TecnoCampus
Escola Superior
Politècnica

Centre adscrit a la
upf. Universitat
Pompeu Fabra
Barcelona

Grau en Enginyeria Informàtica de Gestió i Sistemes d'Informació

Concerts

Memòria

Sergi Obiols Olcina

Tutora: Catalina Juan Nadal

2020/2021

TecnoCampus
Mataró-Maresme

Dedicatòria

A la meva família, especialment la meva mare.

Agraïments

A amics i familiars per les propostes a millores i funcionalitats durant el projecte.

Abstract

This project aims to develop a mobile application and a management dashboard for artists that allows them to manage concerts and music festivals. Users, from the mobile application, will be able to buy, attend and view detailed information about the concerts. To achieve the initially established objectives, a set of requirements and use cases are defined, in addition to studying the most suitable technological tools for development.

Resum

Aquest projecte té com a objectiu el desenvolupament d'una aplicació mòbil i un dashboard de gestió per artistes que els hi permeti gestionar concerts i festivals musicals. Els usuaris, des de l'aplicació mòbil, poden comprar, assistir i visualitzar informació detallada dels concerts. Per assolir els objectius establerts inicialment, es defineixen un conjunt de requisits i casos d'ús, a més d'estudiar les eines tecnològiques més idònies per al desenvolupament.

Resumen

Este proyecto tiene como objetivo el desarrollo de una aplicación móvil y un dashboard de gestión para artistas que les permite a ellos gestionar conciertos y festivales musicales. Los usuarios, desde la aplicación móvil, pueden comprar, asistir y visualizar información detallada de los conciertos. Para alcanzar los objetivos establecidos inicialmente, se definen un conjunto de requisitos y casos de uso, además de estudiar las herramientas tecnológicas más idóneas para el desarrollo.

Índex

Índex de figures

Índex de taules

Glossari de termes

AWS Amazon Web Services

Sistema cloud d'Amazon que proporciona plataformes, emmagatzematge i APIs. [1]

S3 Sistema cloud d'emmagatzematge de fitxers d'Amazon. [2]

REST Representional State Transfer. [3]

Estil arquitectònic per proporcionar estàndards entre sistemes informàtics a la xarxa, facilitant la comunicació dels sistemes entre ells. [3]

JWT Json Web Token. [4]

Es un estàndard obert basat en JSON proposat per IETF (RFC 7519) per a la creació de tokens d'accés que permeten la propagació d'identitat i privilegis.

1. Introducció

1.1 Objecte del projecte

El treball en qüestió sorgeix per donar resposta a les necessitats de difusió de concerts musicals per part de productores i artistes.

El principal objectiu és desenvolupar una aplicació mòbil per Android, tant per artistes com per usuaris, juntament amb un Dashboard que poden iniciar sessió tots aquells artistes que volen crear un esdeveniment o concert, i poder especificar més informació sobre l'esdeveniment.

Es recullen les següents dades a l'hora de crear un concert:

- **Descripció i data del concert**. En aquesta part es recull el nom del concert, la data i una breu descripció.

- **Cover o imatge del concert**. Una imatge identificadora de l'esdeveniment que els usuaris veuen des de l'aplicació.

- **Lloc on es durà a terme**. Es mostra un mapa on l'artista o equip que ho gestiona pot ubicar el concert.

- **Descripció de l'establiment o espai**. Cal especificar les facilitats que disposa la ubicació. Que millor que poder adjuntar imatges sobre el lloc on es fa el concert.

  L'artista té l'opció de posar en venta diferents tipus d'entrades a diferents preus o aplicar un descompte.

- **Entrades.** Aquesta secció l'artista pot afegir tants tipus d'entrades com vulgui que venen determinades per els següents camps:

    - **Nom.** Per identificar el tipus d'entrada, com per exemple, VIP, estandar.

    - **Descripció.** Una breu explicació de les avantatges o del que inclou aquesta entrada.

    - **Número d'entrades disponibles.** Per limitar el nombre màxim d'assistents.

    - **Preu**. Les entrades tenen el seu preu corresponent.

- **Artistes que hi participen**. L'artista pot organitzar un concert sol o especificar quins altres artistes també hi participen.

D'altra banda, l'aplicació mòbil es divideix en dos rols, el rol d'usuari i d'artista.

Un usuari de l'aplicació és aquell que vol assistir o té intenció d'assistir a un concert creat per un artista.

També cal contemplar l'opció que aquest usuari només vol consultar informació d'algun concert en concret o veure els que hi ha més propers a la seva ubicació actual.

Pel que fa el rol d'artista a l'aplicació, un artista és un usuari que s'ha registrat com a artista des del dashboard.

Les seves funcionalitats són completament diferents de les d'un usuari. Un artista pot veure activitat relacionada amb els concerts que ha organitzat o participa juntament amb altres artistes, com per exemple, veure els comentaris que han escrit els usuaris que han assistit als seus concerts, veure quantes entrades han comprat i de quin concert.

Amb aquest projecte es vol aconseguir unificar tots els treballs dels artistes en una aplicació específica, tal com fa l'aplicació Spotify amb les cançons, en aquest cas, amb els concerts. Que un usuari pugui saber en tot moment on, quan, qui i preus per assistir a un concert en qüestió de segons.

Per assolir els objectius, s'ha d'interioritzar els processos interns, dissenyar i implementar el software.

1.2 Context

Actualment el mercat és escàs, però es poden llistar grans aplicacions que poden complementar el nostre objectiu principal. A continuació es llisten juntament amb una breu descripció funcional:

- **Tripadvisor**
    - Aplicació mòbil per planificar les vacances i reservar un viatge. Gran ventall de recomanacions durant la teva estada, des d'un passeig en bici a algun dels parcs de la ciutat que tinguis planejat viatjar, fins a un vol per sobre de Nova York.
    - Gran quantitat d'usuaris registrats i més de cent milions de descàrregues que ajuden a saber si un esdeveniment és de qualitat gràcies al sistema de comentaris i recomanacions.

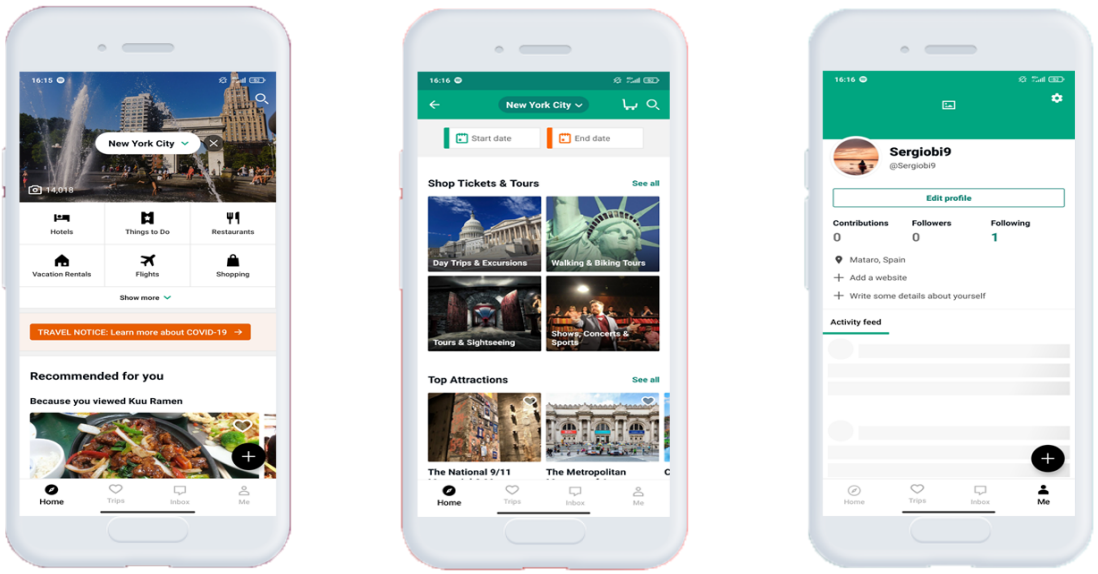

Fig 2.1 Captures de pantalla aplicació Tripadvisor. Font: Elaboració pròpia

- **Eventbrite**

    - Aplicació mòbil per descobrir pròxims esdeveniments a prop teu. Gran llistat d'esdeveniments a escollir. Pots registrar-te a un i obtenir la teva entrada instantàniament.

    - Aplicació bastant completa. No és necessari iniciar sessió per tal de poder buscar el que més t'agrada. No tenen un mapa per mostrar els esdeveniments en punts i poder-te ubicar.

    - Disposen d'una gran quantitat d'usuaris i deu milions de descàrregues.

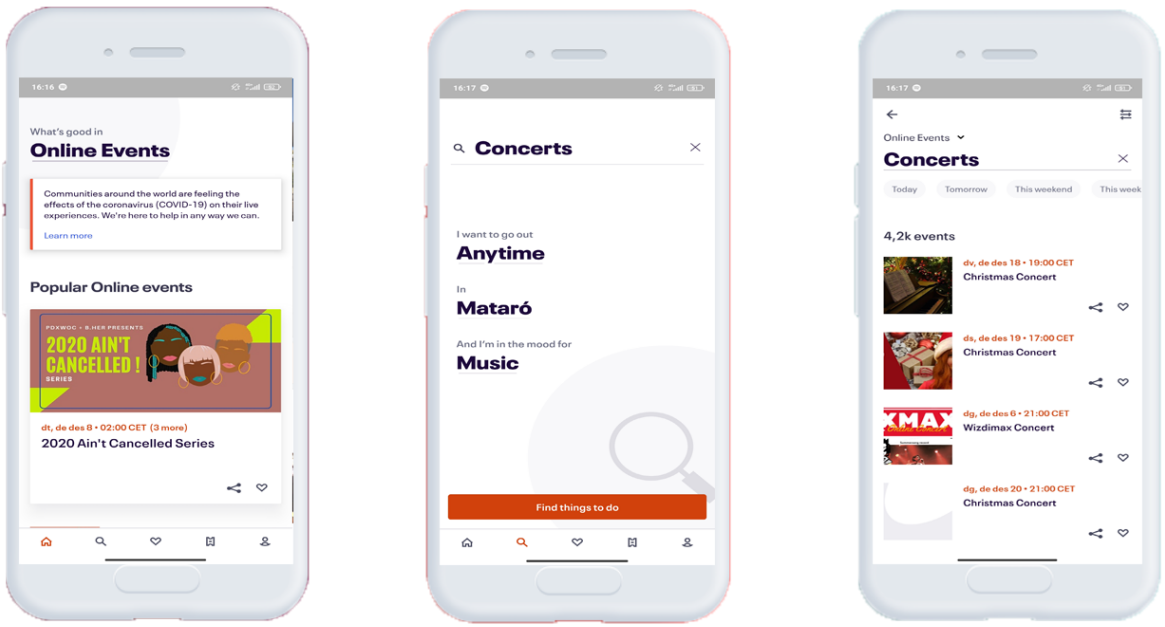

Fig 2.2 Captures de pantalla aplicació Eventbrite. Font: Elaboració pròpia

- **Meetup**

  - Aplicació mòbil per poder gaudir del que més t'agrada on vulguis, sigui online o en persona. Disposen d'una gran quantitat d'usuaris i deu milions de descàrregues.

  - Meetup també t'ajuda a crear una xarxa professional, a descobrir una comunitat tecnològica, a crear una marca personal, entre altres. Pots establir una conversa entre participants o dirigir-te al grup de persones que assisteixen al teu esdeveniment en cas de ser propietari.

  - No disposa d'un mapa per mostrar gràficament els punts on es situen els pròxims esdeveniments.

  - Gran llistat i propostes per assistir a esdeveniments, però poc focalitzat, pot acabar saturant l'usuari.

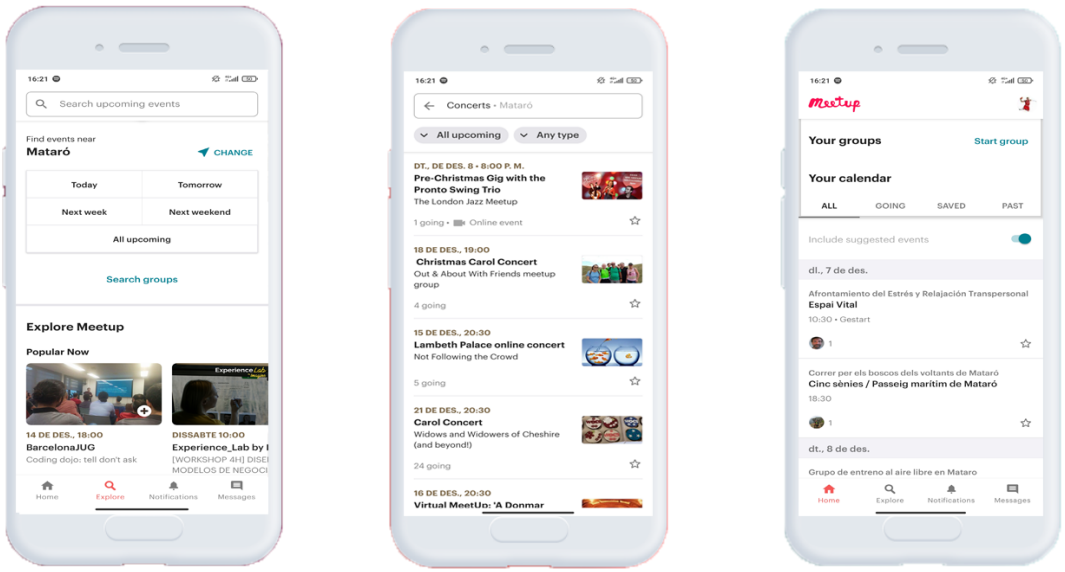

Fig 2.3 Captures de pantalla aplicació Meetup. Font: Elaboració pròpia

- **Facebook local**

    - Aplicació similar a les anteriors però de gran prestigi, ja que forma part de Facebook, un gegant en xarxes socials i difusió de contingut. Està en fase inicial.

    - És molt genèrica, un usuari pot descobrir entre quins són els millors restaurants de la teva zona, on poder prendre unes copes amb els teus amics, on poder passar una bona estona en un parc d'atraccions, fins a quins esdeveniments hi ha a l'escola dels teus fills.

    - Com es pot observar disposa d'un mapa, un gran avantatge, però com s'ha comentat anteriorment, hi ha molta varietat a l'hora de filtrar i per tant, l'usuari potser no s'arriba a decidir i saturar encara més l'usuari.

    - És una gran aplicació si l'usuari està obert a propostes, però de segur que moltes coses passen desapercebudes.

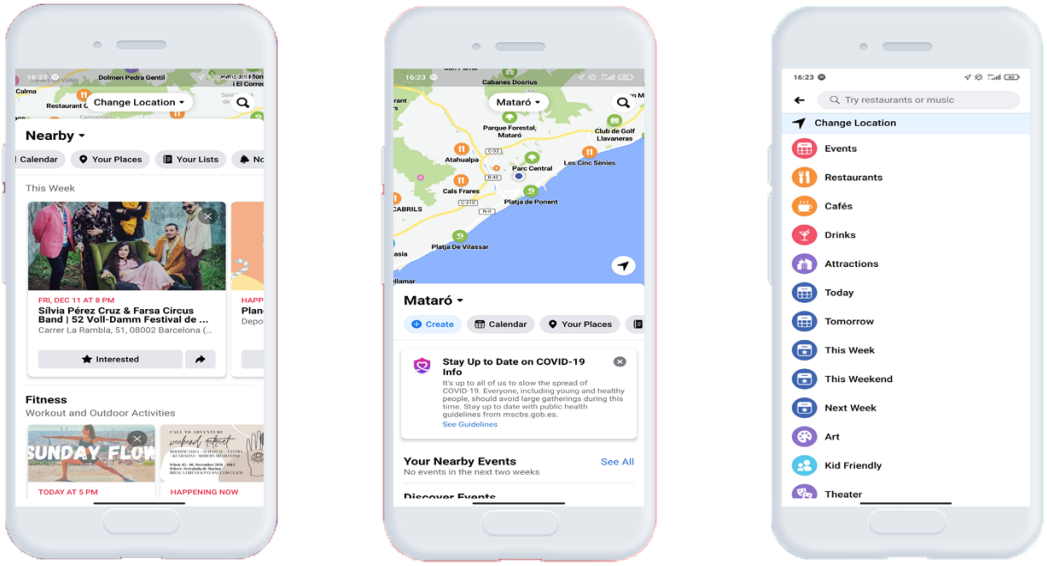

Fig 2.4 Captures de pantalla aplicació Facebook Local. Font: Elaboració pròpia

- **Ticketmaster**

    - Aplicació de compra d'entrades. Gran varietat de temàtiques, concerts, esdeveniments esportius, obres de teatre, festivals.

    - La majoria d'entrades són codis QR, que són llegits en el moment de l'esdeveniment.

    - És una aplicació amb molts usuaris, més de cent mil descàrregues.

    - No disposa d'un mapa per ubicar els llocs dels esdeveniments. Té una interfície bastant agradable i aconseguida.

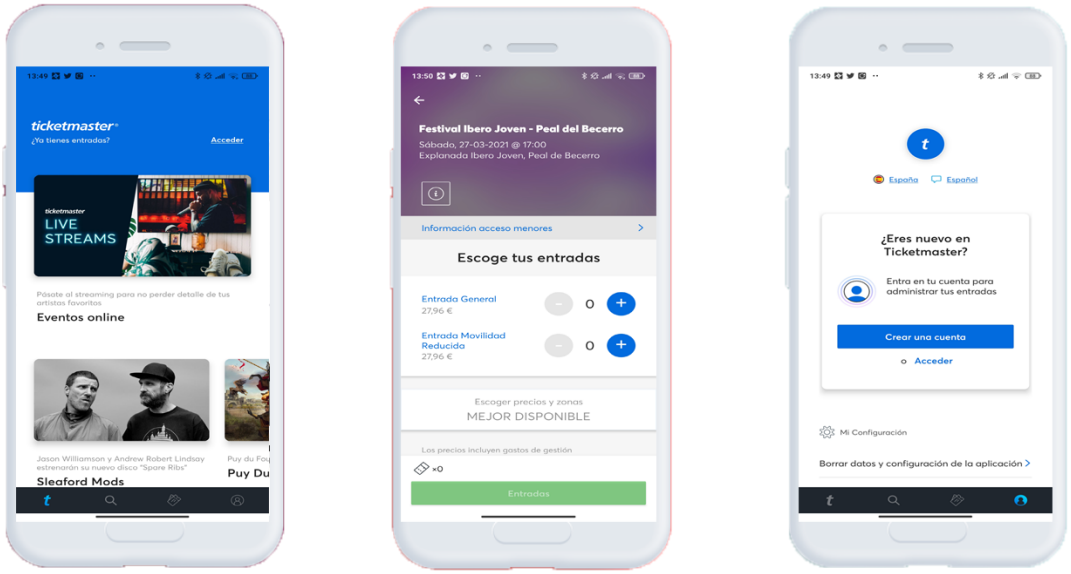

Fig 2.5 Captures de pantalla aplicació Ticketmaster. Font: Elaboració pròpia

S'ha detectat que actualment tots aquells usuaris que desitgen assistir a un concert no compten amb una plataforma on la seva única preocupació sigui triar la ubicació i data exacta.

És cert que existeixen aplicacions, com les mostrades anteriorment, que ofereixen grans llistats de concerts registrats i poden no complir amb les expectatives, però entre tants filtres costa focalitzar el que realment l'usuari està buscant, o no ho pot veure representat en un mapa.

1.3 Antecedents

La relació entre usuaris i artistes està experimentant una transformació digital.

La transformació ve donada gràcies a les noves tecnologies i l'estat de l'art de les eines de desenvolupament en aquest context.

Aquest canvi, té les seves conseqüències, totes aquelles persones que assistien en llocs de venda d'entrades per a concerts es veuen afectades per la situació actual de la Covid-19. Actualment tots els sectors es digitalitzen i a poc a poc, deixen d'existir els llocs de venda d'entrada físics.

1.3.1 Antecedent d'àrees de l'usuari

A continuació es mostren les aplicacions mòbils descrites anteriorment:

- Tripadvisor
- Eventbrite
- Meetup
- Facebook local
- Ticketmaster

Després d'analitzar i estudiar les següents aplicacions es conclou que:

- Són grans aplicacions que poden arribar a ser útils a l'usuari, ofereixen gran diversitat i, en algunes d'elles, hi ha la possibilitat de reservar entrades en cas d'assistir a concerts, o algun altre esdeveniment.

- No totes ofereixen un mapa on mostrar els llocs que hi haurà concerts pròxims.

- Totes les aplicacions ofereixen aplicació mòbil i pàgina web.

1.4 Necessitats d'informació

Per garantir que els objectius del projecte s'assoleixen, s'ha de tenir coneixements respecte a:

- Desenvolupament d'aplicacions Android.
- Desenvolupament de serveis REST.
- Desenvolupament web pel dashboard.
- Desenvolupament de base de dades.
- Autentificació Oauth.

A la secció de desenvolupament s'explica el perquè de cada punt i la seva elecció.

2. Objectius

2.1 Objectius del projecte

Amb la finalitat d'obtenir un producte final que s'adapti a les necessitats del projecte, s'han de complir els següents objectius:

2.1.1 Objectius del producte

- Aplicació mòbil
    - Oferir una aplicació per a la possibilitat d'assistir a concerts.
    - Consultar concerts.
    - Implementar un buscador de concerts.
    - Implementar un buscador d'artistes.
    - Suggerir concerts a partir de gustos musicals de l'usuari.
    - Seguir artistes.
    - Actualitzar les dades dels usuaris.
    - Mostrar un mapa d'ubicacions de concerts.
    - Reservar entrades.
    - Dissenyar una bona interfície *user friendly*.
    - Valorar concerts assistits.
    - Implementar autentificació.

- Dashbaord
    - Llistar els concerts fets per l'artista.
    - Llistar els concerts que participa l'artista.
    - Llistar els usuaris que han assistit a un concert de l'artista.
    - Crear un formulari de creació d'un concert.
    - Dissenyar una bona interfície *user friendly.*
    - Implementar autentificació.
    - Actualitzar les dades dels artistes.
- API i Base de dades
    - Connectar aplicació mòbil i dashboard.

2.1.2 Objectius del client

- **Objectius de l'artista**
    - Obtenir més audiència.
    - Aconseguir més seguidors.
    - Aconseguir influència.
    - Aconseguir digitalitzar-se.
    - Aproximar la relació amb els seus seguidors.
    - Mantenir la seva informació actualitzada.
- **Objectius de l'usuari**
    - Disposar d'una aplicació concretament per assistir a concerts.
    - Disposar d'una aplicació per consultar informació de concerts.
    - Reservar entrades.
    - Seguir artistes.

- Conèixer concerts propers a la seva ubicació.

2.1.3 Públic potencial o target

- Els beneficiaris de l'aplicació mòbil són:
  - Usuaris:
    - Perfil demogràfic: Totes les persones, tant homes com dones, sense mínim d'edat i d'arreu del món.
    - Perfil sociocultural: Totes les persones, sense restriccions socioculturals, apte per a tothom.
    - Perfil digital: Totes les persones amb un mínim d'experiència en navegació per aplicacions i amb smartphone que disposi de centre d'aplicacions.
  - Artistes:
    - Perfil demogràfic: Tots els artistes, tant homes com dones o grups musicals, amb edat mínima de setze anys i d'arreu del món.
    - Perfil sociocultural: Totes les persones, sense restriccions socioculturals, apte per a tothom.
    - Perfil digital: Tots els artistes amb un mínim d'experiència en navegació per aplicacions.
- Els beneficiaris del dashboard:
  - Artistes
    - Facilitant la creació d'un concert que es veurà mostrat en l'aplicació mòbil
    - Obtenció de nous oients amb informació d'on són, quina és la seva edat, gènere, gustos musicals, entre d'altres.

3. Abast

El producte final s'obté com a resultat del disseny i desenvolupament d'aquest projecte.

L'objectiu principal es divideix en dues àrees. L'àrea de l'usuari i de l'artista. Pel que fa a l'àrea de l'usuari ha de poder visualitzar, reservar, assistir, viure experiències i rebre suggeriments de concerts que poden ser del seu gust.
D'altra banda, l'àrea de l'artista l'objectiu és fer notar una millora en el seu expedient musical, aconseguir una millor audiència, nous oients i digitalització dels seus concerts.

4. Metodologia

La metodologia de desenvolupament escollida és *Agile*.

Agile es basa en el desenvolupament iteratiu i incremental, on els requisits i solucions evolucionen amb el temps segons la necessitat del projecte.

Cada iteració del cicle de vida inclou planificació, anàlisi de requisits, disseny, codificació, proves i documentació/desplegament.

Basant-se en la seva experiència conjunta de desenvolupar programari i ajudar els altres a fer-ho, van dissenyar un manifest, anomenat *Agile Manifesto* [5]. Principalment valoren els següents valors:

- **Individus i interaccions** sobre processos i eines.
- **Programari de treball** sobre documentació completa.
- **Col·laboració del client** en la negociació de contractes.
- **Respondre al canvi** seguint un pla.

Fig 4.1 Esquema de la metodologia Agile. Font: Elaboració pròpia

Aquesta metodologia ofereix els següents avantatges, entre d'altres:

- Flexibilitat per a definir funcions prioritàries i establir objectius.
- La seva organització permet identificar ràpidament quines són les tasques prioritàries en cada moment, sense necessitat de perdre temps.
- Eliminació de tasques innecessàries. Si es prioritza les tasques d'un procés, es poden saber quines tasques tenen un major pes, quines resulten secundàries o, fins i tot, innecessàries. Aquesta distinció ajuda a centralitzar esforços i unificar criteris d'actuació.
- Es pot obtenir feedback dels usuaris i artistes en les primeres fases del projecte

En resum, durant el desenvolupament del projecte s'ha seguit la metodologia *Agile*, es divideixen les grans tasques en subtasques que s'han d'anar completant en sprints (aproximadament dues setmanes).
En cas que en un sprint no es completi la feina o hi hagi canvis, es valoren els canvis en el següent.

S'ha seguit un procés definit de planificació, anàlisi de requisits, disseny, codificació, proves i documentació/desplegament.

Durant el desenvolupament del projecte es fa ús del programari de sistema de control de versions GIT. Concretament s'utilitza la plataforma Github.

5. Definició de requeriments funcionals i tecnològics

5.1 Definició de requeriments funcionals

Els requeriments funcionals són un dels pilars per a desenvolupar qualsevol projecte de qualsevol àrea, en aquest cas software. Descriuen el comportament que ha de tenir.

Una bona definició d'aquests ajuda a ser eficient en el temps, a tenir estructurat el projecte i tenir lligades noves funcionalitats de cara al futur.

La metodologia que s'ha arribat a la conclusió per redactar els requeriments funcionals és la tècnica de les cinc W's (5 Whys) [4]. Es basa en l'obtenció de requisits a partir de les respostes a cinc preguntes.

- **What**: Què es necessita?
- **Why**: Per què és necessari aquest requisit?
- **Who**: Per a qui és l'objectiu?
- **Where**: On s'utilitza?
- **When**: Quan s'utilitza?

L'ús d'aquesta tècnica aporta una presa de requeriments completa i detallada. A continuació es mostren els requeriments en format llista i responent les cinc preguntes:

Registrar un nou artista al dashboard

- **What**: Possibilitat de registrar un nou artista.
- **Why**: Per poder tenir accés al dashboard i a les seves funcionalitats.
- **Who**: Per a qualsevol artista que no estigui registrat prèviament.
- **Where**: Al dashboard per artistes.
- **When**: La primera vegada que un artista no registrat accedeixi al servei.

Verificar un artista o usuari donades les seves credencials

- **What**: Donat un nom d'usuari o correu electrònic i una contrasenya, el sistema comprova si aquest està registrat al sistema
- **Why**: Per poder tenir accés al dashboard o a l'aplicació mòbil.
- **Who**: Per a qualsevol artista o usuari ja registrat prèviament.
- **Where**: Al dashboard per artistes o aplicació mòbil.
- **When**: Un artista o usuari desitja accedit als serveis de dashboard o aplicació mòbil i prèviament ja s'ha registrat.

Autentificar artista o usuari (Token).

- **What**: Poder autentificar i autoritzar un artista o usuari.
- **Why**: Per poder fer un ús segur, amb token, de les peticions REST.
- **Who**: Servei de registre i autenticació.
- **Where**: En les crides dels serveis.
- **When**: Quan es vulgui recuperar, inserir, modificar o eliminar a través d'una petició REST.

Proporcionar un sistema de creació de concerts

- **What**: El dashboard ha de comptar amb una funcionalitat essencial per a facilitar la creació i modificació de concerts.
- **Why**: Aquesta funcionalitat és clau per al projecte. Facilita la interacció amb els artistes i els usuaris, un cop creat un concert, ho poden veure visualitzat a l'aplicació mòbil.
- **Who**: Per a qualsevol artista prèviament registrat i autentificat.
- **Where**: Al dashboard per artistes.
- **When**: Un artista desitja crear un nou concert o modificar algunes dades.

Proporcionar un sistema de reserva d'entrades

- **What**: L'aplicació mòbil ha de comptar amb una funcionalitat essencial per a poder assistir a un concert.
- **Why**: Aquesta funcionalitat és clau per al projecte. Un cop l'usuari ha reservat una entrada es generarà un codi QR únic com a identificador que servirà per poder assistir-hi.
- **Who**: Per a qualsevol usuari prèviament registrat i autentificat que vulgui assistir a un concert.
- **Where**: Aplicació mòbil.
- **When**: Un usuari que estigui reservant una entrada per assistir a l'esdeveniment.

Consultar els concerts que un usuari assistirà

- **What**: Poder consultar els concerts que un usuari assistirà.
- **Why**: Aquesta funcionalitat serà necessària per poder cancel·lar una reserva i donar opció a altres usuaris a assistir.
- **Who**: Per a qualsevol usuari prèviament registrat i autentificat que tingui mínim una reserva.
- **Where**: Aplicació mòbil.
- **When**: Un usuari que hagi reservat una entrada per assistir a l'esdeveniment.

Valorar els concerts que un usuari ha assistit

- **What**: Poder valorar els concerts que un usuari ha assistit.
- **Why**: Aquesta funcionalitat serà necessària per poder donar una opinió i valoració sobre l'artista el qual l'usuari ha assistit al seu concert. És vital recopilar informació i valoració d'usuaris que hi han assistit prèviament als concerts d'algun artista, per poder mostrar a nous usuaris que valoren els seus oients.
- **Who**: Per a qualsevol usuari prèviament registrat i autentificat que hagi assistit a un concert.
- **Where**: Aplicació mòbil.
- **When**: Un usuari que hagi assistit a un esdeveniment prèviament.

5.2 Definició de requeriments tecnològics

Protegir les dades

- **What**: Totes les peticions entre servidor, dashboard i aplicació mòbil es fan sota un URL amb certificat SSL i HTTPS. Tot usuari haurà d'estar identificat per modificar dades o visualitzar gran part d'elles.
- **Why**: Per la seguretat.
- **Who**: Tots els serveis.
- **Where**: En el REST.
- **When**: Quan es faci qualsevol petició.

Aconseguir un bon manteniment

- **What**: Tots els serveis i dependències han d'estar actualitzats a les noves versions, tant API com el dashboard i aplicació mòbil.
- **Why**: Evitar problemes de competitivitat i mètodes obsolets.
- **Who**: Tots els serveis.
- **Where**: En el REST, dashboard i aplicació mòbil.
- **When**: Cada mes es farà un control d'actualització de versions i dependències en cas que sigui necessari.

Generar còpies de seguretat

- **What**: En cas de problemes amb la base de dades o algun servei, poder recuperar fàcilment la informació amb les còpies de seguretat (*backups)* que s'han anat fent prèviament.
- **Why**: En cas de virus o atac que alteri la base de dades, poder actuar correctament sense perdre tota la informació emmagatzemada.
- **Who**: Tots els serveis, principalment la base de dades.
- **Where**: Base de dades.
- **When**: Cada dia.

Ser escalable

- **What**: És important mantenir una correcta estructura per poder escalar el projecte en un futur.
- **Why**: Per assegurar que els serveis poden créixer pròsperament, afectant el més mínim el rendiment d'aquests.
- **Who**: Tots els serveis.
- **Where**: Arquitectura de l'API, el dashboard i aplicació mòbil.
- **When**: Quan s'implementen noves funcionalitats.

6. Desenvolupament

Aquesta secció mostra el procés de disseny i desenvolupament dut a terme en aquest projecte.

Inicialment es mostren un seguit de casos d'ús dels processos més importants.

Un cop acabada la secció de casos d'ús es recullen les diferents tecnologies i aplicacions escollides per implementar el projecte. Estan dividides en apartats, valorades amb punts a favor i en contra juntament amb una breu conclusió del perquè de la decisió.

Cada apartat explicatiu de les tecnologies emprades es mostren detalls de com s'han anat implementant des de l'inici fins al final o punt en qüestió, per tal de mostrar d'una manera clara i concisa el funcionament del sistema.

6.1 Definició de casos d'ús

A continuació es descriuen els casos d'ús per tal d'identificar les interaccions que es produeixen entre un sistema i els seus actors.

6.1.1 Registrar usuari

Identificador	RU
Actor	Usuari
Precondició	L'usuari ha de tenir accés a internet. L'usuari no ha d'existir a la base de dades.
Descripció	El sistema ha de permetre registrar un nou usuari. Es demanen les següents dades: correu electrònic, contrasenya, data de naixement, gènere i nom. Es registrarà posteriorment a la base de dades.
Seqüència normal	1. L'usuari no ha iniciat sessió. 2. L'usuari es troba a la pantalla d'iniciar sessió. 3. L'usuari es dirigeix a crear un compte. 4. L'usuari introdueix el correu electrònic. 5. El sistema valida si el correu electrònic ja existeix. 6. L'usuari procedeix a omplir les dades restants. 7. L'usuari procedeix a registrar-se amb el botó. 8. El sistema registra un nou usuari a la base de dades. 9. L'aplicació redirigeix l'usuari a la pantalla de preferències de gèneres musicals i artistes.
Postcondició	L'usuari ja pot veure la pantalla principal de l'aplicació, un cop seleccionats els gèneres musicals i artistes a seguir.
Excepció	6. Si l'usuari introdueix alguna dada no acceptada es notificat amb un missatge explicatiu. 8. Si el sistema no pot registrar l'usuari aquest es notificat amb un missatge en pantalla.

Taula 6.1 Cas d'ús del registre d'un usuari. Font: Elaboració pròpia

6.1.2 Registrar artista

Identificador	RA
Versió	V.1.0
Actor	Artista
Precondició	L'artista ha de tenir accés a internet. L'artista pot no existir a la base de dades.
Descripció	El sistema ha de permetre registrar un nou artista. Es demanen les següents dades: correu electrònic, nom artístic, contrasenya, bio, data de naixement, gènere, país i foto de perfil. Es registrarà posteriorment a la base de dades.
Seqüència normal	1. L'artista no ha iniciat sessió. 2. L'artista es troba a la pantalla d'iniciar sessió. 3. L'artista es dirigeix a crear un compte. 4. L'artista procedeix a omplir les dades. 5. L'artista procedeix a registrar-se com a artista amb el botó un cop seleccionada la foto de perfil. 6. El sistema valida les dades i registra un nou artista. 7. L'aplicació redirigeix l'artista a la pàgina principal.
Postcondició	L'artista ja pot veure la pantalla principal i veure les funcionalitats.
Excepció	6.1 Si el correu electrònic o nom artístic ja existeix, i ja és artista, mostrarà un missatge en pantalla "artista existent". 6.2 Si el correu electrònic ja existeix, però és d'un compte d'usuari, es registra l'artista i el compte passa a ser d'artista.
Comentaris	Un usuari existent pot registrar-se com a artista.

Taula 6.2 Cas d'ús del registre d'un artista. Font: Elaboració pròpia

6.1.3 Iniciar sessió al dashboard per artistes

Identificador	ISDA
Versió	V.1.0
Actor	Artista
Precondició	L'artista ha de tenir accés a internet. L'artista no ha d'existir a la base de dades. L'artista ha d'accedir via web.
Descripció	El sistema ha de permetre identificar un artista i permetre accedir a les seves estadístiques i perfil.
Seqüència normal	1. L'artista no ha iniciat sessió. 2. L'artista es troba a la pantalla d'iniciar sessió. 3. L'artista introdueix les seves credencials 4. El sistema vàlida si les credencials són correctes i existeix un artista amb el correu introduït. 5. L'aplicació redirigeix l'artista a la pàgina principal.
Postcondició	L'artista ja pot veure la pantalla principal i veure les funcionalitats.
Excepció	2. Si el sistema no identifica l'artista, aquest es notificat amb un missatge en pantalla.

Taula 6.3 Cas d'ús d'inici de sessió al dashboard. Font: Elaboració pròpia

6.1.4 Crear concert des de l'aplicació mòbil

Identificador	CC
Versió	V.1.0
Actor	Artista
Precondició	L'artista ha de tenir accés a internet. L'artista ha iniciat sessió.
Descripció	El sistema ha de permetre registrar un nou esdeveniment. Es demanen les següents dades sobre l'esdeveniment i el lloc. Es registrarà posteriorment a la base de dades.
Seqüència normal	1. L'artista es troba a la pantalla principal. 2. L'artista es dirigeix a crear un concert. 3. L'artista procedeix a omplir les dades. 4. L'usuari procedeix a registrar el concert amb el botó un cop especificat el nombre d'assistents. 5. El sistema registra un nou concert a la base de dades. 6. L'aplicació penja les imatges del concert a S3. 7. L'aplicació redirigeix l'artista a la pantalla de llistat de concerts.
Postcondició	L'artista ja pot veure estadístiques, activitat i gestionar el seu concert. L'usuari ja podrà buscar el concert i comprar entrades.
Excepció	4. Si l'usuari introdueix alguna dada no acceptada es notificat amb un missatge explicatiu. 6. Si el sistema no pot registrar el concert, l'artista es notificat amb un missatge en pantalla.

Taula 6.4 Cas d'ús de la creació d'un concert. Font: Elaboració pròpia

6.1.5 Comprar entrades

Identificador	CE
Versió	V.1.0
Actor	Usuari
Precondició	L'usuari ha de tenir accés a internet. L'usuari ha iniciat sessió.
Descripció	El sistema ha de permetre comprar i reservar entrades per a un concert.
Seqüència normal	1. L'usuari ha buscat el concert que vol assistir. 2. L'usuari especifica el número d'entrades que vol reservar. 3. L'usuari procedeix a comprar amb el botó de comprar. 4. El sistema valida si el número d'entrades que es volen reservar supera les màximes que es poden reservar. 5. El sistema registra una nova reserva d'entrades. 6. L'aplicació redirigeix l'usuari a l'apartat d'entrades comprades.
Postcondició	L'usuari ja pot veure les entrades comprades a l'apartat d'entrades de l'aplicació.
Excepció	4. Si el sistema no pot fer la reserva, l'usuari es notificat amb un missatge en pantalla.

Taula 6.5 Cas d'ús de la compra d'entrades. Font: Elaboració pròpia

6.1.6 Valorar concerts assistits

Identificador	VCA
Versió	V.1.0
Actor	Usuari
Precondició	L'usuari ha de tenir accés a internet. L'usuari ha iniciat sessió. L'usuari ha assistit a un concert.
Descripció	El sistema ha de permetre valorar un concert un cop finalitzat des de l'aplicació mòbil.
Seqüència normal	1. L'usuari es troba a la secció del seu perfil 2. L'usuari accedeix a l'apartat de concerts assistits. 3. L'usuari selecciona el concert a valorar. 4. L'usuari escriu un missatge i valora el concert entre zero i cinc estrelles. 5. El sistema registra o actualitza una valoració d'un concert. 6. L'usuari es redirigit a l'apartat de concerts assistits.
Postcondició	L'usuari veu la seva valoració actualitzada a l'apartat de concerts assistits. L'artista podrà veure la valoració, juntament amb altres, a la pantalla principal de l'aplicació.
Excepció	5. Si el sistema no pot registrar o actualitzar la valoració, l'usuari es notificat amb un missatge en pantalla.

Taula 6.6 Cas d'ús de valoració de concerts assistits. Font: Elaboració pròpia

6.2 Desenvolupament aplicació mòbil

Per al desenvolupament de l'aplicació mòbil s'ha fet ús d'Android Studio.

Android Studio és l'entorn de desenvolupament integrat (IDE) oficial per al desenvolupament d'aplicacions per Android. Principalment s'ha triat Android perquè actualment, en el sector d'Smartphones, és el sistema operatiu que més predomina.

Fig 6.1 Distribució de sistemes operatius en Smartphones. Font: Statcounter

S'ha valorat l'opció de desenvolupar una aplicació híbrida, que serveixi per a tots els sistemes operatius, no només per Android. Segons la recerca feta, una aplicació híbrida és la combinació perfecta en cas d'una aplicació senzilla i es vulgui llençar el més aviat possible. D'altra banda, no es pot obtenir un control total del sistema operatiu. En cas de desenvolupar una aplicació que es necessari alguns permisos es recomana utilitzar llenguatge natiu.

Native App	Hybrid App
Programming languages required (Java for Android, Objective-C or Swift for iOS)	HTML, CSS, JavaScript
A separate code for each platform	Single codebase
High performance	Lower performance
Requires more time, money, and expertise	Faster, cheaper and easier to develop
Great user experience	Good user experience
Enhanced security	Good security
Easy to customize	Poor customizability
Direct access to native device features	Requires specific plugins
Easier to test and debug	Simpler to maintain

Fig 6.2 Comparativa entre aplicacions natives i híbrides. Font: Scand

Per tant, s'ha desenvolupat en llenguatge natiu, ja que com a mínim és necessari el permís d'ubicació. Això comporta que l'aplicació només esta disponible per mòbils amb sistema operatiu Android. D'aquesta manera, tal com s'ha comentat anteriorment, es pot garantir un control totalitari de l'aplicació amb les funcionalitats del sistema operatiu per oferir una bona experiència d'usuari i seguretat.

6.2.1 Permisos

Els permisos necesaris per a una bona experència d'usuari són els següents.

Usuari:

- Permís d'ubicació
- Permís d'internet

Artista

- Permís d'ubicació
- Permís d'internet
- Permís de lectura d'imatges

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

Fig 6.3 Permisos de l'aplicació móbil. Font: Elaboració pròpia

6.2.2 Segmentació de l'aplicació

Tal i com s'ha comentat i determinat en anteriors punts, l'aplicació esta dividida en dos rols. El rol d'usuari i artista.

Les funcionalitats de l'aplicació per als usuaris esta dividida en cinc parts:

- **Mapa**. L'usuari pot veure en un mapa, donades les seves coordenades, les ubicacions de quins són els pròxims concerts de la seva zona. S'ha aplicat un filtre d'interval de dates i de distància.

- **Buscador general**. Aquesta funcionalitat té dues seccions. Un buscador de concerts i d'artistes.

  El buscador de concerts, mostra tots els concerts pròxims, juntament amb dos filtres que l'usuari pot modificar, un interval de data i un buscador que retorna tots els concerts que compleixin amb el terme que l'usuari esta buscant.

  L'altra secció, només busca artistes, disposa d'un buscador que retorna tots els artistes que compleixen amb el terme que l'usuari esta buscant. Si l'usuari pot veure el seu perfil prement la seva imatge.

- **Home.** Aquesta és la pantalla principal que es mostra quan l'usuari obra l'aplicació i ha iniciat sessió. Es mostren concerts proxims donada l'ubicació de l'usuari, concerts suggerits a partir dels seus gustos, artistes que li podrien agradar i concerts populars, es a dir, concerts que han venut casi totes les entrades.

- **Entrades.** Aquesta secció mostra totes les entrades que l'usuari ha comprat per assistir als concerts. Si l'usuari prem alguna pot veure els QR corresponents que serveixen per validar les entrades a l'hora d'assistir al concert.

- **Perfil**. L'usuari pot veure i actualitzar les seves dades personals. Podrà consultar informació sobre els esdeveniments que ha reservat o comprat entrades. Valorar el concert un cop finalitzat per poder oferir bones referències a altres usuaris.

L'artista pot veure més informació des del dashboard. Però l'aplicació per part d'un artista es divideix en tres seccions.

- **Home.** És la pantalla principal que es mostra quan un artista ha iniciat sessió i obra l'aplicació. Es pot visualitzar l'activitat relacionada amb els seus concerts i comentaris recents.

- **Gestió de concerts.** Aquesta secció es divideix en quatre parts:

    - **Crear concert.** L'artista pot organitzar un nou concert.

    - **Concerts creats.** Es mostren en format carousel tots els concerts que l'artista ha creat.

    - **Concerts que hi colabora.** Es mostren els concerts, en format carousel, en els que l'artista hi participa però no els ha creat ell o ella.

    - **Concerts finalitzats.** Es mostren en format carousel els concerts, que donada la data actual, s'han finalitzat.

- **Perfil**. L'artista pot veure i actualitzar les seves dades personals i d'artista.

6.3 Desenvolupament API REST

Per la part de desenvolupament de l'API s'ha fet ús d'Spring [9]. Spring és un framework de codi obert per Java.

L'elecció d'utilitzar Spring per desenvolupar l'API és conseqüència dels coneixements adquirits a la universitat i per tal d'aplicar-los en un projecte des de zero.

Spring destaca per les següents especificacions:

- Escalable.
- Seguretat. Les peticions es poden limitar per rols. Per exemple, un usuari no pot crear un concert
- Gran comunitat que l'utilitza.
- Open source.
- MVC.
- Microserveis.

6.3.1 Punts d'accés

Per complir amb els requisits, es dissenyen i implementen els següents punts d'accés.

Per obtenir i actualitzar dades d'usuaris i artistes

http://adreçaServidor/user/create

- Funcionalitat: Crear un usuari.
- Tipus de petició: POST
- Body: Tipus user
- Caracter: Public
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Dades de l'usuari en format JSON.

http://adreçaServidor/user/{userId}

- Funcionalitat: Agafar les dades d'un usuari.
- Tipus de petició: GET
- Paràmetres:
    - userId
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Dades de l'usuari en format JSON

http://adreçaServidor/artist/follow/{artistId}/{userId}/follow

- Funcionalitat: Seguir un artista.
- Tipus de petició: GET
- Paràmetres:
    - artistId
    - userId
    - follow. De tipus boolean, true per seguir i false per deixar de seguir.
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Dades de l'artista en format JSON.

http://adreçaServidor/artist/info/{currentDate}/{artistId}/{userId}

- Funcionalitat: Agafar les dades professionals d'un artista juntament amb la foto de perfil i pròxims concerts.
- Tipus de petició GET
- Paràmetres:
    - currentDate
    - artistId
    - userId
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Dades de l'artista en format JSON.

http://adreçaServidor/artist/create

- Funcionalitat: Crear un artista.
- Tipus de petició: POST
- Body: Tipus artist
- Caracter: Public
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Dades de l'artista en format JSON.

http://adreçaServidor/user/preferences/save

- Funcionalitat: Crear preferències d'usuari per seguir artistes i gèneres musicals.
- Tipus de petició: PUT
- Body: Tipus userPreferences
- Caracter: Private
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
- Resposta: Dades de les preferències de l'artista en format JSON.

Per obtenir i actualitzar dades de conerts

http://adreçaServidor/concert/create

- Funcionalitat: Crear un concert.
- Tipus de petició: POST
- Body: Tipus concertRegister
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Dades del concert en format JSON.

http://adreçaServidor/concert/home/suggestions/{userId}/{currentDate}

- Funcionalitat: Obtenir suggerències de concerts per a un usuari.
- Tipus de petició: GET
- Paràmetres:
    - userId
    - currentDate
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Llistat en format JSON de concerts suggerits.

http://adreçaServidor/concert/home/popular/{userId}/{currentDate}

- Funcionalitat: Obtenir concerts populars per a un usuari.
- Tipus de petició: GET
- Paràmetres:
    - userId
    - currentDate
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Llistat en format JSON de concerts populars.

http://adreçaServidor/concert/map/latitude/{userLatitude}/longitude/{userLongitude}/radius/{radius}/currentDate/{currentDate}/startDate/{startDate}/endDate/{endDate}

- Funcionalitat: Obtenir concerts propers a partir de l'ubicació de l'usuari.
- Tipus de petició: GET
- Paràmetres:
    - userLatitude. Tipus double
    - userLongitude. Tipus double
    - radius. Tipus double
    - currentDate. Tipus String
    - startDate.Tipus String
    - endDate. Tipus String
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Llistat en format JSON de concerts propers.

http://adreçaServidor/concert/info/{userId}/{concertId}

- Funcionalitat: Obtenir informació detallada d'un concert.
- Tipus de petició: GET
- Paràmetres:
    - userId. Tipus String
    - concertId. Tipus String
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Dades del concert en format JSON.

Per comprar i administrar entrades

http://adreçaServidor/booking/create

- Funcionalitat: Crear un reserva.
- Tipus de petició: POST
- Body: Tipus registerBooking
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Format JSON amb missatge d'informació.

http://adreçaServidor/booking/all/user/{userId}/{currentDate}

- Funcionalitat: Obtenir reserves fetes per un usuari.
- Tipus de petició: GET
- Paràmetres:
    - userId
    - currentDate
- Caracter: Privat
- Headers: Token de sessió
- Resposta HTTP:
    - 200. Petició correcte
    - 500. Error de sistema
- Resposta: Llistat en format JSON de les reserves actives fetes per un usuari.

Per valorar concerts

http://adreçaServidor/concert/rating/post/{currentDate}

- Funcionalitat: Publicar una valoració d'un concert assistit.
- Tipus de petició: PUT
- Paràmetres:
  - currentDate. Tipus String
- Body: Tipus ratingSimplified
- Caracter: Private
- Headers: Token de sessió
- Resposta HTTP:
  - 200. Petició correcte
  - 500. Error de sistema
- Resposta: Valoració en format JSON.

http://adreçaServidor/concert/rating/all/userId/{userId}/{currentDate}

- Funcionalitat: Obtenir valoracions fetes o per fer d'un usuari.
- Tipus de petició: GET
- Paràmetres:
  - userId. Segon
- Caracter: Private
- Headers: Token de sessió
- Resposta HTTP:
  - 200. Petició correcte
  - 500. Error de sistema
- Resposta: Llistat en format JSON de les valoracions fetes, o encara per fer, d'un usuari.

6.4 Desenvolupament dashboard per artistes

Per al desenvolupament del dashboard s'ha utlitzat Angular.

Angular [6] és un framework de disseny i una plataforma de desenvolupament d'aplicacions. Principalment s'utilitzen llenguatges de programació web, HTML i CSS o SCSS, juntament amb TypeScript.

S'ha valorat altres alternatives, com React i Vue. Els dos són frameworks també amb millor rendiment i flexibilitat comparat amb Angular.

Tot i que, algunes comparatives recomanen React [7] o Vue [8] per sobre d'Angular, realment és un framework bastant utilitzat i amb molta projecció de cara el futur, és fàcil d'utilitzar i organitzat.

	React	ANGULAR	Vue.js
	A declarative, efficient, and flexible JavaScript library for building user interfaces.	One framework. Mobile & desktop.	A progressive, incrementally adoptable JavaScript framework for building UI on the web.
	116 993 ⭐	59 302 ⭐	121 050 ⭐
Original author	Jordan Walke	Miško Hevery	Evan You
Developers	Facebook	Google	
Initial release	May 29, 2013	October 20, 2010	February 2014
Npm weekly downloads	3 940 035	433 361	709 943
Size	109.7 KiB production 774.7 KiB development	167 kB production 1.2 MB development	30.67 KB production 279 KB development
Easy to learn	Medium	Learn TypeScript	Yes
Coding speed	Normal	Slow	Fast
Documentation	✓	✓	✓
Performance	✓	✓	✓
Startup time	Quick	Longer due to its large codebase	Quick
Complete web apps	Needs to be integrated with many other tools	Can be used on standalone basis	Requires third party tools
Data binding	Uni-directional	Bi-directionall	Bi-directional
Rendering	Server side	Client side	Server side
Model	Virtual DOM	MVC	Virtual DOM
Code reusability	No, only CSS	Yes	Yes, CSS and HTML
When to use	Production, custom UI apps	Production, esp. enterprise apps with Material UI	Startups, production
Big companies	Facebook, Yahoo, Netflix, Atlassian, KhanAcademy	Netflix, Upwork, PayPal, TheGuardian	Facebook, Alibaba, Adobe, Grammarly, GitLab

Fig 6.4 Comparativa entre React, Angular i Vue. Font: Un poco de Java

Alguna de les avantatges d'Angular és el seu fàcil manteniment. Al utilitzar TypeScript, qualsevol canvi que s'hagi de fer a l'aplicació es pot dur a terme ràpidament i sense errors.

El futur és inestable. Amb JavaScript es pateixen canvis sovint i és molt complicat aprendre totes les novetats que van sorgint. Amb Angular, generalment, hi ha molts menys canvis que amb JavaScript.

Per tant, el desenvolupament amb Angular és escalable, de fàcil manteniment, estable i pràctic.

6.5 Cloud

Per el Cloud del projecte s'han estudiat dues possibilitats.

- AWS. Cloud d'Amazon
- Firebase. Cloud de Google

AWS destaca principalment per les següents caracteristiques:

- **Facilitat d'ús**. Ràpidament pots desplegar una aplicació.
- **Flexible**. Permet seleccionar el sistema operatiu, el llenguatge de programació, la plataforma d'aplicacions web, la base de dades, així com la resta de serveis que es necessiti.
- **Rentable**. Només es paga per el seu ús.
- **Confiança**. És una infraestructura informàtica global escalable i segura.
- **Escalabilitat**. Amb les eines d'AWS, les aplicacions podren ampliar-se o reduir-se segons la demanda.
- **Seguretat.**

Firebase enfatitzar les següents qualitats:

- **Rentable**. Inicialment és gratuït i amb un gran marge per començar a pagar, i després es paga per el seu ús .
- **Base de dades en temps real.** Envia automàticament esdeveniments a les aplicacions quan les dades canvien.
- **Seguretat.**
- **Es de Google.**
- **Desplagament ràpid.**

Després de la comparativa entre AWS i Firebase. L'elecció ha sigut dificil però finalment s'ha decantat per AWS.

AWS és el més utilitzat amb gran majoria per empreses com a servei Cloud i juntament amb els coneixements bàsics que es tenen finalment s'ha escollit AWS per tal de millorar-los i ampliar-los durant el desenvolupament del projecte.

De la gran quantitat de serveis que ofereix AWS. S'ha fet ús de la part d'instàncies (Elastic beanstalk) per penjar l'API, i S3 Bucket, per guardar les imatges dels concerts, fotos de perfil dels usuaris i artistes.

6.6 Backend

S'ha triat MongoDB per desenvolupar una base de dades NoSQL ja que no hi han coneixements previs, el projecte és una molt bona oportunitat per començar i aprendre.

MongoDB és un sistema de base de dades NoSQL, orientat a documents i de codi obert.

En lloc de guardar les dades en taules, tal com es fa en les bases de dades relacionals, MongoDB guarda estructures de dades BSON (una especificació similar a JSON) amb un esquema dinàmic, fent que la integració de les dades en certes aplicacions sigui més fàcil i ràpida

L'aplicació esta formada principalment de Clusters, permet a una base de dades escalar horitzontalment en molts servidors o replicar dades, millorant així el rendiment.

Dintre dels Clusters es creen les bases de dades, que estan formades de Collections, en base de dades SQL s'anomenarien entitats o taules.

A continuació és mostra el diagrama de classes utilitzat per muntar l'infraestructura.

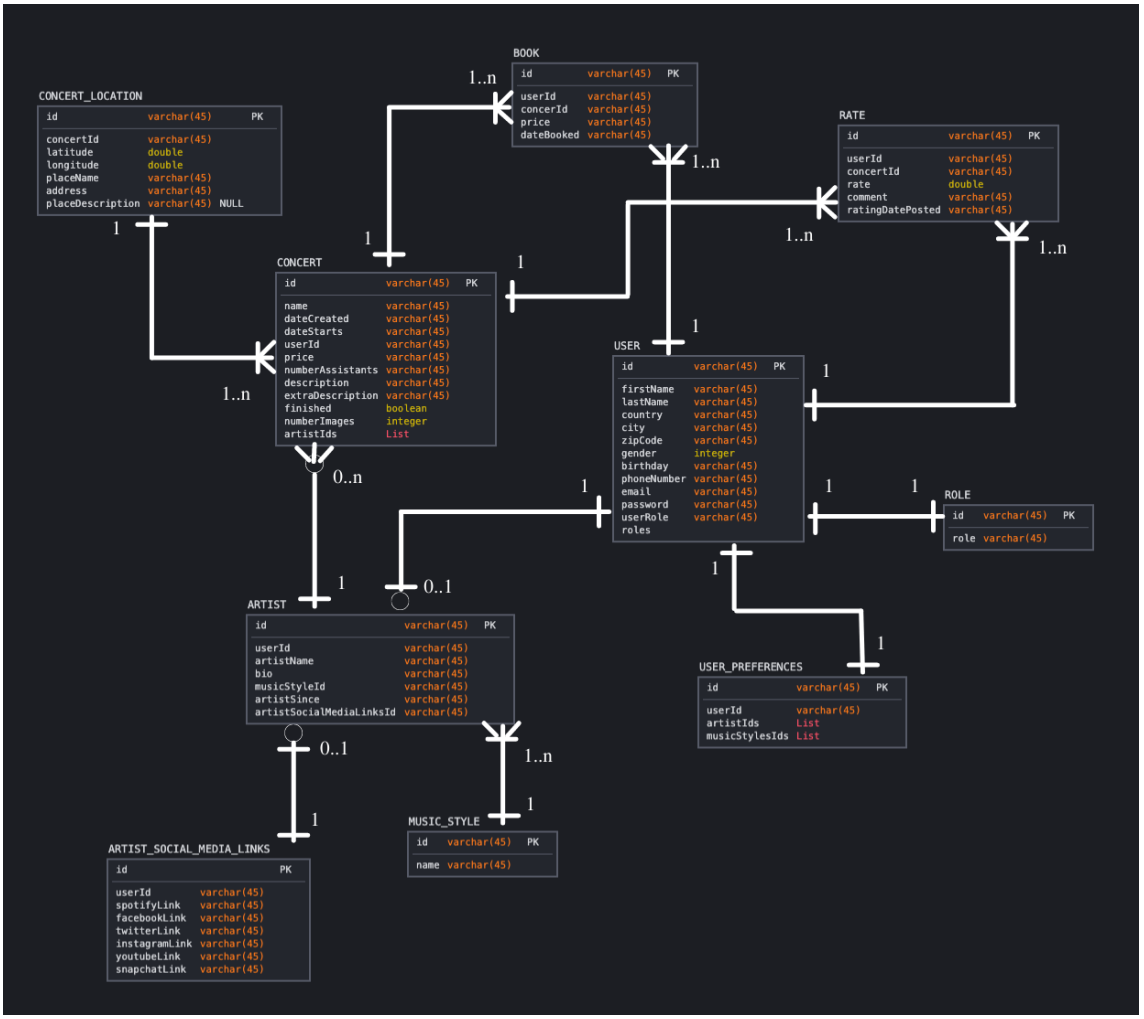

Fig 6.5 Diagrama de classes del projecte. Font: Elaboració pròpia.

Els usuaris són el centre de tot, aquests poden convertir-se en artistes o directament convertir-se en artistes, pero implicitament són creats també com a usuaris. La seva diferència és en el rol, tant un cop l'altre veuen seccions diferents en l'aplicació mòbil i l'usuari no registrat com a artista no té accés al dashboard.

Un artista pertany únicament a un génere musical però un génere musical pot tenir varietat d'artistes. Un artista és pot donar a coneixer a través de xarxes socials. Els links que proporciona són visibles per els usuaris quan accedeixen al perfil de l'artista des de l'aplicació mòbil.

Un usuari pot seguir a més d'un artista, si segueix un artista nou i aquest forma part d'un génere que l'usuari no en te preferència, es registra aquest nou génere per mostrar-ne sugerències i artistes relacionats a seguir.

La part més important és la creació de concerts, aquests són creats per artistes. Un concert pot ser creat per un artista i aquest artista pot haver creat un, o encara no. Es separen les dades d'ubicació del concert per no carregar l'entitat Concert.

Els concerts poden ser valorats per usuaris un cop finalitzats, únicament es demana valorar de l'u al cinc amb estrelles i un comentari.

Els usuaris poden reservar entrades per a un o més concerts. En cas d'oferir serveis "prèmium" de pagament es planteja l'ús d'Stripe. Stripe proporciona la infraestructura tècnica, de prevenció de frau i bancària necessària per operar sistemes de pagament en línia. És una de les últimes integracions previstes per fer.

Un cop l'usuari és registrat es registren també les preferències d'aquest, compostes principalment per artistes i generes músicals a seguir.

6.6.1 Connectar API amb MongoDB

Per tal de connectar la base de dades amb l'API del projecte es necessita, primer de tot, crear un compte i iniciar sessió a MongoDB. Un cop fet el procés es mostra una pantalla com la següent.

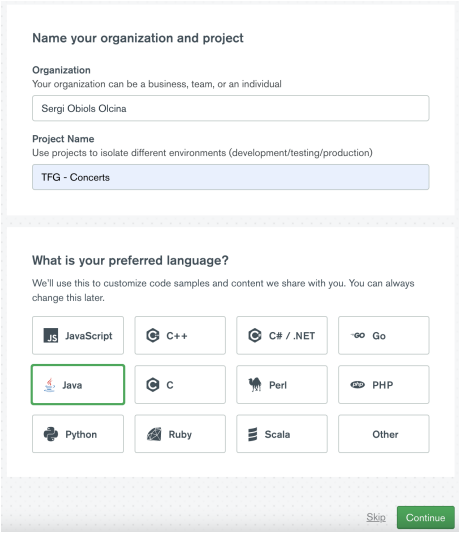

Fig 6.6 Secció de registre MongoDB. Font: Elaboració pròpia

A continuació es selecciona el pla gratuït. Més endevant, quan es gestionen més dades cal canviar de plan a un de dedicat i plantejar-se la funcionalitat de còpies de seguretat diàries per nou euros mensuals.

Seguidament només cal omplir, les seccions del proveidor de Cloud juntament amb la regió, i el nom del Cluster.

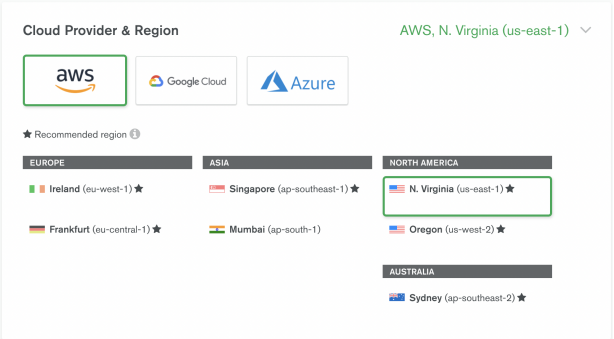

Fig 6.7 Secció de selecció Cloud del MongoDB. Font: Elaboració pròpia

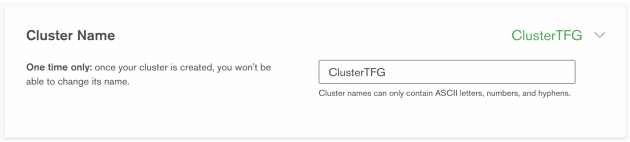

Fig 6.8 Secció del nom del Cluster de MongoDB. Font: Elaboració pròpia

Després d'uns minuts d'espera per la configuració de l'entorn ja es pot veure i interactuar amb la pantalla principal, on el més important és el botó de connectar.

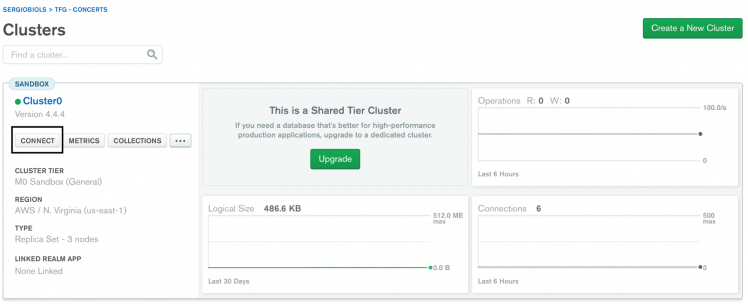

Fig 6.9 Pantalla principal MongoDb. Font: Elaboració pròpia

Un cop dins de l'apartat de connectar es demana afegir una connexió IP, per ara la 0.0.0.0/0 per no especifiar cap en concret. I crear un usuari per tal de donar accés.

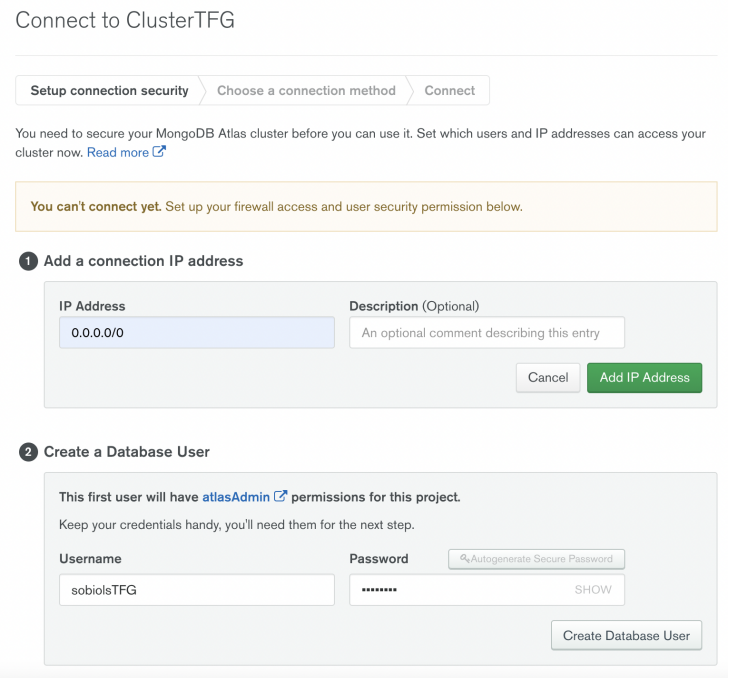

Fig 6.10 Secció de l'apartat de connectar MongoDb. Font: Elaboració pròpia

El següent a fer és especificar com es vol connectar la base de dades. És a gust del desenvolupador, en aquest cas s'ha seleccionat *connectar la teva aplicació.*

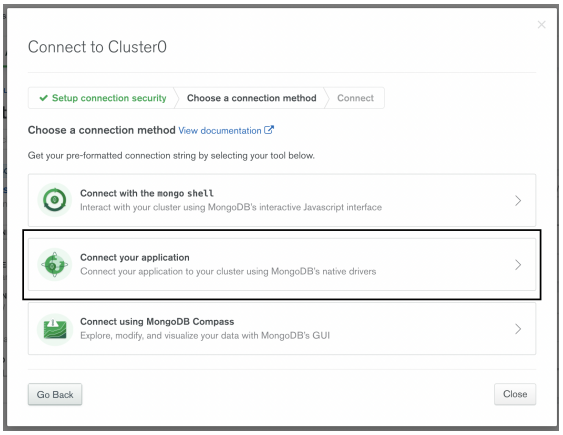

Fig 6.11 Secció de l'apartat del mètode de connectar MongoDb. Font: Elaboració pròpia

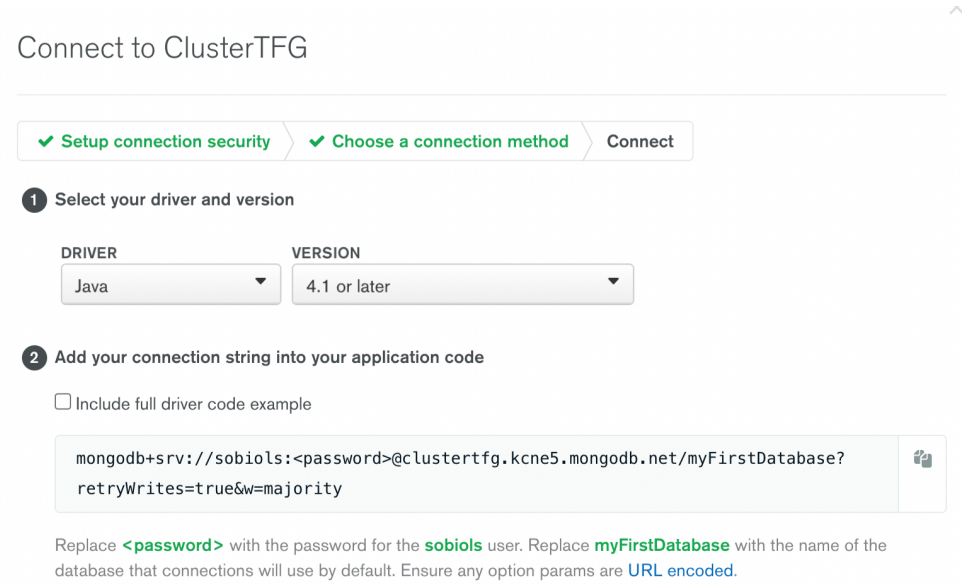

Fig 6.12 Secció final de l'apartat de connectar MongoDb. Font: Elaboració pròpia

Ara ja es pot connectar la base de dades amb l'API, finalment cal dirigir-se al projecte d'Intelij. Primer de tot cal afegir la següent dependència al fitxer *pom.xml* i sincronitzar-lo.

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-mongodb</artifactId>
</dependency>
```

Fig 6.13 Dependència de MongoDb. Font: Elaboració pròpia

Per acabar, s'ha de modificar el fitxer *application.properties* afegint la següent linea. <*Password*> és la especificada en la secció de registrar un usuari d'accés a la base de dades. myFirstDatabase és el nom de la base de dades per defecte, es pot canviar dins del Cluster.

```
spring.data.mongodb.uri=mongodb+srv://sobiols:<password>@clustertfg.kcne5.mongodb.net/myFirstDatabase?retryWrites=true&w=majority
```

Fig 6.14 String per a connectar MongoDB amb API. Font: Elaboració pròpia

6.6.1.1 Registrar usuari a la base de dades

Per a registrar un usuari o qualsevol entitat a MongoDB és necessari crear la seva entitat, el seu repository i el controller amb l'endpoint en qüestió.

Primer de tot es crea la classe *User* amb els atributs corresponents, seguint el diagrama de classes, a la carpeta *Entities*. Cal afegir l'anotació:

@Document(collection="nomDeLaCollection").

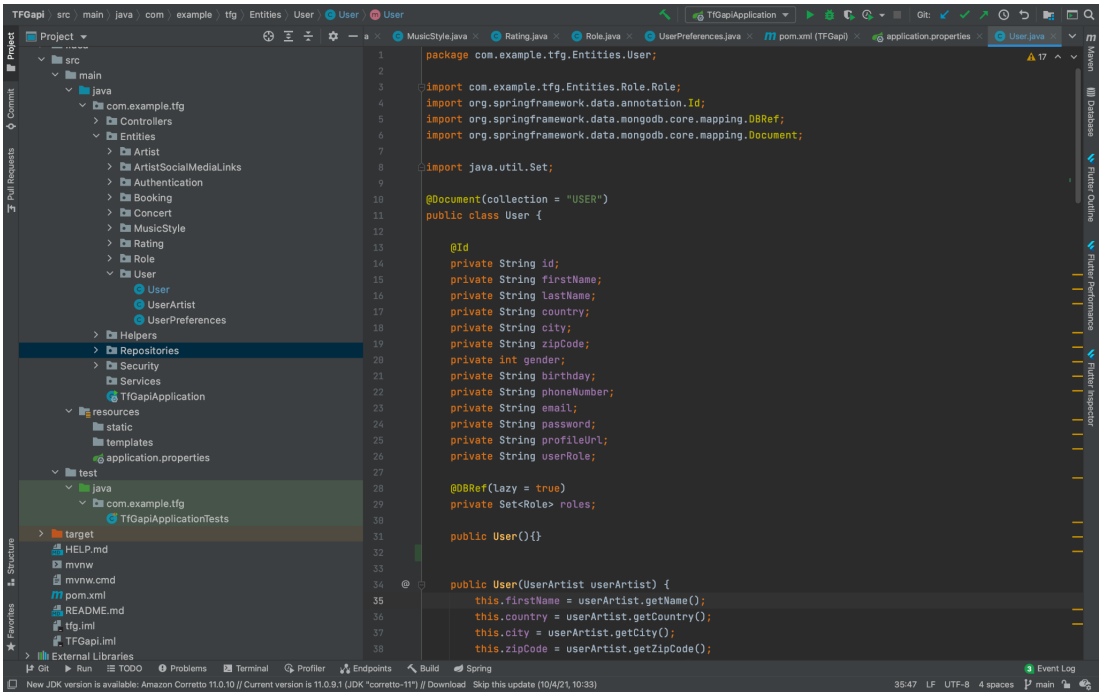

Fig 6.15 Classe User. Font: Elaboració pròpia

Seguidament es crea l'interficie *UserRepository.* Amb la dependència de MongoDB es pot afegir *extends MongoRepository<User,String>* que permet agilitzar i simplificar les consultes.

Cal remarcar que MongoDB ajuda a nombrar les funcions depenentment dels atributs de la classe en qüestió per tal de processar les consultes tal i com es mostra a continuació.

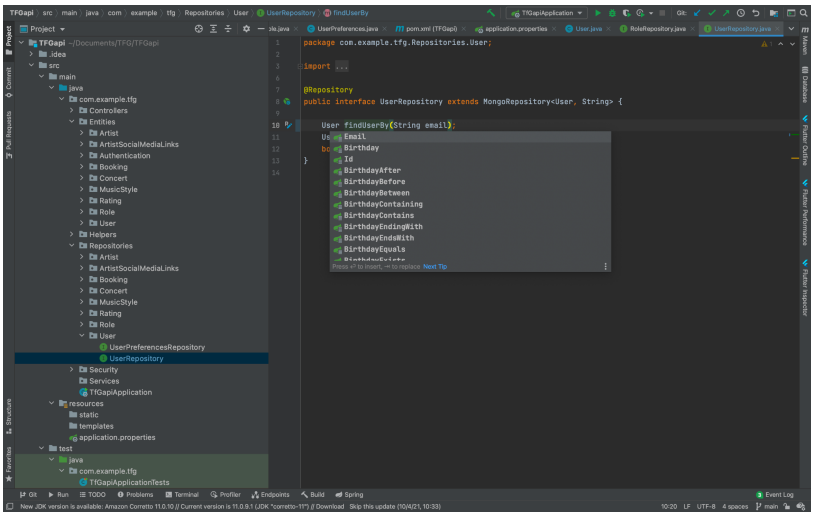

Fig 6.16 Autocompletat d'una consulta MongoDB. Font: Elaboració pròpia

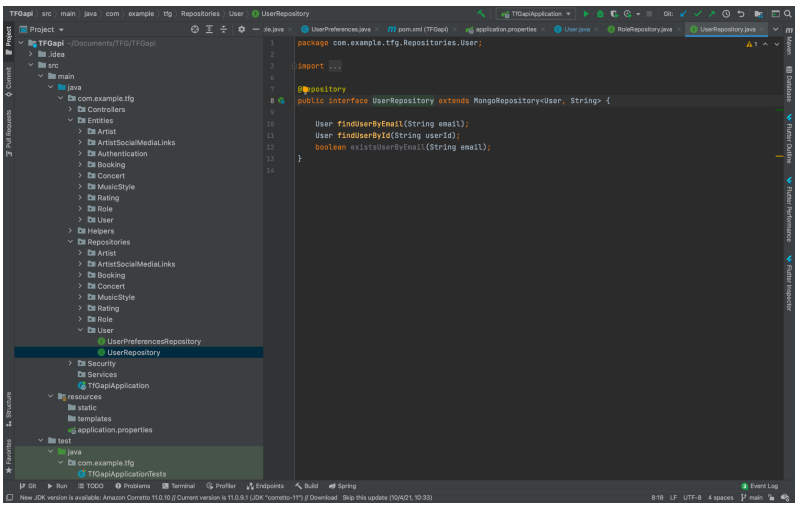

Fig 6.17 Interfície UserRepository. Font: Elaboració pròpia

El següent que cal fer és crear el *UserController* a la carpeta *Controllers*. Permet crear les peticions de l'API REST, en aquest cas per a un *User*. Calen les anotacions de *@CrossOrigin* de localhost per tal de redirigir el trafic, *@RestController* per especificar que és un *Controller* i *@RequestMapping* per determinar l'endpoint de les peticions.

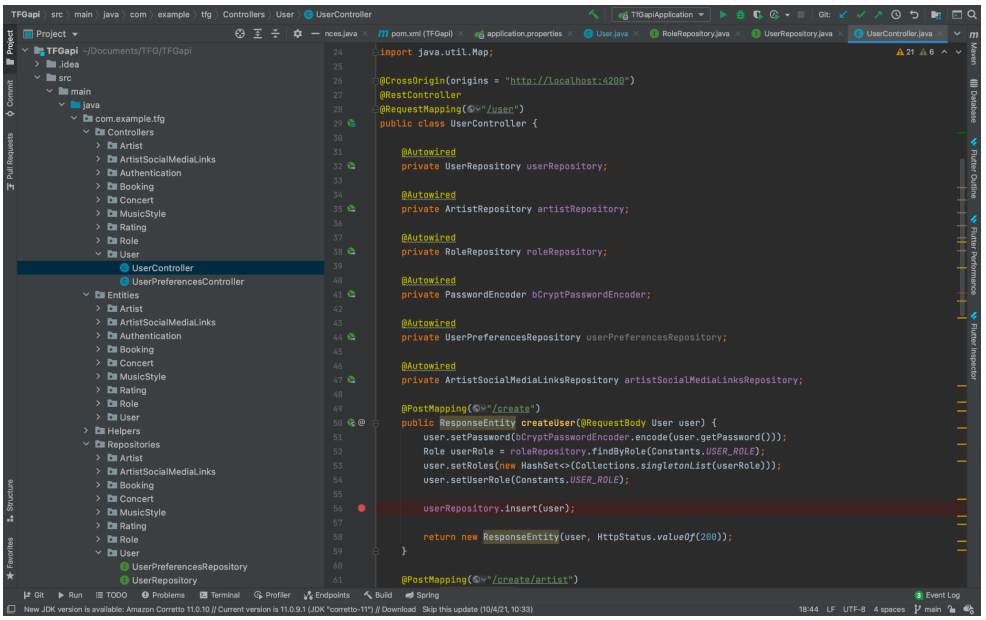

Fig 6.18 Mostra de codi de UserController. Font: Elaboració pròpia

Per tal de registrar un nou usuari cal crear la petició en qüestió, aquesta serà de tipus POST, rebrà com a parametre un body de tipus *User* i al endpoint *adreçaServidor/user/create.*

```
@PostMapping("/create")
public ResponseEntity createUser(@RequestBody User user) {
    user.setPassword(bCryptPasswordEncoder.encode(user.getPassword()));
    Role userRole = roleRepository.findByRole(Constants.USER_ROLE);
    user.setRoles(new HashSet<>(Collections.singletonList(userRole)));
    user.setUserRole(Constants.USER_ROLE);

    userRepository.insert(user);

    return new ResponseEntity(user, HttpStatus.valueOf(200));
}
```

Fig 6.19 Petició de registre d'un User. Font: Elaboració pròpia

El següent pass cal executar l'API en localhost i port 5000 per testejar-ho. Per especificar que l'API es propagui en el port 5000 cal afegir la següent linea al fitxer *application.properties*.

```
server.port=5000
```

Fig 6.20 Atribut per executar API al port 5000. Font: Elaboració pròpia

Seguidament cal provar la petició de registre d'usuari, per fer-ho cal l'aplicació Postman. Es crea una nova petició de tipus POST a *http://localhost:5000/user/create* amb *Headers* de *Content-Type : application/json* i el següent body.

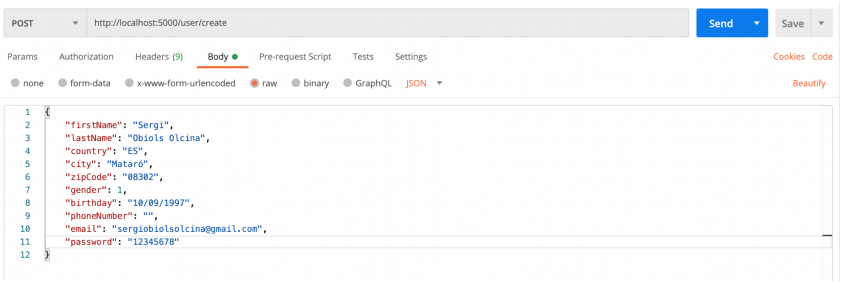

Fig 6.21 Petició registrar usuari des de Postman. Font: Elaboració pròpia

Finalment, un cop executada la petició aquesta retorna l'usuari creat com a JSON, cal anar a MongoDB, entrar a *Clusters*, a continuació a *Collections* i ja podrem veure la Collection *User* amb l'usuari registrat correctament a la base de dades.

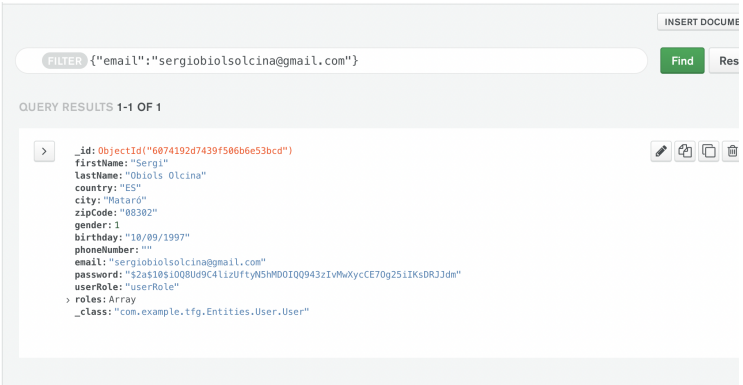

Fig 6.22 User registrat a MongoDb. Font: Elaboració pròpia

6.6.2 Connectar API amb AWS

AWS engloba una gran quantitat de serveis per poder realitzar diferents tipus d'activitats en el Cloud.

Entre ells cal destacar Elastik beanstalk i S3 bucket. El primer serveix per poder propagar l'API del projecte, i el segon terme per poder emmagatzemar fitxers, tal i com fotos de perfil dels artistes.

Per tal de connectar l'API del projecte amb AWS es necessita, primer de tot, crear un compte i iniciar sessió a AWS. En cas de no tenir compte, es demana omplir un seguit de dades personals juntament amb una targeta de crèdit, que ara per ara no hi han cobraments.

Un cop fet el procés es mostra una pantalla principal com la següent.

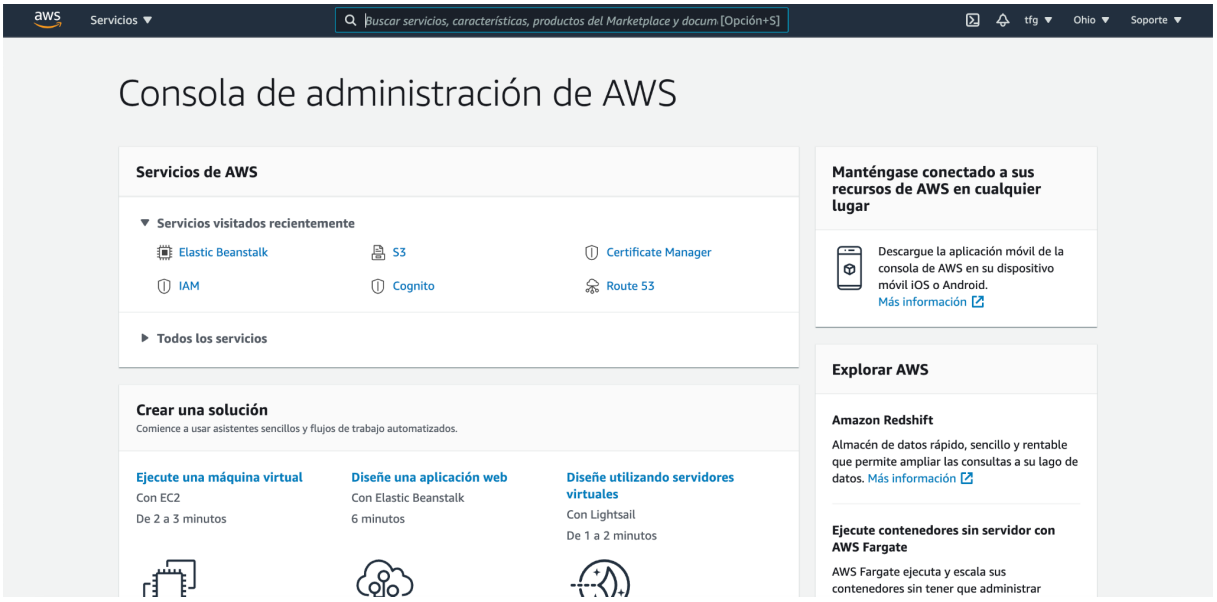

Fig 6.23 Pantalla principal AWS. Font: Elaboració pròpi

6.6.2.1 Creació de l'entorn AWS

Al ser el primer cop, s'ha de buscar Elastic Beanstalk per tal de poder crear una instància i administrar en un futur l'API del projecte.

Un cop dins la secció d'Elastic Beanstalk cal crear un nou entorn i seleccionar un entorn de servidor web.

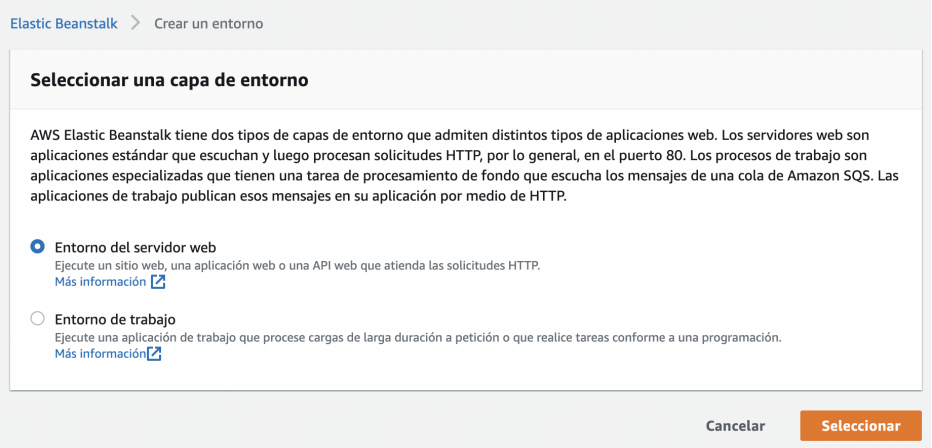

Fig 6.24 Primer apartat creació entorn AWS. Font: Elaboració pròpia

Seguidament es demana el nom de l'aplicació, informació sobre l'entorn, la plataforma (en el cas del projecte en qüestio s'ha seleccionat Java.

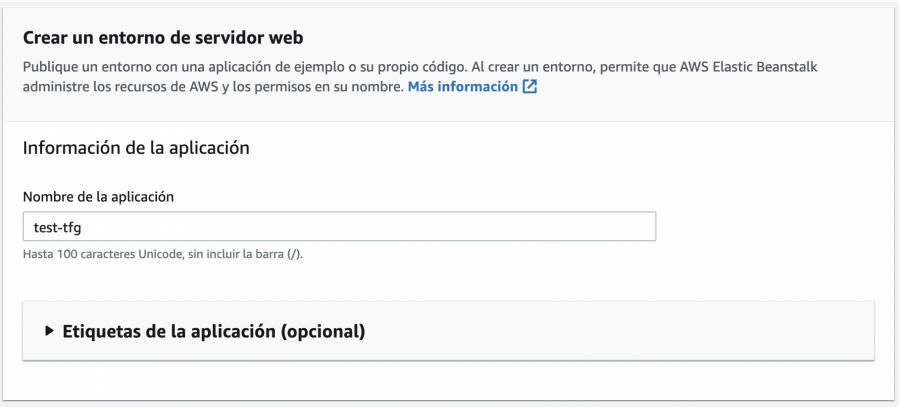

Fig 6.25 Selecció del nom en el procés de creació entorn AWS. Font: Elaboració pròpia

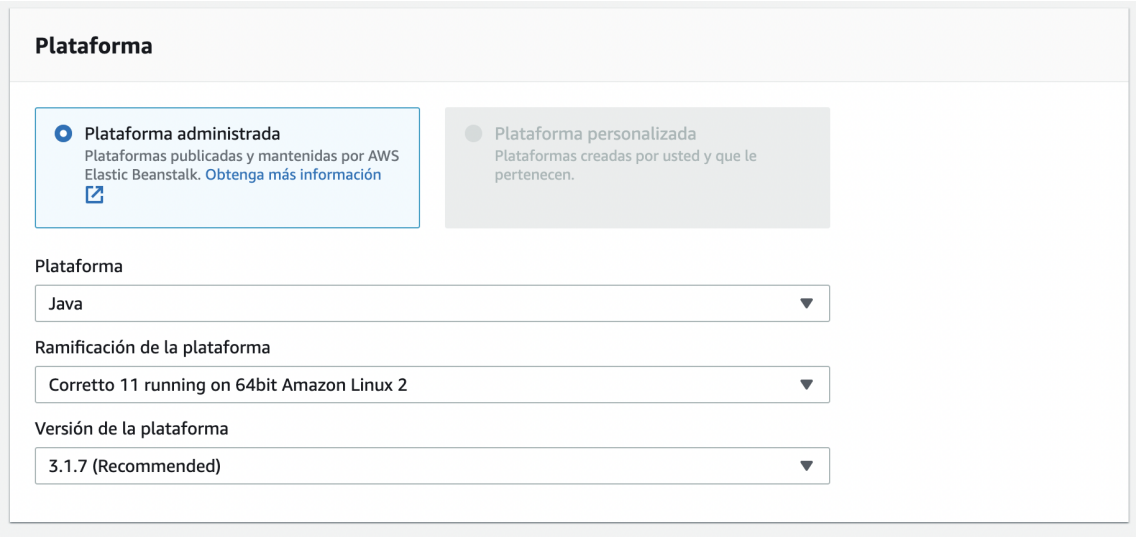

Fig 6.26 Selecció de la plataforma en el procés de creació entorn AWS.
Font: Elaboració pròpia

Finalment, un cop creat l'entorn i passats uns deu minuts per la configuració inicial, es pot visualitzar la següent pantalla.

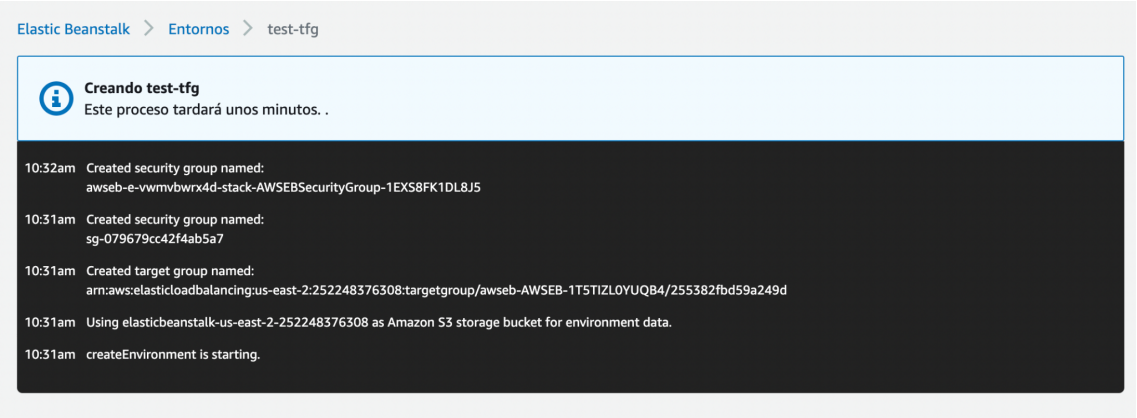

Fig 6.27 Procés de creació de l'entorn AWS. Font: Elaboració pròpia

Al finalitzar el procés de configuració inicial ja es pot veure correctament la pantalla principal de gestió de l'entorn.

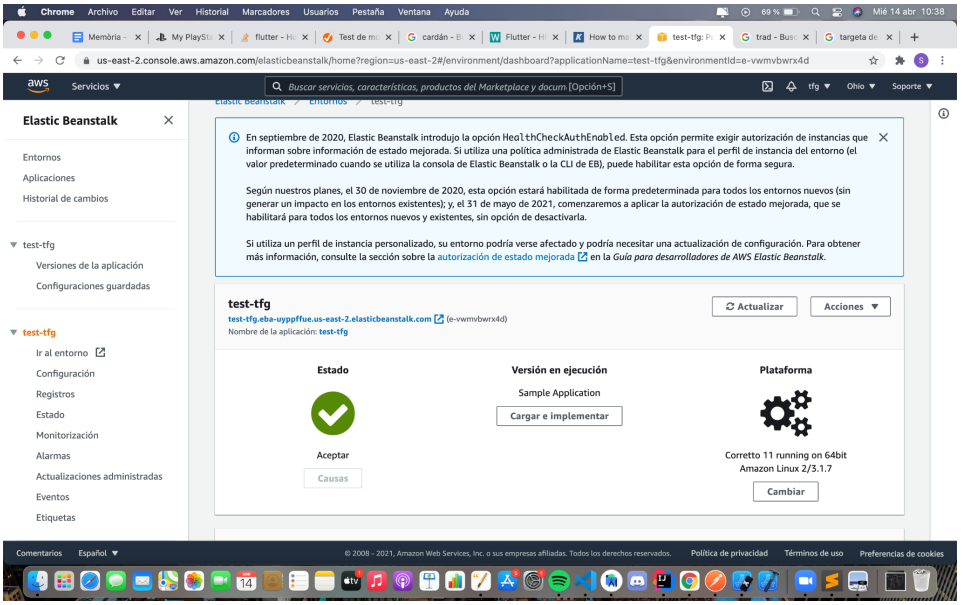

Fig 6.28 Pantalla de gestió de l'entorn Elastic Beanstalk AWS. Font: Elaboració pròpia

Per poder comprovar que ja s'ha propagat tot correctament i ja podem fer ús dels serveis, cal obrir la URL generada sota el nom de l'aplicació.

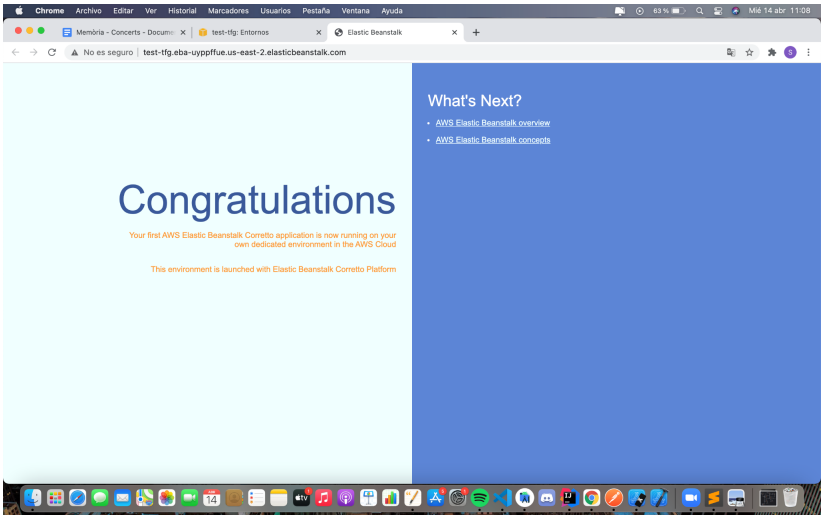

Fig 6.29 Root de l'aplicació AWS. Font: Elaboració pròpia

6.6.2.2 Deploy a AWS

Un cop feta la configuració inicial d'Spring i tenir instalat Intelij es pot procedir a crear un fitxer .jar per a poder carregar-lo i implementar-lo en l'entorn del projecte creat en la secció prèvia.

Intelij és IDE escollit per dur a terme el projecte tal i com s'ha definit anteriorment. Una de les característiques més útils a l'hora de la creació del package és els shortcuts de Maven de la dreta de l'aplicació.

Primer de tot cal fer un *clean*, elimina les dependències i importacions del projecte.

Seguidament es procedeix a executar el *compile,* un cop s'ha fet un *clean* afegeix les depèndencies existents i noves, també reconeix els imports del projecte.

Finalment, s'executa el *package,* aquesta comanda genera un arxiu .jar a la carpeta del projecte, dintre de la carpeta *target.*

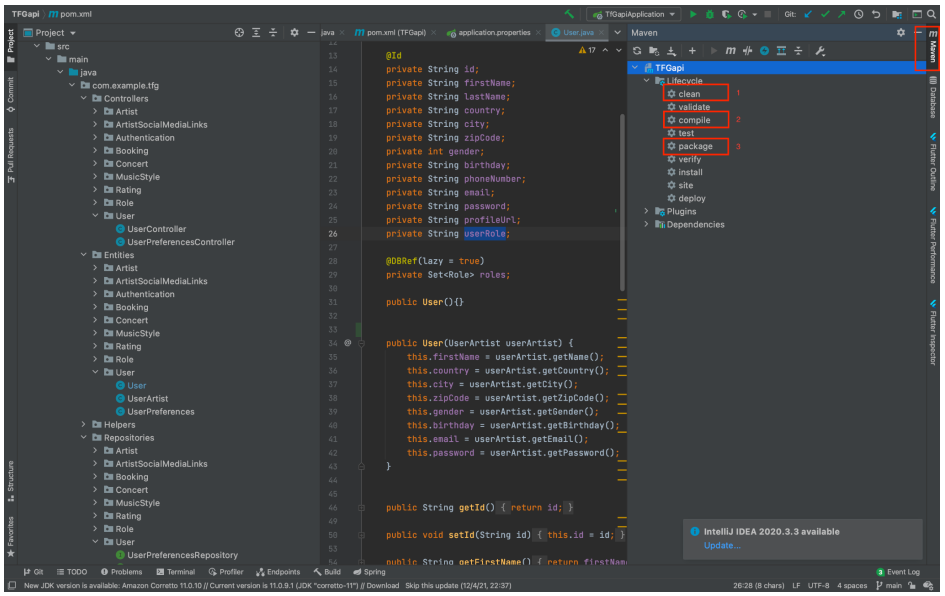

Fig 6.30 Procés de la creació del package API. Font: Elaboració pròpia

El procés és ràpid, el següent pas a fer és dirigir-se a AWS, a la pantalla principal de gestió d'Elastic Beanstalk i seleccionar *cargar e implementar.*

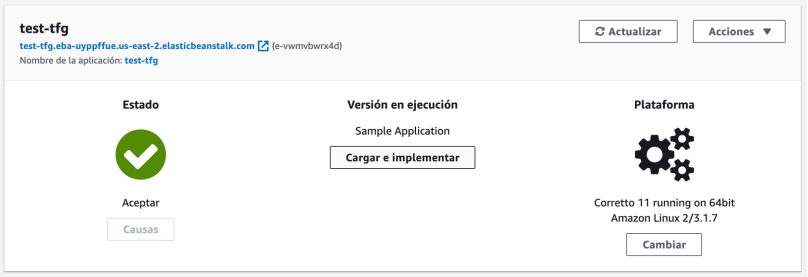

Fig 6.31 Procés de deploy del package API. Font: Elaboració pròpia

A continuació es selecciona el fitxer generat en el procés de package de l'API, a la carpeta *target* i es defineix una etiqueta identificadora de la versió del deploy.

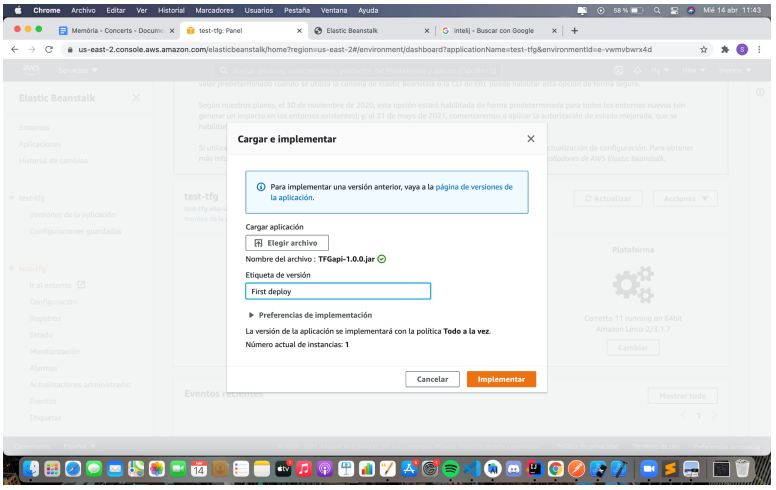

Fig 6.32 Procés final del deploy del package API. Font: Elaboració pròpia

Un cop finalitzada l'implementació, l'entorn propaga l'API al servidor un cop actualitzat. Cal esperar un minut aproximadament.

Per comprovar que tot ha sortit bé és recomenable anar actualitzant amb el botó definit.

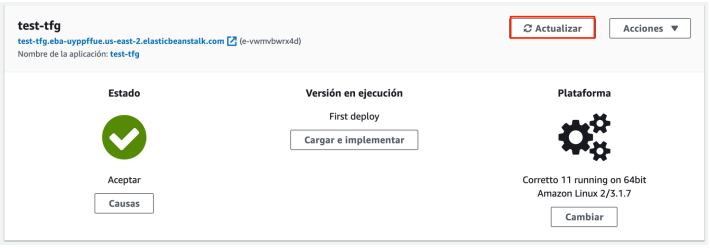

Fig 6.33 Procés final del deploy del package API. Font: Elaboració pròpia

6.6.2.3 Validació del deploy

És important, després d'haver actualitzat l'entorn, que el servidor segueix actiu i no retorna codis HTTP 500 de resposta.

Per comprovar-ho, s'agafa una petició d'exemple. En aquest cas la *adreçaServidor/user/{userId}* que rep un *userId* existent com a paràmetre per tal d'obtenir l'informació de l'usuari en qüestió.

```
@GetMapping("/{userId}")
public ResponseEntity getUserById(@PathVariable String userId) {
    User user = userRepository.findUserById(userId);
    return new ResponseEntity(user, HttpStatus.valueOf(200));
}
```

Fig 6.34 Petició getUserById de l'API. Font: Elaboració pròpia

Al buscar *l'usuari* creat a la secció 6.6.1.1 la resposta ha de ser un 200 i l'informació de l'usuari.

Fig 6.35 Resposta de la petició GET d'un usuari per Id. Font: Elaboració pròpia

7. Disseny

Aquesta secció mostra el disseny de pantalles dut a terme en aquest projecte de l'aplicació mòbil i el dashboard.

7.1 Aplicació mòbil

Tal i com en prèvies seccions s'ha comentat, l'aplicació esta dividida en dues seccions, la d'usuari i la d'artista. Per tant, cada un d'ells tenen funcionalitats i pantalles diferents.

S'ha dividit aquest punt en dues seccions i per cada una, en rols d'usuari i d'artista.

Les seccions de l'aplicació per cada un dels rols s'han especificat a la secció 6.3.2.

7.1.1 Rol d'usuari

Un usuari de l'aplicació és aquell que vol assistir, té intenció d'assistir a un concert creat per un artista o vol consultar informació sobre un concert.

A continuació és mostren captures d'algunes de les funcionalitats més importants de l'aplicació mòbil per part d'un usuari.

7.1.1.1 Registrar usuari

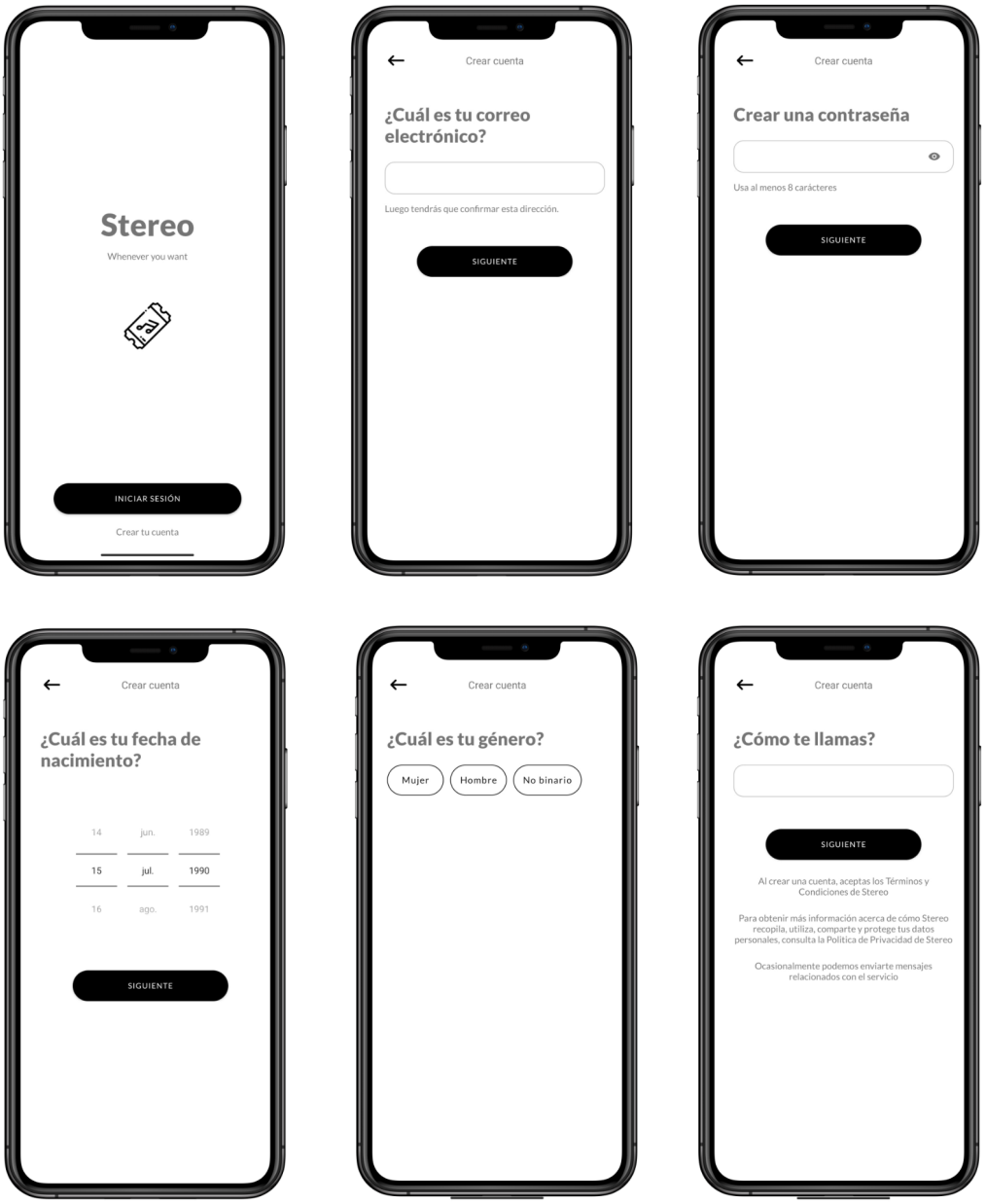

Fig 7.1 Captures de pantalla del procés de registre d'usuari. Font: Elaboració pròpia

7.1.1.2 Preferències de l'usuari

Fig 7.2 Registre de preferències musicals d'un usuari. Font: Elaboració pròpia

7.1.1.3 Pantalla d'inici

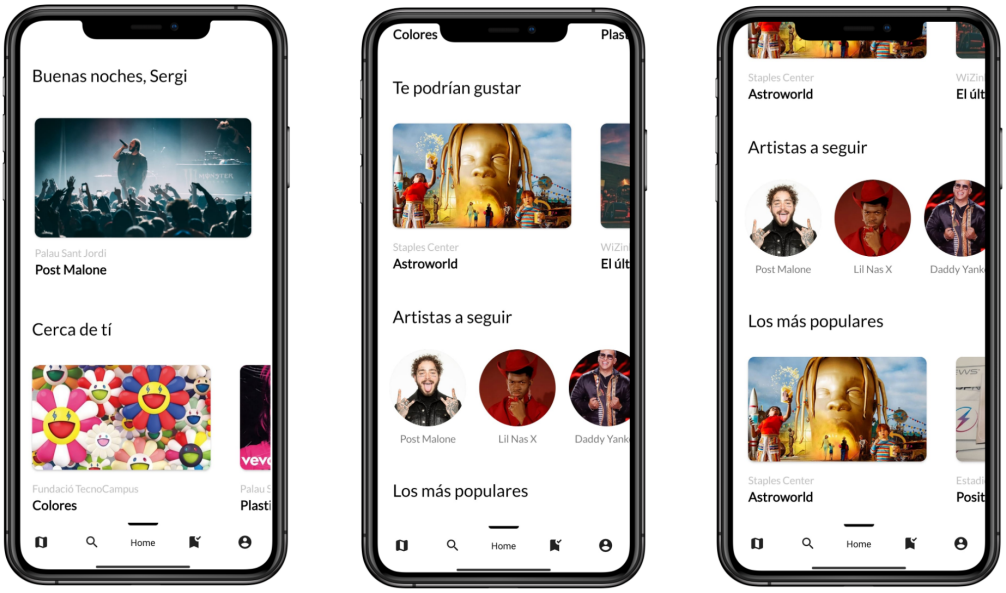

Fig 7.3 Pantalla principal d'un usuari. Font: Elaboració pròpia

7.1.1.4 Mapa de concerts

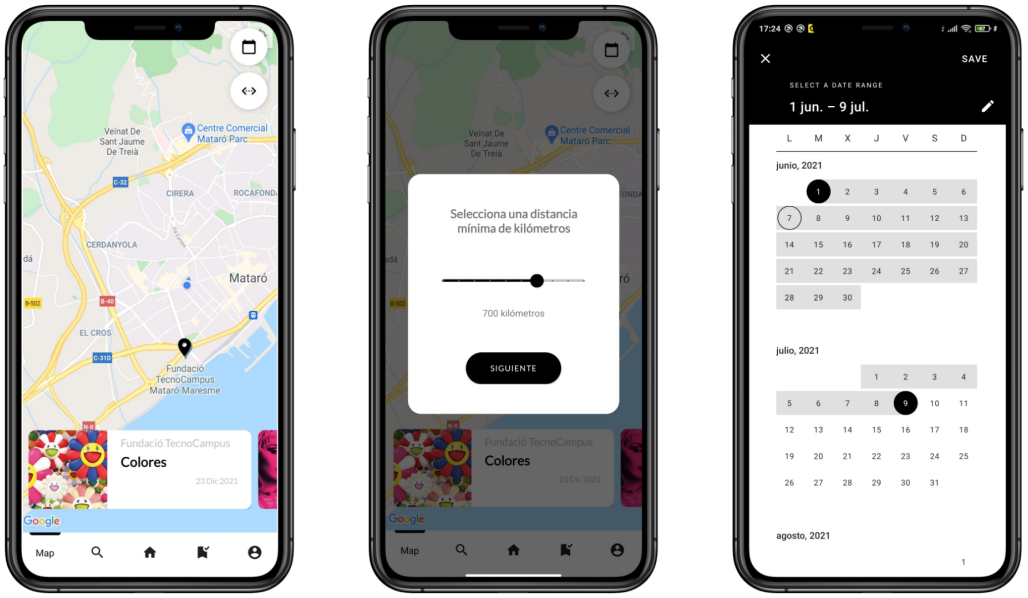

Fig 7.4 Mapa de concerts. Font: Elaboració pròpia

7.1.1.5 Buscador de concerts

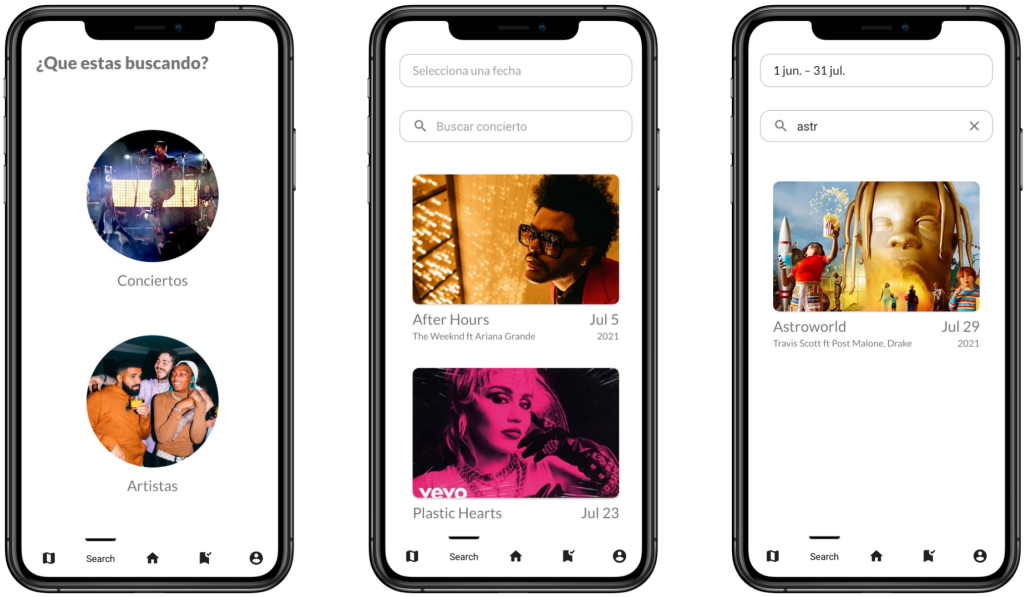

Fig 7.5 Buscador de concerts. Font: Elaboració pròpia

7.1.1.6 Buscador de d'artistes

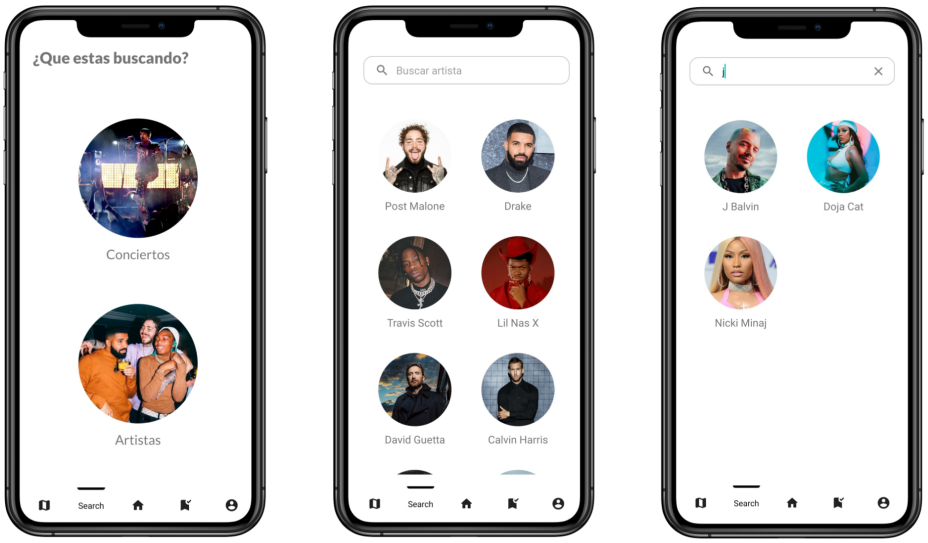

Fig 7.6 Buscador d'artistes. Font: Elaboració pròpia

7.1.1.7 Detalls d'un concert

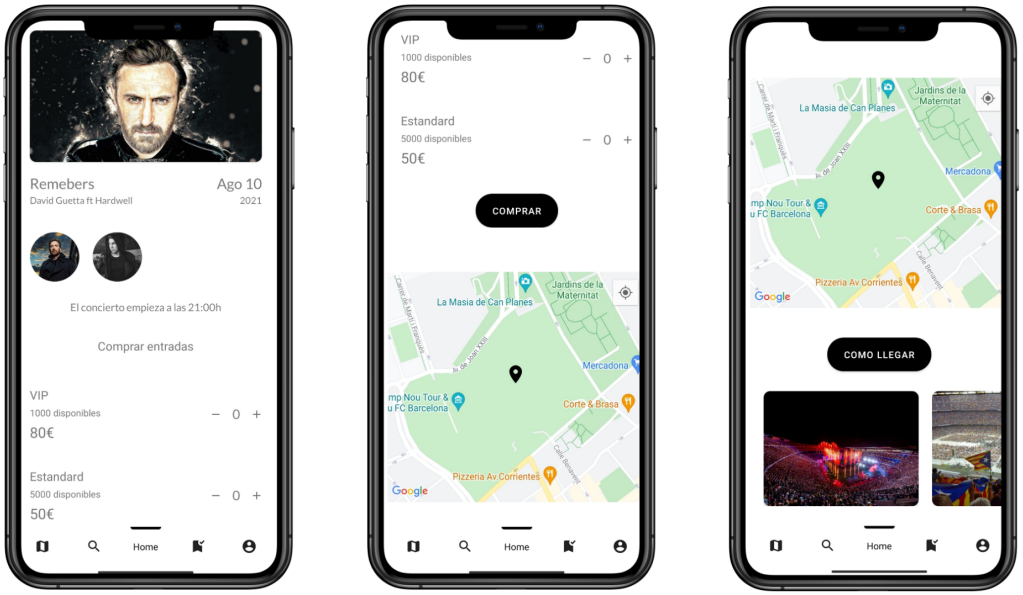

Fig 7.7 Detalls d'un concert. Font: Elaboració pròpia

7.1.1.8 Comprar entrades

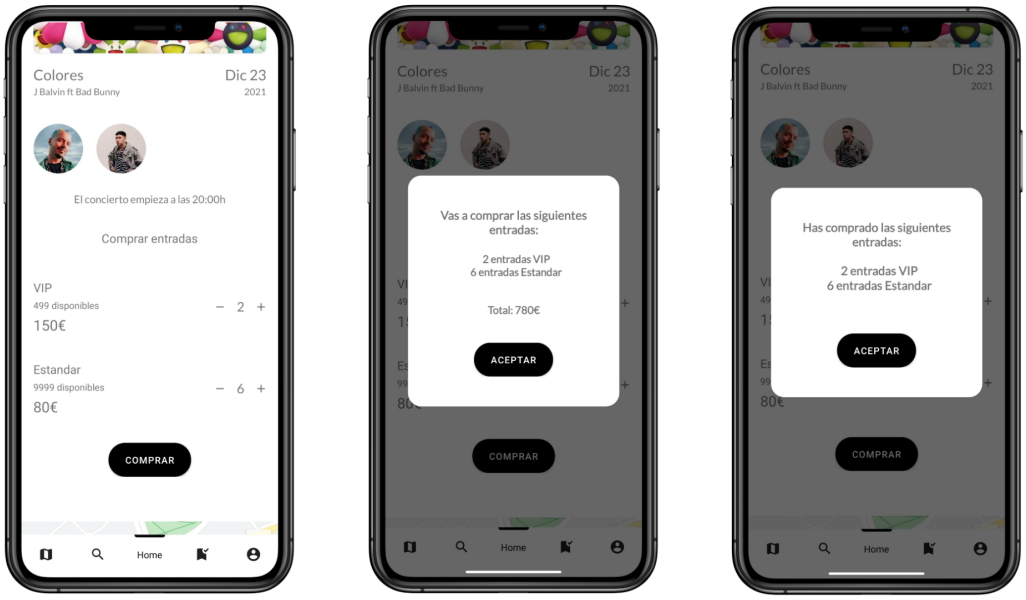

Fig 7.8 Procès de compra d'entrades. Font: Elaboració pròpia

7.1.1.9 Entrades comprades

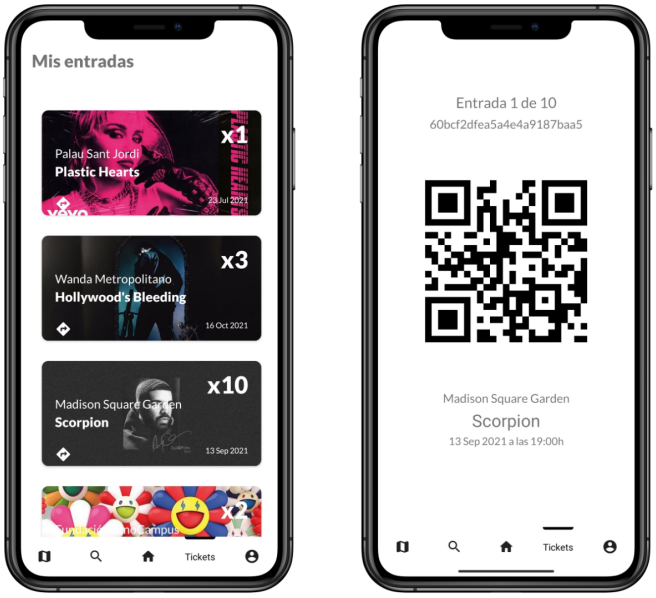

Fig 7.9 Llistat d'entrades comprades d'un usuari. Font: Elaboració pròpia

7.1.1.10 Valorar concerts assistits

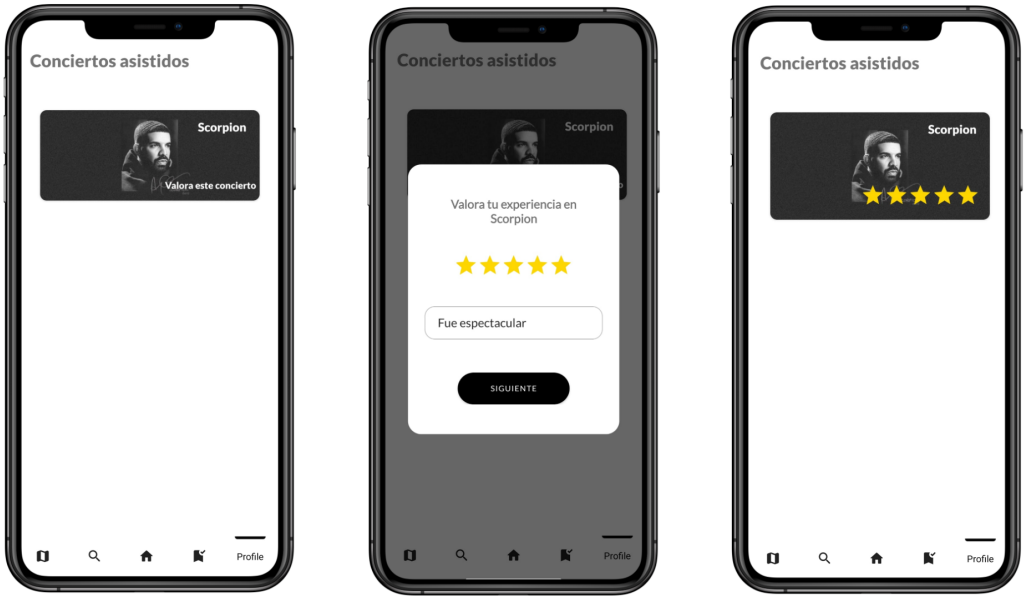

Fig 7.10 Valoració d'un concert assistit per un usuari. Font: Elaboració pròpia

7.1.1.11 Perfil d'un artista

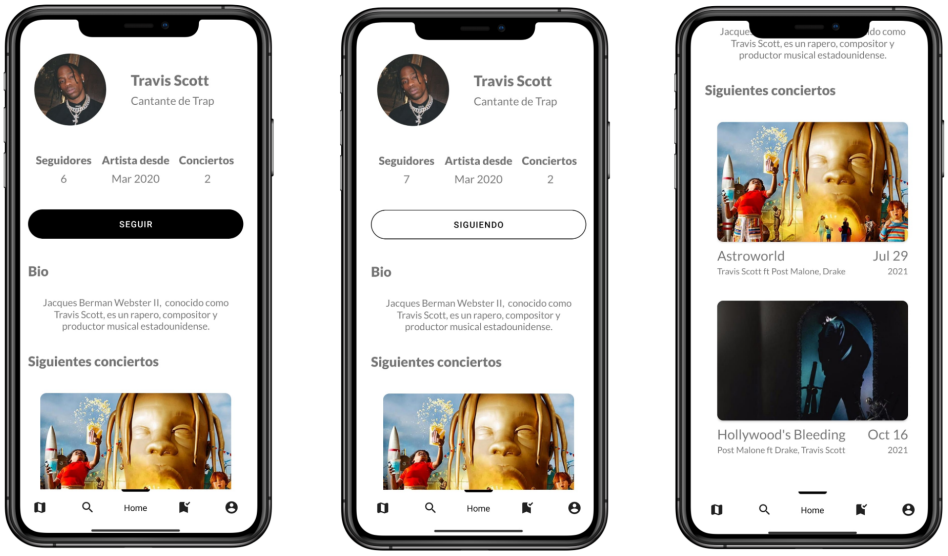

Fig 7.11 Perfil d'un artista. Font: Elaboració pròpia

7.1.2 Rol d'artista

Un artista de l'aplicació és un usuari que s'ha registrat com a artista des del dashboard.

Un artista pot crear un concert en un lloc i moment determinat juntament amb el preu i el nombre d'entrades.

A continuació és mostren captures d'algunes de les funcionalitats més importants de l'aplicació mòbil per part d'un artista.

7.1.2.1 Pantalla d'inici

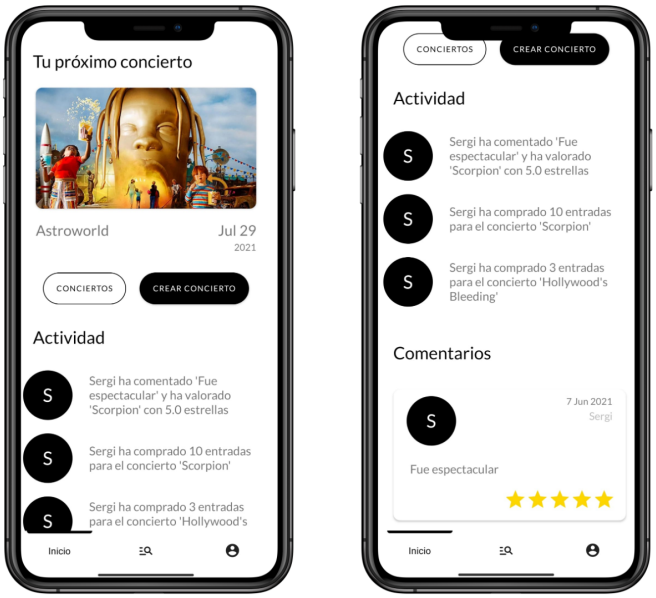

Fig 7.12 Pantalla d'inici d'un artista. Font: Elaboració pròpia

7.1.2.2 Creació d'un concert

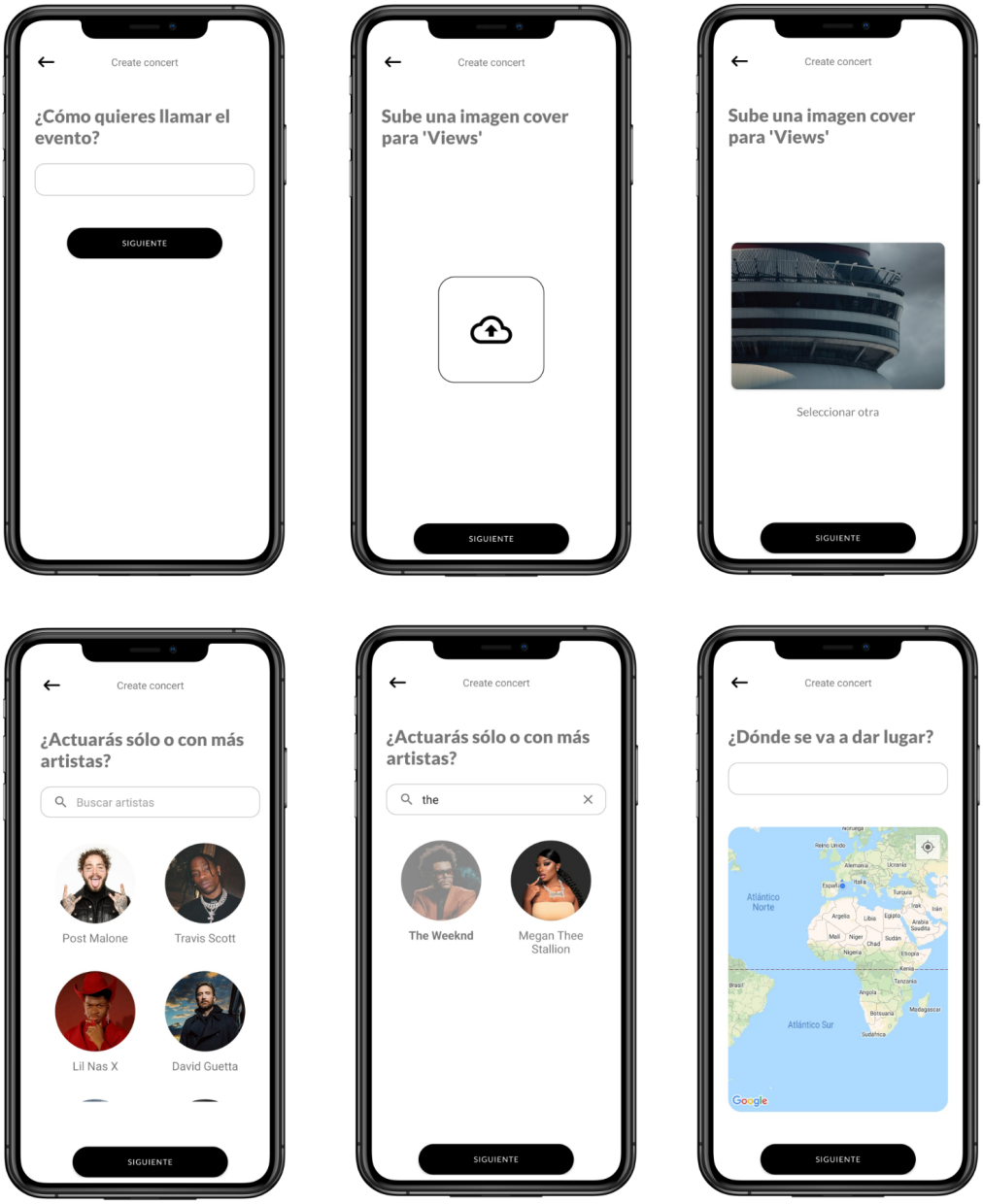

Fig 7.13 Creació d'un concert, primera part. Font: Elaboració pròpia

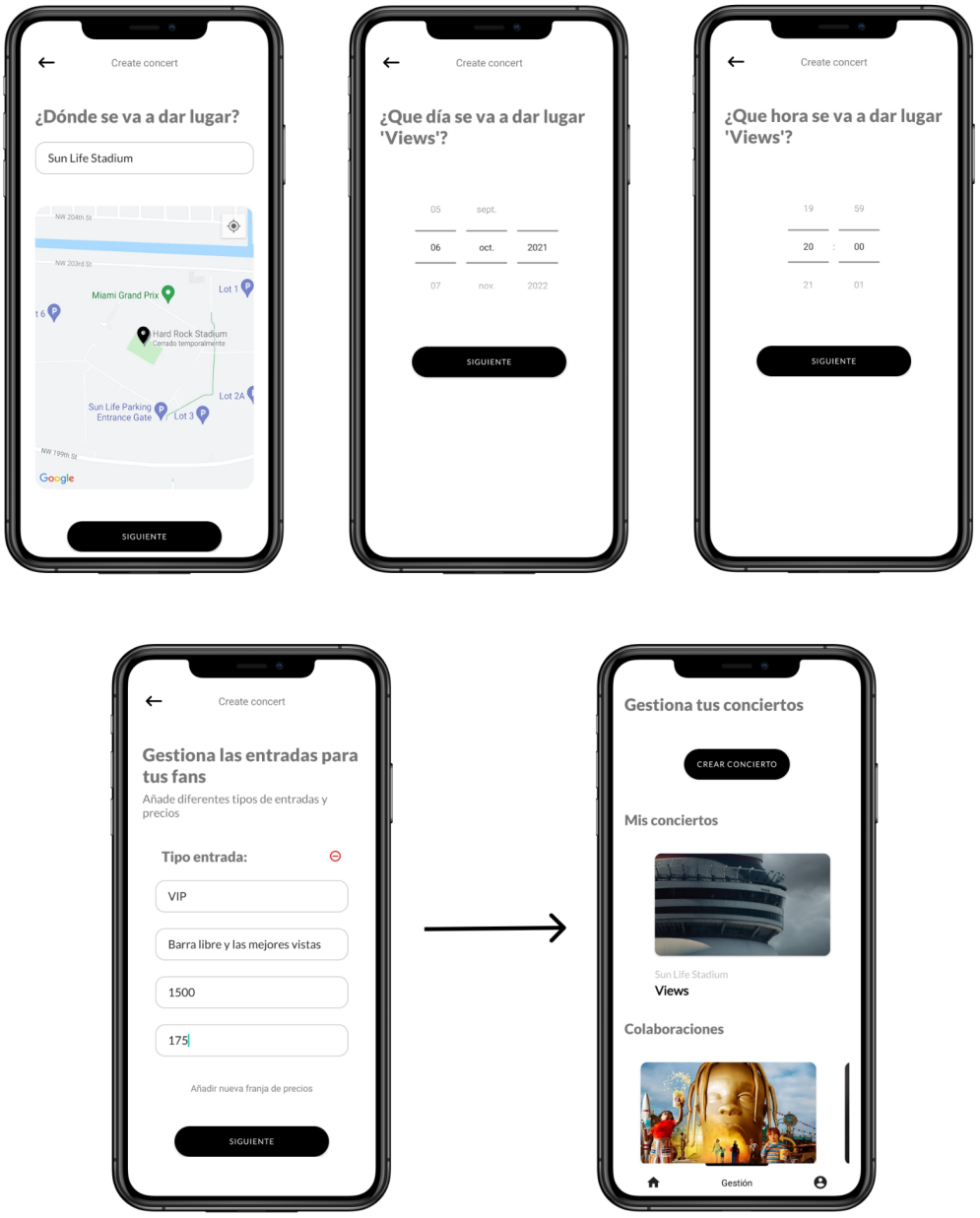

Fig 7.14 Creació d'un concert, segona part. Font: Elaboració pròpia

7.1.2.3 Pantalla de gestió de concerts

Fig 7.15 Gestió de concerts d'un artista. Font: Elaboració pròpia

7.2 Dashboard

Un usuari de l'aplicació no pot accedir al dashboard. Per tant, s'ha de convertir en artista.

El dashboard disposa de les mateixes funcionalitats d'un artista de l'aplicació mòbil.

A continuació es mostren captures d'algunes de les funcionalitats més importants del dashboard per artistes.

7.2.1 Pantalla d'inici de sessió

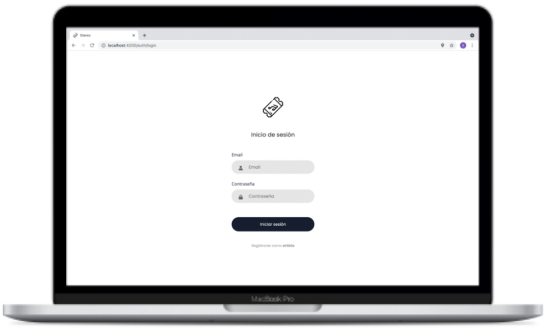

Fig 7.16 Pantalla de d'inici de sessió. Font: Elaboració pròpia

7.2.2 Pantalla d'inici

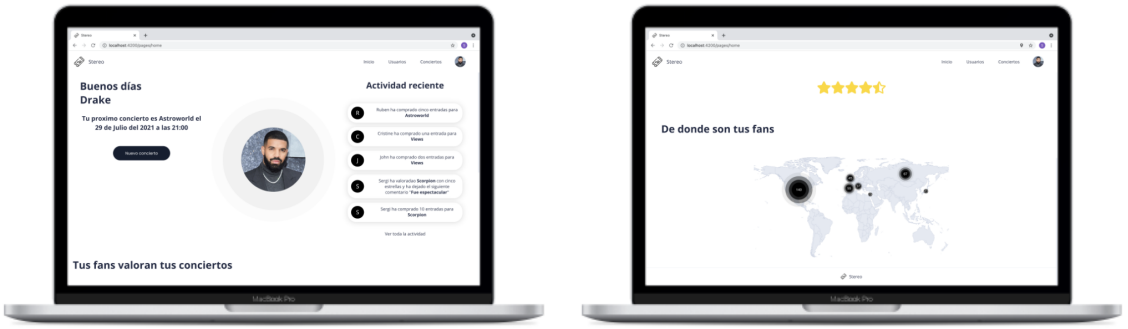

Fig 7.17 Pantalla principal del dashboard. Font: Elaboració pròpia

7.2.3 Registrar artista

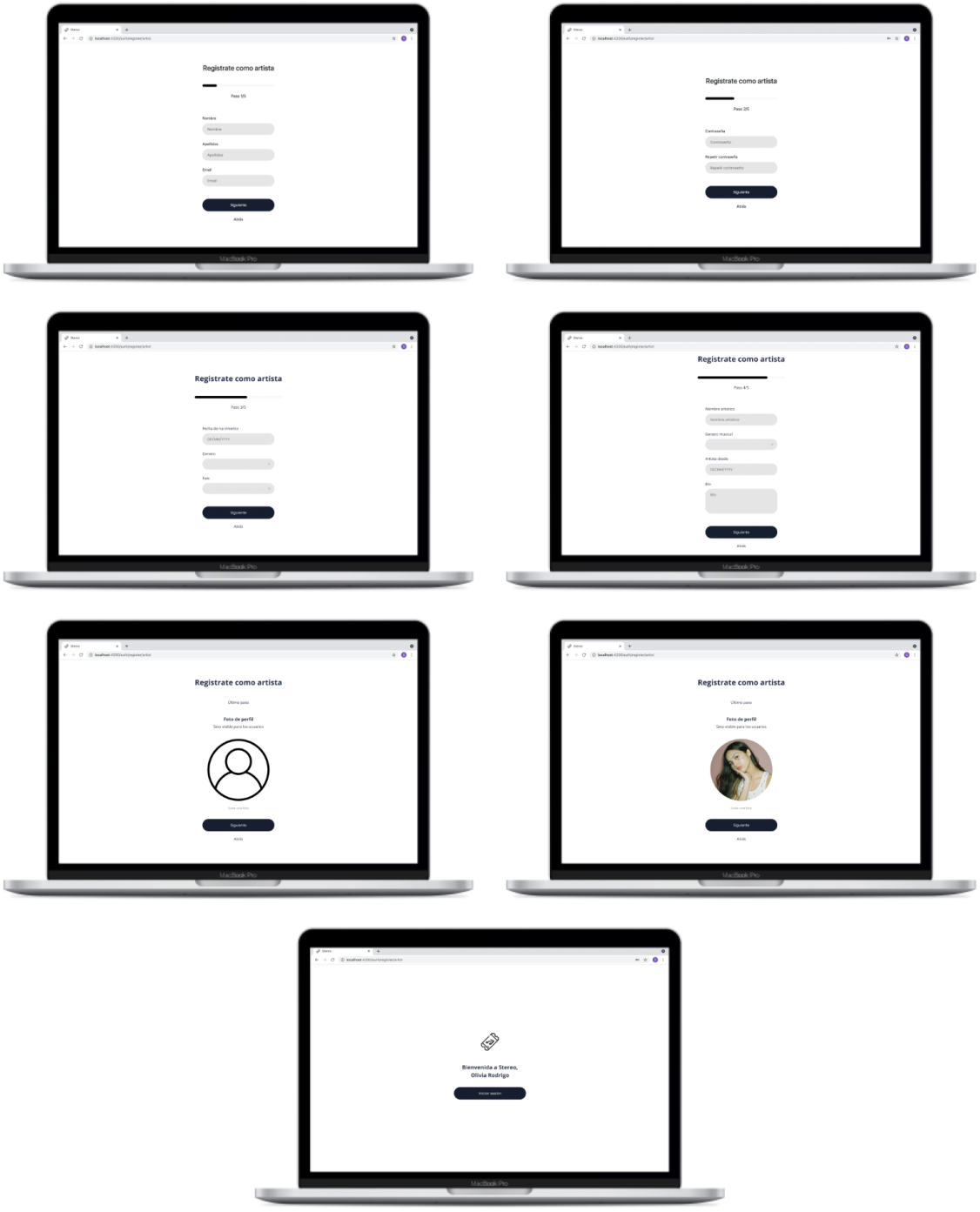

Fig 7.18 Registre d'un artista. Font: Elaboració pròpia

7.2.4 Pantalla de gestió de concerts

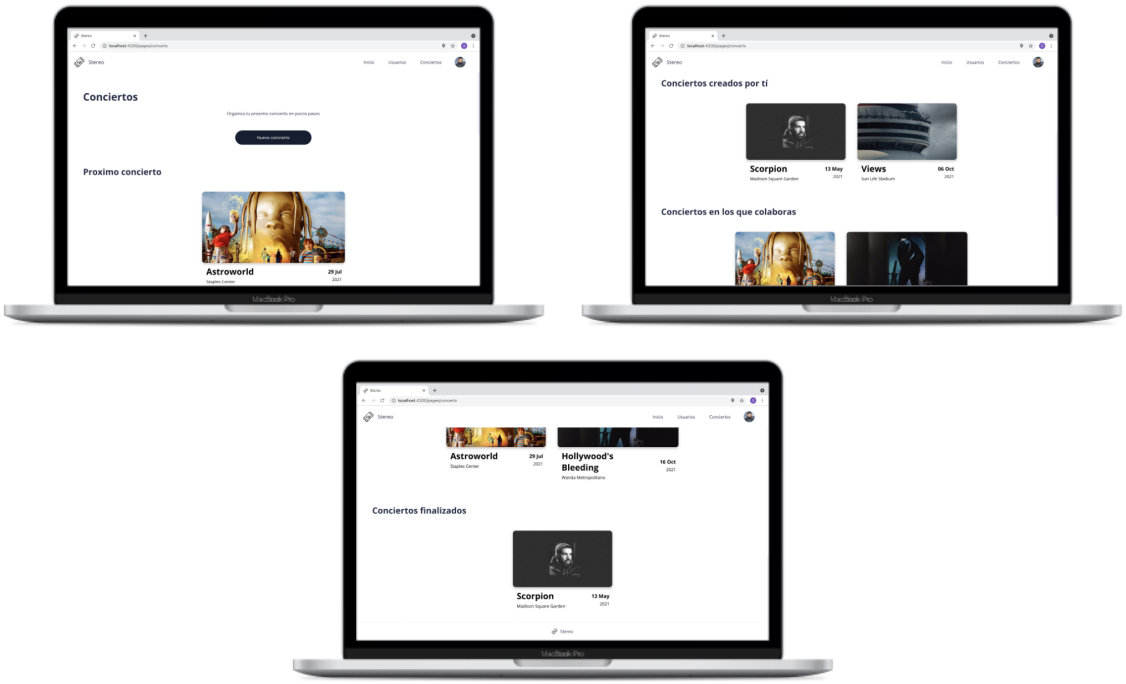

Fig 7.19 Gestió dels concerts d'un artista. Font: Elaboració pròpia.

7.2.5 Pantalla d'usuaris

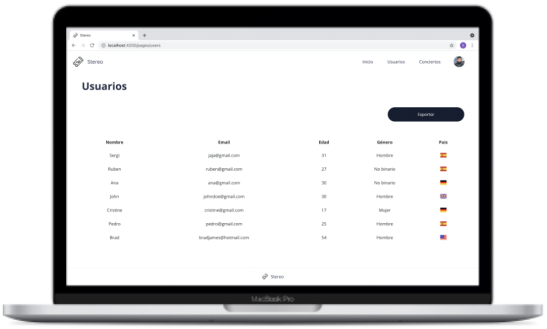

Fig 7.20 Llistat d'usuaris que assisteixen als concerts de l'artista. Font: Elaboració pròpia

8. Conclusions

Un cop finalitzat el projecte es conclou que és viable econòmicament com tecnoloògicament.

S'han completat totes les fases establertes en la fase de planificació satisfactoriament, algunes més complertes que d'altres, però totes complint les seves funcionalitats.

S'ha aconseguit un producte final que proporciona als usuaris una aplicació mòbil per Android per poder consultar concerts, reservar entrades i seguir artistes per estar atents a possibles nous concerts.

D'altra banda, els artistes ja poden comptar amb l'aplicació mòbil totalment especifica per ells i accedir al dashboard per tal de poder visualitzar i processar millor les dades relacionades amb els seus concerts.

Tant l'aplicació com el dashboard, la base de dades i l'API han estant dissenyats de manera que si s'afegeixen noves funcionalitats i entitats al disseny del projecte sigui totalment escalable i només faci falta afegir els canvis corresponents.

Un cop finalitzat el projecte es conclou que també hagues sigut viable i possible desenvolupar l'aplicació per Android i per iOS. Actualment el sector de les aplicacions hibrides esta en creixement i en constant millora.

Durant tot el procés de realització del projecte s'ha aconseguit millorar els coneixements d'Android, AWS, Angular, MongoDB i Spring de tal manera de que s'ha pogut desenvolupar satisfactoriament una aplicació i un dashboard des de zero amb aquestes tecnologies.

9. Ampliacions

Al tractar-se d'un treball de final de grau. El projecte es dona per concluit. Però hi han algunes ampliacions que s'han tingut en compte durant el procès de desenvolupament.

A continuació es detallen algunes ampliacions:

- Registrar un nou rol d'usuari que s'encarregui de llegir i validar les entrades quan un usuari assiteix i té entrades del concert de l'aplicació.

- Integrar els pagaments amb Stripe. Ara per ara tota reserva és gratuïta.

- Implementar un control de registre d'artistes. Cal identificar tot artista que es registra, poder contactar amb ells i d'alguna manera validar que l'artista en qüestió és real.

- Implementar la seguretat Oauth.

- Implementar push notifications per enviar notificacions de l'activitat d'un artista que segueixi l'usuari.

- Afegir traduccions.

10. Bibliografia

[1] What is AWS? [en línia] [consulta: 21 de Febrer de 2021].
Disponible a: What is AWS?

[2] What is AWS S3 bucket? [en línia] [consulta: 24 de Febrer de 2021].
Disponible a: What is AWS S3 bucket

[3] What is REST? [en línia] [consulta: 25 de Febrer de 2021].
Disponible a: What is REST?

[4] Remember the 5 W's [en línia] [consulta: 2 de Gener de 2021].
Disponible a: Remember the 5 W's | IT Best Practices | Nebraska

[5] Agile Manifesto [en línia] [consulta: 9 de Gener de 2021].
Disponible a: Manifesto for Agile Software Development

[6] What is Angular? [en línia] [consulta: 11 de Gener de 2021].
Disponible a: Introduction to the Angular Docs

[7] What is React? [en línia] [consulta: 1 de Març de 2021].
Disponible a: Introduction to React

[8] What is Vue? [en línia] [consulta: 1 de Març de 2021].
Disponible a: Introduction to Vue

[9] What is Spring? [en línia] [consulta: 5 de Març de 2021].
Disponible a: Spring Framework