ENERQUIZ

TENS EL POWER?

MEMÒRIA

Cognom	Nom	Responsable	Upc e-mail	Taiga	GDrive	GitHub
Caminal	Imanol	Development 2	imanol.caminal@estudiantat.upc.edu	imanolcami	imanol.caminal@estudiantat.upc.edu	kuu02Real
Camps	Maria	Services	maria.camps@estudiantat.upc.edu	mcampsll	maria.camps@estudiantat.upc.edu	mcampsll
Domènech	Arnau	Inception 1	arnau.domenech@estudiantat.upc.edu	arnaudome	arnau.domenech@estudiantat.upc.edu	arna1d0me
Heras	David	Inception 2	david.heras.fernandez@estudiantat.upc.edu	davidheras	david.heras.fernandez@estudiantat.upc.edu	David-Heras
Pombero	Daniel	Demo	daniel.pombero@estudiantat.upc.edu	Daniips	daniel.pombero@estudiantat.upc.edu	Daniips
Roma	Isaac	Development 3	isaac.roma.granado@estudiantat.upc.edu	isaacroma	isaac.roma.granado@estudiantat.upc.edu	isaacroma
Ye	Sheng	Development 1	sheng.ye@estudiantat.upc.edu	guionwind	sheng.ye@estudiantat.upc.edu	guionwind

Índex

1. Requirements

1.1. General conception of the project

El projecte es centra en el desenvolupament d'una aplicació mòbil enfocada a promoure un ús responsable i sostenible de l'energia i a conscienciar als habitants de Catalunya sobre el malbaratament energètic.

Això ho aconseguirem mitjançant un "Quiz" amb una temàtica de preguntes relacionada amb la sostenibilitat, l'ús responsable de l'energia i el consum d'aquesta a Catalunya, concretament a cada municipi. Les preguntes podran variar segons la ubicació de l'usuari, ja que seran preguntes més enfocades al municipi on es trobi. Amb aquestes preguntes l'usuari podrà aconseguir punts per competir amb altres jugadors i guanyar premis que estaran destinats a promoure l'ús del transport públic.

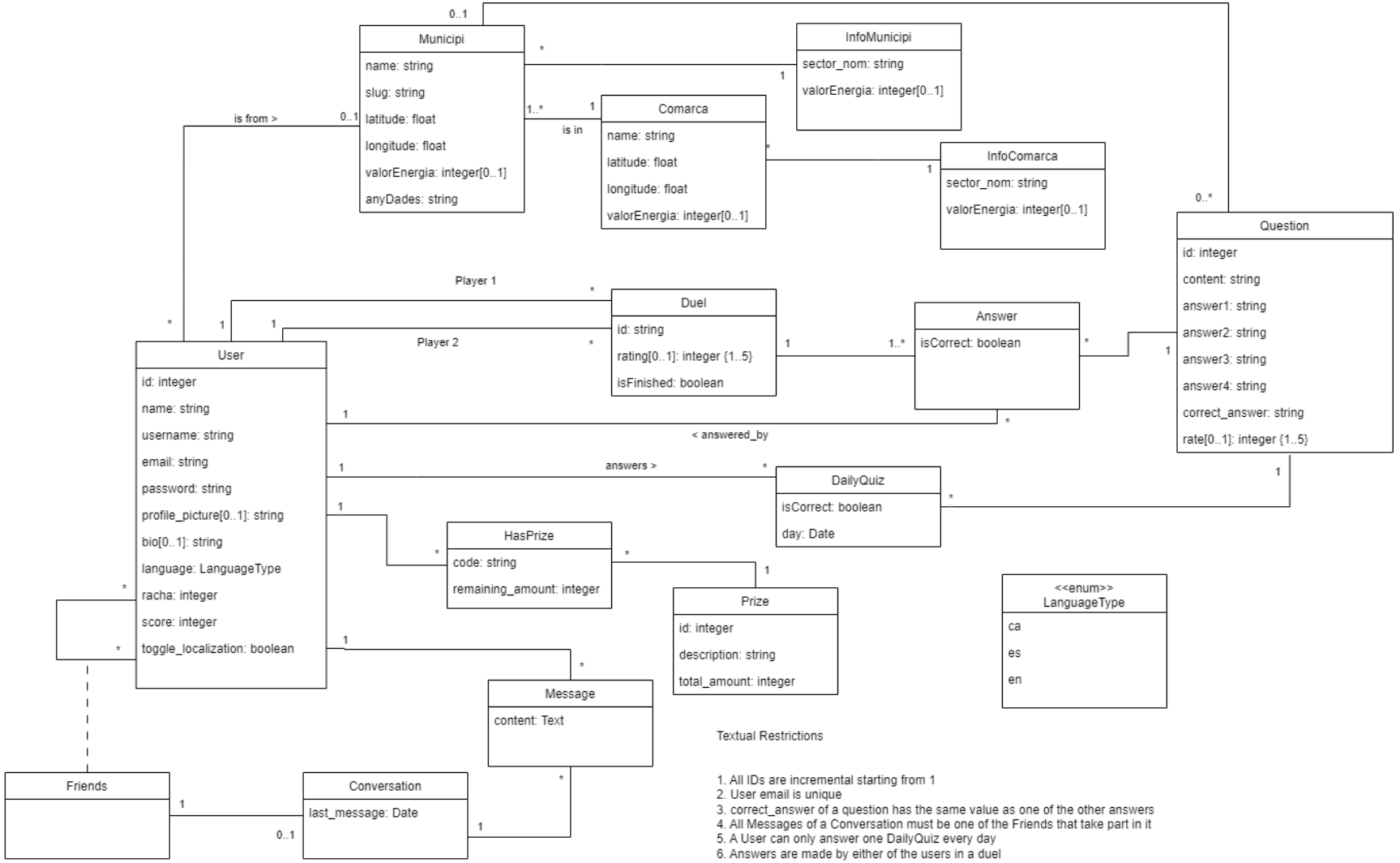
L'aplicació també permetrà a l'usuari navegar per un mapa interactiu de Catalunya en el que, per cada municipi, es mostrarà el consum energètic de la zona i com es reparteix aquest consum entre els diferents sectors. La informació del mapa serà molt útil als usuaris per tal de respondre correctament alguna de les preguntes.

1.2. “NOT” list

Funcionalitats IN	Funcionalitats OUT
• Sistema de qüestionaris sobre sostenibilitat • Premi per als millors jugadors cada mes • Mapa que mostra els consums energètics de diferents municipis i comarques • Xat a temps real amb amics • Login • Poder filtrar per municipis i comarques • Sistema multi idioma • Sistema de gestió d’usuari (perfils d’usuari) • Geolocalització per a oferir preguntes personalitzades • Rànquing de puntuació pels usuaris • Afegir i eliminar amics • Poder compartir esdeveniments a les xarxes socials • Valorar partida • Calendari que mostra quants dies ha fet l’usuari el daily quiz • Login amb les xarxes socials • Mostrar els punts de càrrega per municipi i estacions bicing • Partides multijugador • Xat amb històric • Moviment lliure pel mapa	• Abast de municipis fora de Catalunya • Sistema d’indicacions per mitjà de veu • Discriminació segons certificat energètic • Xat a temps real amb suport tècnic • Mostrar els edificis menys contaminants de cada municipi • Gràfic que classifica als municipis segons la seva valoració • Valorar rival • Calendari amb diferents activitats • Poder reportar queixes sobre l’aplicació

Funcionalitats UNSOLVED
• Mostrar els sectors més avançats en energies renovables de cada municipi • Sistema de notificacions(reps una notificació si tens una sol·licitud d’amistat, un nou missatge o el Daily Quiz està disponible)

1.3. Conceptual model



1.4. Summary of product backlog in the document

User management	US3, 7(1,1)	US3, 8(1,1)	US3, 10(1, 3)	US3, 11(1, 1)	US3, 12(1, 1)	US3, 13(1, 1)	US3, 14(1, 1)	US3, 15(2, 2)	US3, 16(2, 3)	US3, 17(2, 3)	US3, 18(2, 3)	US3, 19(2, 2)
Map management	US4, 20(1, 1)	US4, 22 (1, 1)	US4, 23 (3, 3)	US4, 52 (3, 3)								
Game management	US5, 24(1, 2)	US5, 25(2, 3)	US5, 26(2, 3)	US5, 27(2, 2)	US5, 28(13, 3)	US5, 30(13, 3)						
Ranking management	US6, 31(3, 3)	US6, 32(3, 3)	US6, 33(3, 3)	US6, 34(3, 3)								

1.5. Non-functional requirements

Els requisits no funcionals són requisits que especifiquen criteris que poden utilitzar-se per jutjar l'operació d'un sistema en lloc dels seus components específics. És a dir es refereixen a tots els requisits que descriuen característiques de funcionament.

1.1.1. Requisits de capacitat d'ús i humanitat

Número	**1**
Tipus de requisit	11a. Requisits de facilitat d'ús
Descripció	Descriu les aspiracions del nostre client sobre la facilitat tindran els usuaris previstos del producte utilitzar-lo. La usabilitat del producte es deriva de les habilitats dels usuaris esperats del producte i de la complexitat de la seva funcionalitat.
Justificació del requisit	El nostre sistema ha de ser fàcil d'utilitzar per tal de que els usuaris no deixin de fer servir l'aplicació al veure que és complicat el seu ús.
Condició de satisfacció	Els usuaris han de poder utilitzar el nostre sistema sense tenir cap tipus de coneixement previ de com es fa servir. És a dir, el sistema ha de ser extremadament intuïtiu.

Número	2
Tipus de requisit	11b. Requisits de personalització i internacionalització
Descripció	Descriu la manera com es pot modificar o configurar el producte per tenir en compte les preferències personals de l'usuari o l'elecció de l'idioma.
Justificació del requisit	Que l'usuari no hagi de lidiar o acceptar les convencions culturals del creador.
Condició de satisfacció	L'usuari podrà canviar l'idioma de l'aplicació així com la seva foto de perfil.

Número	3
Tipus de requisit	11c. Requisits d'aprenentatge
Descripció	Descriu la facilitat amb la que l'usuari aprèn a utilitzar la aplicació
Justificació del requisit	La nostra aplicació ha d'estar feta de forma que els usuaris puguin aprofitar les seves funcionalitats de manera ràpida i sense que suposi un gran esforç aprendre'n.
Condició de satisfacció	Els usuaris han de poder utilitzar l'aplicació completament sense cap tipus de formació previa.

1.1.2. Requisits de rendiment

Número	**4**
Tipus de requisit	12h. Requisits de longevitat
Descripció	L'aplicació s'ha de mantenir funcional en tot moment.
Justificació del requisit	Cal mantenir el sistema actiu durant un període llarg de temps per tal de complir els objectius que ens proposem amb el desenvolupament d'aquesta *App*.
Condició de satisfacció	L'aplicació es mantindrà funcional fins a acabar el projecte que consisteix aquesta assignatura.

1.1.3. Requisits de preservació i suport

Número	**5**
Tipus de requisit	14b. Requisits de suport
Descripció	Aquest requisit es centra en la quantificació del nivell de suport que necessita l'aplicació.
Justificació del requisit	En cas que el usuaris necessitin algun tipus d'ajuda o servei per part de l'equip de suport, ens hem d'assegurar de poder oferir-la o proporcionar solucions i consells per tal de poder solucionar el problema.
Condició de satisfacció	L'usuari tindrà a la seva disposició una pàgina de FAQ, amb la qual l'usuari obtindrà consells per poder resoldre la possible incidència.

1.1.4. Requisits de seguretat

Número	6
Tipus de requisit	15b. Requisits d'integritat
Descripció	Especificació de la integritat requerida de les bases de dades i d'altres arxius, així com del producte en si.
Justificació del requisit	L'aplicació ha de proporcionar dades correctes i íntegres, així com garantizar que les preguntes són correctes.
Condició de satisfacció	Es garantitzarà que les dades ofertes siguin correctes, les preguntes també. En un futur a mig-llarg termini, si els usuaris poden afegir preguntes, hi haurà un sistema per garantir la correctesa de les preguntes.

Número	7
Tipus de requisit	15c. Requisits de privacitat
Descripció	Especifica com el sistema garanteix la privadesa de la informació que emmagatzema dels clients.
Justificació del requisit	L'aplicació ha de complir amb les lleis de protecció de dades vigent. La privacitat és un dret fonamental i un valor important per la nostra empresa.
Condició de satisfacció	Tota la informació sensible serà encriptada i emmagatzemada correctament de manera segura, de forma que no supondrà un risc per l'usuari.

1.6. Treatment of transversal aspects

Aspect	Description	Current status
Geolocalització	S'utilitzarà la geolocalització per tal que, a l'usuari, li apareguin preguntes relacionades depenent d'on estigui.	Mapa implementat tant pel back com pel front. Les preguntes estan classificades per municipi però no s'utilitza per personalitzar preguntes.
Xarxes socials	Inici de sessió amb Google i compartir resultat d'una pregunta per Whatsapp i sms.	Implementada
Xat	Els usuaris podran parlar els uns amb els altres a través del xat.	Implementada
Gamificació	Hi haurà preguntes diàries sobre sostenibilitat disponibles per l'usuari. També podrà fer preguntes, desafiant així, als seus amics.	Implementada la part del front i back de les preguntes del Daily Quiz. La implementació de Duels ha quedat a mitges.
Stakeholders reals	CultuCat i GoVolt, utilitzarà un servei nostre i nosaltres utilitzarem un servei seu, respectivament.	Implementada
Refutació	L'usuari podrà valorar preguntes.	Implementada
Calendari	Hi haurà un calendari on es mostraran els dies els quals l'usuari ha contestat el Daily Quiz, on sortiran els dies en vermell o verd depenent si l'ha respost bé o malament. Blanc pel dia actual, si no ha contestat encara i gris pels dies que falten per arribar.	Implementada
Web-app admin	Hi haurà un administrador que podrà accedir a través de la web per administrar	No implementada.

	l’aplicació.	
Multiidioma	L'usuari podrà seleccionar l'idioma preferit entre el Català, Castellà i Anglès.	Implementada, amb bugs visuals menors.

1.7. Third-party services

Nosaltres oferim a CultuCat un servei on ells ens donen un municipi en concret i nosaltres, segons l’ubicació que ens han donat els hi retornem una pregunta de la nostra base de dades.

Integrem un servei de GoVolt, que ens proporciona la ubicació dels punts de càrrega situats a Catalunya.

Per comunicar-nos entre els altres equips, utilitzem el correu electrònic (gmail) i fem reunions presencials amb els responsables dels altres equips.

2. Methodology

En aquest apartat definirem quina metodologia estem seguint a l’hora de desenvolupar el projecte, així com la que seguirem un cop comencem els sprints i toqui desenvolupar codi.

2.1. Global vision

Per gestionar el nostre projecte i portar un correcte desenvolupament, hem decidit emprar la metodologia Scrum, dividint el projecte en dues fases d’incepció, on realitzarem la documentació necessària per començar el desenvolupament de l’aplicació, i 3 fases de Sprint, en les que es desenvoluparan tant el back-end com el front-end. Per cada fase hi ha un Scrum master que supervisa la feina de l'equip per veure que tothom treballa i es compleixen els terminis d’entrega del projecte. També tenim un responsable que es comunica amb els diferents equips de l’assignatura per tal de demanar funcionalitats necessàries per la nostra aplicació, així com per oferir serveis als altres equips.

Per tal de portar la feina al dia i treballar en equip hem decidit crear un servidor de Discord per tal de poder xatejar amb l’equip i fer reunions si és necessari, per veure quines aportacions noves té cada membre. De moment a classe intentem dividir la feina que ha de fer cada membre per la setmana vinent, i si sorgeixen dubtes o més coses per fer ho

comuniquem per Discord. Cap a l'inici del primer sprint, cada membre la de saber les tasques que ha de fer, i ha de completar-les donant preferència a les que són dependències d'altres.

Pel que fa als Sprints, un cop es comenci amb el codi, tenim pensat fer una reunió cada setmana de tot l'equip per comentar quines funcionalitats ha implementat cada membre i resoldre dubtes, així com per organitzar la feina de la setmana vinent. També farem reunions en petits equips, per si varis membres necessiten avançar conjuntament en el desenvolupament de l'aplicació. La divisió de feina la farem amb el Taiga on cada membre tindrà tasques assignades.

2.2. Project management

Com hem comentat anteriorment, per portar a terme el projecte farem servir la metodologia Scrum, on dividirem en diferents Sprints la feina a fer. Per tal de poder seguir aquesta metodologia farem servir l'eina Taiga, que ens permet crear i assignar històries d'usuari a diferents membres de l'equip, així com dividir-les per sprints, cosa que ens serà útil per tal d'organitzar la feina.

Cada història d'usuari tindrà uns punts, corresponents amb la càrrega de treball que creiem pot suposar el seu desenvolupament. La puntuació de les històries d'usuari va ser decidida pels membres de l'equip mitjançant un planning poker, que és una tècnica usada per quantificar l'esforç que pot suposar una determinada tasca. Per tal de poder quantificar bé les històries d'usuari vem decidir entre els membres del grup que 1 punt representaria una hora, per tal de valorar-les segons el temps que creiem que trigarem en fer-les.

Entre els membres de l'equip també hem acordat que significa que una historia d'usuari o un sprint s'hagi realitzat i per tant es doni per tancat, el que es coneix com a Definition of Done. Per tal de considerar una història d'usuari com a acabada s'han d'haver realitzat totes les tasques de la història corresponent i no hi ha d'haver errors ni bugs en el codi. Pel que fa als sprints, els considerarem acabats un cop s'hagin realitzat totes les històries d'usuari contingudes al sprint així com la documentació corresponent, el codi no te ni bugs ni errors i tots els tests realitzats al codi han estat passats amb èxit. No es pot passar al següent sprint fins que no es dóna per tancat el sprint en el que s'està treballant.

2.3. Repository management

Per al control de versions hem decidit utilitzar GitFlow. Aquest és un model de branquillatge que proposa una estratègia de ramificació i generació de versions de productes utilitzant un repositori Git. Tot i que GitFlow ofereix comandes per gestionar aquestes estratègies de manera directa, ràpida i eficient, nosaltres no les utilitzarem d'aquesta manera perquè farem servir també el sistema de pull requests, d'aquesta manera, podem tenir més control del codi que es puja a producció en tot moment.

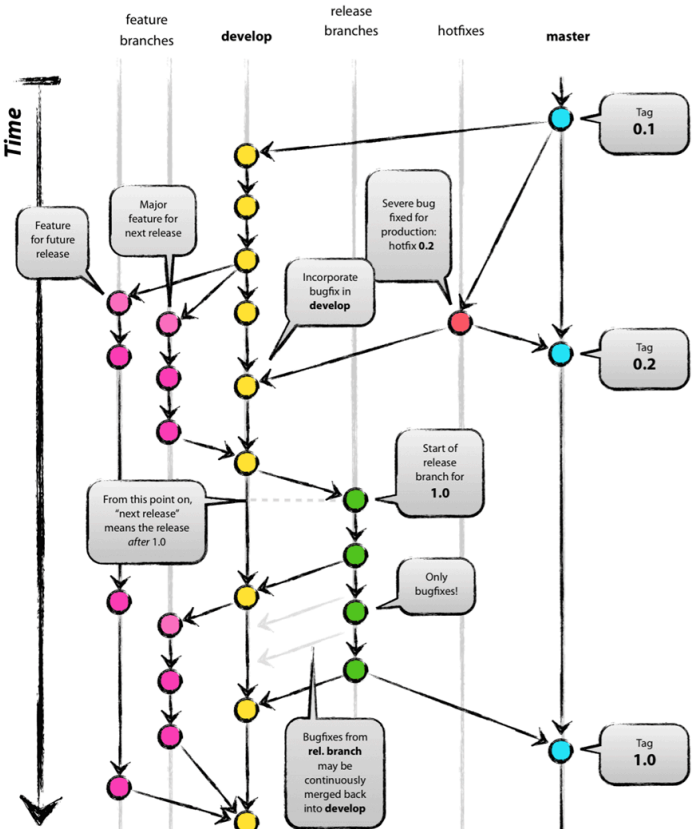
Les pull requests (a partir d'ara PR) són una funcionalitat que ofereix una interfície web intuïtiva per revisar i debatre els canvis proposats abans d' integrar-los al projecte. A través d'aquest mecanisme, quan una branca està llesta, els desenvolupadors notifiquen als membres de l'equip que han acabat una funcionalitat. D'aquesta manera, totes les persones involucrades saben que han de revisar el codi per fusionar-lo amb la branca corresponent.

Com hem comentat, una PR avisa als revisors a través d'una notificació i serveix com a fòrum per debatre sobre una funcionalitat proposada. En ella, els membres de l'equip poden publicar el seu feedback realitzant recomanacions per millorar el codi sempre que ho considerin necessari o simplement acceptar la PR si ho veuen tot correcte. El seguiment de tota aquesta activitat es realitza directament des de la PR a GitHub.

Dins del context de GitFlow, es divideixen les branques en dos tipus principals:

Les **branques principals** són la "main" i "develop". Només existeix una branca de cada tipus. Ningú fa commits directament en aquestes branques, i la incorporació de codi només es permet a través d'una pull request (PR). A més, el nom de la branca coincideix amb el seu tipus, és a dir, "main" o "develop".

- Branca **"main"**:
  - Conté el codi que està actualment en producció o que està a punt de ser-hi.
  - Mai s'elimina aquesta branca.
- Branca **"develop"**:
  - Es genera a partir de "main".
  - Conté el codi que encara no és a producció, però ja té una qualitat suficient, és completa i funcional.



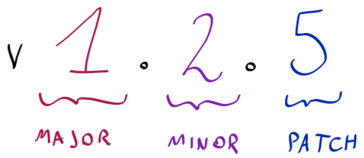
    - El seu propòsit és mantenir "main" net i evitar conflictes.
    - Mai s'elimina aquesta branca.

D'altra banda, les **branques auxiliars** s'utilitzen per a tasques específiques i desenvolupaments temporals. Es poden crear tantes d'aquestes branques com sigui necessari, i el seu propòsit principal és permetre la feina directa en elles sense necessitat de generar una PR. El nom d'aquestes branques es compon del seu tipus i objectiu, que pot ser una funcionalitat o una versió específica. Per exemple, "feature-484-alta-usuario".

- Branca **"feature"**:
    - Es genera a partir de "develop".
    - El seu nom ve determinat pel codi i el nom de la tasca de Taiga, amb "feature/" a l'inici. Per exemple: feature/FAM-454-Alta-usuario.
    - Conté codi actiu que actualment s'està desenvolupant.
    - Es creen per a desenvolupaments curts i no s'han de prolongar.
    - Quan es fa el merge d'aquesta branca a "develop", s'ha d'eliminar del repositori remot.
- Branca **"release"**:
    - Es genera a partir de "develop" quan es finalitza el desenvolupament d'un conjunt de funcionalitats (features).
    - El seu nom ve determinat per la versió del producte, amb "release/" a l'inici. Per exemple: release/10.1.10.
    - Permet "congelar", provar i depurar una versió abans de passar a "main".
    - Només es poden fer canvis de configuració en aquesta branca.
    - És la branca que s'implanta en preproducció per fer les proves de QA.
    - Al final de les proves, es fa el merge amb "main" i amb "develop" (si s'han fet canvis), i es genera una etiqueta (tag) amb aquesta versió.
- Branca **"hotfix"**:
    - Es crea una branca quan es detecta un problema urgent a producció que ha de ser resolt immediatament.
    - El codi d'aquesta branca prové de "main".
    - El nom de la branca pot ser la versió correctora, amb "hotfix/" a l'inici. Per exemple: hotfix/10.1.11.
    - Un cop corregit, es fa el merge a través d'una PR a "main" i a "develop", i es genera una etiqueta (tag) amb aquesta versió.
    - Si hi ha una branca "release" preparada, també es fa el merge en ella per tenir aquesta correcció.

- Un cop fusionada a "main", aquesta versió s'implanta primer a preproducció per fer les proves de QA.
- Si tot està bé, s'implanta a producció.
- Quan es fa el merge d'aquesta branca a "main", s'ha d'eliminar del repositori remot.

Per a l'etiqueta de les versions utilitzarem la següent nomenclatura:



PATCH: Augmenta només quan es corregeixen errors que no modifiquen cap dels mètodes públics, és a dir, no realitzen canvis en el comportament.
MINOR: S'incrementa quan s'afegeix una nova funcionalitat o si algun mètode es marca com a obsolet i sigui compatible amb la versió anterior.
MAJOR: S'incrementa quan es produeix un canvi incompatible amb alguna versió anterior; poden incloure canvis minors i patch

2.4. Communication within the team

Per comunicar-nos entre membres de l'equip hem creat un servidor de Discord on tenim diferents canals de text per poder parlar sobre diferents temes o entre membres específics de l'equip. Hem creat un canal per l'equip de backend, un pel de frontend, un per parlar sobre possibles bugs de l'aplicació i un ultim per organitzar reunions. A part d'aquests canals de text també hem creat varis canals de veu per poder fer reunions entre els membres de l'equip i poder treballar de manera conjunta amb altres membres.

2.5. Quality management

La IDE que utilitzarem per a programar el backend en el nostre projecte és *PhpStorm,* la qual és una IDE desenvolupada per la companyia de *JetBrains*. Més concretament, farem servir les eines d'anàlisi de codi que ens proporcionarà aquesta IDE. Amb aquestes eines podrem detectar errors en el nostre codi de manera més fàcil. Per exemple, es ressaltaran errors de sintaxi, problemes d'estil, problemes de rendiment, etc. A més també podrem autocompletar codi a partir de suggeriments del mateix programa, o fins i tot podrem fer una refactorització de parts del codi perquè aquest sigui més òptim i fàcil de llegir.

També, a part de tot això, amb aquesta IDE tindrem facilitats sobre l'escriptura i execució de proves unitàries, així com informació sobre la cobertura del codi, la qual cosa ajuda a garantir que totes les parts importants del codi estan sent testejades.

La principal IDE que farem servir al front-end, Android Studio, té eines similars.

2.6. Testing strategy

Per poder comprovar que les funcionalitats implementades funcionin correctament, realitzarem tests unitaris i d'integració amb Laravel PEST, un framework elegant per realitzar tests. Hem decidit utilitzar Laravel PEST ja que un membre de l'equip està bastant familiaritzat amb aquest framework i ens ha recomanat el seu ús.

Al front per tal de poder fer test, hem provat les funcionalitats al emulador, i un cop pujades les branques a Github, hem demanat als nostres companys que les provin per poder estar segurs que funcionen abans de fer merge a master o develop. Comprovem casos extrems d'ús, per assegurar-nos de que cap bug important s'escapi.

2.7. Management of configurations

Pel que fa a la gestió de configuracions al back del nostre projecte utilitzarem la carpeta config que ofereix Laravel, ja que conté arxius per configurar diferents aspectes de l'aplicació. L'accés local a la base de dades es gestiona amb l'arxiu .env, que es troba a la carpeta arrel.

Al front, s'utilitzaran les eines de React Native, que és el framework en el que treballarem.

Amb aquestes pràctiques podem gestionar eficientment les configuracions en el nostre projecte, assegurant-nos que cada entorn tingui les configuracions adequades.

2.8. Interaction with colleagues

La major part de les interaccions i comunicacions amb altres equips es van dur a terme principalment per correu electrònic amb els representants de negociacions. Tot i que es van mantenir algunes trobades en persona durant les sessions de classe, aquestes van ser relativament curtes i es van utilitzar principalment per aclarir detalls o qüestions puntuals. En

resum, la comunicació predominant es va dur a terme a través del correu electrònic, amb algunes interaccions informals i breus cara a cara.

2.9. Bug management

Gestionarem els bugs del nostre sistema mitjançant l'ús del nostre servidor de Discord, i en menor mesura amb els issues de Taiga. L'avantatge d'utilitzar Discord és que podem parlar directament amb la persona encarregada de solucionar-lo. Normalment, aquesta persona serà la persona encarregada de desenvolupar la part del sistema on s'ha trobat el *bug*.

Pel bugpost s'explicarà quina funcionalitat té el problema i amb quin model de dades està relacionat. A través del canal de text *bugs* de Discord. Aquest canal, com tots els del servidor, el pot veure tot l'equip, i és una manera d'assegurar-nos que tothom estigui al corrent.

2.10. NFRs treatment

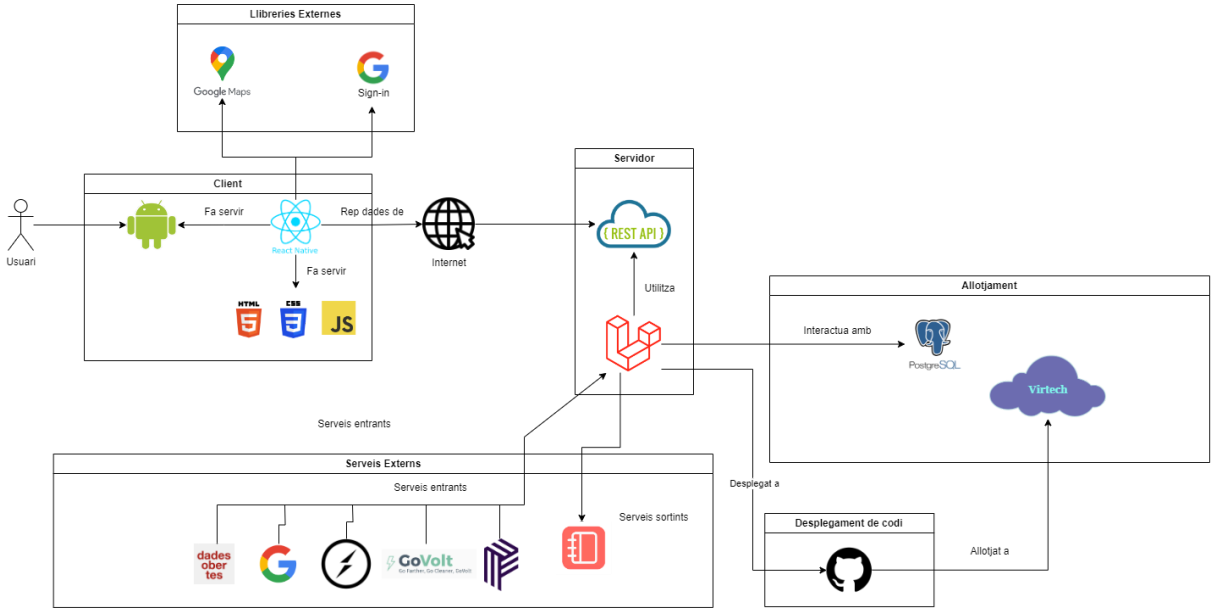
Per tal de tractar els requisits no funcionals del projecte seguirem una sèrie de passos. El primer que farem serà elaborar una llista de tots els requisits no funcionals que seran rellevants al nostre projecte. Un cop identificats els classificarem segons la seva importància i prioritat per a l'èxit del projecte. També avaluarem l'impacte de cada un a l'arquitectura i el disseny de l'aplicació. Seguidament establirem mètriques específiques i quantificables per avaluar el compliment de cada requisit, i establirem objectius clars per cada requisit no funcional. A l'hora de fer el disseny de l'arquitectura de l'aplicació, l'adaptarem per tal de que compleixi els requisits identificats. Per últim realitzarem proves per verificar el compliment dels requisits no funcionals.

3. Technical description

3.1. Overall conception of the architecture

3.1.1. Physical architecture

A continuació mostrem el diagrama de l'arquitectura física que defineix el nostre projecte.



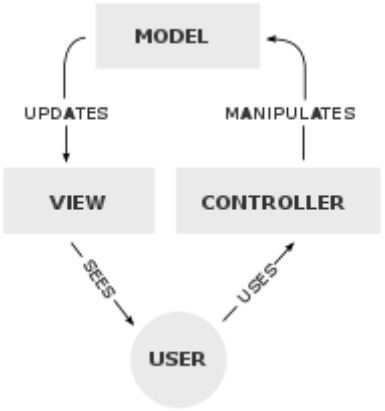
Pel que fa al front, hem fet servir React Native, tecnología la qual també fa servir CSS, HTML i JavaScript. Les llibreries externes implementades que utilitza el front actualment són les de Google Maps i Google Sign-in.

Pel que fa a per la part del servidor, hem continuant usant Laravel amb PhpStorm com a IDE principal per al back, i de moment tenim com a serveis entrants les dades obertes de Catalunya. Per a fer la interacció amb la base de dades hem fet servir una base de dades relacional amb MySQL com a driver. Això ho hem fet gràcies al fet que Laravel treballa naturalment amb MySQL.

Per la connexió entre Front i Back hem decidit seguir amb Virtech, ja que després de provar Docker i AWS, és l'eina que ens ha semblat més senzilla i amb la que hem pogut aconseguir fer el desplegament del backend.

3.1.2. Architectural pattern(s) applied

L'arquitectura de EnerQuiz està basada en el patró Model-View-Controller (MVC), on les dades estan representades per models, tractades pels controladors i mostrades per les vistes. El backend es composa dels models i els controladors. Això també inclou la base de dades on desem la informació de la nostra app. El front és el conjunt de vistes que formen la interfície gràfica entre l'usuari i el servidor web, i que s'encarrega de comprovar i enviar les peticions dels usuaris cap als controladors.



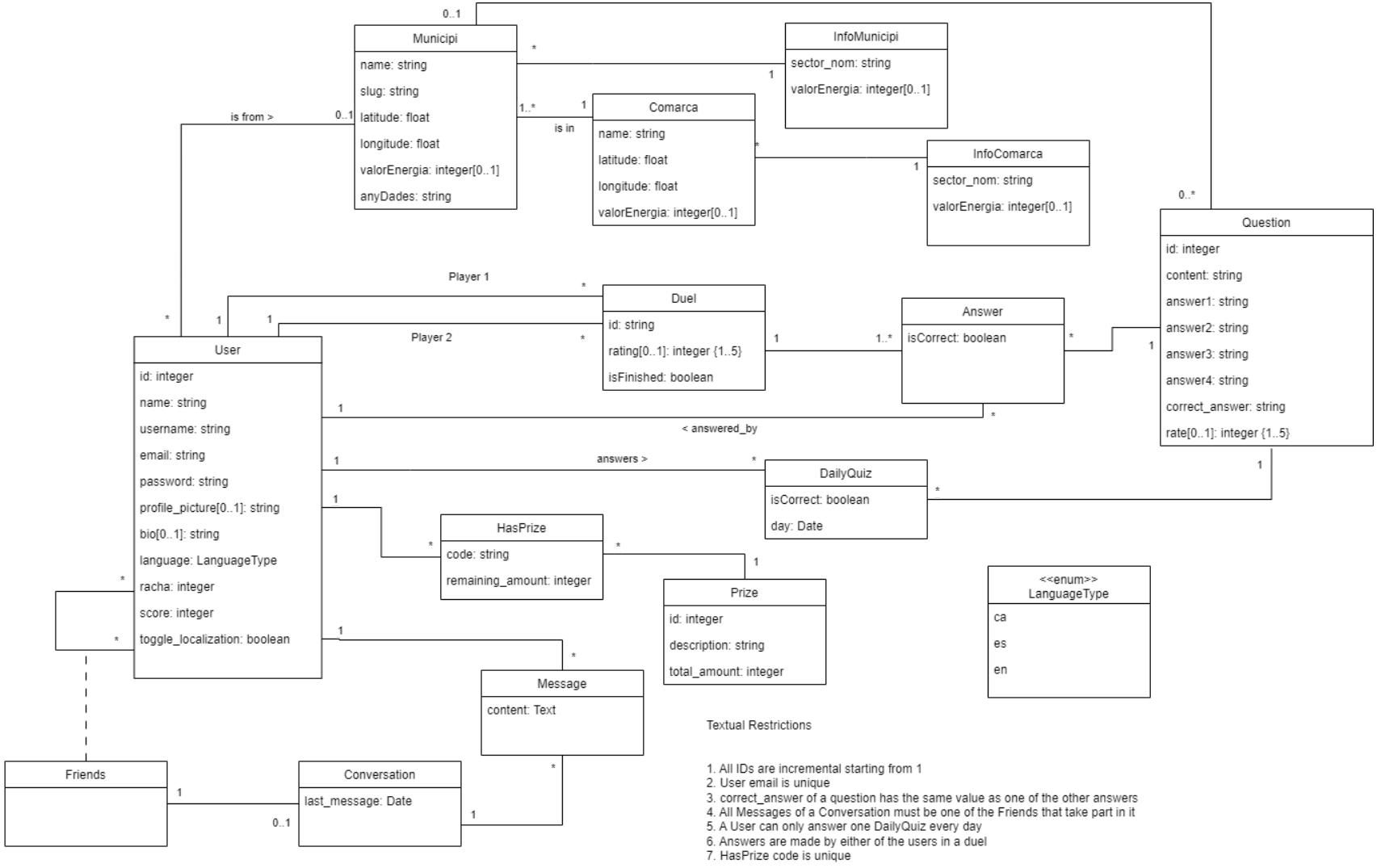
Fem servir el patró MVC ja que ha estat àmpliament utilitzat per a desenvolupar aplicacions web, un cas molt similar a la nostra aplicació. Laravel, el nostre framework de back, està pensat precisament per a fer servir aquesta arquitectura, i per tant té moltes funcionalitats que li donen suport. Per tant, és una forma pràctica i convenient de estructurar el nostre programa. A més, ens permet reutilitzar codi, facilitar el testing i separar bé les funcionalitats, seguint els principis de la programació orientada a objectes.

3.2. Domain layer

3.2.1. Design patterns applied

En la realització del projecte hem aplicat el patró de disseny Adaptador, a l'hora de tractar les dades proporcionades per dades obertes per tal de poder-les utilitzar sense problema a la nostra aplicació. Hem aplicat aquest patró ja que ens ha fet falta tractar els noms de les comarques i municipis per poder fer funcionalitats amb ells sense problema, com per exemple a l'hora de passar la funcionalitat a un altre grup, on hem hagut de fer un slug dels noms de municipis i comarques, substituint els espais per '_'.

3.3. Database diagram (UML)



3.4. Instrumentation and list of technologies

3.4.1. IDEs

3.4.1.1 Backend

Hem decidit que l'entorn de desenvolupament que utilitzarem pel backend de la nostra aplicació será PhpStorm. PhpStrom és un entorn de desenvolupament dissenyat específicament per al desenvolupament d'aplicacions web i projectes en PHP. Aquest IDE va ser desenvolupat per JetBrains.

Al ser desenvolupat per JetBrains, no tindrem problemes en aconseguir-ho, ja que ofereixen llicències per estudiants.

PhpStorm ofereix avantatges en comparació a altres IDES, entre ells destaquen:

- Suport PHP d'alta qualitat: PhpStorm ofereix un suport excepcional per a PHP, incloent auto completament intel·ligent, inspecció de codi entre altres.
- Depuració eficaç: PhpStorm proporciona eines com Xdebug per facilitar la detecció i correcció d'errors.
- Compatibilitat: PhpStorm és compatible amb tecnologies web com html, css o javascript i també amb altres frameworks com Laravel.
  També es compatible amb eines de control de versions com git, svn…

També té altres avantatges com el suport continu, la fàcil integració amb bases de dades o les eines de refactorització que ofereix.

3.4.1.2 Frontend

En el frontend de la nostra aplicació utilitzarem dos entorns de desenvolupament, Visual Studio Code i Android Studio. Hem decidit usar aquests dos ja que cada membre de l'equip tenia preferències diferents.

Visual Studio Code és un entorn de desenvolupament molt popular creat per Microsoft. Entre les seves característiques principals trobem que soporta una gran varietat d'extensions que ens poden ajudar molt a l'hora de programar. També ofereix un editor de text avançat capaç de reconèixer molts llenguatges, entre ells els que utilitzarem, i que per tant ofereix autocompletat intel·ligent per fer-nos més senzilla l'escriptura de codi. També té moltes altres característiques com suport per a desenvolupament web, terminal integrada e integració amb Git.

Andoid Studio és un entorn de desenvolupament creat per Google que es basa en IntelliJ IDEA. Entre les seves característiques principals trobem que ofereix emuladors i dispositius virtuals per tal de provar les aplicacions. Al igual que Visual Studio Code, ofereix també un

editor de text avançat. Té moltes més característiques com integració amb Google Services, depuració integrada i sistema de construcció Gradle.

3.4.2. Llenguatges

Els llenguatges que hem decidit pel nostre projecte són Laravel (PHP) com a framework pel backend; i React Native amb Tailwind CSS, TypeScript i HTML pel frontend.

Pel que fa a Laravel, és un framework PHP conegut per la seva robustesa i escalabilitat. Proporciona una estructura organitzada que facilita el desenvolupament d'aplicacions web i APIs. Ofereix funcionalitats de seguretat incorporades, com a la protecció contra atacs CSRF (*Cross-site request forgery)* i XSS(*cross-site scripting)*, a més d'una ORM (*Object-Relational Mapping*) anomenada Eloquent que facilita la interacció amb base de dades, reduint la complexitat de les consultes SQL. A part de totes les qualitats i facilitats que ens ofereix utilitzar aquesta tecnologia, una de las raons clau de la nostre elecció ha sigut que un dels integrants de l'equip la fa servir a la feina i està molt familiaritzat.

En el cas de React Native per al frontend va ser una decisió ràpida i sòlida ja que té la capacitat de desenvolupar aplicacions mòbils de manera eficient per a la plataforma Android i també per web, aquest enfocament de desenvolupament multiplataforma estalvia temps i recursos. A més, ofereix un bon rendiment i una experiència d'usuari àgil ja que, fa servir components natius de cada plataforma. A part, disposa d'una comunitat activa i una gran quantitat de llibreries que facilita la integració de funcionalitats addicionals a la nostra aplicació.

En quant a les tecnologies de disseny i estil, hem seleccionat Tailwind CSS, aquest marc de disseny utilitari es destaca per la seva eficiència, ja que es basa en classes reutilitzables i personalitzables que acceleren el desenvolupament de la interfície d'usuari. Typescript també serà utilitzat per millorar la qualitat del nostre codi, ja que afegeix tipat estàtic, ajudant a identificar errors abans de l'execució del codi. A més, completarem aquesta combinació amb HTML per estructurar i millorar l'experiència d'usuari.

Per concloure, la combinació de Laravel al backend i React Native amb Tailwind CSS, Typescript i HTML al front ens proporciona una base sòlida per al desenvolupament del nostre projecte. Aquesta elecció ens permetrà avançar de manera eficient, garantir la seguretat de les dades i oferir una experiència d'usuari de qualitat en dispositius mòbils i

web. A més, el suport actiu i l'abundància de recursos disponibles en les comunitats d'aquestes tecnologies seran un valor afegit per al nostre projecte.

3.4.3. Gestió de bases de dades

En quant a la gestió de bases de dades utilitzarem MySQL. MySQL és un sistema de gestió de base de dades relacionals de codi obert i d'alt rendiment.

3.5. APIs

Per al nostre projecte hem utilitzat una API REST endpoint de dades obertes proveïdes per l'Institut Català d'Energia (ICAEN). Més concretament, hem utilitzat dades relacionades sobre el consum d'energia elèctrica per municipis i sectors de Catalunya. Per a accedir a aquestes dades hem fet servir una API endpoint atorgada per la mateixa generalitat. Aquesta endpoint ens atorga un total de 45000 tuples aproximadament, on es contenen les dades sobre el consum elèctric de la gran majoria de municipis i sectors de Catalunya. Analitzant una mica les dades ens vam adonar que hi havia moltes tuples on es donava la mateixa informació energètica, però en anys diferents. Per tant, vam decidir emprar les dades més noves datades de l'any 2022.

Amb aquestes dades hem aconseguit guardar a la nostra base de dades la següent informació: el consum elèctric total de totes les comarques de Catalunya, el consum elèctric total de tots els 947 municipis de Catalunya, el consum total de diferents sectors de tots els municipis i el consum total de diferents sectors de les comarques. Amb tota aquesta informació hem pogut crear diversos models de preguntes per a la nostra app. Cal mencionar que d'algunes de les comarques no hem pogut obtenir les dades del sector de 'Transport' ja que certes comarques han decidit mantenir les dades en secret. Per tant, a algunes tuples en comptes de sortir el valor energètic es retorna el missatge: 'Dada subjecta a secret estadístic'.

També estem utilitzant Google Maps per tot el que fa referència a la funcionalitat del mapa. De Google també utilitzem la API per fer login amb les credencials del compte de Google. També utilitzem la API de Whatsapp per poder compartir via aquest la resposta d'una pregunta.

Per últim, tenim els serveis als stakeholders del grup de PES. Utilitzem un servei extern, provinent de GoVolt, que ens envien la ubicació de tots els punts de càrrega a Catalunya, i

oferim un servei extern a CultuCat, on li proporcionem una pregunta sobre l'ús d'energia a Catalunya sobre el municipi que ells ens demanen en la seva crida.

3.6. Development tools and working environment

En el desenvolupament de la nostra aplicació hem utilitzat diferents eines, que varien segons el Frontend i el Backend. A continuació es troba l'explicació de cada una de les eines usades.

3.6.1 Frontend

Al Frontend hem fet servir React Native per desenvolupar l'aplicació ja que és un framework molt conegut i utilitzat que s'usa per crear aplicacions mòbils de manera casi nativa. Ara bé això només és el framework de desenvolupament, per poder escriure codi hem usat Javascript, Typescript, Tailwind CSS, que és un framework de CSS, i HTML ja que són els llenguatges principals de React Native.

Per tal de poder programar, hem utilitzat Visual Studio Code i Android Studio, depenent de les preferències de cada membre de l'equip.

3.6.2 Backend

Al Backend hem utilitzat Laravel, que és un framework de PHP que s'ha tornat molt popular a la comunitat de desenvolupadors, i que ja era conegut per un dels membres de l'equip. Per programar hem utilitzat l'IDE PHPStorm.

3.6.3 Deploy

Per fer el deploy, hem usat Virtech, on hem fet el desplegament del backend en un servidor, al qual ens connectem desde el frontend per mitjà dels endpoints per fer les crides necessàries. Pel que fa al deploy del frontend hem generat una apk, un executable que funciona en dispositius mòbil Android o simuladors d'Android.