopilació de dades altament escalable provinent d'una xarxa d'estacions de qualitat de l'aire.
- L'anàlisi de les dades recopilades i la creació de diferents gràfiques i mapes per poder visualitzar els resultats obtinguts.

Els conceptes previs emprats en aquest projecte són: Amazon Web Services (AWS), Amazon API Gateway, AWS Lambda, Amazon RDS, Amazon EC2, Amazon Cognito, qualitat de l'aire, Zones de la Qualitat de l'Aire (ZQA) i Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica (XVPCA).

5.1 Amazon Web Services (AWS)



Imatge 5 Logotip de Amazon Web Services

Amazon Web Services (AWS) és una empresa del grup Amazon que ofereix serveis de computació al núvol (cloud computing) a través d'Internet (Imatge 5). Actualment, és una de les empreses líders en el seu sector i competeix directament amb altres serveis com Microsoft Azure i Google Cloud Platform. AWS ofereix solucions pràcticament per a totes les necessitats que puguin sorgir per a la computació al núvol, des de diferents tipus de bases de dades SQL i NoSQL fins a solucions d'intel·ligència artificial i blockchain (Wikipedia, 2019b).

Per a aquest projecte s'han utilitzat varis serveis dels que ofereixen. S'ha escollit aquesta empresa perquè a part de ser una de les líders en el seu sector també té una comunitat de desenvolupadors que ofereix molta documentació i suport.

AWS ofereix durant el primer any als nous clients uns quants serveis de forma gratuïta perquè els clients puguin testejar els serveis. La majoria dels serveis utilitzats en el projecte estan a dins el període gratuït de AWS, tot i això, hi ha alguns serveis que no ho estan i s'han hagut de pagar a part. Per altre banda, al crear el compte AWS s'ofereixen 40 € de crèdit per gastar en els seus serveis. A part, si el client és estudiant o professor d'universitat pot obtenir fins a 110 € de crèdit addicional a través del paquet educatiu que ofereix GitHub (GitHub, 2019).

Al moment d'escollir els serveis de AWS, és important saber que AWS té diferents centres arreu del món i ofereix els seus serveis per regions. Això significa que no tots els serveis s'ofereixen a totes les localitzacions, i que els preus varien en funció del lloc.

Els serveis de AWS utilitzats en aquest projecte són els següents: Amazon API Gateway, AWS Lambda, Amazon RDS, Amazon EC2 i Amazon Cognito.

5.1.1 Amazon API Gateway



Imatge 6 Logotip de AWS API Gateway

Amazon API Gateway (Imatge 6) és un servei completament administrat que facilita als desenvolupadors la creació, la publicació, el manteniment, el monitoratge i la protecció d'API a qualsevol escala (AWS, 2019f).

Amb tan sols uns clics a la consola d'administració de AWS es poden crear API REST i API websocket perquè les aplicacions obtinguin accés a les dades dels serveis backend.

API Gateway gestiona l'acceptació i processament de fins a centenars de milers de crides simultànies. També ofereix la possibilitat d'afegir control d'autorització i accés a través de tokens.

Aquest servei s'ha utilitzat per fer el disseny de l' API per enviar i rebre dades provinents de la xarxa d'estacions de qualitat de l'aire del projecte RoMi.

5.1.2 AWS Lambda



Imatge 7 Logotip de AWS Lambda

AWS Lambda (Imatge 7) permet executar codi sense la necessitat de configurar ni administrar cap servidor. També es coneix amb el terme *serverless computing* (AWS, 2019g).

Amb aquest servei únicament s'ha de programar el codi, i aquest ja s'executa sense cap necessitat de configurar un servidor ni el seu manteniment. L'altre avantatge d'aquest servei és que suporta diferents llenguatges de programació, entre ells: Python, Java, Go, C# i Ruby. El problema és que encara no suporta R. Per aquest motiu, totes les funcions Lambda creades per a aquest projecte s'han programat amb Python.

A més a més, Lambda ofereix una gran integració amb els altres serveis d'AWS. L'avantatge d'aquest servei és que només s'ha de pagar per l'ús que es fa.

Aquest servei s'ha utilitzat per a les següents tasques:

- Programació com a backend de totes les crides de la API.
- Creació del bot de Telegram.
- Obtenció de les dades meteorològiques de Meteocat.

- Obtenció de les dades de qualitat de l'aire de la Generalitat de Catalunya.

5.1.3 Amazon RDS



Imatge 8 Logotip de Amazon RDS

Amazon Relational Database Service (Amazon RDS) (Imatge 8) és un servei de AWS que permet configurar i escalar fàcilment una base de dades relacional sense la necessitat de configurar ni administrar cap servidor. També s'encarrega de la gestió de les actualitzacions i les còpies de seguretat.

Amazon RDS està disponible per a diversos tipus d'instàncies de bases de dades (optimitzades per memòria, rendiment o operacions d'escriptura/lectura) i ofereix la possibilitat de configurar una base de dades entre els 6 motors de bases de dades més utilitzats a l'actualitat. Aquests són: Amazon Aurora, PostgreSQL, MySQL, MariaDB , Oracle Database i SQL Server (AWS, 2019h).

Per a aquest projecte, s'ha escollit aquest servei per a la creació de la base de dades per emmagatzemar totes les dades recollides. En concret, s'ha escollit el motor de base de dades MySQL.

5.1.4 Amazon EC2



Imatge 9 Logotip de Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2) (Imatge 9) és un servei de AWS que permet configurar i escalar fàcilment un o varis servidors virtuals. També s'encarrega de la gestió de les actualitzacions del sistema operatiu i les còpies de seguretat (AWS, 2019i).

Amazon EC2 està disponible per a diversos tipus d'instàncies de servidors (optimitzades per memòria, rendiment o operacions d'escriptura/lectura) i ofereix la possibilitat de configurar el sistema operatiu que es vulgui.

Per a aquest projecte s'ha utilitzat aquest servei per a configurar les dues pàgines webs creades: RoMi Box i Respira.CAT.

5.1.5 Amazon Cognito



Imatge 10 Logotip de Amazon Cognito

Amazon Cognito (Imatge 10) és un servei que permet de forma fàcil i senzilla incorporar a les aplicacions: control d'accés, inici de sessió i inscripció. També és un sistema altament escalable que permet la creació de milions de comptes nous i inicis de sessió de forma simultània (AWS, 2019j).

Per a aquest projecte s'ha utilitzat aquest servei per el control d'accés a la API amb l'ús de tokens.

5.2 Qualitat de l'aire

A Catalunya el responsable d'avaluar la qualitat de l'aire és el Servei de Vigilància i Control de l'Aire, a partir de les dades recollides amb la Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica (XVPCA).

L'avaluació de la qualitat de l'aire es fa d'acord amb la legislació vigent (Directiva 2008/50/CE, Reial Decret 102/2011, etc.). Aquesta legislació requereix, en primer lloc, avaluar la qualitat de l'aire per zones. Per això s'ha dividit Catalunya en 15 zones de qualitat de l'aire segons les emissions i les condicions de dispersió.

El fet de dividir el territori ha permès optimitzar el nombre de punts de mesurament, ja que si es mesura dins d'una zona en els diferents entorns que hi ha, la resta de punts de la zona seran equivalents als nivells mesurats en alguna de les estacions de mesura (Generalitat de Catalunya, 2019a).

5.2.1 Zones de Qualitat de l'aire (ZQA)

Les Zones de Qualitat de l'Aire (ZQA) són zones creades en el territori de Catalunya de tal manera que cada una d'elles sigui representativa de la qualitat de l'aire de tota l'àrea que la comprèn (Generalitat de Catalunya, 2019b).

Per tenir una qualitat d'aire semblant s'ha de tenir en compte que la superfície que la forma sigui homogènia respecte a l'orografia, la climatologia, la densitat de població i el volum d'emissions industrials i de trànsit.

A Catalunya s'han creat un total de 15 zones de qualitat de l'aire. En la Imatge 11 es poden veure les diferents ZQA que hi ha a Catalunya.



Imatge 11 Mapa de les Zones de Qualitat de l'Aire a Catalunya

Les prediccions de qualitat de l'aire que realitza diàriament la Generalitat de Catalunya són a nivell de ZQA.

5.2.2 Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica (XVPCA)

La Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica (XVPCA) és un conjunt d'estacions que detecten els diferents nivells d'immissió dels principals contaminants. Va ser creada per la Llei 22/1983, de 21 de novembre, definida per l'Ordre de 20 de juny de 1986 i actualment està adscrita administrativament al Departament de Territori i Sostenibilitat de Catalunya. Té una estructura piramidal amb la base formada pels punts de mesurament i el vèrtex amb el Centre Receptor i Coordinador de Dades (Generalitat de Catalunya, 2019c).

Aquesta xarxa està formada per estacions de mesurament manuals i automàtiques. Les estacions manuals són les que necessiten la intervenció d'algú per el càlcul de la contaminació. Les estacions automàtiques realitzen els mesuraments automàticament sense la intervenció humana.

Les dades obtingudes en aquest treball provenen de les 76 estacions de mesurament automàtiques de la XVPCA.

La Generalitat de Catalunya ha creat una pàgina web per poder visualitzar l'estat de contaminació de l'aire en temps real (Generalitat de Catalunya, 2019d).

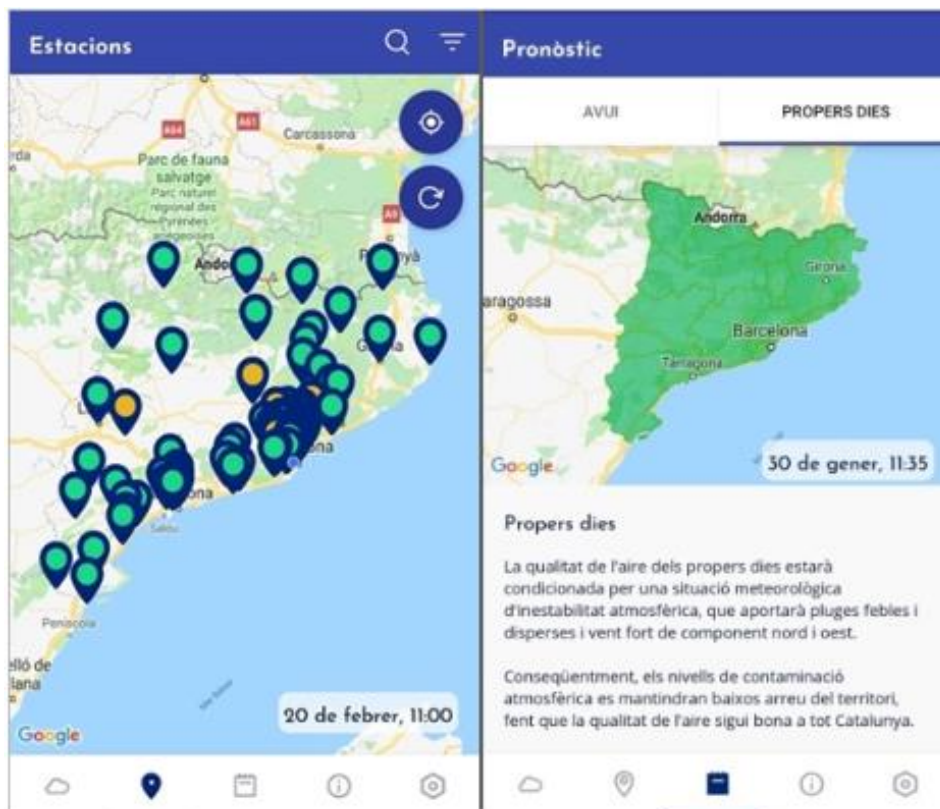
En la Imatge 12 es pot veure el mapa de les estacions automàtiques de la XVPCA. El color de cada una d'elles indica la qualitat de l'aire. En aquesta pàgina web també es pot veure l'històric de les dades per a cada estació fins a un màxim de 9 dies anteriors al dia actual.



Imatge 12 Mapa de les estacions de la XVPCA de la web de Gencat

Per altra banda, la Generalitat de Catalunya també ha creat una aplicació mòbil anomenada Aire.Cat que també permet visualitzar l'estat actual de les estacions de la XVPCA, aplicar filtres per els contaminants O_3 , NO_2 i PM_{10} , i també ofereix la predicció del dia actual i la del dia següent a nivell de ZQA. Per contra, no ofereix la possibilitat de veure l'històric de les dades de cada estació a diferència de la versió web (Generalitat de Catalunya, 2019e).

En la Imatge 13 es pot veure a l'esquerra una captura de pantalla de l'aplicació amb el mapa de totes les estacions a Catalunya. A la dreta, es pot veure la predicció de la qualitat de l'aire del dia següent amb els colors de la qualitat per cada ZQA i una petita descripció.



Imatge 13 Captures de pantalla de l'aplicació Aire.Cat

6. Requisits del sistema

6.1 Requisits bàsics

El disseny del sistema de recopilació de dades ha de permetre:

- Recopilar les dades provinents d'una xarxa d'estacions de qualitat d'aire.
- Recopilar dades provinents de fonts externes.
- Consultar les dades de contaminació en temps real.
- Consultar l'històric de les dades de cada estació.

El disseny de la API ha de permetre:

- Obtenir, crear, modificar i eliminar mesures.
- Obtenir, crear, modificar i eliminar usuaris.
- Obtenir, crear, modificar i eliminar estacions.

6.2 Requisits específics

Com a requisits específics tenim els següents:

- Crear un sistema de recopilació de dades altament escalable amb un cost de manteniment mínim.
- Creació d'un sistema per visualitzar l'estat de les estacions en temps real
- Recopilar dades provinents de noves estacions.
- Visualitzar els valors de contaminació en temps real de la xarxa d'estacions de qualitat d'aire de RoMi.
- Visualitzar els valors de contaminació en temps real de la xarxa d'estacions de qualitat d'aire de la Generalitat de Catalunya.
- Visualitzar l'històric de les dades de contaminació a nivell d'estació i diferents nivells regionals de les estacions de la Generalitat de Catalunya.

7. Estudis i decisions

7.1 Estudi dels serveis de AWS

Per a la realització d'aquest treball s'ha utilitzat AWS per a la construcció del sistema de recopilació de dades altament escalable. Com ja s'ha explicat anteriorment, el motiu pel qual s'ha escollit AWS ha sigut perquè AWS és una de les empreses líders i pioneres en aquest sector. A part, també ofereix un període de prova per provar els seus productes i crèdit addicional per provar els productes no inclosos en el període de prova.

És important saber que AWS té diferents centres arreu del món i ofereix els seus serveis per regions. Això significa que no tots els serveis s'ofereixen a totes les localitzacions, ni tampoc que els seus preus són els mateixos per a totes les regions.

En aquest treball s'han escollit els servidors de AWS localitzats a Irlanda, ja que són els que ofereixen més serveis a un menor preu a Europa. Els preus dels serveis d'aquesta regió s'han descrit anteriorment a l'apartat 2.

7.2 Estudi de l'escalabilitat de la base de dades

Un sistema de recopilació de dades altament escalable significa que la base de dades s'ha de poder escalar, de tal forma que pugui acceptar peticions d'escriptura provinents de milers d'estacions, i peticions de lectures provinents de milers de clients.

Per aconseguir això, Amazon RDS permet escalar tant horitzontalment com verticalment la base de dades (Yap, 2016).

Per escalar verticalment la base de dades simplement s'ha d'assignar una instància amb molta més potència de càlcul i més memòria RAM. AWS ofereix la possibilitat d'escollir entre 18 tipus diferents d'instàncies optimitzades en funció de les necessitats de càlcul o memòria.

En la Imatge 14 es pot veure una part dels diferents tipus de instàncies que s'ofereixen.

Modify DB Instance: Im1ks7xpixmry6w

Instance Specifications

DB Engine Version	MySQL 5.6.27 (default) ▼
DB Instance Class	db.t2.small — 1 vCPU, 2 GiB RAM ▼
Multi-AZ Deployment	db.t2.micro — 1 vCPU, 1 GiB RAM db.t2.small — 1 vCPU, 2 GiB RAM db.t2.medium — 2 vCPU, 4 GiB RAM db.t2.large — 2 vCPU, 8 GiB RAM
Storage Type	db.m4.large — 2 vCPU, 8 GiB RAM db.m4.xlarge — 4 vCPU, 16 GiB RAM db.m4.2xlarge — 8 vCPU, 32 GiB RAM db.m4.4xlarge — 16 vCPU, 64 GiB RAM db.m4.10xlarge — 40 vCPU, 160 GiB RAM
Allocated Storage*	db.m3.medium — 1 vCPU, 3.75 GiB RAM db.m3.large — 2 vCPU, 7.5 GiB RAM db.m3.xlarge — 4 vCPU, 15 GiB RAM db.m3.2xlarge — 8 vCPU, 30 GiB RAM
Settings	db.r3.large — 2 vCPU, 15 GiB RAM db.r3.xlarge — 4 vCPU, 30.5 GiB RAM db.r3.2xlarge — 8 vCPU, 61 GiB RAM db.r3.4xlarge — 16 vCPU, 122 GiB RAM db.r3.8xlarge — 32 vCPU, 244 GiB RAM
DB Instance Identifier	db.m2.xlarge — 2 vCPU, 17.1 GiB RAM db.m2.2xlarge — 4 vCPU, 34 GiB RAM
New Master Password	
Network & Security	
Security Group	

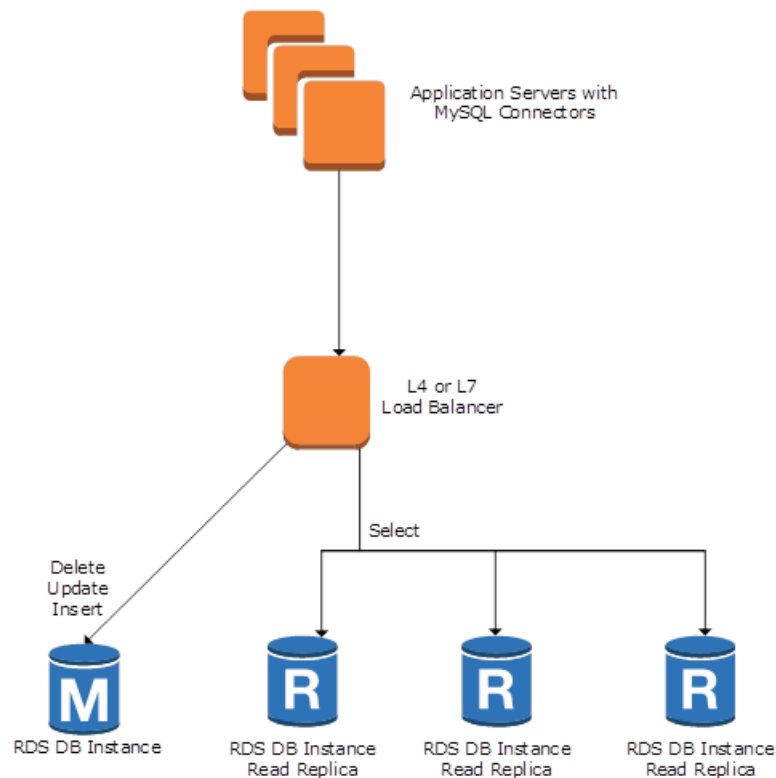
Imatge 14 Diferents tipus de instància del servei Amazon RDS.

Per a aquest projecte, no s'ha necessitat escalar horitzontalment, ja que el volum de dades no ho requereix. Tot i així, com que el sistema s'ha dissenyat sempre tenint en ment el concepte d'escalabilitat, amb aquest servei de cares a un futur serà molt fàcil escalar la base de dades horitzontalment.

Per escalar horitzontalment la base de dades, Amazon RDS ofereix la possibilitat de crear repliques de la base de dades principal de forma molt fàcil i senzilla. Amb els motors de base de dades MySQL, PostgreSQL i MariaDB es poden configurar fins a un màxim de 5 repliques.

Pel motor de base de dades Amazon Aurora es poden configurar fins a 15 repliques. Aquest motor de base de dades és pròpia de AWS i és una versió millorada i optimitzada de PostgreSQL.

En la Imatge 15 es pot veure un exemple de una base de dades amb 3 rèpliques i un balancejador de càrrega entre les aplicacions i la base de dades.



Imatge 15 Exemple de base de dades amb 3 rèpliques.

Per crear les rèpliques (esclaus), Amazon RDS crea una instantània de la base de dades principal (mestre) i posteriorment, quan es realitza un canvi a la base de dades principal, automàticament rèplica aquest canvis a les rèpliques configurades.

En el cas que la base de dades mestre falli, llavors es pot configurar que la base de dades esclau es converteixi en mestre.

Per a aquest projecte, s'ha configurat una única base de dades rèplica per rebre les peticions de consulta provinents de les dues pàgines webs creades.

Tot i que pel volum de dades actual no és necessari configurar cap rèplica, s'ha configurat per comprovar que realment funciona. D'aquesta forma es pot oferir un sistema altament escalable per quan es processin moltes més dades en un futur pròxim.

7.3 Origen de les dades de la qualitat de l'aire

Per obtenir les dades de qualitat de l'aire de fonts externes, es van extreure del portal de dades obertes de la Generalitat de Catalunya. El conjunt de dades s'anomena "Dades d'immissió dels punts de mesurament de la Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica". Aquest dataset conté dades des de l'any 1991 fins a l'actualitat, i la seva freqüència d'actualització és diària. Està format per 5,23 milions de files i 68 columnes (Dades Obertes Catalunya, 2019).

A més a més, té l'avantatge que es proporciona una API per als desenvolupadors que facilita molt la consulta de les dades (Departament de Territori i Sostenibilitat, 2019).

Tot i això, el problema principal d'aquest dataset és que només s'actualitza amb una freqüència diària. Per a l'anàlisi de les dades no hi havia cap problema, però si es volia mostrar una pàgina web amb la qualitat de les dades en temps real llavors aquest no era el dataset correcte.

Posteriorment es va investigar, en el codi font de l'aplicació web de Gencat, l'origen de les dades que mostren en el mapa de la qualitat de l'aire en temps real (explicat a l'apartat 5). També es van analitzar les connexions que realitzava l'aplicació mòbil Aire.Cat per veure des d'on obtenia les dades de qualitat de l'aire que mostrava a l'aplicació.

En tots dos casos es va trobar el mateix origen de dades. Aquest són uns fitxers que es generen diàriament i que tenen un període de vida de com a màxim 9 dies. És per aquest motiu que la pàgina web de Gencat només permet veure l'històric de les dades de fins a 9 dies anteriors al dia actual.

L'última actualització de dades sempre es troba en el següent fitxer:

http://guiaweb.gencat.cat/web/shared/json/quaire/estatCQA_.json

Llavors, si es vol mirar l'històric dels dies anteriors (fins a 9 dies anteriors al dia actual) s'ha d'afegir al final de la URL la data i hora en el format YYYYMMDDHH.

Per exemple, si es vol obtenir les dades del dia 22/08/2019 a les 12:00h s'ha de escriure la següent URL:

http://guiaweb.gencat.cat/web/shared/json/quaire/estatICQA_2019082212.json

Al consultar aquests fitxers s'obtindrà un fitxer en format JSON amb totes les mesures de totes les estacions de mesurament automàtic de la XVPCA.

En cas de voler obtenir les dades de pronòstic del dia d'avui es pot fer a través de la següent URL:

http://guiaweb.gencat.cat/web/shared/json/quaire/pronostic_AVUI.json

I per obtenir la predicció del dia següent:

http://guiaweb.gencat.cat/web/shared/json/quaire/pronostic_DEMA.json

En els dos fitxers de predicció hi ha una petita descripció respecte el pronòstic de la qualitat de l'aire, així com totes les ZQA en format GeoJSON amb el color de la qualitat de cada una de les zones.

El temps de generació de cada fitxer és d'entre 20 i 30 minuts. Això significa que les dades de la qualitat d'aire de una hora en concret, estaran disponibles al cap de 20-30 minuts després de l'hora de mesurament.

Per a aquest projecte, s'ha creat una tasca automàtica amb una freqüència d'execució horària al minut 30 de cada hora. Aquest procés es connecta als servidors de Gencat i obté les dades de la hora en la qual s'està executant.

Pot ser que en algun cas el fitxer no s'hagi generat durant els primers 20-30 minuts. Per solucionar aquest problema, també s'ha programat una altre tasca que s'executa cada 2 dies que obté totes les dades dels últims 3 dies i les torna a carregar a la base de dades. En cas que faltin dades a alguna hora, aquest procés s'encarregarà d'afegir-les a posteriori.

La programació d'aquestes tasques s'explica a l'apartat 9.

7.4 Origen de les dades meteorològiques

Per obtenir les dades meteorològiques de fonts externes inicialment es va trobar una primera opció a la pàgina web OpenWeather (OpenWeather, 2019).

Aquest pàgina web ofereix una API i diferents plans de pagament. El pla gratuït només ofereix 60 crides com a màxim per minut. Es van realitzar algunes proves per obtenir les dades meteorològiques però el format i la freqüència de les dades obtingudes no era del tot adequada per a aquest projecte.

Posteriorment, es va provar d'obtenir les dades de l'Agència Estatal de Meteorologia (AEMET) a través de l'API que ofereixen. Es va crear un compte al sistema i llavors es va obtenir una clau per poder fer peticions a la API (AEMET OpenData, 2019).

Tot i ser una API gratuïta, les dades que s'ofereixen tenen una freqüència diària i no horària. Per aquest motiu es va descartar.

Llavors, es van consultar els serveis que ofereix Meteocat (Servei Meteorològic de Catalunya) i es va trobar que també ofereixen una API per a consultar les dades meteorològiques de Catalunya (Meteocat, 2019).

L'accés a l'API és gratuït per als ciutadans que volen fer un ús personal de les dades que genera l'SMC, estudiants i/o centres de recerca, i per a l'administració. L'únic problema és que està limitat a un màxim de 750 peticions al mes. En cas de voler realitzar més peticions llavors s'ha de pagar.

En aquest cas, sí teníem les dades que volíem per al projecte, el problema era que amb 750 peticions al mes només podíem obtenir dades d'una única estació amb una freqüència horària.

Finalment, analitzant el codi font dels Ginys que ofereix Meteocat, es van trobar els fitxers a on es generen diàriament tots els valors meteorològics per a tots els municipis de Catalunya amb una freqüència horària.

Els valors meteorològics que es poden obtenir en aquests fitxers són:

- Estat del cel
- Temperatura
- Temperatura xafogor
- Precipitació acumulada
- Humitat relativa
- Direcció del vent
- Velocitat del vent

Cal remarcar que aquestes dades no són els valors reals mesurats, sinó una predicció molt acurada dels valors reals. Amb el mateix mètode també es pot obtenir la predicció meteorològica dels dos dies següent a l'actual de tots els municipis de Catalunya.

La base de la URL per obtenir les dades és la següent:

<http://static-m.meteo.cat/ginys/>

Llavors s'ha d'afegir la resta de la URL en funció del tipus de valor meteorològic que es vulgui obtenir.

Per exemple, en el cas de voler obtenir els valors de la temperatura la URL seria la següent:

models/postProcessament/variables/temperature/intervals/{INTERVAL}/temperature-{DIA}_{INTERVAL}h.json

A l'anterior URL s'ha de substituir el camp {INTERVAL} per el valor 1, això significa un interval de 1 hora. Llavors el camp {DIA} s'ha de substituir en funció del dia que es volen les dades. Per exemple, si es volen les dades d'avui s'ha de posar un 1, les de demà un 2 i les de passat demà un 3.

La programació d'aquestes tasques s'explica a l'apartat 9.

7.5 Anàlisi de les dades de la qualitat de l'aire

L'anàlisi de dades inicialment estava pensat per utilitzar les dades provinents de la xarxa de sensors del projecte RoMi. Degut a varis motius, al final s'ha realitzat l'anàlisi de les dades de qualitat de l'aire utilitzant les dades provinents de la xarxa d'estacions de qualitat de l'aire de la Generalitat de Catalunya.

El motius pels quals no s'ha realitzat l'anàlisi amb les dades de la xarxa de sensors de RoMi són:

- Nombre d'estacions molt limitat i variabilitat entre les dades molt baixa ja que totes (a excepció d'una), estan a Girona ciutat.
- Únicament es mesuren els contaminants PM 2.5 i PM 10.
- El fet que totes estiguin a Girona fa molt difícil trobar una correlació entre els diferents nivells de qualitat d'aire i les dades atmosfèriques, ja que totes les estacions són molt pròximes entre elles.
- Dades sense filtrar i sense cap comprovació.
- Freqüència de mesurament irregular i molt elevada.

Els motius pels quals s'ha decidit fer l'anàlisi utilitzant les dades de la xarxa d'estacions de qualitat d'aire de la Generalitat de Catalunya són:

- Nombre d'estacions molt gran i cobertura a tot el territori de Catalunya.
- Alta variabilitat entre les dades degut a les diferències geogràfiques i climatològiques.
- Dades comprovades i validades prèviament amb sistemes automàtics.
- Dades amb una freqüència horària regular.
- Es mesuren molts tipus de contaminants com per exemple: ozó, PM 10, diòxid de nitrogen i benzè..
- El fet de tenir cobertura a tot el territori de Catalunya és molt millor per correlacionar la qualitat d'aire amb les dades climatològiques.

8. Anàlisi i disseny del sistema

8.1 Disseny del sistema de recopilació de dades (SRD)

El disseny del sistema de recopilació de dades (SRD) s'ha realitzat en 4 etapes descrites a continuació.

8.1.1 Primera etapa del disseny del sistema de recopilació de dades

La configuració inicial del projecte es va realitzar utilitzant un servidor virtual (VPS) contractat a l'empresa 1&1. En aquest servidor virtual es va instal·lar el sistema operatiu Ubuntu amb Plesk. Plesk és una plataforma d'allotjament web que permet administrar i gestionar diferents pàgines webs a través de un panell de control.

En aquesta primera etapa es va contractar el domini www.romi.gg, es va crear un lloc web i una primera versió de base de dades per recollir les dades provinents de les estacions de la xarxa de RoMi.

En aquesta primera versió de base de dades només es va crear una única taula anomenada *punts_gps* per recollir totes les dades.

En la imatge 16 es pot veure el disseny de la primera etapa.

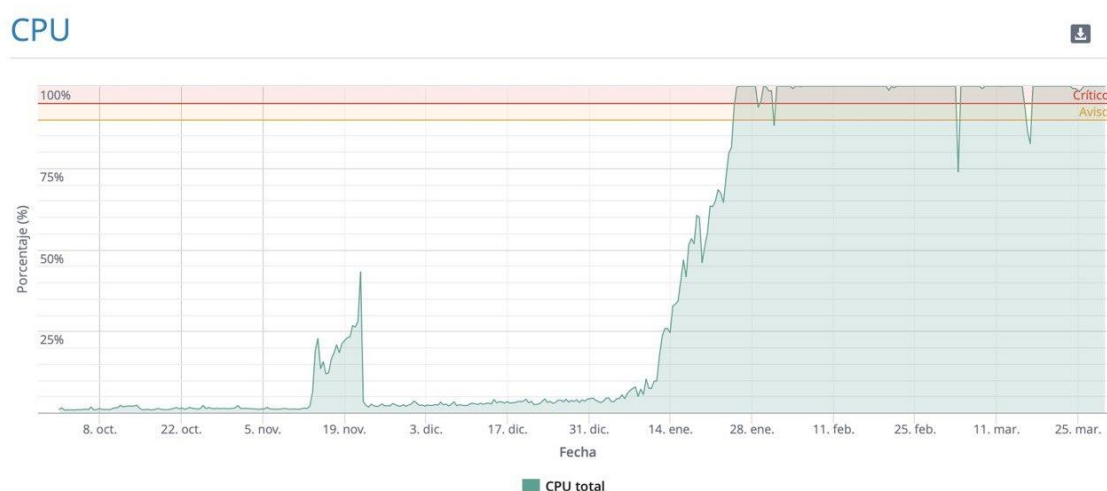


Imatge 16 Disseny de la primera etapa del SRD.

En aquesta etapa, quan la xarxa d'estacions es va a incrementar a més de 5 estacions, es van començar a veure els primers problemes de rendiment de la base de dades i del servidor. També es va detectar que el sistema d'enviament

de dades proposat inicialment no era gens eficient per a les consultes excessives a la base de dades.

En la Imatge 17 es pot veure la gràfica del rendiment de la CPU del servidor, des del inici del projecte a finals d'octubre del 2018 fins a finals de març del 2019. Com es pot observar, a la segona quinzena del mes de gener del 2019 es van afegir més estacions a la xarxa d'estacions de RoMi, i això va repercutir en el rendiment del servidor fins a bloquejar-lo completament.



Imatge 17 Gràfica d'ús de la CPU del servidor VPS utilitzat a la primera etapa del projecte.

Un dels motius pels quals es va col·lapsar el servidor, a part de la ineficiència dels protocols definits, va ser el fet de tenir un servidor amb 1vCPU i 1 GB de RAM per al sistema operatiu i la base de dades.

8.1.2 Segona etapa del disseny del sistema de recopilació de dades

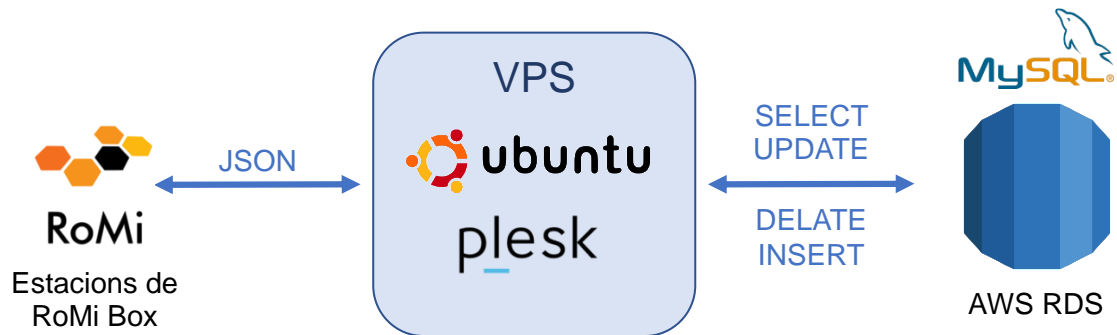
A la segona etapa del disseny era primordial solucionar per una banda el protocol d'enviament de dades, per fer-lo més eficient, i per l'altra solucionar els problemes de rendiment del servidor.

Tenint en ment el concepte d'escalabilitat, es va decidir separar la base de dades del servidor. D'aquesta forma, no s'utilitzen els mateixos recursos per executar el servidor i la base de dades.

El fet de desacoblar la base de dades del servidor permet escalar la base de dades posteriorment sense dependre de cap altre servidor ni servei.

Per a la configuració de la base de dades es va escollir el servei de AWS anomenat AWS RDS, i en concret es va configurar el motor de base de dades MySQL. Com ja s'ha explicat anteriorment a l'apartat 7, aquest servei permet escalar la base de dades de forma horitzontal i vertical.

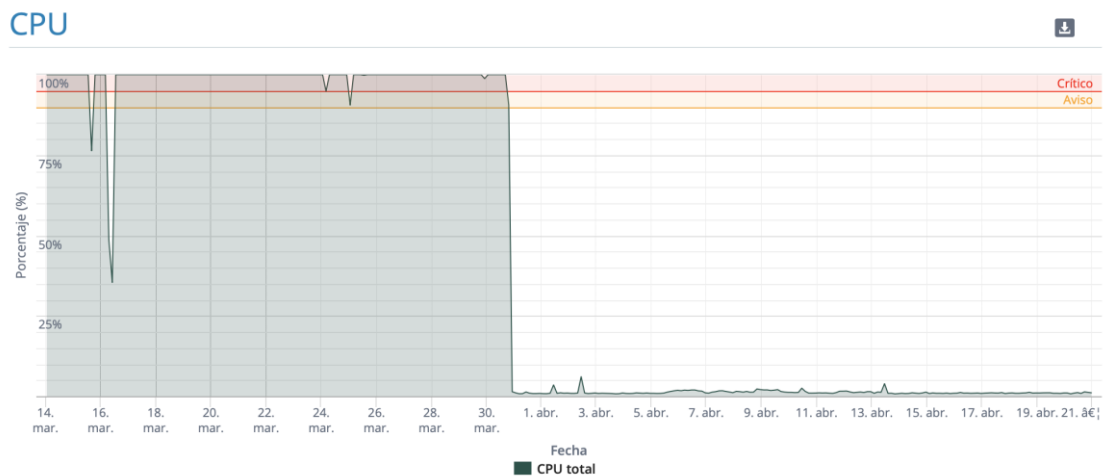
El disseny de la segona etapa es troba representat en la Imatge 18.



Imatge 18 Disseny de la segona etapa del SRP.

Al separar la base de dades del servidor, i millorar el protocol d'enviament de les dades per fer-lo més eficient, es va notar una millora en el rendiment del servidor VPS.

En la Imatge 19 es pot veure el rendiment del servidor VPS, i en concret, el moment en el qual es va configurar la base de dades externa a principis del mes d'abril del 2019.



Imatge 19 Gràfica d'ús de la CPU del servidor VPS utilitzat a la primera etapa del projecte.

A la segona etapa, el sistema ja estava força estable per funcionar amb un nombre petit d'estacions.

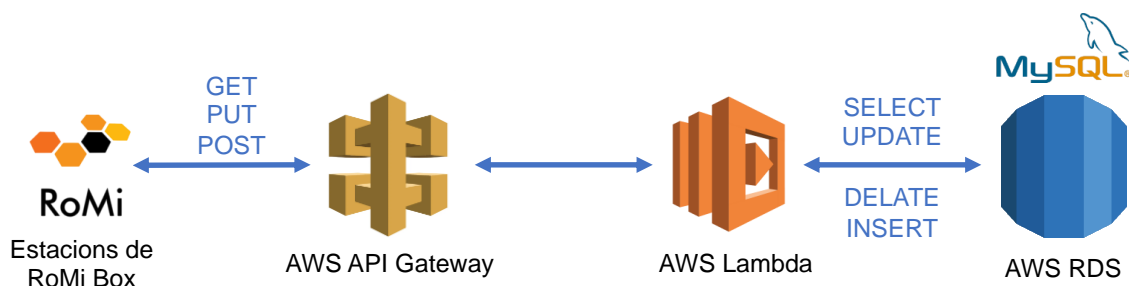
8.1.3 Tercera etapa del disseny del sistema de recopilació de dades

Com que l'objectiu del sistema era que fos altament escalable i que es pogués adaptar a nous sensors, es van incorporar dues millores. Per una banda, la creació d'una API REST per a l'enviament de les dades, i per l'altre, un nou disseny de la base de dades més dinàmic i que acceptés noves mesures de nous sensors. Fins a l'etapa 2, la base de dades només estava pensada per rebre dades dels contaminants PM2.5 i PM10.

Per a la creació de l'API REST es van estudiar diferents alternatives, però la millor opció va ser crear-la utilitzant el servei que ofereix AWS anomenat AWS API Gateway. Aquest servei a part de permetre crear una API REST de forma molt fàcil, també suporta milers de crides simultànies, i només s'ha de pagar per el nombre de consultes realitzades.

El backend de cada crida de l'API es va programar utilitzant el servei AWS Lambda. Aquest servei, tal i com s'ha explicat anteriorment, ofereix la possibilitat de programar petits talls de codi sense la necessitat de configurar cap servidor i només es paga en funció del nombre d'execucions realitzades.

El disseny de la tercera etapa es troba representat en la Imatge 20.



Imatge 20 Disseny de la tercera etapa del SRD.

Al final de la tercera etapa el projecte ja tenia una primera versió d'un sistema altament escalable totalment serverless.

8.1.4 Quarta etapa del disseny del sistema de recopilació de dades

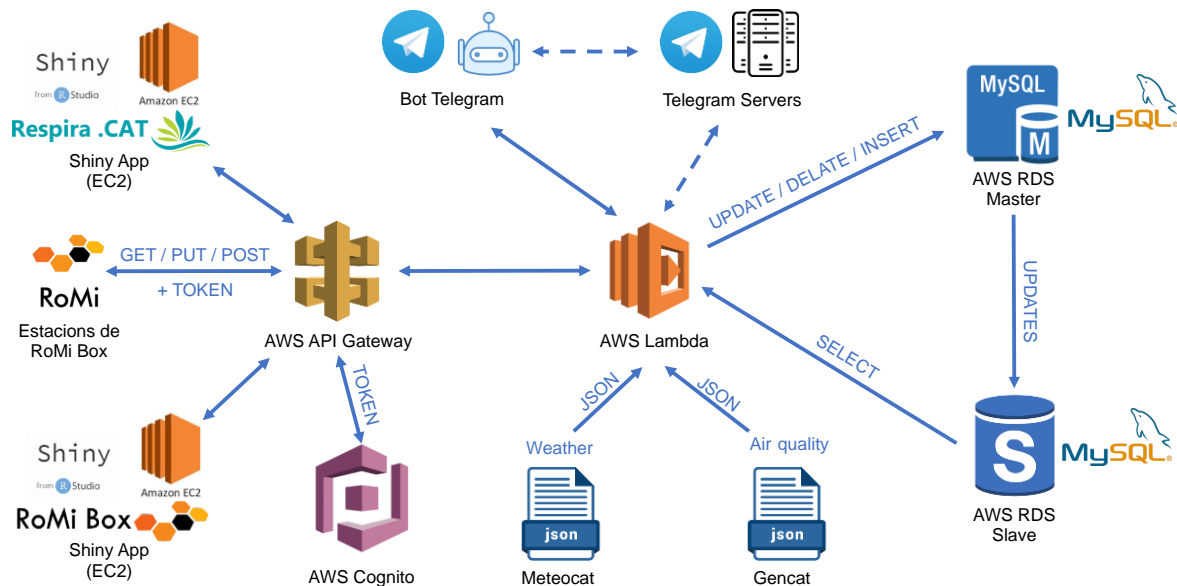
Tot i finalitzar la tercera etapa amb un sistema altament escalable, el sistema encara es podia millorar.

Un dels primers problemes detectats va ser la falta de seguretat a l'accedir a l'API. Per aquest motiu es va afegir el servei de AWS anomenat AWS Cognito, amb el qual es generen claus d'autenticació (tokens). Gràcies a l'ús dels tokens es pot controlar el nombre de crides realitzades per cada client de l'API, ja siguin estacions o altres aplicacions.

En aquesta etapa es van programar les funcions per obtenir les dades de qualitat de l'aire i meteorològiques provinents de la Gencat i Meteocat, respectivament.

També es va crear un bot de Telegram per veure l'estat de les estacions en temps real. Es van registrar els dominis www.romibox.com i www.respira.cat, i es van configurar i crear les dos pàgines webs Shiny.

En la Imatge 21 es pot veure representat el disseny de la quarta etapa.



Imatge 21 Disseny de la quarta etapa del SRD.

Tot això suposava per a la base de dades moltes peticions de consultes provinents del bot de Telegram i les dos pàgines webs (clients), així com moltes peticions d'escriptura provinents de la xarxa d'estacions de RoMi. La solució

radicava en escalar verticalment la base de dades assignant més recursos, o en escalar horitzontalment la base de dades creant una o més repliques.

En aquest cas, es va optar per crear una rèplica de lectura. D'aquesta forma, totes les peticions de lectura provinents de les dos pàgines web Shiny i del bot de Telegram realitzarien les consultes a la base de dades rèplica. D'altra banda, totes les estacions de qualitat de l'aire realitzarien les peticions d'escriptura a la base de dades mestre.

A la quarta etapa ja es va tenir un sistema altament escalable, segur i preparat per a processar les peticions provinents de milers d'estacions i de les aplicacions que consulten les dades.

8.2 Disseny de la base de dades

El disseny de la base de dades es va fer tal manera que permetés afegir nous tipus de mesures de forma dinàmica sense haver de modificar l'estructura de la base de dades. A més, el disseny havia de permetre acceptar diferents mesures de diferents estacions. També es va crear una taula d'usuaris pensant en un futur pròxim, on els clients que vulguin tenir la seva pròpia estació de qualitat de l'aire pugin veure només les dades de les seves estacions.

En la Imatge 22 es mostra el disseny de les taules, els camps que contenen, tipus de dada, les claus primàries i les claus foranes.



Imatge 22 Disseny de la base de dades.

8.3 Disseny de la API REST

Al moment de dissenyar l'API REST es va tenir present el disseny de la base de dades.

Per a cada endpoint (URL) de la API REST hi ha definides 4 accions:

- **GET:** Per obtenir dades.
- **POST:** Per crear noves dades.
- **PUT:** Per editar les dades.
- **DELETE:** Per eliminar les dades.

També s'ha tingut en compte el control de versions de la API, motiu pel qual tots els endpoints comencen amb el codi de la versió API.

Per al disseny de la API s'han identificat 3 recursos:

- **Usuaris:** Es poden crear, modificar, llegir i eliminar. Un usuari pot tenir una o més d'una estació.
- **Estacions:** Es poden crear, modificar, llegir i eliminar. Cada estació té les seves mesures.
- **Mesures:** Es poden crear, modificar, llegir i eliminar. Cada mesura pertany a una estació.

Els mètodes definits per a la funció GET són els següents (Imatge 23):

PATH	GET
/ {api-version} /users/	Obtenir llista d'usuaris
/ {api-version} /users/ {user-id} /	Obtenir informació de l'usuari
/ {api-version} /users/ {user-id} /stations/	Obtenir llista d'estacions de l'usuari
/ {api-version} /stations/	Obtenir llista d'estacions
/ {api-version} /stations/ {station-id} /	Obtenir informació de l'estació
/ {api-version} /stations/ {station-id} /measures/	Obtenir llista de mesures de l'estació
/ {api-version} /stations/ {station-id} /measures/ {measure-id} /	Obtenir informació de la mesura

Imatge 23 Definició de l'API REST pel mètode GET

Els mètodes definits per a la funció POST són el següents (Imatge 51):

PATH	POST
/ {api-version} /users/	Crear nou usuari
/ {api-version} /users/ {user-id} /	
/ {api-version} /users/ {user-id} /stations/	
/ {api-version} /stations/	Crear nova estació
/ {api-version} /stations/ {station-id} /	
/ {api-version} /stations/ {station-id} /measures/	Crear nova mesura
/ {api-version} /stations/ {station-id} /measures/ {measure-id} /	

Imatge 24 Definició de l'API REST pel mètode POST

Els mètodes definits per a la funció PUT són el següents (Imatge 52):

PATH	PUT
/ {api-version} /users/	
/ {api-version} /users/ {user-id} /	Actualitzar la informació del usuari
/ {api-version} /users/ {user-id} /stations/	
/ {api-version} /stations/	
/ {api-version} /stations/ {station-id} /	Actualitzar la informació de l'estació
/ {api-version} /stations/ {station-id} /measures/	
/ {api-version} /stations/ {station-id} /measures/ {measure-id} /	Actualitzar la informació de la mesura

Imatge 25 Definició de l'API REST pel mètode PUT

Els mètodes definits per a la funció DELETE són el següents (Imatge 53):

PATH	DELETE
/ {api-version} /users/	
/ {api-version} /users/ {user-id} /	Eliminar usuari
/ {api-version} /users/ {user-id} /stations/	
/ {api-version} /stations/	
/ {api-version} /stations/ {station-id} /	Eliminar estació
/ {api-version} /stations/ {station-id} /measures/	
/ {api-version} /stations/ {station-id} /measures/ {measure-id} /	Eliminar mesura

Imatge 26 Definició de l'API REST pel mètode DELETE

Els endpoints amb la descripció de color vermell significa que no estan definits per el mètode, ja que el disseny de la API no ho permet. La implementació de la API s'explicarà en el següent apartat.

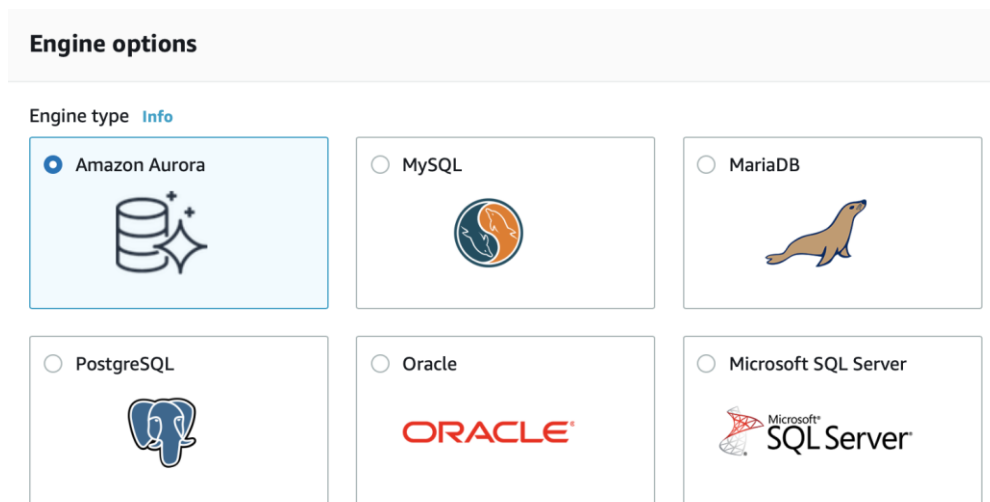
9. Implementació

9.1 Configuració de la base de dades

Per començar la configuració de la base de dades primer s'ha d'anar al servei de Amazon RDS a través del panell de control de AWS. Allà s'haurà de pulsar el botó *Create database*. Llavors s'obrirà una pantalla nova amb tots els paràmetres necessaris per configurar una base de dades.

Primer de tot, AWS ofereix la opció de fer una configuració estàndard o una configuració més fàcil amb menys opcions per configurar. Per aquest projecte es va escollir la configuració estàndard.

Posteriorment, s'ha d'escollir el motor de base de dades que es vol executar (Imatge 27).



Imatge 27 Tipus de motos de base de dades per configurar a Amazon RDS

Per a aquest projecte s'ha escollit el motor de base de dades MySQL i en concret la versió MySQL 5.6.43.

El següent que s'ha de configurar és l'identificador de la instància de la base de dades. Per a aquest projecte simplement s'ha posat el nom de *romi*.

Llavors s'han de definir les credencials d'accés de l'administrador de la base de dades.

Seguidament, s'ha de configurar el tipus d'instància que es vol crear. Amazon RDS ofereix fins a 18 tipus diferents de instàncies optimitzades segons la potència de càlcul o memòria. Per a aquest projecte s'ha escollit la opció *Burstable classes* i després el tipus d'instància *db.t2.micro* (Imatge 28).

DB instance size

DB instance class [Info](#)
Choose a DB instance class that meets your processing power and memory requirements. The DB instance class options below are limited to those supported by the engine you selected above.

Standard classes (includes m classes)
 Memory Optimized classes (includes r and x classes)
 Burstable classes (includes t classes)

db.t2.micro
1 vCPUs 1 GiB RAM Not EBS Optimized

Include previous generation classes

Imatge 28 Configuració del tipus de instància al crear una nova base de dades a Amazon RDS

Aquest tipus de instància està inclosa dins el període de prova de AWS, i ofereix 1vCPU i 1 GB de RAM. Per a l'inici del projecte aquesta potència de càlcul i memòria és més que suficient. Posteriorment, si de cares al futur es necessita més potència, es pot canviar el tipus de instància de forma molt fàcil sense la necessitat de tornar a instal·lar-ho i configurar-ho tot.

Després de configurar el tipus de instància, s'ha de configurar el tipus i capacitat del disc que s'utilitzarà per guardar totes les dades (Imatge 29).

Storage

Storage type [Info](#)
General Purpose (SSD)

Allocated storage
20 GiB
(Minimum: 20 GiB, Maximum: 16384 GiB) Higher allocated storage **may improve** IOPS performance.

ⓘ Provisioning less than 100 GiB of General Purpose (SSD) storage for high throughput workloads could result in higher latencies upon exhaustion of the initial General Purpose (SSD) IO credit balance. [external link for more details.](#)

Storage autoscaling [Info](#)
Provides dynamic scaling support for your database's storage based on your application's needs.

Enable storage autoscaling
Enabling this feature will allow the storage to increase once the specified threshold is exceeded.

Imatge 29 Configuració del tipus de disc al crear una nova base de dades a Amazon RDS

Amazon RDS ofereix dos tipus d'emmagatzematge:

- *General Purpose (SSD)*: És un tipus de emmagatzematge de tipus general per càrregues de treball normals. Ofereix una capacitat de fins a 3.000 IOPS (operacions de lectura i escriptura per segon).
- *Provisioned IOPS (SSD)*: És un tipus de emmagatzematge preparat per a càrregues de treball molt elevades d'escriptura i lectura. Ofereix una flexibilitat de subministrament de dades que van des de 1.000 fins a 30.000 IOPS (operacions de lectura i escriptura per segon).

La millor opció de les dos és *Provisioned IOPS (SSD)*, però lògicament aquesta opció també és molt més cara. Per tant, la més adient per el projecte de les nostres característiques en una fase inicial és el tipus *General Purpose (SSD)*.

Llavors s'ha de configurar la capacitat que es vol assignar al disc. El mínim permès és de 20 GB i el màxim és de 16384 GB. Per a aquest projecte s'ha escollit el mínim permès, que és 20 GB.

Amazon RDS també permet marcar l'opció d'ampliar el disc de forma dinàmica a mesura que es necessiti més capacitat. Aquesta opció és molt útil en els casos de tenir pics de molta alta capacitat, ja que llavors s'augmentarà la capacitat dinàmicament i només es pagarà per l'espai utilitzat durant el període que s'ha fet servir. Tot i això, per a aquest projecte no fa falta i, per tant, no s'ha marcat.

Seguidament, AWS ofereix la possibilitat de crear una altre base de dades rèplica que en cas que falli la principal llavors les peticions s'enviaran a la rèplica (Imatge 30).

Availability & durability

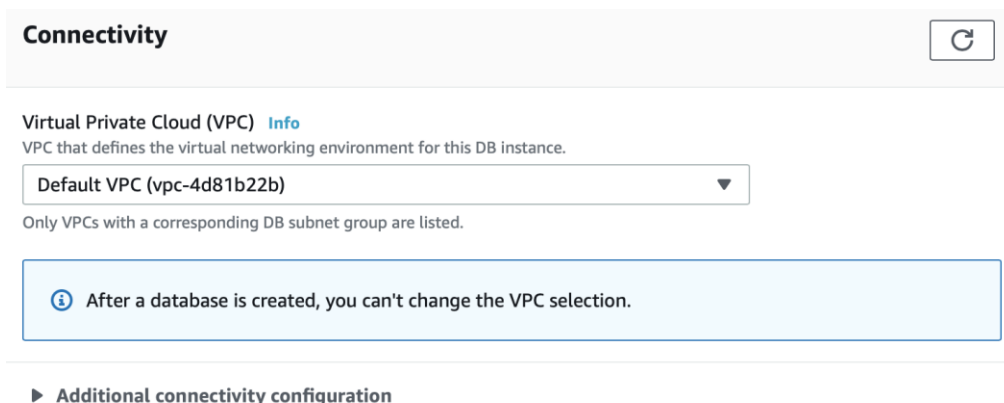
Multi-AZ deployment [Info](#)

- Create a standby instance (recommended for production usage)
Creates a standby in a different Availability Zone (AZ) to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups.
- Do not create a standby instance

Imatge 30 Configuració de una base de dades rèplica al crear una nova base de dades a Amazon RDS

En aquest cas, per a aquest projecte es va marcar l'opció de no crear la base de dades de rèplica. Tot i això, en el següent apartat s'explica com crear una rèplica de lectura d'una base de dades ja existent.

Tot seguit, només queda configurar el tipus de xarxa VPC (*Virtual Private Cloud*) a on volem configurar la nostra base de dades (Imatge 31).



Connectivity ↻

Virtual Private Cloud (VPC) [Info](#)
VPC that defines the virtual networking environment for this DB instance.

Default VPC (vpc-4d81b22b) ▼

Only VPCs with a corresponding DB subnet group are listed.

i After a database is created, you can't change the VPC selection.

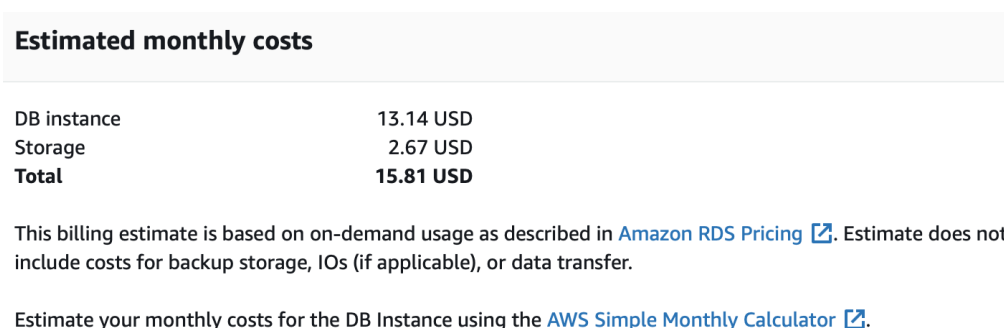
► [Additional connectivity configuration](#)

Imatge 31 Configuració de la xarxa VPC al crear una nova base de dades a Amazon RDS

La xarxa VPC s'utilitza per connectar els diferents serveis de AWS entre ells. En aquest cas, com que tots els serveis que utilitzem són per al mateix projecte ho posarem a la xarxa VPC que ve per defecte.

Finalment, AWS ofereix l'opció d'afegir unes configuracions addicionals com per exemple la creació de còpies de seguretat (per defecte activat), el temps que es vol fer còpies de seguretat. (per defecte 7 dies), generar informes del rendiment i errors de la base de dades, etc.

En aquest punt, AWS calcularà una estimació del cost mensual de la base de dades configurada (Imatge 32).



Estimated monthly costs

DB instance	13.14 USD
Storage	2.67 USD
Total	15.81 USD

This billing estimate is based on on-demand usage as described in [Amazon RDS Pricing](#). Estimate does not include costs for backup storage, IOs (if applicable), or data transfer.

Estimate your monthly costs for the DB Instance using the [AWS Simple Monthly Calculator](#).

Imatge 32 Estimació del cost mensual de la base de dades creada

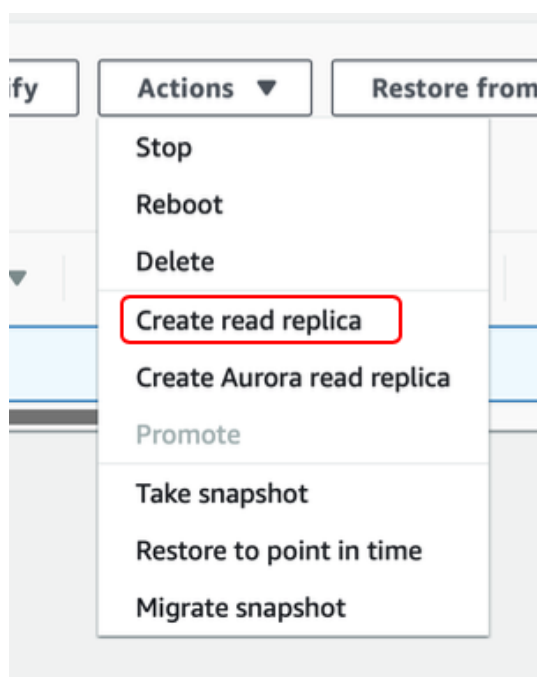
Per a la configuració escollida per a aquest projecte el cost mensual estimat és de 15,81 USD.

Amb tot això, ja tindrem la configuració finalitzada i només quedarà pulsar el botó *Create database* situat al final de tot per crear la base de dades.

9.1.1 Configuració de la rèplica de la base de dades

Amazon RDS ofereix la possibilitat de crear repliques de lectura d'una forma molt fàcil a través del panell de control de AWS (AWS Amazon RDS, 2019).

Primer de tot s'ha de seleccionar la base de dades que es vol replicar i posteriorment clicar en el menú *Actions* i escollir la opció *Create read replica* (Imatge 33).



Imatge 33 Opció del menú de Amazon RDS per crear repliques de lectura

Un cop s'ha clicat l'opció llavors s'obrirà una nova pàgina a on es podran configurar totes les especificacions de la nova base de dades que es crearà com a rèplica de l'original.

Per defecte, copiarà la configuració de la base de dades de la qual s'ha creat la rèplica i únicament s'haurà de assignar un identificador a la nova instància.

Per a aquest projecte com que l'identificar de la base de dades mestre és romi s'ha posat d'identificador romi-rèplica.

També cal saber que per defecte en les bases de dades del tipus rèplica no es realitzen còpies de seguretat. Tot i així, també es pot programar que es facin còpies de seguretat.

Finalment, un cop s'ha assignat un identificador i s'ha configurat ja es pot clicar el botó final de *Create read rèplica*, que automàticament crearà la rèplica amb la configuració entrada.

Llavors a la llista de base de dades ja es podrà veure la nova base de dades creada. També hi haurà una columna anomenada *Role* que indicarà si la base de dades és del tipus *Master* o *Rèplica* (Imatge 34).

DB identifier ▲	Role ▼	Engine ▼	Region & AZ ▼
romi	Master	MySQL Community	eu-west-1b
romi-replica	Replica	MySQL Community	eu-west-1c

Imatge 34 Llistat de les bases de dades configurades a Amazon RDS

Amb això ja es tindrà la base de dades configurada i preparada per rebre les peticions de lectura. L'actualització de les dades de la base de dades rèplica es farà automàticament quan es modifiqui la base de dades mestre.

9.2 Configuració de la API REST amb Amazon API Gateway

Per a la configuració de la API REST s'ha utilitzat el servei de AWS anomenat Amazon API Gateway.

Per crear una API REST amb aquest servei primer de tot s'ha d'anar al panell de control de AWS i fer clic al servei API Gateway. Després des de la pàgina de API Gateway s'haurà de polsar la opció *Create API*.

Llavors s'obrirà una nova pantalla amb la configuració tal i com es mostra a la següent Imatge 35.

Choose the protocol

Select whether you would like to create a REST API or a WebSocket API.

REST WebSocket

Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

New API Clone from existing API Import from Swagger or Open API 3 Example API

Settings

Choose a friendly name and description for your API.

API name*	<input type="text" value="My API"/>
Description	<input type="text"/>
Endpoint Type	<input type="text" value="Regional"/> ⓘ

Imatge 35 Configuració al crear una nova API amb el servei Amazon API Gateway

Primer de tot s'ha d'escollir si es vol fer una API del tipus REST o WebSocket. Per a aquest projecte s'ha fet una API del tipus REST.

Seguidament, AWS ofereix la possibilitat de crear la API des de nou, copiar-la d'una ja existent, importar-la des de Swagger o Open API 3 o crear-ne una d'exemple. En aquest punt, s'escollirà l'opció de crear una nova API.

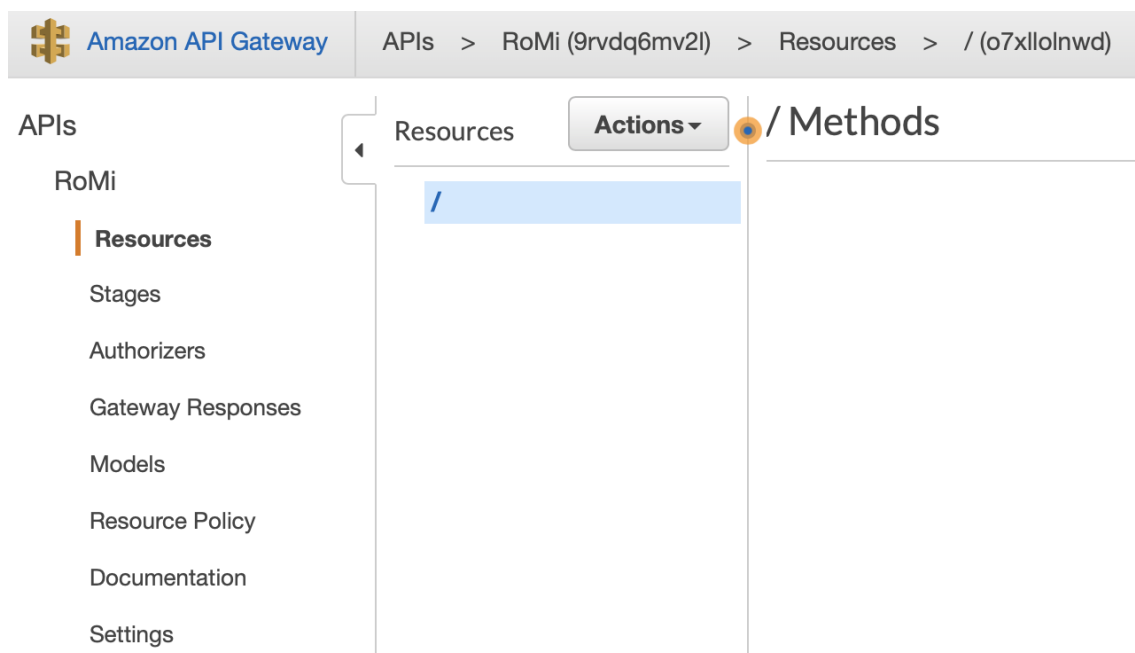
Finalment només quedarà especificar el nom, escriure una petita descripció i escollir el tipus de endpoint.

El tipus de endpoint pot ser de tres tipus diferents:

- *Regional*: Les API amb endpoint del tipus *regional* són accessibles a tot Internet i s'executen a la regió seleccionada per al usuari.
- *Edge optimized*: Les API amb endpoint del tipus *edge optimized* estan desenvolupades utilitzant una xarxa de CloudFront.
- *Private*: Les API amb endpoint del tipus *private* només són accessibles des de la xarxa VPC (*Virtual Private Cloud*) al qual s'ha configurat.

Per a la realització d'aquest projecte, s'ha escollit la opció *Regional*.

Amb això, ja es podrà pulsar el botó *Create API* i automàticament ja es crearà la API en blanc. Un cop creada, s'obrirà una pantalla a on es poden començar a definir tots els endpoints que s'han dissenyat a l'apartat anterior.



Imatge 36 Pàgina de configuració de la API recent creada

Primer de tot s'ha d'afegir la versió de la API. Per fer-ho es clicarà el botó *Actions* i després *Create Resource*. Això obrirà una nova pantalla a on s'haurà d'especificar el nom del recurs i el path.

A la següent Imatge 37 es pot veure els camps necessaris per a la creació d'un recurs a la API.

New Child Resource

Use this page to create a new child resource for your resource. 

Configure as [proxy resource](#)



Resource Name*

My Resource

Resource Path*

/ my-resource

You can add path parameters using brackets. For example, the resource path **{username}** represents a path parameter called 'username'. Configuring **/{proxy+}** as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to **/foo**. To handle requests to **/**, add a new ANY method on the **/** resource.

Enable API Gateway CORS



Imatge 37 Creació de un nou recurs per a la API

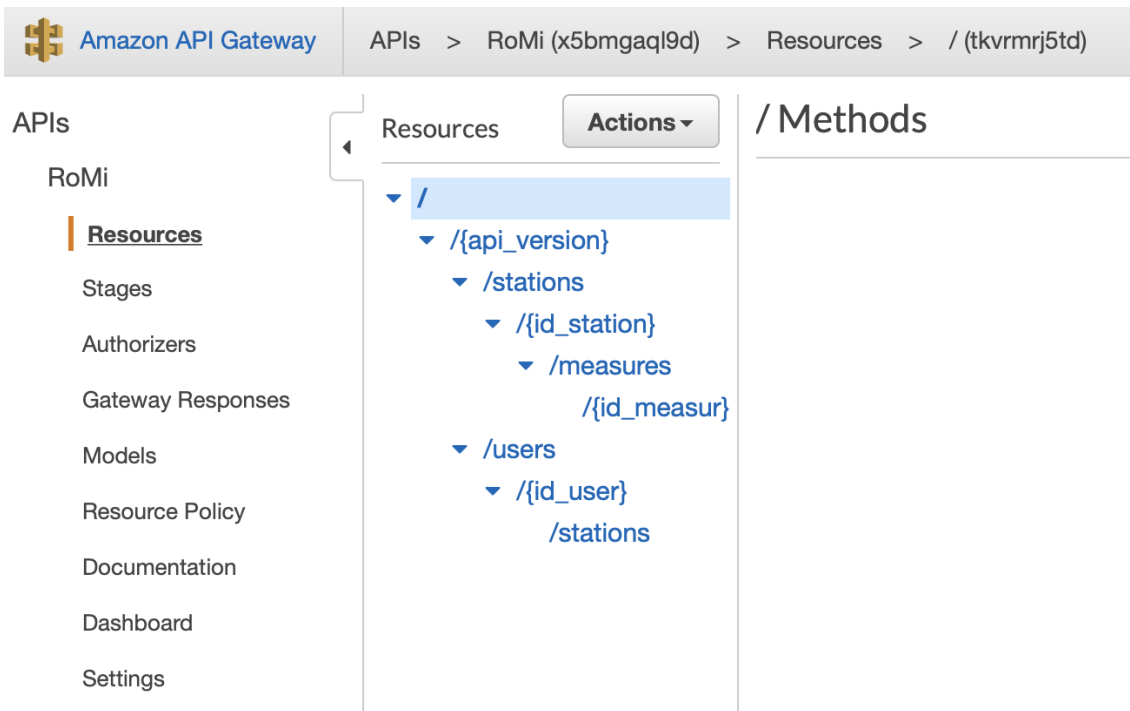
En aquest cas es posarà com a nom *api-version* i com a path *{api-version}*, llavors es polsarà el botó de *Create Resource*.

Un cop creat, s'ha de seleccionar el recurs creat i tornar a seguir el mateix procediment seguit anteriorment i crear dos nous recursos, un per les estacions anomenat *stations* i un altre per als usuaris anomenat *users*. És important destacar que en aquest cas no s'ha d'escriure el path entre claus.

Quan s'escriu el recurs de la API entre claus ("{ i }") significarà que aquest camp de la API s'utilitza per passar valors, normalment identificadors. Per exemple, en el cas de la versió de la API, aquest camp es substituirà per el valor "v1" per indicar la versió 1 de la API.

Aquest procediment d'afegir recursos es realitzarà fins a completar tots els endpoints definits en el disseny de la API.

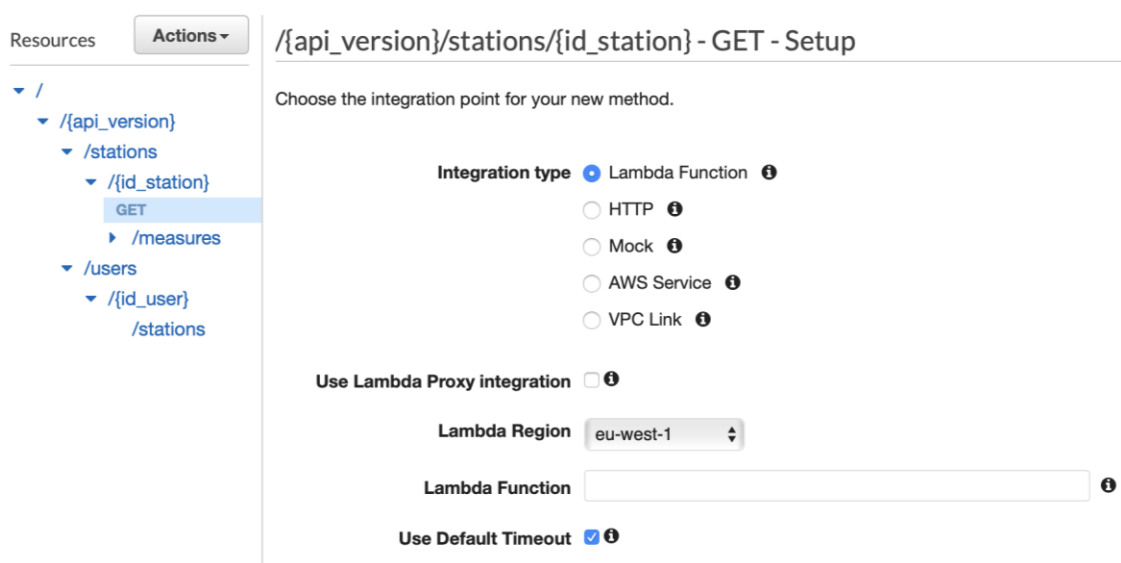
Un cop s'han definits els endpoints es veurà una estructura com la de la següent Imatge 38.



Imatge 38 Pàgina de configuració de la API amb els recursos afegits

Un cop s'han afegit tots els recursos de tots els endpoints, només falta afegir per cada endpoint els mètodes (GET, POST, PUT o DELETE) que tingui definits.

Per afegir els mètodes s'ha de seleccionar l'últim recurs de cada endpoint i tot seguit polsar l'opció *Actions* i després *Create Method*. Després sortirà un desplegable i s'haurà d'escollir quin mètode es vol definir. Un cop s'ha escollit si es vol fer el mètode GET, POST, PUT o DELETE s'obrirà la següent pantalla.



Imatge 39 Creació d'un nou mètode per a un endpoint de la API

Aquí s'haurà d'especificar que es vol executar una funció Lambda, per tant, s'haurà de marcar la primera opció anomenada *Lambda Function*.

També s'ha d'activar la opció *Use Lambda Proxy Integration*. Amb aquesta opció es passarà tot el contingut del header de la petició realitzada a la API a la funció Lambda, d'aquesta forma es podran recollir els paràmetres definits a la URL.

La *Lambda Region* es deixarà la que està per defecte.

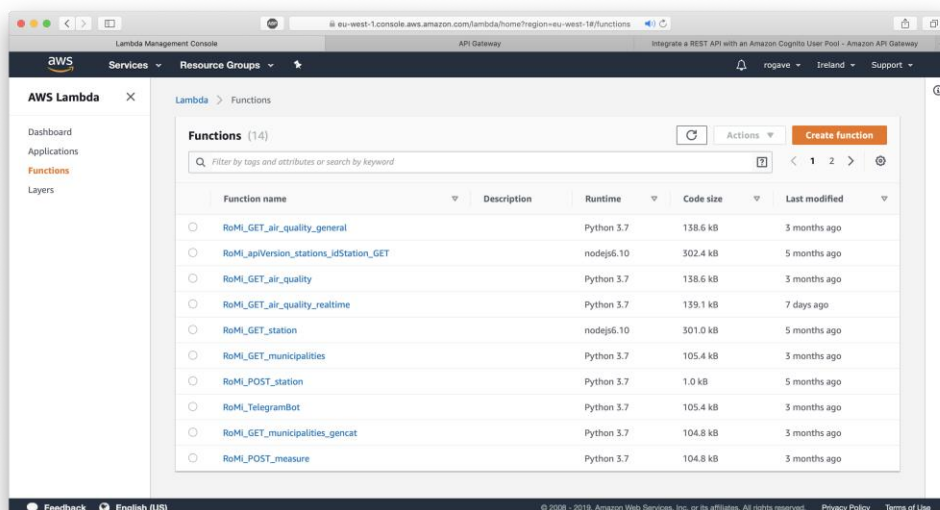
Finalment, només quedarà assignar la funció Lambda de l'endpoint amb el mètode seleccionat.

Això significa que per cada endpoint i mètode que tingui cada un d'ells s'haurà de definir una funció Lambda per processar les peticions de la API. En aquest projecte no s'han programat els endpoints necessaris per el correcte funcionament del sistema a la quarta etapa del disseny.

9.2.1 Programació dels endpoint de la API amb AWS Lambda

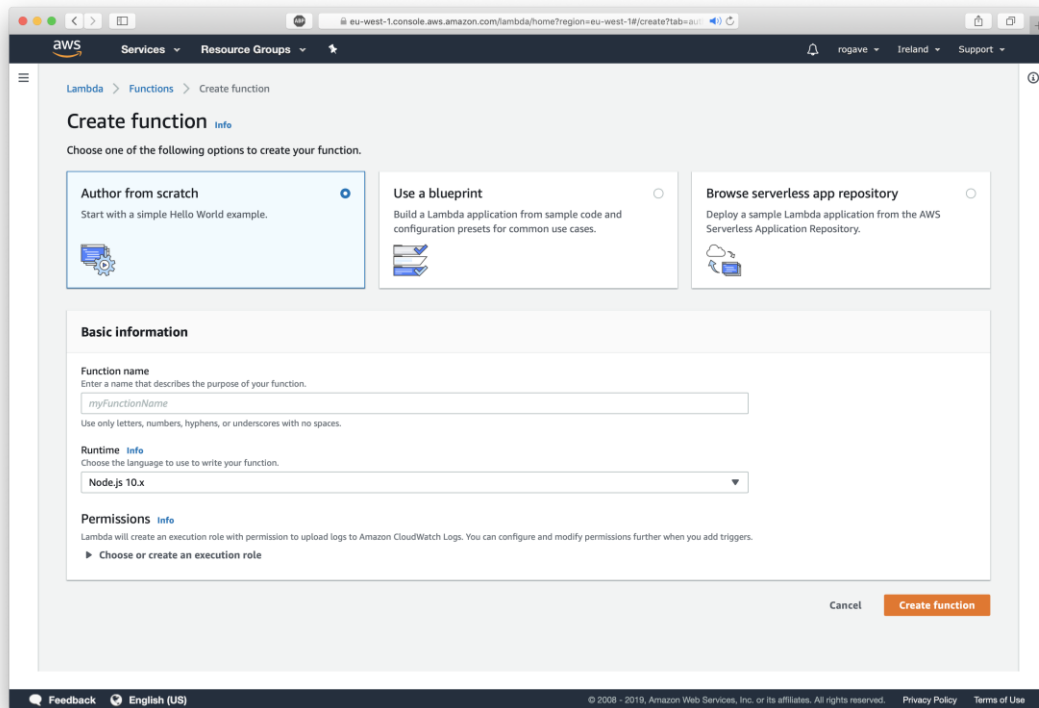
Per configurar els endpoints s'utilitzarà el servei de AWS Lambda. Aquest servei es pot configurar a través del panell de control de AWS.

Primer de tot s'anirà al servei AWS Lambda i es farà clic a l'opció *Functions* del menú de l'esquerra. Allà podrem veure el llistat de funcions Lambda creades.



Imatge 40 Llistat de funcions creades al servei AWS Lambda

En aquesta pantalla es polsarà el botó de d'alt a la dreta que posa *Create function*. Tot seguit, s'obrirà una nova pantalla a on s'haurà de configurar la nova funció que volem crear (Imatge 41).



Imatge 41 Pàgina per configurar una nova funció Lambda al servei AWS Lambda

Primer de tot s'ha d'especificar el nom de la funció, després el llenguatge de programació i versió del llenguatge que s'utilitzarà per programar la funció.

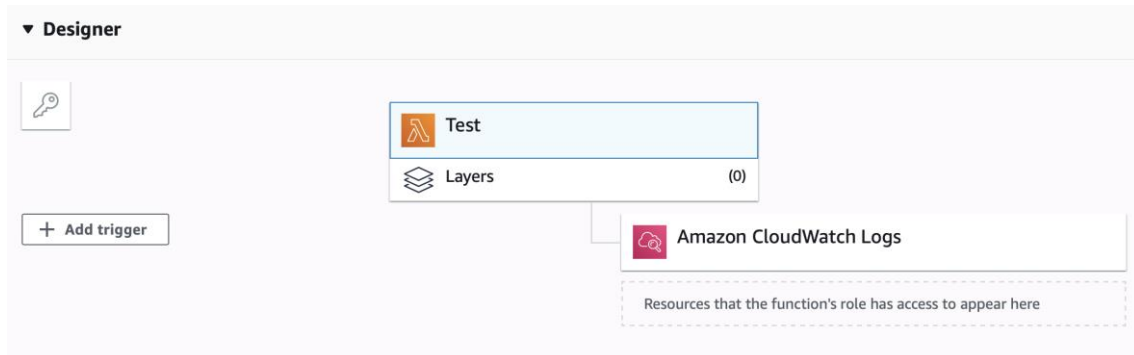
Per a aquest projecte, totes les funcions Lambda que s'han programat s'han fet utilitzant el llenguatge de programació Python, i en concret, la versió 3.7.

Fet això, ja es pot polsar el botó *Create function* i es crearà la funció Lambda.

Un cop creada es podran veure les diferents parts que componen una funció Lambda.

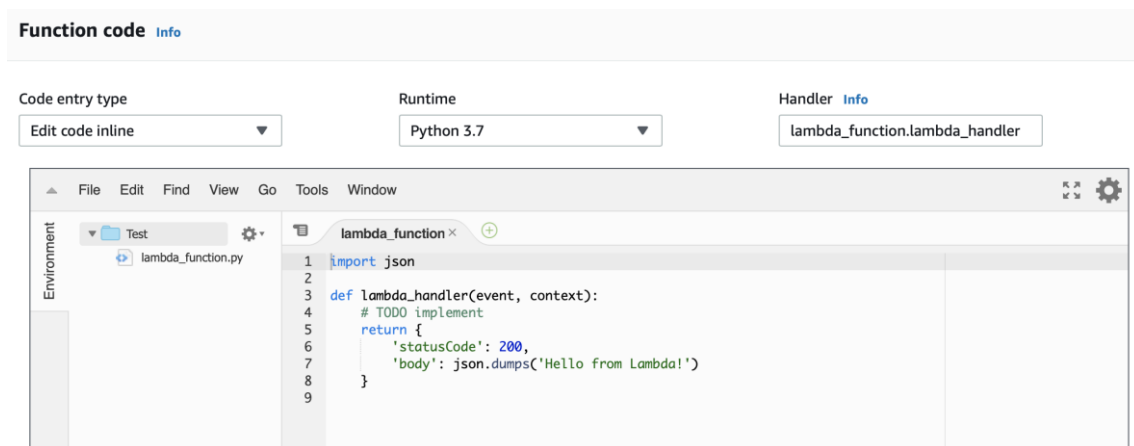
Primer de tot, es té el disseny de la funció Lambda. En el disseny es poden veure totes les connexions d'entrada i sortida amb altres serveis de AWS que té la funció Lambda.

El servei que ve per defecte connectat a la funció Lambda és Amazon CloudWatch Logs, que permet fer seguiment del funcionament i rendiment de la funció (Imatge 42).



Imatge 42 Disseny per defecte de la funció Lambda

Després del disseny hi ha un editor online per poder escriure la funció que es vol executar (Imatge 43).



Imatge 43 Editor online per programar la funció Lambda

També ofereix la possibilitat de carregar el codi amb un zip o a través de Amazon S3 Bucket, aquesta opció és molt útil en el cas que s'hagin d'adjuntar llibreries amb el codi. En aquest projecte per fer les connexions a la base de dades amb Python s'ha necessitat afegir la llibreria *pymysql*, per tant, s'ha utilitzat la opció de pujar el codi amb un zip.

Després del disseny hi ha més paràmetres de configuració de la funció Lambda, com per exemple el rol d'execució, el temps d'execució màxim i el màxim de memòria permesa, tipus de xarxa VPC, etc.

El més important és el rol d'execució, en el rol s'han de definir els diferents permisos de connexió que tindrà la funció Lambda. De moment, el rol que s'ha definit únicament té permís per accedir al servi de Amazon CloudWatch Logs, llavors falta afegir el permís per accedir al servei de Amazon RDS per així poder realitzar operacions a la base de dades.

Per afegir el permís de Amazon RDS s'ha de fer clic al menú anomenat *Execution role* i llavors fer clic a *View the role on the IAM console*.

Al obrir la consola IAM es pot veure la següent pantalla (imatge 44).

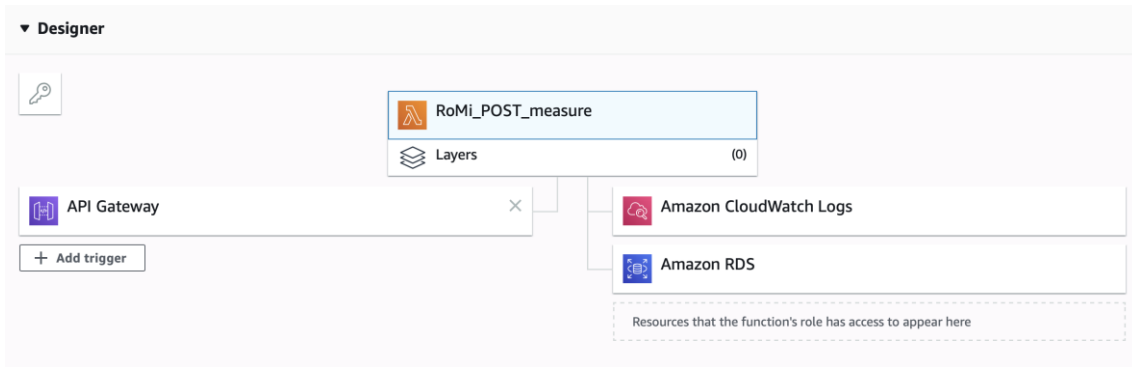
The screenshot shows the IAM console interface for a role. At the top, there is a 'Summary' section with a 'Delete role' button. Below this, several key-value pairs are displayed: Role ARN (arn:aws:iam::514380617171:role/service-role/Test-role-d0b2wy4a), Role description (with an 'Edit' link), Instance Profile ARNs (with a copy icon), Path (/service-role/), Creation time (2019-09-01 23:15 UTC+0200), and Maximum CLI/API session duration (1 hour, with an 'Edit' link). Below the summary, there are tabs for 'Permissions', 'Trust relationships', 'Tags', 'Access Advisor', and 'Revoke sessions'. The 'Permissions' tab is active, showing a dropdown for 'Permissions policies (1 policy applied)'. There is an 'Attach policies' button and an 'Add inline policy' button. A table lists the attached policy: 'AWSLambdaBasicExecutionRole-14b9e2be-f4be-45cc-9920-7ea9652e930c' with a 'Policy type' of 'Managed policy'. At the bottom, there is a section for 'Permissions boundary (not set)'.

Imatge 44 Resum del rol des de la consola IAM

Per afegir un nou permís en al rol s'ha de fer clic a botó *Attach policies* i buscar per el rol anomenat *AmazonRDSFullAccess*. Un cop s'ha trobat el rol, s'ha de fer clic i després pulsar el botó *Attach policy*.

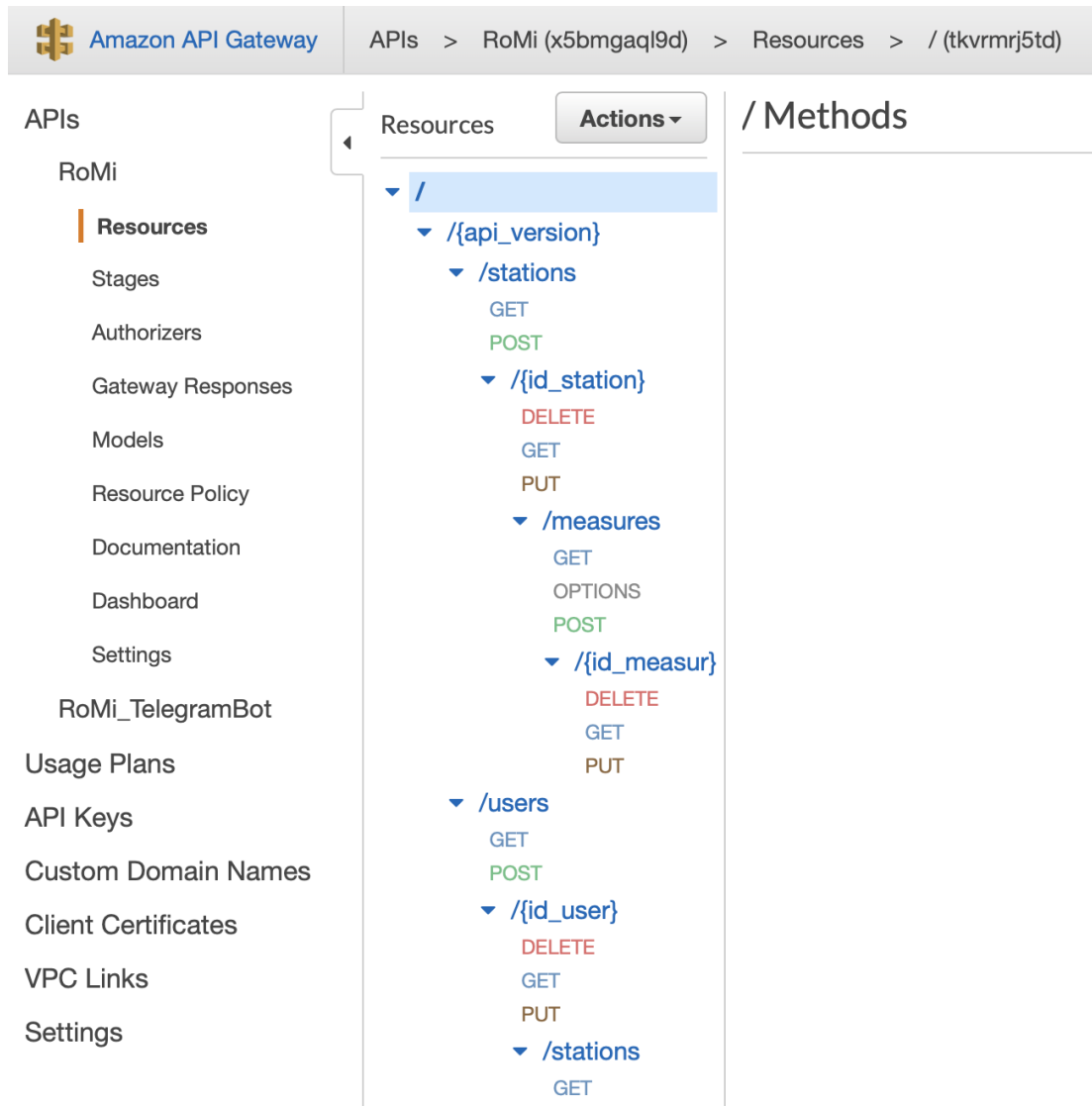
Amb aquest rol ja es tindrà accés al servei Amazon RDS des de la funció Lambda. A aquest punt, només faltará programar la funció perquè realitzi les operacions necessàries a la base de dades i llavors assignar la funció creada a l'endpoint de la API.

Un cop assignada la funció al endpoint de la API el disseny de la funció tindrà un trigger que serà el servei API Gateway, tal i com es pot veure a la següent imatge.



Imatge 45 Disseny de la funció Lambda amb els serveis afegits

El procés de creació de la funció Lambda s'haurà de repetir per a tots els mètodes definits en al disseny de la API REST. Finalment, la API serà tal i com es pot veure a la següent Imatge 46.

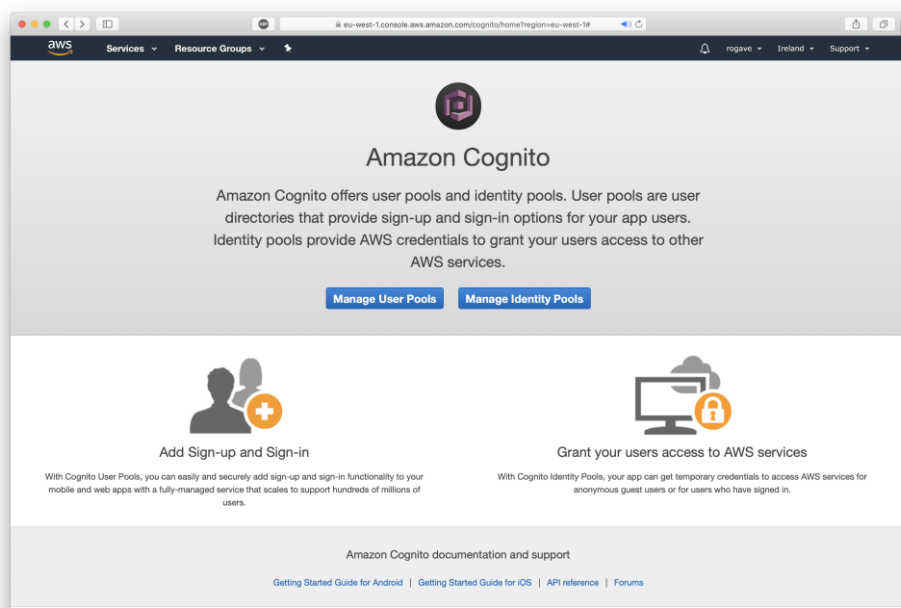


Imatge 46 Pàgina de configuració de la API amb els recursos i mètodes afegits

9.2.2 Configuració del control de seguretat amb AWS Cognito

Amazon API Gateway ofereix la possibilitat d'afegir controls de seguretat per controlar l'accés a la API i tenir registres del nombre de peticions que realitza cada client. Per afegir el control de seguretat s'utilitzarà un altre servei de AWS anomenat AWS Cognito (AWS Cognito, 2019).

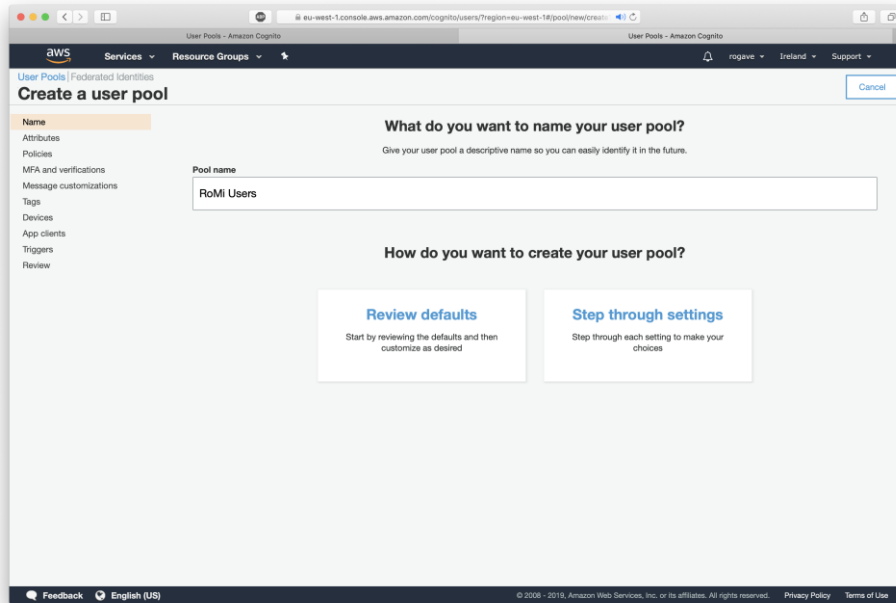
Per configurar el control d'accés amb token primer de tot s'haurà de crear un grup de usuaris (*User pool*) en al servei AWS Cognito. Per fer-ho, s'obrirà el panell de control de AWS i s'accedirà al servei de AWS Cognito. (Imatge 47)



Imatge 47 Pantalla inicial del servei AWS Cognito

A la pàgina inicial del servei AWS Cognito s'haurà de polsar la opció *Manager User Pools* i tot seguit a la nova pantalla clicarem al botó de d'alt a la dreta que posarà *Create a user pool*.

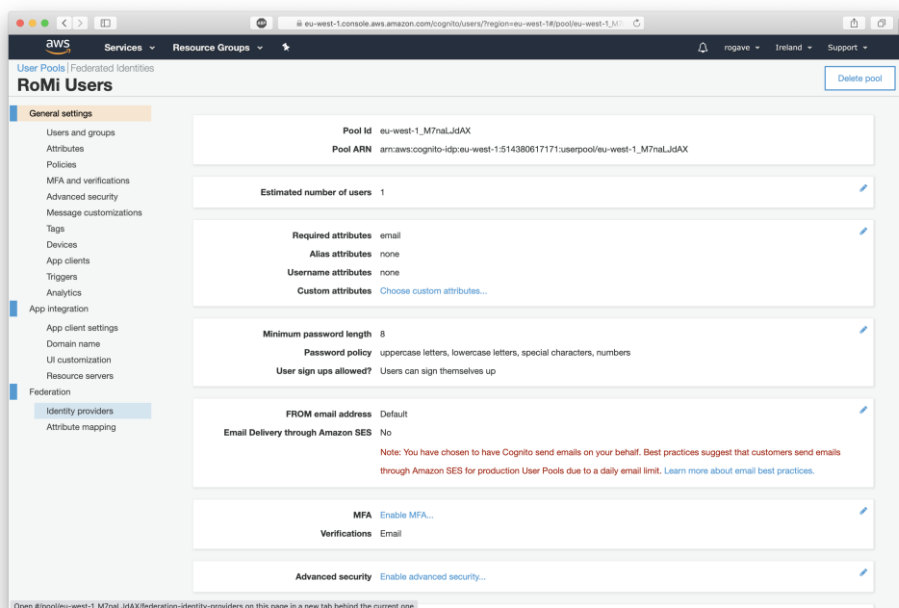
Al crear un nou grup de usuaris es demana que s'introdueixi el nom del grup, en aquest cas s'ha posat *RoMi Users*.



Imatge 48 Procés de creació d'un nou grup de usuaris

Després de posar el nom s'escollirà la opció *Review defaults* i posteriorment es polsarà el botó situat al final que posa *Create pool*. Amb aquestes passes ja s'haurà creat el grup de usuaris.

Un cop creat ja es podrà veure a la llista de la pantalla inicial de AWS Cognito. Posteriorment, fent clic al grup creat s'obrirà la següent pantalla (Imatge 49).



Imatge 49 Configuració del grup de usuaris creat a AWS Cognito

En aquesta pantalla s'haurà de polsar el menú de l'esquerra amb el nom *Users and groups*. Aquí és on es podrà crear un nou usuari o importar una llista d'usuaris. Per a aquest projecte s'ha creat un usuari de prova.

Per crear l'usuari s'ha de polsar la opció *Create user* i posteriorment emplenar totes les dades que es demanen.

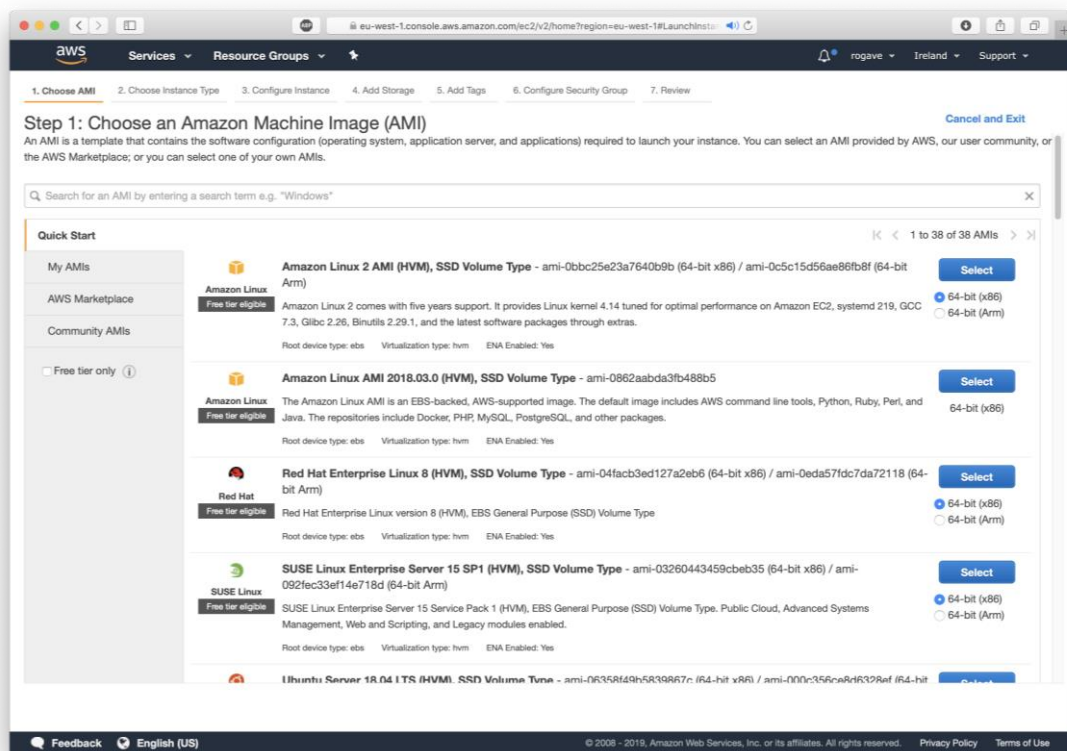
Per cada usuari es crearà un token que posteriorment es podrà utilitzar per autenticar-se al realitzar les crides a la API.

9.3 Configuració dels servidors Shiny

9.3.1 Creació de la instància EC2 a AWS

Les dues pàgines webs que s'han creat per a aquest treball (RoMi Box i Respira.CAT) estan fetes amb Shiny i programades en R. Cada una d'elles funciona a una instància de AWS utilitzant el servei EC2 (*Elastic Compute Cloud*). En el cas de la pàgina web Respira.CAT s'utilitza un servidor més potent que a RoMi Box, ja que és necessària una major potència per carregar els mapes de les ZQA, comarques i municipis.

Per començar la configuració dels dos servidors Shiny primer s'ha d'escollir el tipus de sistema operatiu que es vol dins el servei EC2 que ofereix AWS (Imatge 50).

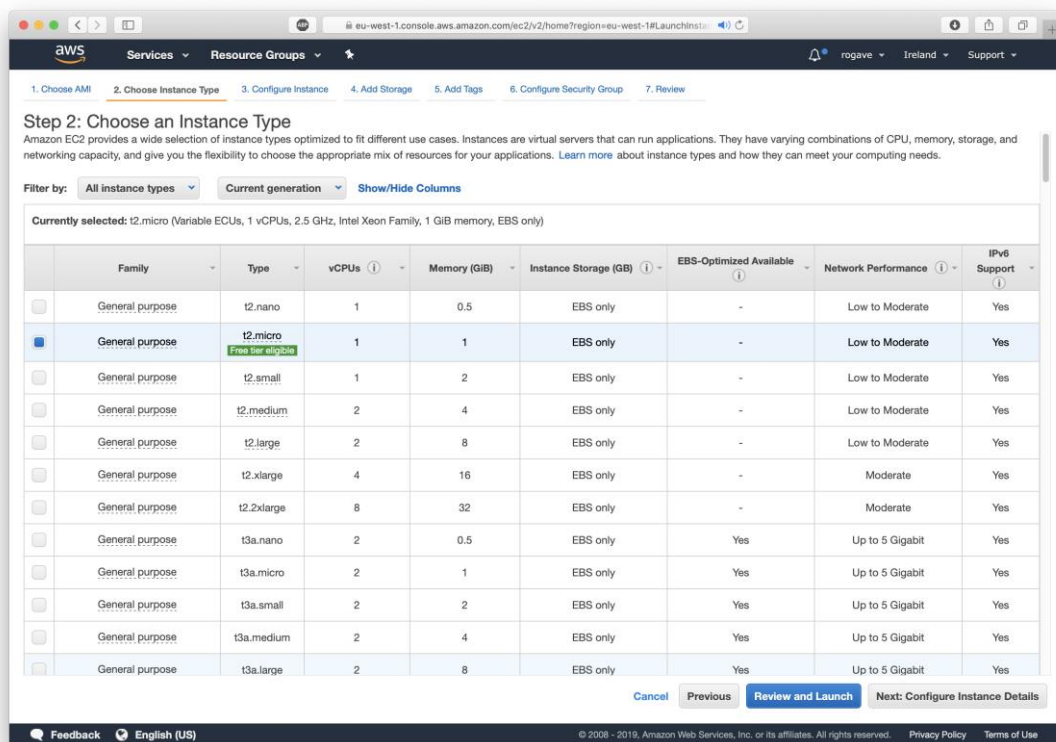


Imatge 50 Pàgina per configurar el sistema operatiu de la nova instància en el servei EC2 de AWS.

Tant per a RoMi Box com per a Respira.CAT s'ha escollit el sistema operatiu Ubuntu 16.04 LTS.

Es va estudiar la possibilitat d'escollir el sistema operatiu Amazon Linux 2 AMI, que és un sistema operatiu basat en Linux, dissenyat per a Amazon i especialment optimitzat per funcionar a AWS. El problema principal és que la majoria de documentació per instal·lar i treballar amb Shiny només estava disponible per al sistema operatiu Ubuntu.

Un cop s'ha escollit el sistema operatiu el següent pas és escollir el tipus d'instància que es vol configurar (Imatge 51).

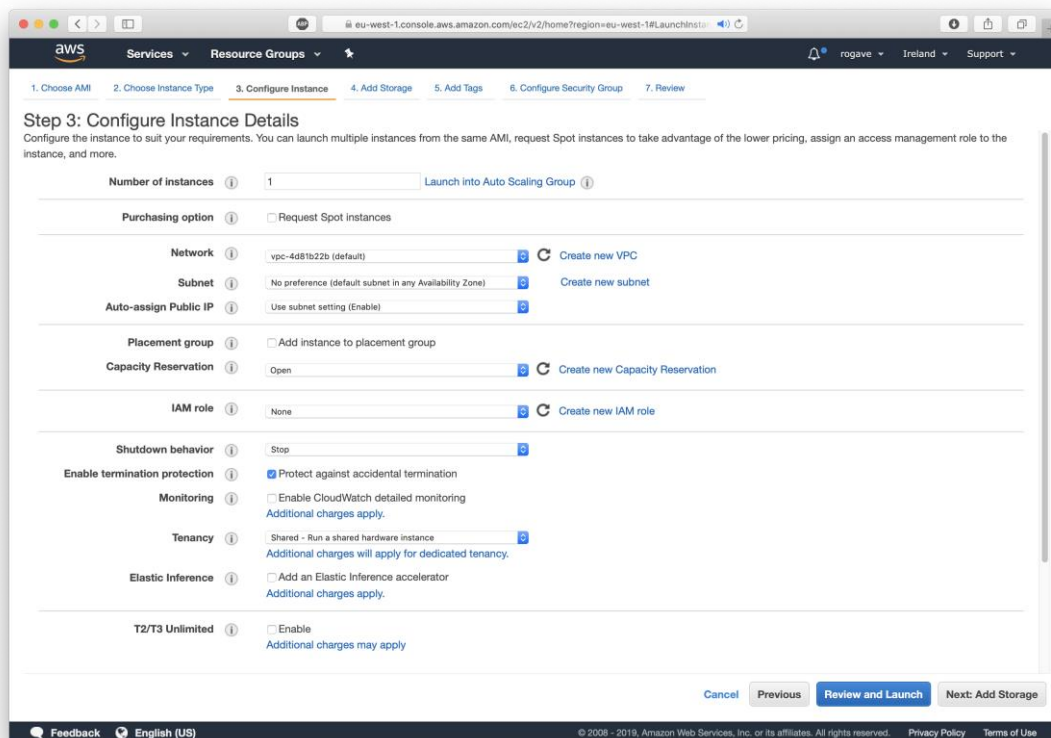


Imatge 51 Pàgina per configurar el tipus d'instància en el servei EC2 de AWS.

En el cas de RoMi Box s'ha escollit la instància *t2.micro*. Aquesta instància és gratuïta dins el període de prova del programa anomenat *Free Tier*, que ofereix AWS en als nous clients durant el primer any. Les especificacions d'aquestes instàncies són 1 vCPU (CPU virtuals) i 1 GB de memòria RAM.

Per a Respira.CAT s'ha escollit la instància *t3a.small*, que té 2vCPU i 2 GB de memòria RAM. Aquesta no està inclosa dins el període gratuït de AWS i, per tant, s'ha de pagar.

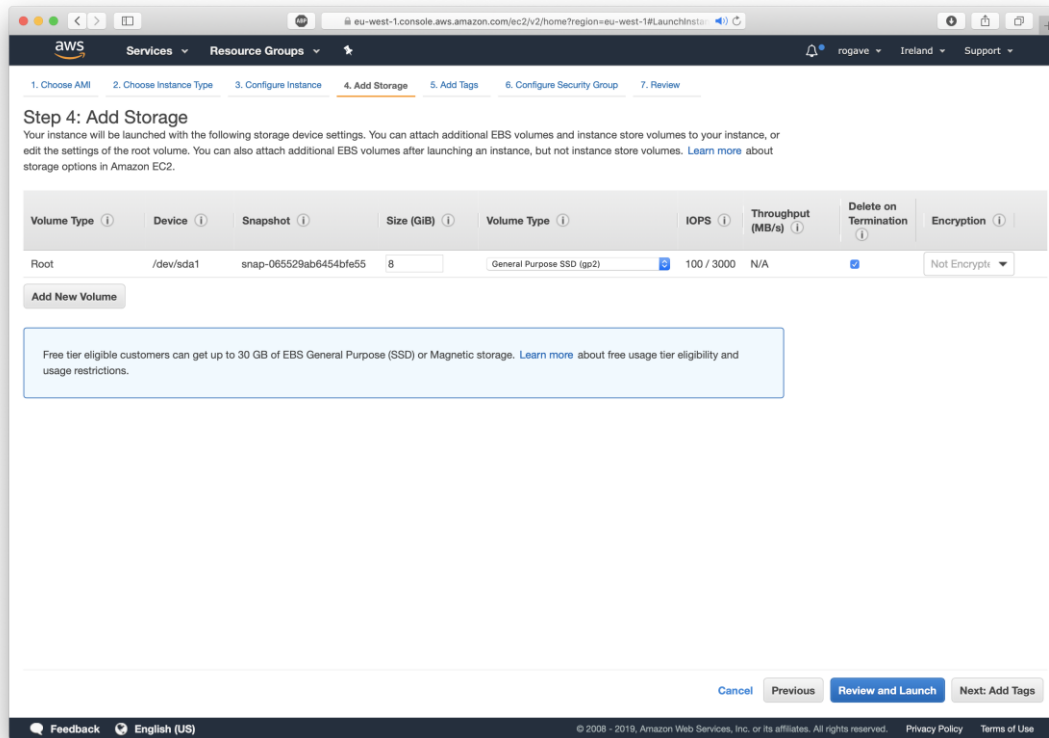
En el següent pas s'han de configurar els detalls de la instància (Imatge 52). Això inclou el nombre de instàncies que es vol crear, el tipus de xarxa i sub-xarxa interna, els permisos d'accés, etc.



Imatge 52 Pàgina per configurar els detalls de la instància en el servei EC2 de AWS.

En aquest punt s'ha deixat tot per defecte, a excepció de marcar l'opció que permet la protecció contra la eliminació de la instància de forma accidental.

En el quart pas s'han de configurar els volums de memòria que es volen assignar a la instància. AWS ofereix tant la possibilitat d'afegir diferents volums de diverses mides, com d'escollir el tipus: SSD o magnètic (Imatge 53).

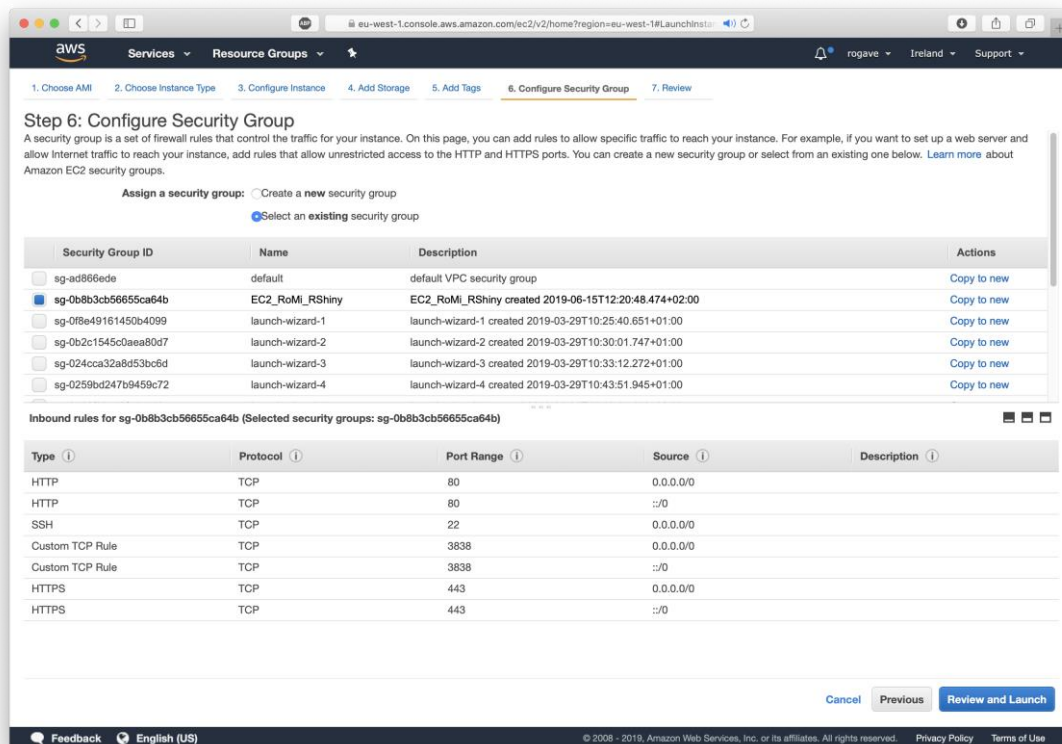


Imatge 53 Pàgina per configurar els discs de memòria de la instància en el servei EC2 de AWS.

Per a aquest projecte només s'ha utilitzat un disc de 8GB SSD per a cada instància.

En el següent pas es dona la possibilitat d'afegir "tags". Això és especialment útil en el cas de tenir moltes instàncies de diferents projectes, ja que així pots filtrar de forma més ràpida les instàncies. A més, et permet filtrar per projecte entre els diferents serveis que s'utilitzen a AWS.

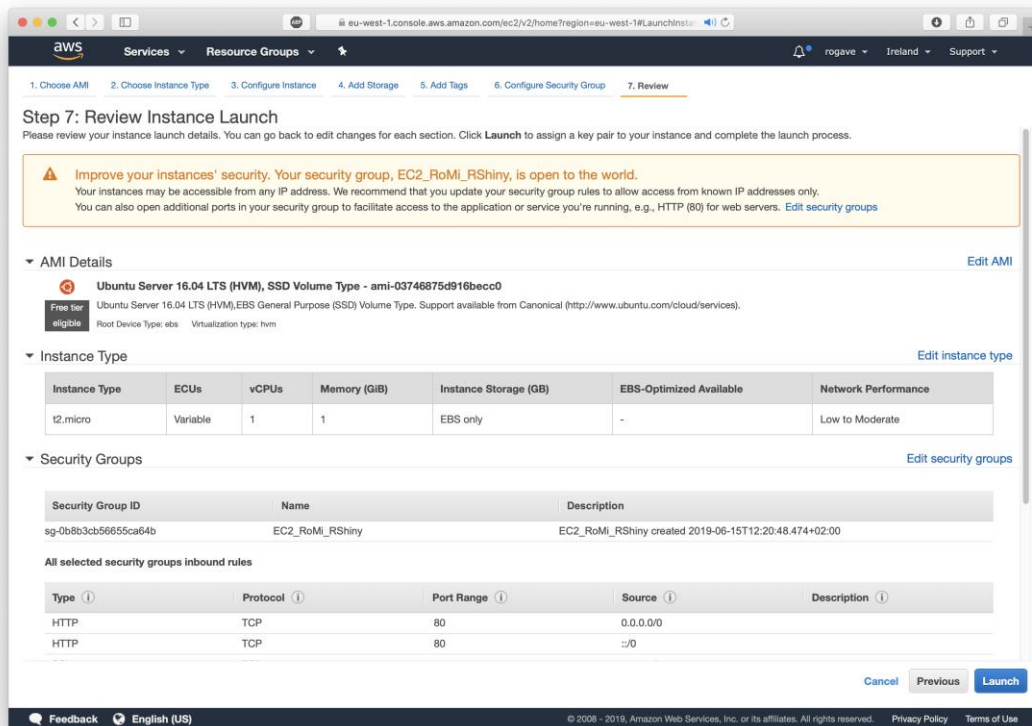
Finalment, només queda assignar el grup de seguretat per a la instància. Això és molt important, ja que en el grup es defineixen els permisos d'entrada i sortida del servidor (Imatge 54).



Imatge 54 Pàgina per configurar el grup de seguretat de la instància en el servei EC2 de AWS.

Per a les dues instàncies s'ha creat un grup anomenat "EC2_RoMi_RShiny", on es permet l'entrada i sortida de totes les adreces IP, i on s'han obert els ports 22, 80, 443 i 3838.

Finalment, s'ha de repassar que tot s'hagi configurat correctament i ja es pot iniciar la instància (Imatge 55).



Imatge 55 Pàgina final del procés de configuració d'una instància en el servei EC2 d'AWS.

9.3.2 Instal·lació de Nginx

Les pàgines webs que s'han creat funcionen amb Shiny, i aquestes, per defecte, funcionen al port 3838. Això significa que si un usuari vol accedir a la nostra pàgina web, ha de posar el nom del domini o la IP del servidor i després aquest port.

Com que no és gens pràctic, s'ha optat per redirigir el tràfic d'entrada del port 80 al port 3838 de forma interna. De forma tècnica vindria a ésser un proxy invers. Amb aquest canvi, quan un usuari posi el nom del domini en el buscador, automàticament es carregarà la pàgina web Shiny (encara que aquesta funcioni en el port 3838 de forma interna en al servidor).

Per configurar un proxy invers s'ha utilitzat Nginx. Nginx és de programari lliure, de codi obert, multiplataforma i àmpliament utilitzat com a servidor web i també com a proxy invers d'alt rendiment a tot al món (Nginx, 2019).

Per començar amb la instal·lació primer de tot és molt important actualitzar el sistema (How To Install Nginx on Ubuntu 16.04, 2016).

```
$ sudo apt-get update
```

Un cop s'ha actualitzat, ja es pot instal·lar Nginx:

```
$ sudo apt-get install nginx
```

Posteriorment, s'ha de configurar el *firewall* del sistema operatiu perquè permeti accés al servei de Nginx. Per fer-ho s'ha d'indicar si es vol permetre que Nginx obri el port 80 (HTTP), 443 (HTTPS) o tots dos.

En aquest cas, com que posteriorment es configurarà un certificat TLS/SSL, s'obriran els dos ports.

```
$ sudo ufw allows 'Nginx Full'
```

Per comprovar que el servidor Nginx funciona en els dos ports, s'ha de posar la IP al navegador, o també amb el domini (en el cas de tenir-lo ja configurat amb la IP pública del servidor). En el marc d'aquest projecte s'ha configurat el domini <http://respira.cat>, que com es pot observar, funciona correctament (Imatge 56).



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Imatge 56 Pàgina per defecte de Nginx.

9.3.3 Creació del certificat SSL/TLS amb Let's Encrypt

Per defecte, les pàgines web Shiny només estan pensades per treballar amb HTTP. En aquest projecte, al ja tenir un proxy invers amb Nginx, es pot configurar la pàgina web perquè funcioni amb HTTPS. D'aquesta forma, la connexió és segura entre la pàgina web i el client.

Per configurar Nginx perquè funcioni amb HTTPS són necessàries dues coses. Primer, s'ha de generar un certificat SSL/TLS per fer la connexió segura. Aquest certificat pot ser autogenerat o emès per una entitat certificadora. L'avantatge dels autogenerats és que es pot crear sense dependre de ningú, però l'inconvenient és que el navegador detectarà que no està validat per cap entitat certificadora, i avisarà als usuaris que la connexió amb la pàgina web no és segura.

L'altre opció és generar un certificat amb una entitat certificadora. Generalment s'ha de pagar, però existeix una entitat certificadora, anomenada Let's Encrypt, que genera certificats de forma gratuïta sempre que es compleixin uns requisits. L'únic que demanen és que la sol·licitud de generar el certificat ha de venir del servidor a on apunta el domini que es vol certificar. Això significa que la IP pública del servidor des d'on es vol certificar, i la IP de la resolució DNS del domini han de ser la mateixa direcció IP. Si això es compleix llavors es pot generar el certificat.

Per generar el certificat amb Let's Encrypt existeixen diferents vies per fer-ho. La més fàcil és utilitzant *Certbot*, que facilita molt la generació i gestió dels certificats (How To Secure Nginx with Let's Encrypt on Ubuntu 16.04, 2017). Aquesta és la via que s'ha empleat en el projecte.

Primer de tot s'ha d'afegir el repositori de *certbot* a la llista de repositoris.

```
$ sudo add-apt-repository ppa:certbot/certbot
```

Un cop afegit, s'ha d'actualitzar la llista de paquets del nou repositori que s'ha afegit.

```
$ sudo apt-get update
```

Ara només falta instal·lar el client *Certbot* amb la següent comanda.

```
$ sudo apt-get install python-certbot-nginx
```

En aquest punt *Certbot* ja està preparat per generar el certificat SSL/TLS, però abans s'ha d'acabar de configurar el proxy invers Nginx amb el domini que volem certificar. Per fer-ho, s'ha d'editar el fitxer de configuració anomenat *default*.

```
$ sudo nano /etc/nginx/sites-available/default
```

En aquest fitxer s'ha d'especificar el nom del domini i el subdomini *www* per a cada domini. Per exemple, en el cas del servidor amb el domini *respira.cat* seria:

```
server_name respira.cat www.respira.cat
```

Per comprovar que el fitxer de configuració està correcta podem utilitzar la següent comanda.

```
$ sudo nginx -t
```

Quan la configuració està correcta, ja es pot reiniciar el servei de Nginx.

```
$ sudo systemctl reload nginx
```

Amb aquest canvi *Certbot* ja es capaç de reconèixer el domini que es vol certificar.

Finalment, ja es pot iniciar el procés de generació del certificat utilitzant *Certbot*. Per fer-ho s'ha d'escriure la següent comanda amb el nom del domini i subdominis que es vulguin certificar. En el cas del domini *respira.cat* aquesta comanda seria:

```
$ sudo certbot --nginx -d respira.cat -d www.respira.cat
```

Amb aquesta comanda s'executa el *Certbot* amb el plugin de *Nginx*, especificant que es vol certificar el domini *respira.cat* i *www.respira.cat*.

La primera vegada que s'executa *Certbot* es pot configurar un electrònic per rebre avisos de seguretat o renovació del nostre domini. Posteriorment, s'han d'acceptar els termes del servei de *Let's Encrypt*. Un cop acceptats, *Certbot* es

connectarà als servidors de *Let's Encrypt* i comprovarà que el domini apunta a la IP pública del servidor.

En cas que el procés de validació finalitzi correctament, es pregunta si es vol redirigir tot el tràfic HTTP a HTTPS i eliminar l'accés via HTTP.

```
Please choose whether or not to redirect HTTP traffic to HTTPS, removing HTTP access.
-----
1: No redirect - Make no further changes to the webserver configuration.
2: Redirect - Make all requests redirect to secure HTTPS access. Choose this for
new sites, or if you're confident your site works on HTTPS. You can undo this
change by editing your web server's configuration.
-----
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): █
```

Això dependrà de cada usuari, si vol o no redirigir tot el tràfic. En el cas de RoMi Box i Respira.CAT s'ha escollit la opció de redirigir tot el tràfic HTTP a HTTPS.

Amb aquest últim pas es finalitza el procés de generació del certificat SSL/TLS, i *Certbot* dirà si tot ha anat bé, a on s'ha guardat la clau pública i privada, i la data d'expiració del certificat.

IMPORTANT NOTES:

- Congratulations! Your certificate and chain have been saved at:
/etc/letsencrypt/live/respira.cat/fullchain.pem
Your key file has been saved at:
/etc/letsencrypt/live/respira.cat/privkey.pem
Your cert will expire on 2019-11-20. To obtain a new or tweaked version of this certificate in the future, simply run certbot again with the "certonly" option. To non-interactively renew *all* of your certificates, run "certbot renew"
- Your account credentials have been saved in your Certbot configuration directory at /etc/letsencrypt. You should make a secure backup of this folder now. This configuration directory will also contain certificates and private keys obtained by Certbot so making regular backups of this folder is ideal.
- If you like Certbot, please consider supporting our work by:

Donating to ISRG / Let's Encrypt: <https://letsencrypt.org/donate>
Donating to EFF: <https://eff.org/donate-le>

En aquest punt, si es torna a visualitzar la pàgina web, es pot veure que la connexió ja està encriptada.



Thank you for using nginx.

Imatge 57 Pàgina per defecte de Nginx amb el certificat SSL/TLS

L'únic problema que tenen els certificats de *Let's Encrypt* és que només tenen un temps de vida de 3 mesos. Això significa que cada 3 mesos s'han de renovar els certificats. Per sort, el procés de renovació també es programa de forma automàtica amb *Certbot*. El que farà serà renovar qualsevol certificat que estigui a 13 dies o menys d'expirar i el renovarà automàticament. Aquest procés s'executarà dues vegades per dia. Per comprovar el procés de renovació ho podem fer amb la següent comanda.

```
$ sudo certbot renew --dry-run
```

Si el procés finalitza sense cap error significarà que tot s'ha configurat correctament, i que *Certbot* s'encarregarà de renovar automàticament els certificats i de reiniciar *Nginx*. En el cas que el procés donés algun error llavors *Certbot* enviaria un correu a l'adreça de correu que s'ha especificat a l'inici de la configuració.

9.3.4 Instal·lació de R i Shiny server

Com ja s'ha explicat prèviament, les pàgines webs estan fetes amb Shiny i programades amb R. Per això, primer de tot s'ha d'instal·lar R amb les següents comandes (Installing Shiny Server Open Source, 2019).

```
$ sudo apt-get install r-base
```

Un cop s'ha instal·lat R ja es pot instal·lar el paquet de R anomenat *shiny*.

```
$ sudo su - -c "R -e \"install.packages('shiny',  
repos='https://cran.rstudio.com/')\""
```

A continuació s'ha d'instal·lar el servidor Shiny. Abans però, és obligatori instal·lar *gdebi*, ja que és necessari per instal·lar el servidor Shiny i les seves dependències.

```
$ sudo apt-get install gdebi-core
```

Posteriorment, s'ha de descarregar i instal·lar el servidor Shiny.

```
$ wget https://download3.rstudio.org/ubuntu-  
14.04/x86_64/shiny-server-1.5.9.923-amd64.deb
```

```
$ sudo gdebi shiny-server-1.5.9.923-amd64.deb
```

Durant el procés d'instal·lació es van experimentar problemes per falta de memòria a la instància que només tenia 1 GB de memòria RAM. Per solucionar els problemes de memòria es va crear un fitxer que va actuar com a memòria swap (Internal compiler error, s.f.).

Per crear el fitxer de memòria swap s'han d'executar les següents comandes:

```
$ sudo dd if=/dev/zero of=/swapfile bs=64M count=16
```

```
$ sudo mkswap /swapfile
```

```
$ sudo swapon /swapfile
```

Un cop creat, ja es poden executar les comandes per instal·lar els paquets de R. Al finalitzar, s'ha d'eliminar el fitxer amb les següents comandes:

```
$ sudo swapoff /swapfile
```

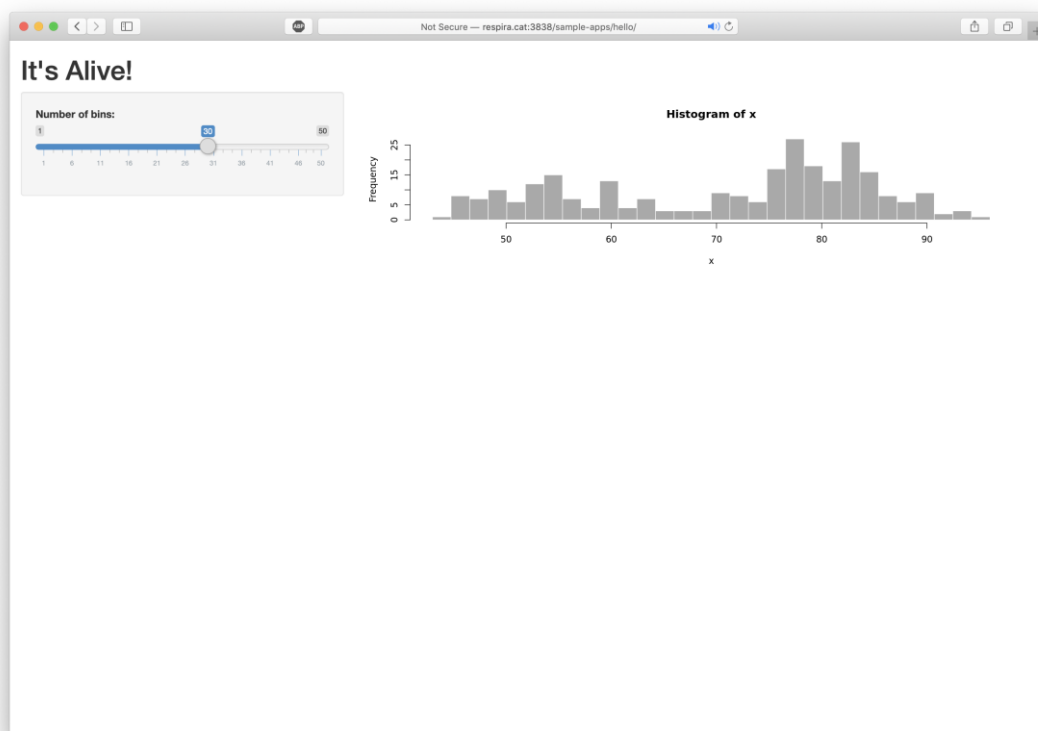
```
$ sudo rm /swapfile
```

En aquest punt, si s'ha pogut instal·lar correctament ja es podrà veure funcionar l'aplicació Shiny que ve per defecte a l'instal·lar Shiny server.

Si es posa al navegador l'adreça IP pública del servidor o el nom del domini més el port 3838 i el subdomini `/sample-apps/hello` es podrà veure la pàgina per defecte. En el cas del domini `respira.cat` la URL seria la següent:

<http://respira.cat:3838/sample-apps/hello/>

En la Imatge 58 es pot veure la pàgina per defecte de Shiny server.



Imatge 58 Pàgina per defecte a l'instal·lar Shiny server.

Al finalitzar la instal·lació del paquet *shiny* i instal·lar Shiny server s'haurà creat una nova carpeta en el directori `/srv/shiny-server/`. En aquest directori serà a on s'haurà de crear una nova carpeta i posar tots els fitxers de la nostra aplicació Shiny a dins.

Com que el desenvolupament de l'aplicació Shiny s'ha realitzat utilitzant el control de versions Git amb GitHub, simplement s'ha de fer un *clone* del repositori per obtenir tots els fitxers de l'aplicació Shiny.

Tant RoMi Box com Respira.CAT s'han fet amb Shiny i comparteixen part del disseny, però les dades i filtres que es mostren són totalment diferents. Per a aquest motiu s'ha treballat amb un únic repositori Git i s'ha creat una branca per a cada pàgina web. D'aquesta forma s'aconsegueix mantenir un control de versions de cada pàgina web compartint part de codi entre elles.

Per obtenir una còpia del repositori, en cas que aquest sigui públic, guardat a GitHub es pot fer amb la següent comanda.

```
$ sudo git clone https://github.com/robertgv/[user]/project.git
```

En cas que el repositori de GitHub sigui privat podem utilitzar la següent comanda a on s'haurà d'especificar l'usuari i contrasenya (How to clone a private repository in GitHub, 2019).

```
$ sudo git clone https://[user]:[password]@github.com/[user]/project.git
```

Quan es té el segon factor d'autenticació activat en al compte de GitHub s'ha de crear un *token* d'accés personal des del menú de configuració de GitHub i substituir el camp de la contrasenya per al *token* creat.

Posteriorment, s'ha d'assegurar que tots els paquets de R utilitzats en l'aplicació Shiny estan instal·lats. En cas de no haver-ho fet, s'ha de realitzar utilitzant la mateixa comanda empleada anteriorment per instal·lar el paquet de R de Shiny. Un exemple d'instal·lació del paquet *dplyr* seria:

```
$ sudo su - -c "R -e \"install.packages('dplyr',  
repos='https://cran.rstudio.com/')\""
```

Pot ser que alguns dels paquets que s'utilitzin no estiguin disponibles per a la versió actual d'R instal·lada en el servidor. Per comprovar la versió instal·lada es pot fer amb la següent comanda:

```
$ R --version
```

Si hi ha algun problema de versions s'haurà d'actualitzar la versió de R instal·lada en el servidor.

9.3.5 Actualització de R i els paquets instal·lats

En cas que es necessiti actualitzar la versió de R s'han de seguir els següents passos (Updating R on Ubuntu, 2018).

Primer de tot s'ha d'afegir el següent enllaç a la llista de repositoris del sistema. Per fer-ho s'ha d'obrir el fitxer a on hi ha la llista de repositoris.

```
$ sudo nano /etc/apt/sources.list
```

En funció de la versió de Ubuntu s'ha d'afegir la següent línia al final del fitxer.

- Ubuntu 18.04

```
deb https://cloud.r-project.org/bin/linux/ubuntu bionic-  
cran35/
```

- Ubuntu 16.04

```
deb https://cloud.r-project.org/bin/linux/ubuntu xenial-  
cran35/
```

- Ubuntu 14.04

```
deb https://cloud.r-project.org/bin/linux/ubuntu trusty-  
cran35/
```

Després d'afegir l'enllaç s'ha d'actualitzar la llista de repositoris del sistema amb la següent comanda.

```
$ sudo apt-get update
```

Posteriorment, només fa falta tornar a instal·lar R.

```
$ sudo apt-get install r-base r-base-dev
```

Per actualitzar les llibreries d'R ja instal·lades s'ha d'executar la següent comanda.

```
$ sudo apt-get upgrade
```

Pot ser que aquesta comanda no actualitzi totes les llibreries, i que a l'instal·lar noves llibreries de R doni problemes de compatibilitat amb les llibreries ja instal·lades. Per solucionar aquests problemes de compatibilitat el més recomanable és eliminar les llibreries instal·lades amb les versió anterior d'R que generen el conflicte i tornar-les a instal·lar amb la nova versió d'R.

El directori a on s'instal·len les llibreries d'R és el següent:

```
$ cd /usr/local/lib/R/site-library/
```

Finalment, un cop s'han instal·lat totes les llibreries només queda reiniciar el servidor Shiny.

```
$ sudo systemctl restart shiny-server
```

Amb això ja es pot visualitzar la nostra aplicació Shiny funcionant a la pàgina web en al port 3838.

Si al carregar la pàgina en el navegador genera un error o no es carrega, serà de gran utilitat consultar els logs d'error que Shiny server genera. Aquests es poden consultar en al següent directori.

```
$ cd /var/log/shiny-server/
```

En aquest directori hi haurà diferents fitxers de logs ordenats per data i hora de generació.

9.3.6 Configuració del proxy invers amb Nginx

El següent pas serà configurar el proxy invers amb Nginx perquè la pàgina web funcioni únicament amb el domini sense que l'usuari hagi d'especificar el port 3838 (Securing Shiny Server with a Reverse Proxy and SSL Certificate, 2019).

Primer de tot s'ha de modificar la configuració de Nginx perquè reenvii les peticions de la pàgina web en al servidor Shiny a través de WebSocket. Per fer-ho s'ha de editar el fitxer de configuració de Nginx.

```
$ sudo nano /etc/nginx/nginx.conf
```

A dins el bloc *http* s'han d'afegir les següents línies.

```
# Map proxy settings for RStudio
map $http_upgrade $connection_upgrade {
    default upgrade;
    '' close;
}
```

Un cop afegides, s'ha de guardar i tancar el document. Tot seguit, s'ha de crear un nou fitxer de configuració amb el nom de la pàgina web que es vol crear. En el cas de *respira.cat* seria.

```
$ sudo nano /etc/nginx/sites-available/respira.cat
```

En aquest fitxer (Imatge 59) s'han de posar les següents línies substituint el camp *respira.cat* per al nom del domini de la pàgina que es vulgui configurar.


```

server {
    listen 80 default_server;
    listen [::]:80 default_server ipv6only=on;
    server_name respira.cat www.respira.cat;
    return 301 https://$server_name$request_uri;
}
server {
    listen 443 ssl;
    server_name respira.cat www.respira.cat;
    ssl_certificate /etc/letsencrypt/live/respira.cat/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/respira.cat/privkey.pem;
    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers on;
    ssl_ciphers AES256+EECDH:AES256+EDH:!aNULL;

    location / {
        proxy_pass http://respira.cat:3838;
        proxy_redirect http://respira.cat:3838/ https://$host/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection $connection_upgrade;
        proxy_read_timeout 20d;
    }
}

```

Imatge 59 Fitxer de configuració de la pàgina de Nginx.

En aquest fitxer és a on s'especifica que totes les peticions d'entrada al port 80 i 3838 són redirigides al port 443, utilitzant HTTPS amb el certificat SSL/TLS generat anteriorment.

Després s'ha de crear un enllaç simbòlic a dins la carpeta `/etc/nginx/sites-enabled/` fent referència al fitxer que s'acaba de crear.

```
$ sudo ln -s /etc/nginx/sites-available/respira.cat
/etc/nginx/sites-enabled/respira.cat
```

Un cop creat l'enllaç simbòlic ja es pot eliminar l'enllaç simbòlic de la pàgina que ve per defecte al instal·lar Nginx.

```
$ sudo rm -f /etc/nginx/sites-enabled/default
```

Finalment només queda comprovar que la configuració de Nginx està correcte.

```
$ sudo nginx -t
```

Si tot està correcte significarà que ja es pot reiniciar Nginx i comprovar si realment es veu la pàgina web. Amb la següent comanda es pot reiniciar el servei de Nginx.

```
$ sudo systemctl restart nginx
```

El problema és que a l'obrir el navegador sortirà una pàgina en blanc amb el directori de la nostra aplicació Shiny:



Imatge 60 Pàgina inicial amb l'enllaç a l'aplicació Shiny

Si l'usuari fa clic a l'enllaç, se li carregarà l'aplicació Shiny. Però això no és el que es vol, sinó que quan l'usuari entri a la pàgina es carregui automàticament l'aplicació Shiny.

Per solucionar aquest problema s'ha d'editar el fitxer de configuració del servidor Shiny.

```
$ sudo nano /etc/shiny-server/shiny-server.conf
```

I a la línia a on hi ha la definició de `site_dir` s'ha d'afegir al final del path el [nom] del directori de la nostra aplicació Shiny.

```

# Instruct Shiny Server to run applications as the user "shiny"
run_as shiny;

# Define a server that listens on port 3838
server {
  listen 3838;

  # Define a location at the base URL
  location / {

    # Host the directory of Shiny Apps stored in this directory
    site_dir /srv/shiny-server/[nom];

    # Log all Shiny output to files in this directory
    log_dir /var/log/shiny-server;

    # When a user visits the base URL rather than a particular application,
    # an index of the applications available in this directory will be shown.
    directory_index on;
  }
}

```

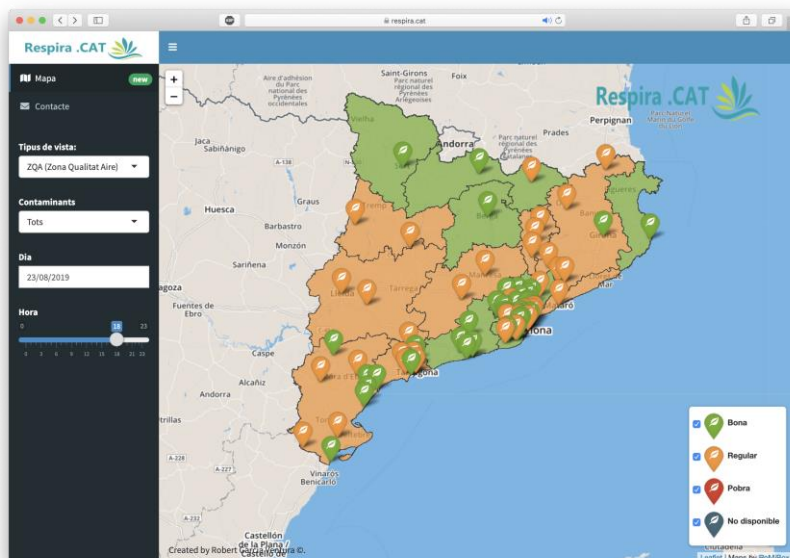
Imatge 61 Fitxer de configuració del servidor Shiny

Un cop s'ha modificat només faltaria reiniciar els serveis del servidor Shiny.

```
$ sudo systemctl restart shiny-server
```

Amb aquest últim canvi ja es podrà veure l'aplicació Shiny funcionant en el domini escollit.

En el cas de *Respira.cat* ja podrem veure la següent pàgina que es mostra en la Imatge 62.



Imatge 62 Pàgina inicial de Respira.CAT

9.3.7 Programar l'actualització automàtica de l'aplicació Shiny

Tal i com s'ha explicat a l'inici, el desenvolupament de les dues aplicacions Shiny s'ha realitzat utilitzant el IDE RStudio local del ordinador i també el RStudio Cloud. Els projectes s'han guardat i sincronitzat utilitzant GitHub.

D'aquesta forma, si es programa una tasca automàtica en al servidor, que cada X temps realitzi una crida als servidors de GitHub per obtenir l'última còpia, llavors no serà necessari connectar-se al servidor per actualitzar la pàgina web cada vegada que es realitzi algun canvi. Únicament serà necessari per instal·lar noves llibreries o solucionar algun error.

Per programar la tasca automàtica primer de tot s'ha de crear un petit script.

```
$ sudo nano /srv/shiny-server/script.sh
```

Posteriorment, s'ha de posar el següent codi substituint el camp *RoMiBox_RShiny* per el nom de l'aplicació Shiny que s'ha creat.

```
#!/bin/bash
```

```
cd /srv/shiny-server/RoMiBox_RShiny  
sudo git pull origin master
```

En aquest cas el que fa l'script és anar a la carpeta de l'aplicació Shiny i executar un pull de la branca *master* que està en els servidors de GitHub.

És molt important afegir al principi del document la primera línia, ja que aquesta informarà al sistema que és un script.

Un cop creat el document, s'han d'afegir els permisos d'execució. Per fer-ho, s'utilitza la següent comanda.

```
$ sudo chmod 700 /srv/shiny-server/script.sh
```

A continuació, s'ha d'executar la programació del script cada cert temps utilitzant el cron del sistema.

Amb la següent comanda s'obra el fitxer de cron.

```
$ sudo crontab -e
```

Al final del fitxer s'ha d'afegir la crida del script. En funció de la freqüència que es vulgui que s'actualitzi canviarà la línia. Per exemple, en el cas de voler que cada 10 minuts s'actualitzi seria la següent línia.

```
*/10 * * * * /srv/shiny-server/script.sh
```

9.4 Anàlisi de les dades

L'anàlisi de les dades s'ha creat amb el llenguatge programació R utilitzant el format R Markdown. La creació de l'anàlisi s'ha dividit en 3 parts principals.

Primer de tot s'han definit els objectius de l'anàlisi i quines són les variables objectius. Per a aquest anàlisi s'han definit com a variables objectius O3 (ozó) i PM10 (nombre de partícules de 10 micròmetres de diàmetre). L'objectiu de l'anàlisi és intentar predir aquests dos contaminants a curt termini a nivell d'estació de qualitat de l'aire. D'aquesta forma, es podrà predir la qualitat de l'aire dels pròxims dies. Per a la predicció de les variables objectius, a part d'utilitzar les dades de la qualitat de l'aire de tot Catalunya, també s'han utilitzat les dades meteorològiques de tots els municipis de Catalunya.

Un cop s'han definit els objectius, s'ha creat el preprocés de les dades. Això implica seleccionar, identificar i preparar les dades. Un cop preparades s'ha creat un primer anàlisi per veure el tipus de dades i les correlacions que existeixen entre les diferents variables.

Cal destacar que la part del preprocés de les dades és una part molt important per a la mineria de dades. Tot i això, també cal dir que és una part que comporta molta feina i pràcticament el 90 % del temps dedicat a l'anàlisi s'ha dedicat en el preprocés de les dades.

Finalment, s'ha fet la modalització de les dades utilitzant dos models. Això significa desenvolupar i entrenar els models, i posteriorment, validar i provar que els models funcionen correctament.

Per a la part de la modalització es van estudiar diferents alternatives però al final es va decidir implementar els models ANCOVA i ARIMA.

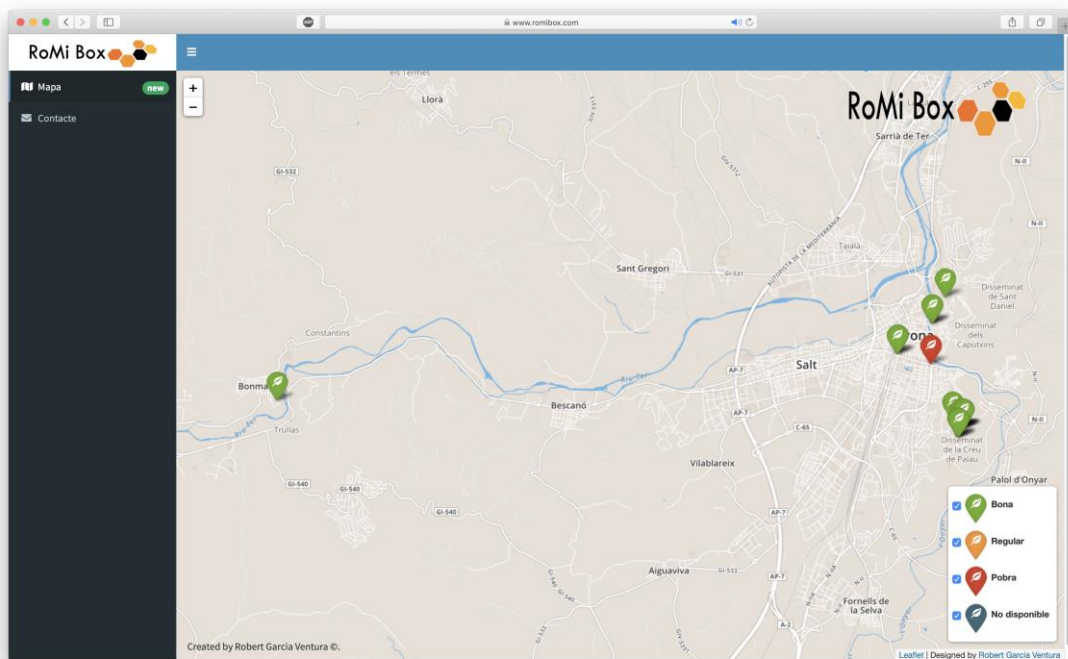
El document de l'anàlisi generat amb tot el codi, gràfiques i resultats es pot trobar adjunt a aquest treball.

10. Resultats

10.1 Pàgina web RoMi Box

A la pàgina web RoMi Box es pot veure la qualitat de l'aire en temps real de les estacions que formen part de la xarxa de RoMi. També s'ha afegit l'estació de la Generalitat de Catalunya situada a l'Escola de Música de Girona, per així poder comparar fàcilment els valors de les estacions de RoMi amb els valors de l'estació de la Generalitat de Catalunya.

A la següent Imatge 63 es pot veure la pàgina web creada:



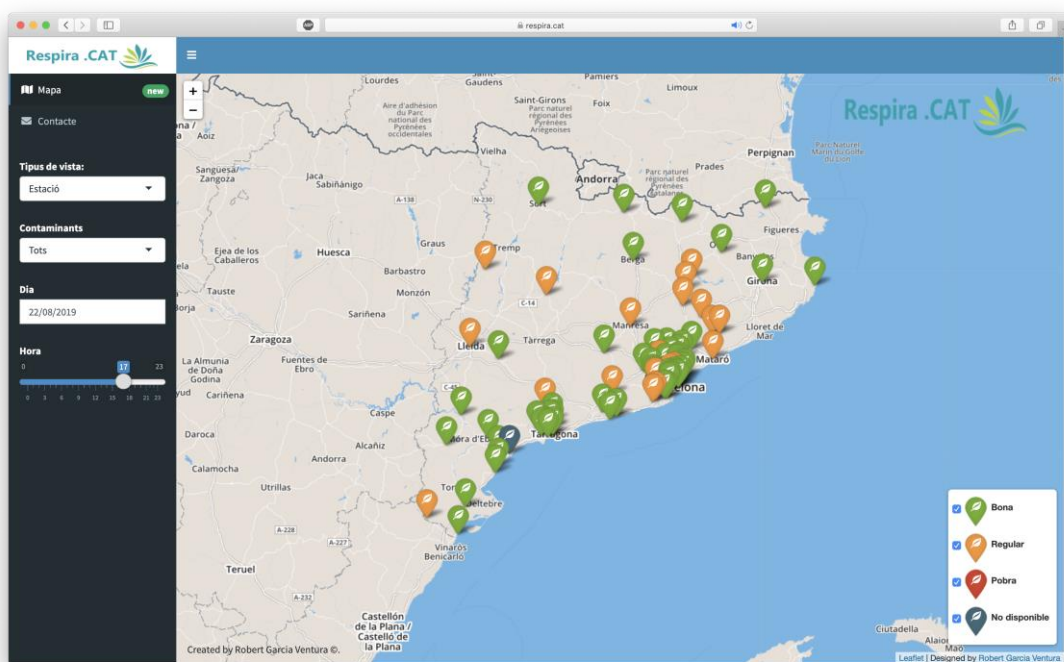
Imatge 63 Pàgina web de RoMi Box

En el menú de l'esquerra s'ha afegit una opció de contacte que permet enviar un correu a la meua adreça de correu electrònic, en cas que algú es vulgui posar en contacte.

10.2 Pàgina web Respira .CAT

A la pàgina web Respira .CAT es pot veure la qualitat de l'aire en temps real de les estacions que formen part de la xarxa d'estacions de la Generalitat de Catalunya.

Al carregar la pàgina web per defecte sempre es carrega l'hora de les últimes dades descarregades a la base de dades (Imatge 64).

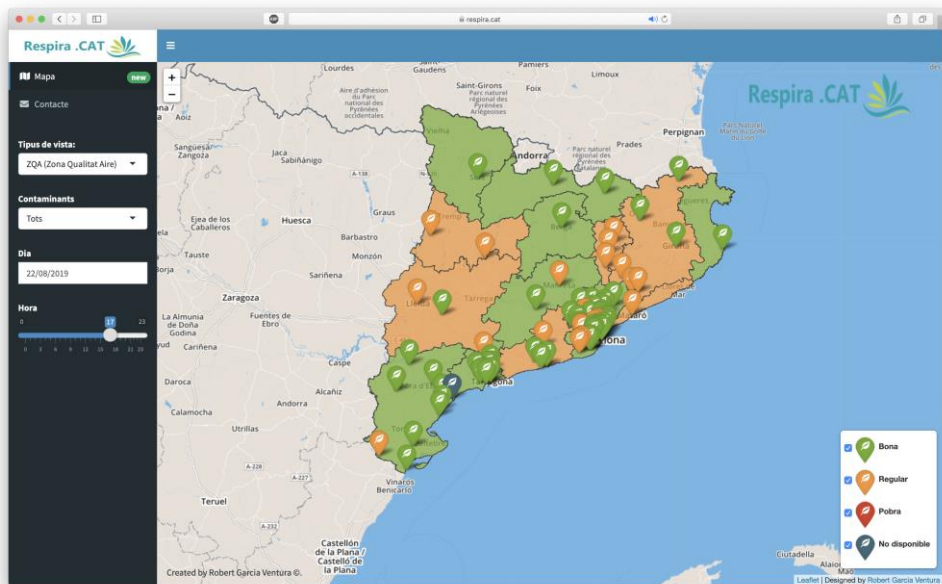


Imatge 64 Pàgina web de Respira .CAT amb el tipus de vista d'estació

A diferència de la pàgina web RoMi Box, en aquesta pàgina sí que es pot consultar l'històric de les dades des del dia en el qual es van començar a recopilar a principis de juny del 2019.

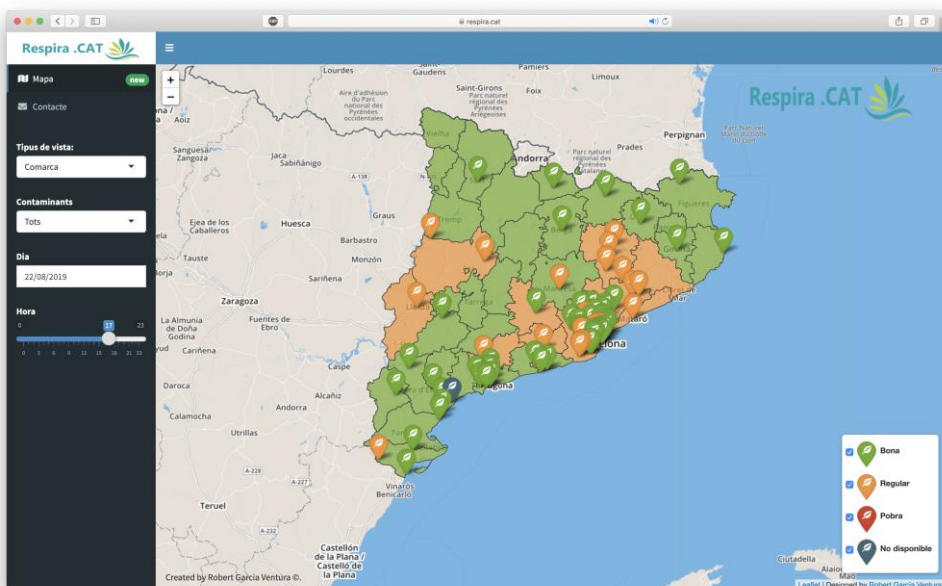
També es pot aplicar un filtre per contaminat, d'aquesta forma es posarà el color de cada estació en funció del contaminant escollit. Per defecte, quan el filtre de contaminants és *Tots* el color de cada estació vindrà determinat per el pitjor valor mesurat de l'estació. Per exemple, si l'estació és de color verd significarà que tots els contaminants mesurats en aquesta estació estan a dins els marges considerats bons.

A la següent Imatge 65 es pot veure el mapa amb el filtre de tipus de vista de ZQA:



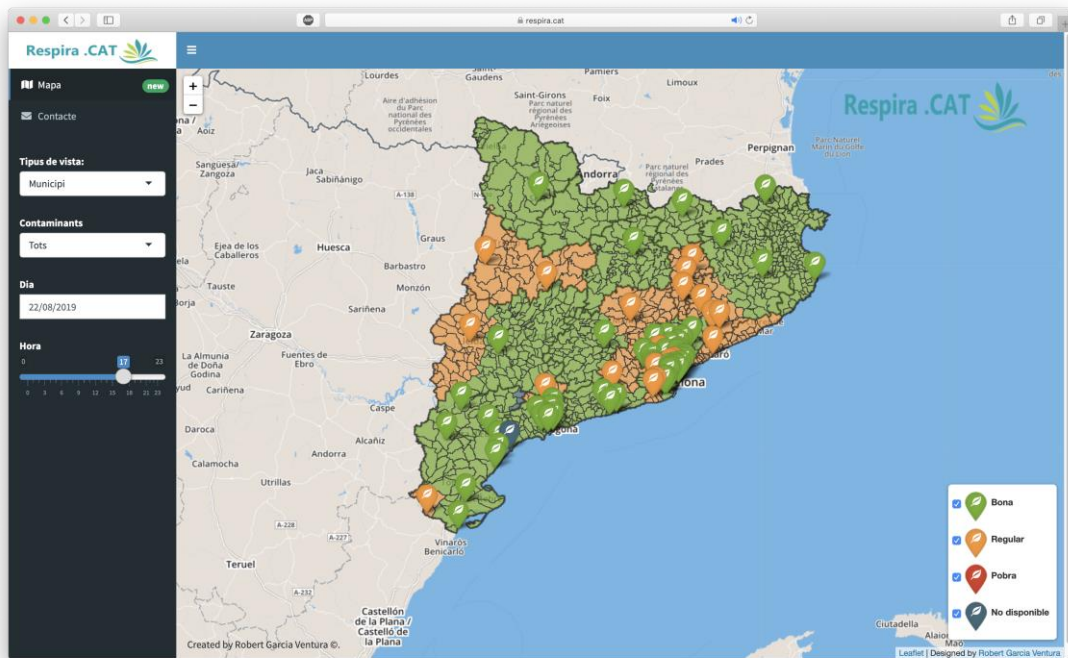
Imatge 65 Pàgina web de Respira .CAT amb el tipus de vista de ZQA

A la següent Imatge 66 es pot veure el mapa amb el filtre de tipus de vista de comarca. En aquest cas, primer s'ha assignat a cada municipi l'estació més pròxima dins la ZQA en la qual es troba el municipi. Un cop es té la contaminació de cada municipi llavors s'ha agrupat per comarca.



Imatge 66 Pàgina web de Respira .CAT amb el tipus de vista de comarca

Finalment, a la següent Imatge 67 es pot veure el mapa amb el filtre de tipus de vista de municipi. En aquest cas, per calcular la qualitat de l'aire de cada municipi, s'ha calculat per a cada municipi l'estació de qualitat d'aire més pròxima dins la ZQA en la qual està cada municipi. Aquesta vista és la que tarda més temps en carregar, ja que es mostren els 947 municipis de tot Catalunya.



Imatge 67 Pàgina web de Respira .CAT amb el tipus de vista de municipi

En aquesta pàgina web, també s'ha afegit al menú de l'esquerra una pàgina de contacte per si alguna persona es vol posar en contacte amb mi.

11. Conclusions

El que s'ha aconseguit amb aquest projecte per una banda és la creació d'un sistema de recopilació de dades altament escalable amb el mínim cost de manteniment.

Per altre banda, mitjançant tècniques estadístiques i de mineria de dades, s'ha creat un anàlisi de la qualitat de l'aire a tot Catalunya on s'ha modelat la qualitat de l'aire amb dades meteorològiques.

Les dades de qualitat de l'aire recollides a la base de dades no han estat suficients per obtenir uns bons resultats en la part del modelatge de l'anàlisi. Això és degut al fet que només es disposen de dades de l'estiu del 2019. Els valors que es volen predir són de finals d'agost del 2019 i les condicions climatològiques durant aquest període han estat molt diferents en comparació a la resta de l'estiu.

En el cas de disposar de moltes més dades a on es representin totes les estacions de l'any és molt més probable que s'obtinguin uns millors resultats aplicant els mateixos models.

Aquest projecte m'ha permès poder aplicar i ampliar un gran conjunt de coneixements, estudiats prèviament durant el màster i alguns d'ells completament nous, en un únic projecte. També m'ha permès aprendre molt més dels serveis que s'ofereixen actualment en la computació al núvol i tots els beneficis que té les estructures serverless.

Personalment, m'ha agradat molt realitzar aquest projecte i la idea és continuar amb el seu desenvolupament per convertir el projecte RoMi en una empresa que ofereixi solucions de mesurament de qualitat de l'aire a baix cost.

En resum, vist els resultats obtinguts, es pot dir que els objectius establerts al inici del projecte s'han complert de forma extraordinària.

12. Treball futur

Tot i que els resultats obtinguts en el projecte han estat molt bons, encara queda molta feina a fer de cares al futur en cas que es vulgui continuar amb el projecte RoMi i comercialitzar-lo.

Les pròximes millores a realitza seran les següents:

- **Aplicació mòbil:** Actualment només es poden veure els valors de la qualitat de l'aire a través de les dues pàgines webs creades. Tot i que aquestes dues pàgines webs també es poden veure en els navegadors dels dispositius mòbils, amb una aplicació mòbil encara es podria millorar molt més la seva usabilitat.
- **Automatitzar la creació de tokens per a la API:** Ara mateix tots els usuaris que vulguin utilitzar la API s'han d'afegir manualment en al servei Amazon Cognito. De cares a un futur, si es vol obrir la API al públic perquè realitzin consultes el procés de creació dels tokens s'haurà d'automatitzar.
- **Millores de les pàgines webs:** Actualment només es poden veure les dades en temps real i l'històric de les dades representades a sobre el mapa. De cares a un futur, també estaria be incorporar un altre apartat del menú per poder visualitzar les dades en gràfiques i comparar els valors de contaminació entre diferents estacions.
- **Millora de la pàgina web RoMi Box:** Aquest pàgina web, a diferència de Respira .CAT, no es pot veure l'històric de les dades ja que les dades encara s'estan enviant a través del sistema antic i no s'està utilitzant la API. Un cop s'actualitzi el codi de les estacions i aquestes enviïn les dades a través de la API, llavors s'haurà de modificar la pàgina web perquè també permeti la visualització de l'històric de les dades.
- **Estacions de qualitat mòbils:** En cas que les estacions de qualitat siguin mòbils i incorporin un sensor GPS, llavors s'haurà de redissenyar la base de dades per poder emmagatzemar mesures amb posició GPS.
- **Anàlisi de dades:** Un cop s'hagin instal·lat més estacions de la xarxa RoMi a diferents localitats i amb més sensors de qualitat de l'aire, llavors es podrà crear una anàlisi utilitzant les dades provinents de la xarxa de sensors de RoMi.

13. Bibliografia

- Contaminació atmosfèrica.* (2017). Recollit de <https://www.eea.europa.eu/es/themes/air/intro>
- AEMA. (2017). *Contaminación atmosférica.* Recollit de <https://www.eea.europa.eu/es/themes/air/intro>
- AEMET OpenData. (2019). *Sistema para la difusión y reutilización de la información de AEMET.* Recollit de <https://opendata.aemet.es/centrodedescargas/inicio>
- Amazon Cognito.* (2019). Recollit de <https://aws.amazon.com/es/cognito/>
- Amazon Cognito Pricing.* (2019). Recollit de <https://aws.amazon.com/cognito/pricing/>
- Amazon API Gateway.* (2019). Recollit de https://aws.amazon.com/es/api-gateway/?nc1=h_ls
- Amazon API Gateway pricing.* (2019). Recollit de <https://aws.amazon.com/api-gateway/pricing/>
- Amazon EC2.* (2019). Recollit de <https://aws.amazon.com/es/ec2/>
- Amazon Elastic Container Service pricing.* (2019). Recollit de <https://aws.amazon.com/ecs/pricing/>
- Amazon RDS for MySQL Pricing.* (2019). Recollit de <https://aws.amazon.com/rds/mysql/pricing/>
- Amazon Relational Database Service (RDS).* (2019). Recollit de <https://aws.amazon.com/es/rds/>
- Amazon Web Services.* (2019). Recollit de https://en.wikipedia.org/wiki/Amazon_Web_Services

- Avaluació de la qualitat de l'aire.* (2019). Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/avaluacio/
- AWS Lambda.* (2019). Recollit de https://aws.amazon.com/es/lambda/?nc2=h_m1
- AWS Lambda pricing.* (2019). Recollit de <https://aws.amazon.com/lambda/pricing/>
- AWS.* (2019c). *Amazon RDS for MySQL Pricing.* Recollit de <https://aws.amazon.com/rds/mysql/pricing/>
- AWS.* (2019a). *Amazon API Gateway pricing.* Recollit de <https://aws.amazon.com/api-gateway/pricing/>
- AWS.* (2019b). *AWS Lambda pricing.* Recollit de <https://aws.amazon.com/lambda/pricing/>
- AWS.* (2019d). *Amazon Elastic Container Service pricing.* Recollit de <https://aws.amazon.com/ecs/pricing/>
- AWS.* (2019e). *Amazon Cognito Pricing.* Recollit de <https://aws.amazon.com/cognito/pricing/>
- AWS.* (2019f). *Amazon API Gateway.* Recollit de https://aws.amazon.com/es/api-gateway/?nc1=h_ls
- AWS.* (2019g). *AWS Lambda.* Recollit de https://aws.amazon.com/es/lambda/?nc2=h_m1
- AWS.* (2019h). *Amazon Relational Database Service (RDS).* Recollit de <https://aws.amazon.com/es/rds/>
- AWS.* (2019i). *Amazon EC2.* Recollit de <https://aws.amazon.com/es/ec2/>
- AWS.* (2019j). *Amazon Cognito.* Recollit de <https://aws.amazon.com/es/cognito/>

AWS Amazon RDS. (2019). *Réplicas de lectura de Amazon RDS*. Recollit de <https://aws.amazon.com/es/rds/details/read-rèplicas/>

Brussel·les porta a Espanya davant la justícia per la contaminació a Madrid i Barcelona. (2019). Recollit de https://elpais.com/sociedad/2019/07/23/actualidad/1563894873_941133.html

Dades Obertes Catalunya. (2019). *Dades d'immissió dels punts de mesurament de la Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica*. Recollit de <https://analisi.transparenciacatalunya.cat/Medi-Ambient/Dades-d-immissi-dels-punts-de-mesurament-de-la-Xar/uy6k-2s8r>

de Miguel, B., & Planelles, M. (24 / julio / 2019). *Brussel·les porta a Espanya davant la justícia per la contaminació a Madrid i Barcelona*. Recollit de https://elpais.com/sociedad/2019/07/23/actualidad/1563894873_941133.html

Departament de Territori i Sostenibilitat. (2019). *Dades d'immissió dels punts de mesurament de la Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica*. Recollit de <https://dev.socrata.com/foundry/analisi.transparenciacatalunya.cat/uy6k-2s8r>

Descarrega't l'aplicació mòbil Aire.Cat. (2019). Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/descarrega_app/

Generalitat de Catalunya. (2019c). *Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica (XVPCA)*. Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/avaluacio/xarxa_de_vigilancia_i_previsio_de_la_contaminacio_atmosferica_xvpca/

Generalitat de Catalunya. (2019a). *Avaluació de la qualitat de l'aire*. Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/avaluacio/

Generalitat de Catalunya. (2019b). *Zones de Qualitat de l'Aire (ZQA)*. Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/avaluacio/xarxa_de_vigilancia_i_previsio_de_la_contaminacio_atmosferica_xvpca/zones_de_qualitat_de_laire_zqa/

Generalitat de Catalunya. (2019d). *Vols saber què respires?* Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/vols-saber-que-respires/

Generalitat de Catalunya. (2019e). *Descarrega't l'aplicació mòbil Aire.Cat*. Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/descarrega_app/

GitHub. (2019). *GitHub Student Developer Pack*. Recollit de <https://education.github.com/pack>

GitHub Student Developer Pack. (2019). Recollit de <https://education.github.com/pack>

How to clone a private repository in GitHub. (2019). Recollit de <https://github.community/t5/How-to-use-Git-and-GitHub/Clone-private-repo/td-p/12616>

How To Install Nginx on Ubuntu 16.04. (2016). Recollit de <https://www.digitalocean.com/community/tutorials/how-to-install-nginx-on-ubuntu-16-04>

How To Secure Nginx with Let's Encrypt on Ubuntu 16.04. (2017). Recollit de <https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-16-04>

Installing Shiny Server Open Source. (2019). Recollit de <https://www.rstudio.com/products/shiny/download-server/>

Integrate a REST API with an Amazon Cognito User Pool. (2019). Recollit de <https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-enable-cognito-user-pool.html>

Internal compiler error. (sense data). Recollit de <https://stackoverflow.com/questions/30887143/make-j-8-g-internal-compiler-error-killed-program-cc1plus>

Meteocat. (2019). *API de Dades Meteorològiques.* Recollit de https://apidocs.meteocat.gencat.cat/#_ga=2.236969416.819219199.1567269619-888264179.1565890743

Nginx. (2019). Recollit de <https://ca.wikipedia.org/wiki/Nginx>

OpenWeather. (2019). *OpenWeather Map.* Recollit de <https://openweathermap.org>

Scrum. (2019). Recollit de <https://ca.wikipedia.org/wiki/Scrum>

Securing Shiny Server with a Reverse Proxy and SSL Certificate. (2019). Recollit de <https://www.digitalocean.com/community/tutorials/how-to-set-up-shiny-server-on-ubuntu-16-04>

SmartCAT Challenge 2018. (2018). Recollit de http://smartcatalonia.gencat.cat/ca/projectes/ciutats_i_regions/smartcat-challenge/edicions-anteriors/smartcat-challenge-2018/

Updating R on Ubuntu. (2018). Recollit de <https://www.r-bloggers.com/updating-r-on-ubuntu/>

Vols saber què respires? (2019). Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/vols-saber-que-respires/

Wikipedia. (2019a). *Scrum.* Recollit de <https://ca.wikipedia.org/wiki/Scrum>

Wikipedia. (2019b). *Amazon Web Services*. Recollit de https://en.wikipedia.org/wiki/Amazon_Web_Services

Xarxa de Vigilància i Previsió de la Contaminació Atmosfèrica (XVPCA). (2019). Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/avaluacio/xarxa_de_vigilancia_i_previsio_de_la_contaminacio_atmosferica_xvpca/

Yap, M. (2016). *Scaling Your Amazon RDS Instance Vertically and Horizontally*. Recollit de <https://aws.amazon.com/es/blogs/database/scaling-your-amazon-rds-instance-vertically-and-horizontally/>

Zones de Qualitat de l'Aire (ZQA). (2018). Recollit de http://mediambient.gencat.cat/ca/05_ambits_dactuacio/atmosfera/qualitat_de_laire/avaluacio/xarxa_de_vigilancia_i_previsio_de_la_contaminacio_atmosferica_xvpca/zones_de_qualitat_de_laire_zqa/