# Plataforma de finançament col·laboratiu per a estudiants universitaris: Desenvolupament i Implementació

Jordi Salazar Badia

**Resum**

En aquest treball final de grau es presenta el disseny i el desenvolupament d'una plataforma de *Crowdfunding* destinada a la comunitat estudiantil de la Universitat Autònoma de Barcelona (UAB). L'objectiu principal de l'aplicació és democratitzar el finançament universitari, permeten als estudiants fer realitat les seves idees i projectes de manera eficient i segura. La plataforma busca fomentar la creativitat i la col·laboració entre els membres de la comunitat universitària, facilitant la cerca i selecció de projectes alineats amb els interessos dels usuaris.

D'altra banda, es du a terme un anàlisi de dues plataformes que han obtingut resultats exitosos seguint aquesta filosofia, com són Kickstarter i Goteo, avaluant les seves característiques principals i funcionament. Aquesta anàlisi serveix com a base per a definir els requisits funcionals i no funcionals de l'aplicació. L'arquitectura del sistema es basa en un model client-servidor monolític, utilitzant Node JS i Express.js per part del servidor i React Native com a front-end per a l'aplicació híbrida.

**Paraules clau**

Crowdfunding, Plataforma digital, Finançament, Emprenedoria , Innovació tecnològica.

**Abstract**

This final degree project presents the design and development of a Crowdfunding platform intended for the student community of the Autonomous University of Barcelona (UAB). The main objective of the application is to democratize university funding, allowing students to bring their ideas and projects to life in an efficient and secure manner. The platform aims to foster creativity and collaboration among members of the university community, facilitating the search and selection of projects aligned with users' interests.

Additionally, an analysis of two platforms that have achieved successful results following this philosophy, such as Kickstarter and Goteo, is carried out, evaluating their main characteristics and functioning. This analysis serves as a basis for defining the functional and non-functional requirements of the application. The system architecture is based on a monolithic client-server model, using Node.js and Express.js on the server side and React Native as the front-end for the progressive web application (PWA).

**Index Terms**

Crowdfunding, Digital Platform, Financing, Entrepreneurship, Technological Innovation.

# 1 Introducció - Context del treball

En el context universitari actual, els estudiants de la Universitat Autònoma de Barcelona (UAB) enfronten una sèrie de desafiaments a l'hora de finançar i promoure els seus projectes. La manca d'una plataforma centralitzada que permet gestionar aquests aspectes de manera eficient i segura representa un obstacle significatiu. Aquest buit dificulta la col·laboració entre els membres de la comunitat estudiantil i limita les oportunitats de finançament per a projectes innovadors.

Actualment, els estudiants han de recórrer a mètodes dispersos i sovint ineficaços per aconseguir fons, com ara l'ús de múltiples xarxes socials, correus electrònics i reunions presencials, la qual cosa pot ser tediós i poc productiu. A més, l'absència d'una eina dedicada complica la visibilitat dels projectes i la possibilitat d'atraure possibles finançadors interessats.

Un altre problema crític és la manca d'un sistema de seguiment transparent. Sense una manera clara i accessible de monitorar el progrés dels projectes i la gestió de les inversions, els finançadors potencials poden ser reticents a contribuir. La transparència i la confiança són fonamentals per a l'èxit de qualsevol iniciativa de Crowdfunding, i la manca d'aquests elements pot afectar la participació i el suport als projectes estudiantils.

Per tant, s'identifica la necessitat de crear una plataforma integral que abordi aquestes deficiències. La proposta és desenvolupar una aplicació que faciliti el finançament col·laboratiu, fomenti la creativitat i proporciona transparència en el procés d'inversió i seguiment de projectes. Aquesta plataforma no només permetrà als estudiants dirigir el capital directament cap a les idees que consideren valuoses, sinó que també crearà un espai col·laboratiu on puguin compartir i exposar els seus projectes, estimulant així la innovació dins de la comunitat universitària.

En resum, l'objectiu principal d'aquesta plataforma és democratitzar el finançament universitari, fent accessible i eficient el procés de finançament de projectes, i promovent un entorn on la creativitat i la col·laboració puguin florir sense les restriccions actuals.

# 2 Anàlisi de plataformes exitoses

En l'actualitat, diverses plataformes han arribat a l'èxit amb un model de negoci semblant al proposat en aquest treball. A continuació, una anàlisi de dues plataformes de gran renom, impulsades pel Crowdfunding.

## 2.1 Kickstarter

L'objectiu principal de la plataforma són projectes de naturalesa creativa, les seves seccions més populars són: Art, Còmics, Artesania, Dansa, Disseny, Moda, Cinema i Vídeo, Menjar, Jocs, Periodisme, Música.

Funciona amb un mètode de finançament del 'tot o res', és a dir, els projectes un cop publicats, han d'establir quin és exactament l'objectiu que persegueixen. D'aquesta manera, només reben la recaptació, si aquesta arriba al 100% de l'objectiu inicial, d'altre banda, no es realitza cap transacció.
Un dels principals beneficis que obtenen els nous projectes que s'inicien en la plataforma és el de tenir a la seva disposició la gran comunitat de la qual disposa la plataforma, de la qual en poden treure molts 'backers' (persones que donen suport al projecte). Tot i això, els creadors de la plataforma insisteixen en el fet que cada projecte hauria de crear marca pròpia utilitzant màrqueting digital i altres eines d'exposició, ja que això fomenta tant al creixement dels mateixos projectes, com al de la plataforma, la qual cosa també acaba sent positiu pels emprenedors. Com podem veure, això genera un cicle de retroalimentació on tots els participants es beneficien del creixement mutu, i aquest és un dels aspectes de la seva popularitat.

Com es pot donar suport a projectes? La realitat és que és senzill, en la pantalla d'inici trobem un buscador, on podem fer recerca per diferents filtres com les categories, o projectes a prop nostre. Això fa fàcil trobar projectes que ens puguin semblar interessants, tothom pot donar suport a qualsevol projecte i no existeix una quota mínima. En fer el pagament, de manera semblant a qualsevol pagament digital, els diners no arriben directament als projectes, sinó que es guarden en forma de 'promesa', ja que si el projecte que volem finançar no arriba a l'objectiu comentat anteriorment, ens tornaran els diners.
D'aquest finançament, la plataforma aplica un 5% d'interessos.

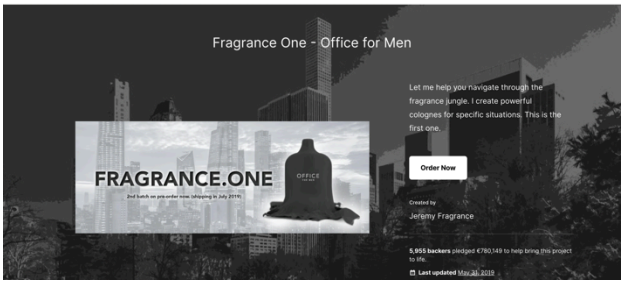
Dades de Kickstarter:

-> 92.838.577 **promeses realitzades.**
-> 252.561 **projectes finançats**.
-> 7.787.971.507 **dòlars en finançament de projectes.**
-> 3.000.000 **de visites mensuals**

Per què finançar projectes a Kickstarter? La resposta no és senzilla, més endavant analitzarem casos d'èxit a la plataforma, però de forma general, la motivació ve donada principalment per dues raons:

- Interès personal en el projecte, sigui pel caràcter innovador o preferències de l'usuari.
- Recompenses i exclusivitat, la plataforma no promet el compliment de les recompenses, tot i això, pren mesures per evitar estafes com actualitzacions periòdiques sobre el seu estat. Cal esmentar les crítiques que ha rebut la plataforma per la poca garantia que dona de com seran utilitzats els diners rebuts per cada projecte, els usuaris demanen més transparència per part d'aquests.
  Per poder entendre la naturalesa dels projectes exitosos dins d'aquest marc de finançament, penso que és molt interessant analitzar dos casos de gran èxit dins de Kickstarter:

1. Fragància Jeremy – 'Office For Men', la marca va llençar una campanya demanant 25.000 euros a la comunitat, el que va acabar sent 780.149 euros de la mà de 5.955 simpatitzants. *("Ejemplos de Kickstarter: 7 campañas de éxito para financiar productos y marcas". Social Media One; n.d.)* Aquest és un exemple perfecte de l'oportunitat per persones que ja tenen una base de fans per explotar el seu potencial, el Jeremy és 'influencer' i la seva popularitat ha contribuït de gran manera a què el seu projecte fos tan popular. Les recompenses d'aquest projecte varien depenent de la donació de l'usuari, que podia triar entre diferents opcions, des de 5 euros, que simplement et permet aportar al projecte desinteressadament, fins a una aportació de 2250 euros, que et garantia a canvi dues ampolles de la fragància.




Font: Kickstarter. "Fragrance One - Office for men by Jeremy Fragrance" https://www.kickstarter.com/projects/jeremyfragrance/fragrance-one-office-for-men

2. Oculus Rift - Step into the Game, un nou i modern cas de realitat virtual (RV) dissenyat específicament per a videojocs. En aquest cas la campanya va començar amb l'objectiu de recaptar 250.000 dòlars, i ja van pels 2.437.429 dòlars, donats per 9.522 'patrocinadors'.


Font: Kickstarter. "Oculus Rift: Step into the game; by Oculus". https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game?ref=discovery

Em sembla interessant com dos projectes considerats de nínxol han pogut tenir un èxit tan gran dins la mateixa plataforma.

### 2.2 Goteo

També veurem la plataforma Goteo, plataforma pública de la ciutat de Barcelona. De la mateixa manera que la proposta vista anteriorment, Goteo també pretén impulsar projectes a través del Crowdfunding. Aquesta semblants, i és que és pública i sense ànims de lucre, té el suport tant de l'Ajuntament de Barcelona com del govern estatal. Un dels aspectes positius de la plataforma és que proporcionen incentius econòmics a les persones amb interès a donar suport als projectes, ja que et permet desgravar bona part de l'aportació econòmica.


Font: Goteo: calculadora fiscal. "A Goteo, mullar-se desgrava". https://ca.goteo.org/calculadora-fiscal

Goteo dona molta importància a la visibilitat de les dades tant d'impacte com d'estadístiques, a continuació veurem les més interessants:

Categories d'interès: és important saber quins són els interessos principals dels usuaris de la plataforma, que podrien ser representatius de la regió, donant molta importància a projectes de caràcter social (20,4%), científic/ tecnològic (16,3%) cultural (15,3%), educatiu (15,1%) i ecològic (13,2%). El nombre de projectes exitosos creix cada any des del primer any 2011, sent la xifra un 88,89% respecte al total el 2023. També ha crescut la recaptació total per a projectes, amb un total de 19.442.135 € des de la seva creació. *("Goteo.org")*

Goteo comparteix una característica amb Kickstarter en la manera de deixar que els projectes estableix diferents tipus d'objectius, al crear un projecte a la seva plataforma et permeten posar un 'mínim' de recaptació, que és el que necessita el projecte com a mínim per a prosperar. També deixen posar un 'òptim' que, seria l'objectiu real que tenen present els propietaris del projecte.

Goteo té molt al cap el seu objectiu principal, generar un impacte social elevat, fent costat a projectes innovadors.

## 3 NATIU, HÍBRID O PWA

Es considera oportú comentar breument el marc actual i les diferents maneres de desenvolupar aplicacions mòbils. Atès que aquest treball s'orienta cap a una aplicació, és necessari introduir les possibles solucions tecnològiques disponibles. Així, es presenten i justifiquen les alternatives principals: aplicacions natives, aplicacions híbrides i aplicacions web progressives (PWA). Cada una d'aquestes opcions té els seus propis avantatges i desavantatges, i la selecció de la solució adequada dependrà dels requisits específics del projecte.

En l'actualitat, el panorama de desenvolupament d'aplicacions mòbils ha evolucionat significativament, oferint una varietat d'enfocaments per crear experiències d'usuari atractives i funcionals. Entre els enfocaments més destacats es troben les 'Progressive Web Apps' (PWA's), les aplicacions híbrides i les aplicacions natives. Cadascuna d'aquestes opcions té les seves pròpies característiques, avantatges i desavantatges, i l'elecció entre elles depèn en gran manera dels requisits del projecte i les necessitats específiques del negoci.


Font: Goteo: pàgina principal. https://ca.goteo.org/

Les PWA's han guanyat popularitat en els últims anys a causa de la seva capacitat per combinar el millor de les aplicacions web i les aplicacions natives. Una PWA és una aplicació web que utilitza tecnologies web estàndard com HTML, CSS i JavaScript, però està dissenyada per funcionar i sentir-se com una aplicació nativa. Una de els principals avantatges de les PWA's és la seva capacitat per funcionar en diferents plataformes i dispositius, la qual cosa les fa altament accessibles per a una àmplia audiència. A més, les PWA's poden funcionar sense connexió, aprofitant les capacitats dels Service Workers per emmagatzemar en memòria cau continguda i dades importants. Això permet una experiència d'usuari fluida fins i tot en condicions de connectivitat intermitent o absència.

D'altra banda, les aplicacions híbrides combinen tecnologies web i natives per crear aplicacions que s'executen en múltiples plataformes. Aquestes aplicacions solen desenvolupar-se utilitzant frameworks com React Native o Iònic, que permeten als desenvolupadors escriure codi en un llenguatge de programació web com JavaScript i després compilar-lo en codi natiu per a iOS i Android. Tot i que les aplicacions híbrides ofereixen avantatges com la capacitat de compartir codi entre plataformes i un desenvolupament més ràpid, sovint s'enfronten a reptes de rendiment i compatibilitat, ja que no poden aprofitar completament les característiques i funcionalitats natives de cada plataforma.

D'altra banda, les aplicacions natives es desenvolupen específicament per a una plataforma en particular, com iOS o Android, utilitzant llenguatges de programació i eines natives proporcionades pels respectius sistemes operatius. Tot i que les aplicacions natives poden requerir més temps i recursos per desenvolupar-se en comparació amb les PWA's i les aplicacions híbrides, ofereixen un rendiment superior i una experiència d'usuari més fluida, ja que poden aprofitar totes les funcionalitats i capacitats natives del dispositiu, com l'accés al maquinari, notificacions push, i optimització de rendiment.

No obstant això, malgrat els avantatges de les aplicacions natives, les PWA's i les aplicacions híbrides estan guanyant terreny a causa de la seva accessibilitat, facilitat de desenvolupament i capacitat per arribar a una audiència més àmplia. A més, les PWA's en particular estan guanyant tracció a causa de la seva capacitat per oferir una experiència d'usuari similar a la d'una aplicació nativa sense la necessitat de descarregar i instal·lar una aplicació des d'una botiga d'aplicacions. Això ha portat a una disminució en l'interès i l'adopció d'aplicacions natives en alguns casos, especialment en mercats on les PWA's i les aplicacions híbrides poden satisfer les necessitats dels usuaris de manera més efectiva i rendible.

En resum, mentre que les aplicacions natives continuen sent una opció viable per a certs casos d'ús que requereixen un rendiment superior i accés complet a les capacitats del dispositiu, les PWA's i les aplicacions híbrides estan guanyant terreny a causa de la seva accessibilitat, facilitat de desenvolupament i capacitat per arribar a una audiència més àmplia. L'elecció entre aquests enfocaments dependrà dels requisits específics del projecte, les necessitats del negoci i les preferències de l'usuari final.

## 4 REQUISITS DEL SOFTWARE

A continuació, llistaré els principals requisits del software per a una aplicació d'aquestes característiques.

*Requisits Funcionals*

Verificació d'Adreça de Correu Electrònic:
Implementar un codi d'enviament al compte de l'usuari per verificar la seva adreça de correu electrònic quan es registra.

Canvi de Contrasenya:
Proporcionar un procediment, mitjançant un codi al compte de correu electrònic, per a permetre als usuaris canviar la seva contrasenya.

Avaluació de Projectes:
Proporcionar una funcionalitat perquè els usuaris puguin valorar els projectes. Això permet als usuaris que no col·laboren directament expressar la seva opinió sobre els projectes. A més, si un projecte no compleix amb les seves promeses, els usuaris poden assignar-hi una puntuació negativa per informar a la comunitat.

Inici de Sessió i Tancament de Sessió:
Permetre als usuaris iniciar sessió de manera segura amb JWT.
Facilitar l'opció de tancar sessió per garantir la privadesa de l'usuari. També han de tenir l'opció d'eliminar el seu compte si així ho desitgen.

Registre de Perfil:
Proporcionar la possibilitat de crear un perfil personalitzat amb nom d'usuari, foto de perfil i una petita descripció. Els perfils seran tots públics per als usuaris registrats, per seguir altres usuaris l'usuari ha d'estar registrat, i la funcionalitat es basa a tenir un seguiment dels projectes d'aquell usuari concret, rebre notificacions, etc.

Pujada de Projectes:
Permetre als usuaris carregar informació detallada sobre nous projectes.
Incloure camps per a títol, descripció, categoria, objectiu de finançament, termini, altres detalls rellevants.

Edició de Projectes Propis:
Els usuaris han de ser capaços d'editar i eliminar els projectes que ells mateixos han creat.

Exploració de Projectes:
Implementar un sistema de cerca que permeti als usuaris trobar projectes segons categories específiques i per títol específic.

Transaccions Financeres:
Habilitar la realització segura de transaccions monetàries.
Integrar passarel·les de pagament fiables per garantir la seguretat de les transaccions. Implementat amb STRIPE

Seguiment de Projectes:
Proporcionar als usuaris una secció dedicada per fer un seguiment dels projectes que han pujat.
Incloure informació detallada sobre l'estat del projecte, actualitzacions i qualsevol canvi rellevant.

Notificacions:
Integrar un sistema de notificacions per informar els usuaris sobre el progrés dels projectes en els quals han invertit, per exemple, avisar quan el projecte ha arribat al 100% de finançament.

Requisits d'Interfície Externa:
Disseny UI/UX:
Realitzar un disseny senzill que faciliti la interacció de l'usuari amb l'aplicació, fent que el contingut sigui l'aspecte més important per damunt de l'atractiu visual.

*Requisits no Funcionals*
Usabilitat:
Dissenyar una interfície d'usuari intuïtiva i fàcil d'utilitzar per acomodar usuaris amb diversos nivells d'experiència tecnològica.

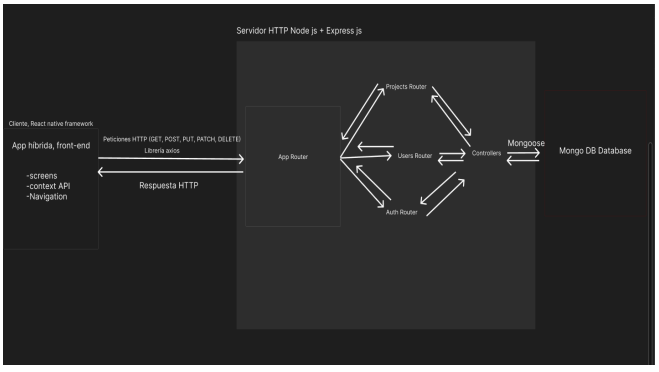
Compliment Legal i Normatiu:
Complir amb les lleis i regulacions locals relacionades amb la privadesa de les dades, les transaccions financeres i el micromecenatge.

Extensibilitat:
Dissenyar el sistema de manera que sigui fàcil de mantenir i actualitzar amb el temps.
Documentar adequadament el codi i els processos per facilitar la gestió del sistema.

Interoperabilitat:
Garantir la interoperabilitat amb sistemes externs, com passarel·les de pagament i serveis de notificació, per a una integració sense problemes.

## 5 ARQUITECTURA DE L'APLICACIÓ

A continuació, es pot observar un diagrama de l'arquitectura, a alt nivell, de l'aplicació:




Font: pròpia. Diagrama de l'arquitectura de l'aplicació

L'aplicació utilitzarà una arquitectura basada en un model client-servidor monolítica. El servidor es basa en Node.js amb Express.js mentre que el front-end estarà desenvolupat amb React Native. La comunicació entre el front-end i el back-end es farà mitjançant peticions HTTP utilitzant la llibreria Axios.

El back-end estarà encarregat de gestionar la lògica de negoci, l'accés a la base de dades, la gestió d'usuaris i la implementació de les funcionalitats requerides. S'utilitzarà Express.js per crear els diferents *endpoints* de l'API que proporcionaran accés a les diferents funcions de l'aplicació. Com a base de dades s'ha decidit utilitzar MongoDB, una base No SQL (not only SQL) basada en documents, a causa de la flexibilitat que ens permet al guardar les dades, i Mongoose, que ens permet interactuar amb la base de dades de forma ràpida i eficient, i validar les dades d'entrada.

El front-end, basat en el framework React Native per desenvolupar una interfície d'usuari nativa per a dispositius mòbils. Mitjançant Axios, farem peticions HTTP al back-end per obtenir i enviar dades, com ara informació d'usuari, detalls dels projectes, notificacions i altres funcionalitats relacionades.

Aquesta arquitectura permetrà una separació clara de responsabilitats entre el front-end i el back-end, facilitant el desenvolupament, la depuració i la mantenibilitat de l'aplicació. La comunicació mitjançant HTTP proporcionarà una integració robusta i escalable entre les dues parts de l'aplicació.

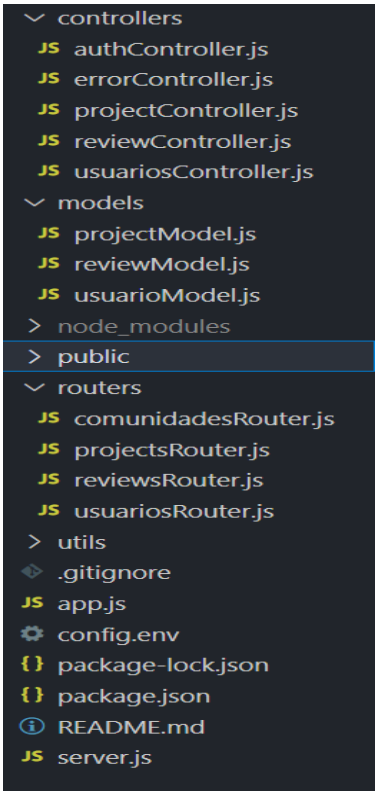
El flux d'una petició des del client per obtenir dades de la

base de dades seria el següent:

El client envia una petició HTTP a un URL específica del servidor. Prèviament, s'ha configurat un enrutador que, segons la petició, sap on ha de redirigir-la perquè un controlador específic pugui gestionar les necessitats (validar la identitat de l'usuari, obtenir o modificar dades de la base de dades, etc.).

També s'ha implementat un controlador d'errors, que rep tots els errors que puguin ocórrer en l'aplicació (tant de codi com d'usuari) i s'encarrega de llençar l'excepció corresponent amb un missatge d'error que es pugui entendre fàcilment.
Amb aquesta estructura, es garanteix que les peticions del client siguin gestionades de manera eficient i segura, assegurant la integritat i la seguretat de les dades.

Per entendre el sistema de fitxers que hi participen, la següent captura mostra el directori del projecte:



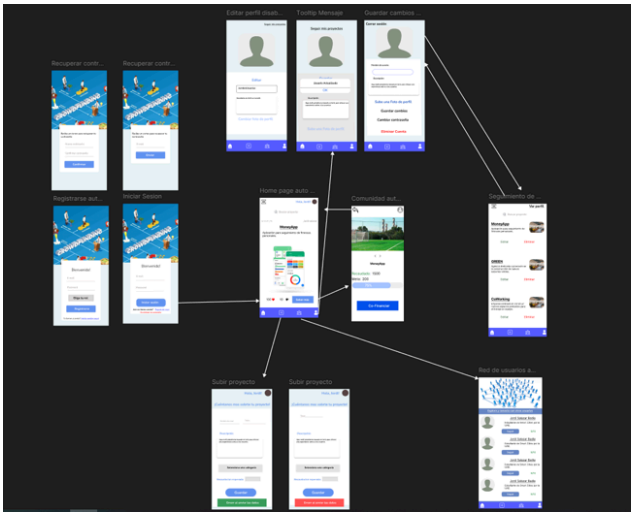

Font: pròpia. Directori del projecte

## 6 DESENVOLUPAMENT I RESULTAT

El desenvolupament de l'aplicació es va dividir en les següents fases, en aquest mateix ordre.

1. Disseny de l'aplicació:

En aquesta primera etapa, ja es tenia clar quin era el problema a resoldre i es va començar a dissenyar el *wireframe* de l'aplicació a Figma, on es van prendre les decisions de com s'havia de veure l'aplicació des d'un punt de vista gràfic per l'usuari final. Es va arribar a la conclusió que el millor era escollir una paleta de colors clara i una interfície UI/UX amigable.
Finalment, es va començar amb el disseny de les pantalles:

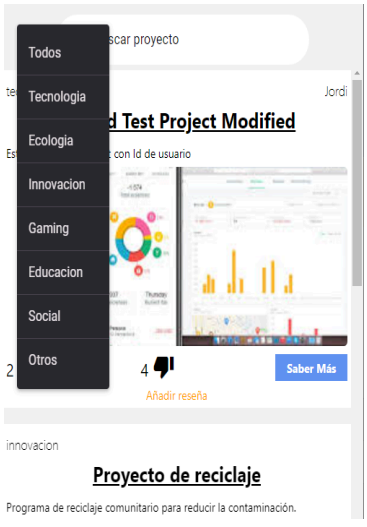


Font: pròpia. Disseny pantalles aplicació.

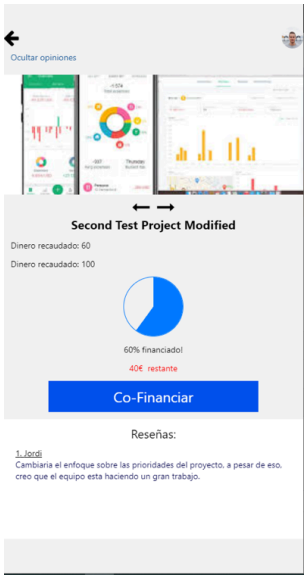
Les fletxes reflecteixen les possibilitats de navegació per l'usuari utilitzant l'aplicació.

**Nota**: La pantalla d''amistats' no ha estat implementada, ja que s'ha afegit per a una possible versió posterior, on els usuaris a part d'administrar i buscar projectes, puguin connectar amb altres usuaris en un estil més semblant a una xarxa social. En un principi, l'aplicació no es va pensar amb la finalitat de ser utilitzada com una xarxa social, tot i així, el fet de poder crear connexions amb altres usuaris podria millorar la confiança en la plataforma, en veure a qui estan finançant d'una manera més transparent.

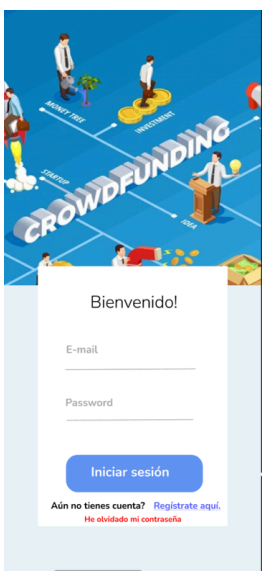
Les següents pantalles són les principals i més importants pel funcionament i objectiu de l'aplicació:

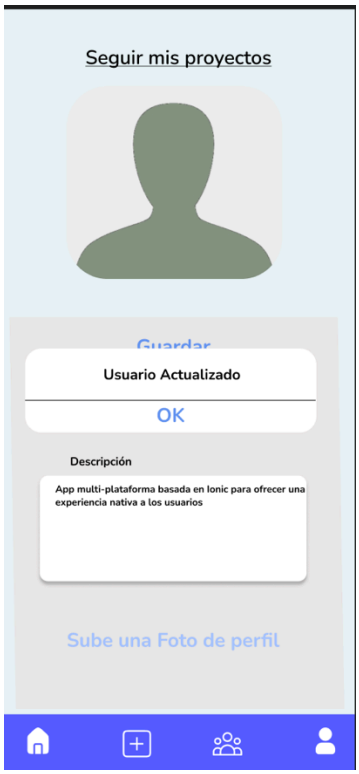
*Pàgina 'Home'*- pàgina inicial de l'aplicació, on els usuaris ja autenticats poden buscar projectes per títol i per categories específiques, I també la pàgina de detalls de cada projecte, on dinàmicament podem veure els projectes individualment, i deixar una opinió, aquesta decisió es va prendre després de veure la falta de confiança que els usuaris tenien en altres plataformes com Kickstarter.




*Pàgina de detalls del projecte* - si un usuari mostra interès en un projecte en particular i fa clic en el botó de 'Saber más', es farà una petició GET al servidor que retornarà la informació del projecte. També es redirigeix a l'usuari a la ruta corresponent de forma automàtica.




*Pàgina 'Log In'*- on els usuaris que ja estan registrats a l'aplicació amb un mail i contrasenya, poden accedir a l'aplicació. L'autenticació és un punt clau de tota aplicació, és per això que s'ha fet ús de l'algoritme Bcrypt per encriptar les contrasenyes dels usuaris, en fer clic a 'Iniciar Sesión', es comprova que la contrasenya que s'ha introduït, un cop encriptada, és la mateixa que la contrasenya que guardem a la base de dades associada al mail en qüestió.



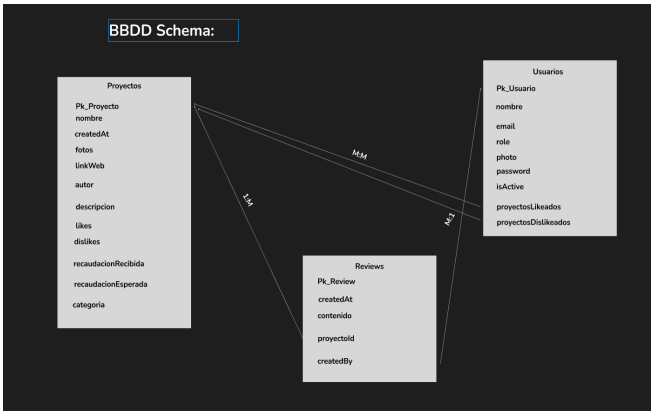

*Pantalla de perfil de l'usuari*- on es pot personalitzar amb una foto de perfil, i es pot actualitzar tant el nom d'usuari com la descripció del perfil fent una petició PATCH al servidor. També es pot fer clic a 'seguir mis proyectos'


*Pàgina dels projectes de l'usuari*- en aquest cas es fa una petició GET al servidor amb l'objectiu de retornar tant els projectes que ha creat l'usuari, com aquells en el que

l'usuari ha donat 'me gusta'.

2. Disseny de la base de dades

Un cop es tenia el disseny de l'aplicació, es va analitzar de tal forma que es podia saber quina informació era necessari guardar, per tant, es va crear un disseny de l'esquema de la base de dades, amb les seves respectives relacions:




Font: pròpia. Disseny base de dades.

El següent pas era implementar el disseny anterior, es va decidir utilitzar MongoDB per guardar la informació. Un cop es va prendre aquesta decisió, s'havia de pensar en com guardar les relacions entre les diferents entitats, en el cas de les 'Reviews', que tenia una relació d'1 a molts, ja que un projecte pot tenir moltes *Reviews, però* cada una d'aquestes només pot pertànyer a un projecte, es va decidir implementar el que es coneix com a 'Parent referencing' és a dir, en l'entitat de *Reviews* guardem l'ID del projecte al qual pertanyen. La relació entre *Reviews* i usuaris és la mateixa, ja que guardem el Id de l'usuari en l'altra taula, en el 'createdBy'.

Per últim, les relacions de M:M (many-to-many) que guarden els usuaris amb els projectes que han donat like o dislike. Aquesta va ser una de les decisions més complicades, finalment es va decidir guardar la informació a la taula d'usuaris, ja que quan volem obtenir la informació de l'usuari al carregar la primera pantalla, ens interessa saber ja a quins projectes ha donat like o no per poder mostrar-ho a la UI, i aquesta necessitat feia que fos interessant guardar-ho a la mateixa taula. D'altra banda, la quantitat de likes i dislikes es guarda en la taula de Projectes en si, com a incremental.

3. Desenvolupament Front-End

Un cop es tenia clar com havia de ser la UI/UX de l'aplicació, es va començar a desenvolupar el codi de la part del client amb el framework React Native + Expo. El primer component que es va crear va ser la barra de navegació, ja que definia les rutes que es podien accedir de l'app, i es va decidir quines eren les rutes privades i públiques. Per a restringir l'accés a les pantalles privades de l'aplicació, es feia ús del Context API de react, que permet guardar un context global d'aplicació, de manera que podíem guardar l'usuari autenticat, si és que aquest existia. Un havíem planificat on guardar la informació de l'usuari, es va començar amb el desenvolupament de les pantalles 'exteriors' o que no necessitaven autorització, és a dir, que eren accessibles a tothom sense necessitat d'autenticar-se. Aquestes pantalles eren Log In i Sign Up, el funcionament d'aquestes dues pantalles era semblant, l'usuari introdueix les seves credencials, ja registrades o noves, i es fa una petició al servidor perquè validi les dades d'entrada, en cas d'èxit, es retorna un JWT (json web token) que es guarda durant la sessió, amb la finalitat d'autoritzar posteriorment a l'usuari a fer peticions al servidor, de manera que el podrem identificar. El JWT accepta tant un *payload* com una clau secreta (aquesta ha de ser sempre privada), gràcies al payload, que en el nostre cas és l'ID de la base de dades de l'usuari concret, ens permet en rebre una petició, amb el Bearer Token com a header, identificar quin usuari fa quina petició, i determinar si té els permisos suficients per rebre el que està demanant. En les dues pantalles, al rebre aquest token es guarda a la 'async storage', semblant al comportament que té l'API de local Storage al navegador.

Un cop ja teníem la manera d'autenticar usuaris, es va seguir amb el desenvolupament de les pantalles que aportaven més valor a l'aplicació, com per exemple la pantalla Home i la de detalls de cada projecte. La pantalla Home tenia un petit inconvenient pel que fa a l'experiència d'usuari, i és que al buscar per títol, s'havia d'esperar a que l'usuari introduís tot el títol per a veure si hi havia algun projecte amb el títol semblant. Això passava ja que cada cop que es feia un canvi en l'input d'entrada, es cridava a una funció que demanava al servidor els projectes amb aquell títol, i per tant fins que no fos exacte no retornava res. Després d'investigar i fer proves, s'ha arribat a la conclusió que una bona solució a aquest problema fer l'ús d'un 'debounce', la lògica és la següent, en comptes d'enviar una petició al servidor cada cop que hi ha un canvi en l'input, es genera un retard en la petició cada cop que hi ha un canvi, així no es fa la petició fins que l'usuari ha acabat descrivint el títol, oferint una experiència molt millor, i amb menys peticions al servidor, el que podria millorar la latència i evitar sobrecàrregues.

4. Desenvolupament Back-End

Per últim, es va programar el servidor en Node JS fent servir Express JS com a framework web i MongoDB com a base de dades NoSql i Mongoose com a connector per la base de dades. Aquest és un stack tecnològic popular per a crear aplicacions MERN (Mongo Express React Node). Node és un entorn de treball que funciona de manera diferent a altres llenguatges o frameworks, ja que és un únic *thread* que funciona de forma asíncrona, i això és important, ja que si bloquejem el fil amb tasques síncrones, perdríem totes les avantatges que ens ofereix.

Per a tenir una bona separació de codi, on cada part s'ocupava d'una responsabilitat, vaig separar-ho de la següent manera:

1. Enrutadors, aquests definien els endpoints de

l'API, importaven les funcions declarades en els controladors, i les assignaven a una acció en concret entre GET/POST/PATCH/DELETE. En total tenim tres routers:
   - Projectes
   - Usuaris
   - Reviews
2. Controladors, per a cada entitat de l'aplicació teníem tant un enrutador com un controlador, aquest s'encarregava d'implementar les funcions que s'utilitzarien per gestionar les peticions dels clients, i retornaria tant els resultats, com un error en cas que la petició no tingui èxit, aquest podria ser un error de servidor, per una mala petició, falta de permisos, etc.
3. Models, per a modelar els esquemes de la base de dades es va fer servir Mongoose, permet definir com ha de ser la nostra base de dades, quines dades ha de tenir, permet validar dades d'entrada i també realitzar accions concretes en la inserció, actualització o eliminació de files de la base de dades. Una de les avantatges d'utilitzar un ODM (Object data modelling) és que fa una abstracció de la implementació de les taules i les seves relacions a la base de dades, el que ens permet aplicar més directe el disseny i les relacions esperades.

A manera d'exemple, a continuació podem veure el model utilitzar per a l'entitat Usuari:

```
const userSchema = mongoose.Schema({
    nombre:{
        type:String,
        required:[true,'El campo nombre es obligatorio'],

    },
    email:{
        type:String,
        required:[true,'El campo email es obligatorio'],
        validator:[validate.isEmail,'Correo no valido'],
        unique:true,
        lowercase:true,

    },
    role:{…
    },
    photo:{…
    },
    password:{…
    },
    passwordConfirm:{…
    },
    isActive:{…
    },
    passwordChangedAt:Date,
    proyectosFinanciados:[{…
    }],
    proyectosLikeados:[{…
    }],
    proyectosDislikeados:[{…
    }],
    passwordResetsToken:String,
    passwordResetsExpires:Date,

})
```

Font: pròpia. Model entitat Usuari.

Com podem veure, es poden materialitzar les relacions que necessita la taula, i generar la validació necessària utilitzant JavaScript.

## 7. CONCLUSIÓ

El treball final de grau ha representat un punt culminant en la meva carrera en desenvolupament, obrint-me els ulls i introduint-me en nous àmbits. Considero que hem aconseguit l'objectiu principal del projecte, que era democratitzar els recursos per fer realitat les millors idees dins de l'àmbit universitari. Durant aquest procés, he tingut l'oportunitat d'interactuar amb una gran varietat de persones dins de la comunitat de la Universitat Autònoma de Barcelona amb una gran motivació per connectar i crear productes o serveis innovadors.

Personalment, aquest projecte també m'ha permès aprofundir en la investigació sobre el desenvolupament mòbil, explorant les diverses alternatives disponibles avui en dia, així com els seus avantatges i inconvenients. En tot moment, he considerat que l'impacte social i educatiu ha estat el punt més important d'aquest TFG, més enllà de les tecnologies utilitzades.

Pel que fa al futur de l'aplicació, espero poder seguir millorant-la tant a nivell visual com funcional. Una de les meves metes és integrar una passarel·la de pagament funcional i segura en una futura versió de l'aplicació. Després d'investigar solucions com Strapi, crec que s'ajustaria al meu cas d'ús, però per manca de temps i experiència, no he pogut implementar-la en aquesta versió de l'aplicació.

En resum, aquest TFG ha estat una experiència enriquidora que m'ha proporcionat una comprensió més profunda del desenvolupament mòbil i m'ha impulsat a cercar solucions innovadores per a les necessitats de la comunitat universitària. Estic emocionat per continuar treballant en aquest projecte i veure com pot evolucionar en el futur per fer un impacte positiu en la comunitat.

## AGRAÏMENTS

M'agradaria agrair al professor Remo Suppi per tot el feedback, consells i reunions que hem tingut i per la disponibilitat que ha mostrat sempre que he tingut qualsevol dubte. També a la família i amics pel suport moral que m'han proporcionat aquests mesos.

## BIBLIOGRAFIA

[1] Aplicaciones web progresivas | MDN Web Docs. https://developer.mozilla.org/es/docs/Web/Progressive_web_apps
[2] Express - Node.js web application framework. https://expressjs.com/
[3] García, Marta. "¿Qué son las aplicaciones nativas, web e híbridas?". ES Design. 06/10/23. https://www.esdesignbarcelona.com/actualidad/diseno-web/apps-nativas-vs-hibridas
[4] Getting started | Axios Docs. https://axios-http.com/docs/intro
[5] Getting started | React Navigation. https://reactnavigation.org/docs/getting-started/
[6] Goteo.org https://ca.goteo.org/
[7] React Native · Learn once, write anywhere. https://reactnative.dev/
[8] Index | Node.js v22.3.0 Documentation. https://nodejs.org/docs/latest/api/

[9] Kickstarter "Haz realidad un proyecto creativo".
https://www.kickstarter.com/?lang=es

[10] MongoDB documentation. MongoDB Documentation.
https://www.mongodb.com/docs/
[11] Mongoose v8.4.3: Getting started.
https://mongoosejs.com/docs/
[12] S Kinski. "Ejemplos de Kickstarter: 7 campañas de éxito para financiar productos y marcas. Social Media Agency." 28/09/23.
https://socialmediaone.es/ejemplos-de-kickstarter-7-campanas-de-exito-para-financiar-productos-y-marcas/

## APÈNDIX

**CODI DEL BACK-END: HTTPS://GITHUB.COM/J-HU960/TFG-BACKEND**

**CODI DEL FRONT-END: HTTPS://GITHUB.COM/J-HU960/TFGFRONTEND**