UNIVERSITAT DE BARCELONA

Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Desenvolupament d'una Web per a una Marca de Roba

Marc Colominas

Director: Jordi José Bazán
Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, 17 de Gener de 2024

Resum

Aquest treball de Fi de Grau s'ha centrat en el desenvolupament de la primera versió d'una plataforma web de comerç electrònic, dissenyada per una marca de roba esportiva.

En el disseny de la interfície d'usuari (frontend), s'ha posat especial atenció en crear una estructura clara que permeti als usuaris explorar fàcilment els diferents productes oferts, així com accedir als detalls d'aquests.

Pel que fa a la gestió de la base de dades i l'API, coneguda com a backend, s'ha desenvolupat per proporcionar al frontend de tots els endpoints necessaris per poder garantir la funcionalitat de la web i gestionar l'emmagatzemen de dades correctament.

Una de les principals característiques de la plataforma és la capacitat de gestionar diferents tipus d'usuaris. A més dels usuaris estàndard, que poden registrar-se, navegar per la web i realitzar comandes, existeix un rol d'usuari administrador. Aquest té la capacitat de modificar les dades dels productes i gestionar els rols de la resta d'usuaris.

S'han utilitzat diverses tecnologies per tecnologies per poder realitzar el projecte, com són: Next.js per generar la web, React per la construcció de l'interfície d'usuari, Flask coma framework pel backend i Firebase per gestionar la base de dades i l'autenticació d'usuaris.

Finalment, s'han plantejat els passos a seguir en cas de voler continuar amb el projecte i fer-lo realitat.

Resumen

Este trabajo de Fin de Grado se ha centrado en el desarrollo de la primera versión de una plataforma web de comercio electrónico, diseñada para una marca de ropa deportiva.

En el diseño de la interfaz de usuario (frontend), se ha puesto especial atención en crear una estructura clara que permita a los usuarios explorar fácilmente los diferentes productos ofrecidos, así como acceder a los detalles de estos.
En cuanto a la gestión de la base de datos y la API, conocida como backend, se ha desarrollado para proporcionar al frontend todos los endpoints necesarios para poder garantizar la funcionalidad de la web y gestionar el almacenamiento de datos correctamente.

Una de las principales características de la plataforma es la capacidad de gestionar diferentes tipos de usuarios. Además de los usuarios estándar, que pueden registrarse, navegar por la web y realizar pedidos, existe un rol de usuario administrador. Este tiene la capacidad de modificar los datos de los productos y gestionar los roles del resto de usuarios.

Se han utilizado diversas tecnologías para poder realizar el proyecto, como son: Next.js para generar la web, React para la construcción de la interfaz de usuario, Flask como framework para el backend y Firebase para gestionar la base de datos y la autenticación de usuarios.

Finalmente, se han planteado los pasos a seguir en caso de querer continuar con el proyecto y hacerlo realidad.

Summary

This Project has focused on developing the first version of an e-commerce web platform, designed for a sportswear brand.

Special attention has been given to the design of the user interface, also known as the frontend, creating a clear structure that allows users to easily explore the various products offered, as well as access their details.

Regarding the management of the database and the API, known as the backend, it has been developed to provide the frontend with all the necessary endpoints to ensure the functionality of the website and manage data storage correctly.

One of the main features of the platform is the ability to manage different types of users. In addition to standard users, who can register, browse the website, and place orders, there is an administrator user role. This role has the ability to modify product data and manage the roles of other users.

Various technologies have been used to carry out the project, such as: Next.js for generating the web, React for building the user interface, Flask as a framework for the backend, and Firebase for managing the database and user authentication.

Finally, the steps to follow have been outlined in case there is a desire to continue with the project and make it a reality.

Índex

1. Introducció

Aquest és el treball de fi de grau pel Grau en Enginyeria Informàtica a la Universitat de Barcelona, en el que intento aplicar una part dels coneixements adquirits durant la carrera.

El treball es basa en la creació d'un *e-commerce* per a una empresa de roba anomenada OkonoFit. OkonoFit és una empresa emergent que ha creat una marca de roba esportiva, és una marca molt recent amb ganes de créixer i fer-se un nom en un sector ja establert.

Tot i que la intenció d'aquest treball no és finalitzar-lo amb la web 100% productiva i funcional, sí que n'és analitzar els passos a seguir per començar a treballar en ella i desenvolupar una primera versió de la web.

1.1 Motivació

Durant la de carrera he tocat diversos llenguatges i he conegut diferents formes de veure la programació, però mai vaig tenir clar cap a quin sector volia conduir la meva carrera professional un cop acabada l'acadèmica.

Això es va resoldre fa relativament poc, quan buscant pràctiques d'empresa on poder anar a treballar i ampliar coneixements a la vegada que començar a agafar experiència en el sector, vaig trobar una oferta com a frontend developer. Aquesta consistia en una formació inicial amb React i posteriorment la incorporació com a becari en un equip professional.
Els mesos de pràctiques em van fer trobar un camí pel qual vull seguir, i és per això que tinc la intenció de continuar formant-me com a desarrollador frontend a la vegada que vull anar agafant coneixements de backend de cara a un dia ser full-stack.

Aquesta és la raó per la qual vaig decidir enfocar el meu treball de fi de carrera en crear una web des de zero, on el repte fos crear tant el frontend com el backend de l'aplicació.

Un cop vaig tenir clar cap a on volia enfocar el treball tecnològicament, necessitava trobar un tema sobre el qual crear la web. Aquí vaig pensar en OkonoFit, una marca de roba propera que acaba de començar i que avui en dia només ven per contacte directe o a través d'Instagram.

1.2 Necessitat i utilitat

Tal com he comentat, OkonoFit és una empresa recentment creada que avui en dia simplement ven els seus productes a gent coneguda a través del boca a boca o mitjançant el seu perfil d'Instagram.
El següent punt per una marca com Okono és ampliar els mètodes de venda, i entre les possibles opcions es troben: obrir una tenda física, trobar un distribuïdor per vendre els productes a altres tendes o obrir un portal de venda online.

Dels 3 punts mencionats, l'empresa descarta el primer a causa de la inversió que s'ha de fer, que actualment, troben inviable. Per tant, toca centrar-se en els punts restants, que poden ser perfectament compatibles.

Així que mentre l'empresa busca un distribuïdor per ampliar les seves vendes i exposició al públic, decideixen contemplar l'opció d'obrir una tenda online.

1.3 Objectius

Fins ara hem vist les raons per les quals faig aquest projecte, ara toca centrar-nos en quins són els objectius establerts a l'inici d'aquest.

L'objectiu principal d'aquest treball es centra a desenvolupar una plataforma web de comerç electrònic que faciliti les vendes de productes a una tenda de roba. Aquesta tenda ha de permetre a un usuari realitzar tot el procés de compra, on inicialment podrà veure un producte que li agradi, a crear una sessió i finalment simular la compra. Els usuaris podran accedir al catàleg de productes i a la informació detallada de cada un d'ells. I, si escau, fer la compra un cop l'usuari hagi estat registrat i hagi iniciat sessió.
També hi haurà un accés per l'administrador de la web on podrà modificar productes, comandes, etc. a la vegada que donar permisos d'administrador a un altre usuari.

A continuació veurem un llistat dels principals objectius del treball:

- Desenvolupar un backend de qualitat, que contingui tota la funcionalitat necessària per nodrir la pàgina web.
- Desenvolupar un frontend de qualitat, que contingui un bon disseny d'interfícies, de fàcil navegació i amb una correcta gestió d'errors.
- Disseny d'una base de dades on guardar totes les dades del projecte.
- Integrar correctament el backend amb el frontend.

Objectius tècnics

- Utilitzar les millors pràctiques de React per desenvolupar una pàgina web.
- Utilitzar les millors pràctiques de Flask per desenvolupar el backend.

Objectius personals

- Continuar formant-me i coneixent a fons el framework React.
- Aprendre a desenvolupar des de zero una pàgina web de qualitat.
- Aprendre el framework Flask.

1.4 Estructura del document

La memòria presentada consta d'11 apartats diferents, a continuació els nombraré i explicaré resumidament en què es basa cada un d'ells:

1. **Introducció:** Es comenta la motivació del projecte, la seva utilitat i s'estableixen els objectius.
2. **Comerç electrònic:** Es fa una introducció al comerç electrònic, veient els diferents tipus i finalment els seus avantatges i desavantatges.
3. **Anàlisi tecnològic:** Es realitza un petit estudi de les diferents possibles tecnologies a utilitzar durant el projecte i s'explica quines han estat escollides.
4. **Planificació del projecte:** En aquesta secció s'explica amb detall la metodologia inicialment plantejada i com ha anat finalment.
5. **Anàlisis sistema:** Es fa un anàlisi del sistema a desenvolupar, definint els casos d'ús i els requisits.
6. **Backend:** S'explica com s'ha desenvolupat el backend per poder portar a terme els casos d'ús plantejats.
7. **Frontend:** S'expliquen les diferents vistes creades per poder portar a terme els casos d'ús plantejats.
8. **Publicació de la web:** Es comenten els diferents possibles hostings a utilitzar i les eleccions finals.
9. **Proves:** S'especifiquen les proves realitzades per revisar el funcionament de la web.
10. **Plans futurs:** Es plantegen els nous casos d'ús i possibles millores a la web en cas de voler continuar amb el projecte.
11. **Conclusions:** Es comenten les conclusions obtingudes al llarg del projecte i es valora si s'han complert els objectius establerts inicialment.
12. **Bibliografia:** Es recol·lecten les diferents referències que s'han utilitzat per desenvolupar el projecte.

2. Comerç electrònic

El comerç electrònic es basa en la compra i venda de productes o serveis a través d'internet, i la transferència de diners i dades per executar aquestes transaccions.
Es tracta principalment de la venda de productes físics en línia, tot i que també podem utilitzar el concepte per descriure qualsevol altre tipus de transacció comercial que es realitzi a través d'internet.

A continuació veurem quins tipus de comerç electrònic trobem, quin és el nostre cas i finalment els avantatges i desavantatges que té un comerç electrònic.

2.1 Tipus

Trobem diferents tipus de comerç electrònic que expliquen el tipus de transaccions que es poden realitzar entre consumidors i empreses.

- B2B (Business to Business): Fa referència a empreses o negocis que venen els seus productes o serveis a altres companyies.
- B2C (Business to Customer): És el perfil més comú, fa referència a una empresa que els seus clients són consumidors individuals.
- C2B (Customer to Business): Fa referència a un consumidor que ven els seus propis productes o serveis a una empresa.
- C2C (Customer to Customer): Es dona quan un consumidor ven a un altre. Les empreses creen portals que conecten consumidors.
- B2G (Business to Government): Les empreses venen a governs o entitats governamentals.
- C2G (Customer to Government): Els consumidors venen a governs o entitats governamentals.
- G2B (Government to Business): Els governs o entitats governamentals venen a empreses.
- G2C (Government to Customer): Els governs o entitats governamentals venen a consumidors.

La plataforma que s'ha creat en aquest treball es tracta d'un B2C, ja que parlem d'una empresa que ven a consumidors finals.

2.2 Avantatges

El comerç electrònic ha crescut de manera exponencial els últims anys, això es deu al gran nombre d'avantatges que té en comparació al comerç tradicional.

- Més exposició: Pel fet que el negoci no es troba en un punt físic sinó que és accessible des de qualsevol navegador web, un usuari pot a accedir-hi sense importar la seva localització.
- 24 h: Com que no és una tenda física on es necessiten treballadors presencialment perquè estigui en funcionament, és accessible per l'usuari les 24 hores del dia.
- Costos reduïts: No és necessari disposar d'un local, amb l'estalvi econòmic que suposa en despeses fixes com lloguer, llum, aigua, etc. A més, hi ha un estalvi amb relació als costs de comercialització, atenció al client o gestió d'inventari, ja que tots aquests es processen d'una manera més ràpida i eficient.
- Millor coneixement dels clients: Quan un client vol comprar online, s'ha de registrar, això implica que tinguem més informació dels clients en comparació al procés de compra a una tenda física. Aquesta informació pot servir a la companyia a ajustar-se a les seves necessitats o millorar l'estratègia de marketing.
- Facilitat per mostrar més productes: En una tenda online no tenim limitacions d'espai, per tant, és més fàcil mostrar un gran catàleg de productes i molta més informació sobre ells.

2.3 Desavantantges

Tot i que és cert que el comerç electrònic té grans avantatges com acabem de veure, també cal comentar que té certs desavantatges:

- Connexió a internet: Tenir connexió a internet i el correcte funcionament d'aquest és un requisit indispensable per utilitzar-lo. Qualsevol error a la xarxa farà inaccessible la plataforma.
- Falta de confiança: Tot i estar molt present en la societat, el comerç electrònic encara és una novetat per a molta gent, això pot implicar una falta de confiança per part dels clients a l'hora d'introduir les seves dades bancàries.
- Productes intangibles: En fer una compra online, el client no pot provar els productes fins que no els rep. Sent aquesta l'única manera de saber si un producte és el que esperava.

3. Anàlisi tecnològic

Al llarg d'aquest apartat es realitzarà un petit estudi de les diferents tecnologies que trobem avui en dia al mercat per desenvolupar un projecte d'aquestes característiques i veurem una explicació de les tecnologies que finalment s'han utilitzat per duu a terme el projecte.

3.1 Mètodes per crear una web

En l'actualitat tenim diferents mètodes per poder llençar una tenda online:

Prestashop

Prestashop és una plataforma de comerç electrònic de codi obert i gratuïta. Permet una personalització avançada i és ideal per aquells que tenen coneixements tècnics en desenvolupament web.

Shopify

Shopify és una solució a comerç electrònic molt popular. És coneguda per la seva facilitat d'ús i ràpida configuració. Ofereix hosting, una varietat de plantilles i un gran sistema de gestió. Ideal per a principiants.

Wordpress

Wordpress és una solució potent per crear tendes online. Ofereix flexibilitat i una gran varietat de temes i plugins per personalitzar la web.

Magento

És una plataforma de comerç electrònic de codi obert coneguda per la seva robustesa i escalabilitat. Ideal per a tendes onlines grans que requereixen personalització avançada. Pot arribar a ser complexa.

BigCommerce

És un altra plataforma tot en un, ofereix una varietat de plantilles i eines de marketing integrades.

Programació personalitzada

Crea una web des de zero que ofereix el nivell màxim de personalització. Implica tenir coneixements avançats en programació web.

En el cas específic del nostre projecte farem la web amb programació personalitzada, ja que ens ofereix la llibertat total de dissenyar i construir cada aspecte de la tenda com desitgem, amb les característiques úniques que volem.
A més, construïm una plataforma que s'adapta a les nostres necessitats actuals, però que serà fàcilment escalable de cara el futur sense les restriccions que imposen la resta de plataformes preexistents. Deixant de banda que en tenir la propietat total del codi elimina dependències de tercers i dona control sobre futures actualitzacions i manteniment.

3.2 Posibles tecnologies

3.2.1 Frontend

En aquest apartat del treball farem un petit anàlisi d'algunes de les tecnologies més populars avui en dia de cara a construir el frontend d'una tenda online.

El frontend és la part d'una aplicació o lloc web que veuen els usuaris i amb la que interactuen directament. Inclou tot el relacionat amb l'interfície d'usuari: disseny, gràfics i com es presenta la informació. El seu objectiu és proporcionar una experiència d'usuari agradable, eficient i accessible.

Vue

És un framework de Javascript per crear interfícies d'usuari. Es basa en HTML, CSS i JavaScript estàndard i proporciona un model de programació declaratiu i basat en components que permet desenvolupar de manera eficient interfícies d'usuari.

Avantatges:

- Corba d'aprenentatge fàcil per principiants
- Documentació clara i detallada
- Rendiment eficient
- Sistema de reactivitat intuïtiu

Desavantatges:

- Comunitat petita
- Menys recursos i llibreries de tercers

Angular

Angular és un framework de codi obert desenvolupat per Google que facilita la creació y programació d'aplicacions web d'una sola pàgina.

Avantatges:

- Framework robust i complet per desenvolupar aplicacions complexes
- Suporta Typescript de forma nativa
- Inclou diferents funcionalitats com routing o formularis incorporat
- Ideal per grans equips i aplicacions empresarials

Desavantatges:

- Corba d'aprenentatge molt gran
- Rendiment inferior comparat amb React i Vue

React

És una llibreria de Javascript de codi obert dissenyada per crear interfícies d'usuari amb l'objectiu de facilitar el desenvolupament d'aplicacions en una sola pàgina. És mantinguda per Facebook i la comunitat de software lliure.

Avantatges:

- Molt utilitzat i suportat
- Components reutilitzables que faciliten el desenvolupament i manteniment
- Molta diversitat de llibreries i eines disponibles

Desavantatges:

- Corba d'aprenentatge alta per a principiants
- Requereix la gestió de l'estat, que pot arribar a ser complexa
- Pot existir excés d'enginyeria en projectes petits

3.2.2 Backend

De la mateixa manera que hem fet amb el frontend, ara analitzarem les tecnologies més utilitzades per construir el backend de la nostra web.

El backend és la part de l'aplicació o lloc web que els usuaris no veuen. És responsable de la gestió de la lògica de l'aplicació, el processament de dades i el rendiment de les operacions de fons. El backend s'encarrega de rebre les sol·licituds del frontend, processar-les i enviar els resultats de tornada al frontend perquè siguin presentats a l'usuari.

Express.js (Node.js)

Express és el framework de backend més popular para Node.js. És minimalista, ràpid i proporciona característiques i eines robustes per desenvolupar aplicacions escalables.

Avantatges:

- Permet utilitzar Javascript tant al Front com al Back, cosa que simplifica el desenvolupament
- Gran quantitat de mòduls i eines disponibles a través de npm
- Asíncron i basat en events, cosa que pot portar a un rendiment eficient.
- Gran comunitat, molt activa

Desavantatges:

- Pot no ser eficient per operacions de CPU intenses
- Maneig d'errors i callbacks poden ser complexos per a principiants.
- Requereix un disseny cuidados per evitar problemes de rendiment i escalabilitat

Django (Python)

Django és un framework web gratuït i de codi obert basat en Python que s'executa en un servidor web. Segueix el patró arquitectònic model-plantilla-vistes.

Avantatges:

- Alt nivell i framework que segueix el patró de disseny MVC
- Inclou característiques de seguretat robusta de forma predeterminada
- Àmpliament utilitzat per aplicacions web, des de petites fins a grans.

Desavantatges:

- Poca flexibilitat en comparació amb altres frameworks
- L'estructura del projecte i configuració pot ser molt gran per a principiants

Spring boot (Java)

És un framework de Java de codi obert basat en microserveis. És especialment útil per als enginyers de programari que desenvolupen aplicacions web i microserveis.

Avantatges:

- Popular en aplicacions empresarials
- Extensa documentació i suport de comunitat
- Facilita la creació d'aplicacions independents
- Ofereix suport per la injecció de dependències

Desavantatges:

- Pot ser excessiu per a projectes petits o simples
- Corba d'aprenentatge molt gran, especialment si no estàs familiaritzat amb Java
- La configuració i estructura del projecte pot arribar a ser complexe

Flask (Python)

Flask és un framework de Python, però considerat com un microframework, ja que és minimalista i simple, oferint l'essencial per crear aplicacions web sense imposar una estructura o dependències particulars.

Avantatges:

- Framework lleuger i minimalista, ofereix molta flexibilitat
- Ideal per projectes petits o mitjans, i per construir microserveis
- Fàcil d'aprendre, especialment si ja es coneix Python
- Ideal per a prototips ràpids i projectes que requereixen un enfocament més simple

Desavantatges:

- Requereix bona configuració manual i selecció d'eines
- Pot no ser adequat en aplicacions molt grans o complexes en comparació a altres frameworks més robustos com Django
- La gestió de dependències i l'estructura del projecte es pot complicar a mesura que el projecte avanci

3.2.3 Base de dades

Un cop vistes les posibles tecnologies a utilitzar per al frontend i el backend de l'aplicació, toca revisar les possibles bases de dades.

Una base de dades és un sistema que permet emmagatzemar, modificar i extreure informació de manera estructurada. És usada pel backend per guardar i recuperar dades. Les bases de dades són fonamentals per la majora d'aplicacions web, ja que permeten la persistència de dades com la informació de l'usuari, inventari de productes, etc.

MongoDB

És una base de datos NoSQL orientada a documents de codi obert i escrit en C++.

Avantatges:

- Excel·lent per manejar gran volum de dades no estructurades
- Integració senzilla en stacks de desenvolupament basat en JavaScript
- Dissenyada per ser escalable i manejar grans càrregues de treball distribuint les dades
- Permet modificacions ràpides de l'esquema de la base de dades, cosa útil en projectes que evolucionen ràpidament

Desavantatges:

- Les transaccions multidocument poden ser més complexes en comparació amb les bases de dades relacionals
- Requereix un disseny cuidados per mantenir la integritat de dades en aplicacions complexes
- Les consultes són menys intuïtives que les de bases de dades SQL

MySQL

És un sistema de bases de dades d'Oracle que s'utilitza per gestionar base de dades. Es basa en l'àlgebra relacional i es fa servir per a l'emmagatzemament de dades de diversos serveis web.

Avantatges:

- És una de les bases de dades relacionals més utilitzades
- Provada en tota mena d'aplicacions
- Compatible amb gran nombre de llenguatges de programació i plataforma
- Comunitat activa i gran quantitat de documentació

Desavantatges:

- Possibles problemes d'escalabilitat en aplicacions grans
- Algunes característiques avançades poden estar més desenvolupades en altres sistemes com PostgreSQL.

PostgreSQL

PostgreSQL és un programari lliure que implementa un sistema de gestió de bases de dades relacional.

Avantatges:

- Suporta tipus de dades i funcionalitats més avançades que a MySQL
- Gran èmfasi en la integritat de les dades i el compliment de les propietats ACID
- Bon maneig de dades geoespacials, JSON entre ells
- Ús i distribució gratuït

Desavantatges:

- Pot ser més difícil d'administrar a causa de les seves característiques avançades
- Amb càrregues de treball molt altes, pot ser menys eficient que altres bases de dades.

Firebase

És una plataforma de desenvolupament d'aplicacions mòbils i webs que ofereixen una varietat de serveis, incloent-hi una base de dades a temps real i una base de dades NoSQL.

Avantatges:

- Integra emmagatzemament al núvol, autenticació d'usuaris entre altres coses
- Permet la sincronització de dades en temps real entre tots els usuaris
- Fàcil integració per aplicacions mòbils i web
- Gestiona automàticament l'escalabilitat

Desavantatges:

- Flexibilitat limitada en comparació amb altres bases de dades
- Pot ser costos a gran escala
- Al ser una plataforma com a servei, tenim una dependència de Google

3.3 Tecnologies escollides

A continuació detallarem les tecnologies escollides per a desenvolupar el projecte.

3.3.1 Frontend

ReactJS

React és una biblioteca de Javascript de codi obert utilitzada per construir aplicacions web renderitzades al costat del client mitjançant el llenguatge JavaScript. Permet crear interficies d'usuari amb l'objectiu de facilitar el desenvolupament d'aplicacions d'una sola pàgina (SPA).

Una de les seves principals característiques és l'utilització de un DOM virtual on es realitzen tots els canvis de la interfície d'usuari, que posteriorment s'apliquen al DOM verdader, accelerant així l'execució de les aplicacions.

Una altra de les seves característiques són els components. Un component a React és una funció que rep props (propietats)i retorna elements que descriuen el que ha d'apareixer en pantalla. Una gran avantage d'utilitzar props és que amb ellas es pot transmitir informació d'un component a un altre, cosa que fa que en cas d'haver un

component pare que conte diferents components fills pugui pasar-lis informació a través de las props i al contrari.

React ha estat utilitzat per desenvolupar grans aplicacions com poden ser: Facebook, Instagram, WhatsApp, Netflix, etc.

NextJS

Next.js és un marc de treball de codi obert per a aplicacions web desenvolupat sobre Node.js que permet facilitar la creació d'aplicacions React amb renderització al servidor o generació de llocs web estàtics. És mantingut per Vercel i una comunitat activa de desenvolupadors.

Next.js està dissenyat per optimitzar l'experiència de desenvolupament i producció, proporcionant funcions com el routing basat en el sistema de fitxers, suport automàtic per a l'estilat amb CSS-in-JS i una configuració mínima per començar. També ofereix optimització de rendiment out-of-the-box, com la divisió automàtica de codi per carregar només el codi necessari per a cada pàgina.

Una de les característiques clau de Next.js és la seva capacitat de pre-renderitzar pàgines, millorant el SEO i la velocitat de càrrega, a més de proporcionar una millor experiència d'usuari.

Tailwind

Tailwind CSS és un framework de CSS que conté classes predefinides per a dissenyar la interfície d'usuari d'una aplicació web. A diferència d'altres frameworks com Bootstrap, que ofereix components amb estil predefinits, Tailwind permet als desenvolupadors dissenyar visualment els seus dissenys mitjançant una àmplia gamma de classes d'utilitat per a cada propietat de CSS, facilitant la creació d'interfícies personalitzades sense haver de sortir de l'HTML.

Tailwind promou el desenvolupament ràpid i eficient, ja que els desenvolupadors poden veure els canvis en l'estilat de manera immediata, cosa que ajuda a iterar dissenys de forma més ràpida. La personalització és un altre punt fort de Tailwind, pel fet que permet als desenvolupadors ajustar el sistema de disseny per adaptar-se als requisits específics del projecte a través d'un fitxer de configuració.

Flowbite

Flowbite és una biblioteca de components basada en Tailwind CSS. Ofereix components d'interfície d'usuari reutilitzables i fàcils de personalitzar, com ara botons, formularis i tarjetes, que s'adapten perfectament a projectes que utilitzen Tailwind. Això permet als desenvolupadors concentrar-se en la lògica de l'aplicació mentre mantenen un disseny coherent i estèticament agradable.

La combinació de Tailwind CSS amb Flowbite pot accelerar encara més el desenvolupament d'interfícies d'usuari, ja que els desenvolupadors tenen accés tant a la flexibilitat de les classes d'utilitat de Tailwind com als components preconstruïts de Flowbite.

3.3.2 Backend

Postman

Postman és una aplicació que ens permet realitzar proves API. Es un client HTTP que ens dóna la possibilitat de testejar 'HTTP requests' a través d'una interfície gràfica d'usuari amb la qual obtindrem diferents tipus de resposta que posteriorment hauran de ser validats.
Ens ajuda a testejar coleccions o catàlegs d'APIs tant per el frontend com per el backend.

Flask

Flask consisteix en un micro Framework centrat en el rendiment y de disseny minimalista.
Flask té una gran versatilitat que ofereix en llenguatge Python, que permet crear APIs y aplicacions web ràpidament en un reduït número de línies de codi.
Es defineix com un micro-framework ja que un cop instal·lat, ofereix tot el necessari per crear una web funcional, a més permet la instal·lació d'extensions, plugins o complements si fos necessari.
De les seves característiques cal destacar que implementa un Model-Vista-Controlador i consta d'un servidor de desenvolupament, un depurador i suport per proves unitàries.

3.3.3 Base de dades

Firebase

Fireabase és una plataforma per al desenvolupament d'aplicacions web i aplicacions mòbil de Google.

És una plataforma ubicada al núvol, integrada amb Google Cloud Platform que utilitza un gran conjunt d'eines per la creació i sincronització de projectes.

Firebase Auth

Firebase Auth es un servei que permet autenticar els usuaris utilitzant únicament codi al costat del client. Inclou la autenticació mitjançant diferents proveïdors com poden ser Facebook, Google, Github…
Així com el mètode classic d'inici de sessió mitjançant un correu electrònic i una contrasenya.

Firestore Database

Cloud firestore es un servei d'emmagatzematge de dades derivat a Google Cloud Platform adaptat a la plataforma de Firebase.
És una base de dades NoSQL.
S'organitza en forma de documents agrupats en coleccions, y en ells podem incluir tants camps com vulguem.

4. Planificació del projecte

4.1 Metodología inicial

En aquesta secció explicare amb detall la metodologia plantejada per executar durant el desenvolupament del projecte, aquesta inclou tant la planificació prèvia al projecte com el desenvolupament d'aquest mateix.

Agile

Per fer aquest treball, he decidit seguir una metodologia basada en Agile, personalitzada segons les meves necessitats i el fet de ser l'únic membre de l'equip en aquest projecte. Agile és un enfocament popular en el desenvolupament de programari que destaca per la seva flexibilitat, adaptabilitat i l'èmfasi en entregar valor durant tot el projecte. A continuació, us explicaré els passos que he fet i com he adaptat aquesta metodologia a la meva situació:

Elaboració d'històries d'usuari

Durant la fase inicial del projecte, he creat una llista d'històries d'usuari que descriuen els objectius i funcionalitats clau que la pàgina web ha de tenir. Aquestes històries d'usuari em donen una visió clara del que vull aconseguir amb aquest projecte i em proporcionen una base sòlida per planificar i prioritzar les tasques.

Implementació de Sprints

He decidit dividir el projecte en Sprints, que són períodes de temps concrets on es realitzen tasques específiques. En el meu cas, cada Sprint dura tres setmanes. Aquesta estructura m'ajudara a treballar de forma més organitzada i sistemàtica, facilitant el seguiment dels avanços.

Planificació de Sprints

Abans de cada Sprint, establir les tasques que vull realitzar durant aquest. Aquesta planificació inclou noves funcionalitats, millores i la revisió de possibles tasques pendents dels Sprints anteriors.

Revisió i adaptació

Quan acaba cada Sprint, reviso el que he aconseguit i analitzo si he complert els objectius que havia establert o si hi ha tasques que encara estan pendents. Així puc

ajustar la planificació del següent Sprint per tractar-les. Aquest enfocament iteratiu i flexible em permet adaptar-me a mesura que el projecte avança, assegurant que es compleixin les necessitats i objectius establerts en les històries d'usuari.

En adoptar aquesta metodologia, espero dur a terme el treball de manera eficient i efectiva, assegurant que la web desenvolupada compleixi amb les expectatives i funcionalitats establertes en un inici.

4.2 Planificació

Abans de començar a programar, és important fer una bona planificació inicial de les tasques a realitzar, i així tenir una idea general que seguir i anar complint amb els temps establerts.

A continuació podem veure les tasques generals a fer durant el projecte, juntament amb el seu ordre d'execució:

1. Desenvolupament de la proposta tecnica
    1.1. Definir projecte
    1.2. Definir la metodologia de treball
2. Interioritzar la idea del projecte
    2.1. Anàlisi sobre les possibles tecnologies a utilitzar
    2.2. Documentació sobre les tecnologies escollides
3. Planificació del projecte
    3.1. Definir tasques inicials
    3.2. Desenvolupar proposta técnica
4. Desenvolupament del projecte
    4.1. Desenvolupament backend
        4.1.1. Disseny backend i base de dades
        4.1.2. Connexió backend amb base de dades
        4.1.3. Desenvolupament tècnic del backend
        4.1.4. Proves del backend
    4.2. Desenvolupament frontend
        4.2.1. Disseny del frontend de la web
        4.2.2. Maquetació frontend de la web
    4.3. Integració de la web
        4.3.1. Desenvolupar la integració del backend amb el frontend
        4.3.2. Proves de la integració
5. Proves finals
6. Documentació i presentació del projecte

4.3 Metodologia final

Inicialment, la metodologia prevista per continuar durant el desenvolupament del projecte era la metodologia Agile. Aquesta decisió es va prendre amb la intenció de donar-li un enfocament professional al projecte, ja que Agile és una metodologia ben establerta i respectada a l'entorn professional, especialment en el desenvolupament de programari i projectes tecnològics.
.
En el meu cas, havia planificat adaptar aquesta metodologia per ajustar-la a un projecte individual, pensant que aquesta adaptació seria factible i em proporcionaria una bona organització del projecte.

Però a mesura que el projecte avançava, m'he trobat desafiaments que m'han portat a desviar-me de la metodologia Agile.
Un dels aspectes fonamentals d'Agile és la planificació i l'execució en sprints i aquesta estructura requereix una planificació detallada i una revisió constant dels objectius, i en un projecte individual, he trobat dificultats per adherir-me a aquesta estructura.
A més, la falta d'un equip de treball significa que no hi ha una diversitat d'opinions durant les fases de revisió i planificació, cosa que és crucial per a l'èxit de la metodologia.
Considero que Agile està dissenyada per ser utilitzada per equips amb rols ben definits i una estructura col·laborativa.

Així i tot, crec que amb una millor organització i plantejament, podria haver seguit aquesta metodologia perfectament i m'hagués sigut molt útil.

Finalment, en lloc de seguir estrictament Agile, el meu enfocament s'ha transformat en un de més adaptatiu, guiat per les necessitats i desenvolupaments diaris del projecte. Aquesta metodologia no estructurada, encara que menys convencional, m'ha permès ser més flexible i respondre de manera més eficaç als reptes imprevistos. En lloc de planificar esprints i revisions regulars, el meu treball s'ha centrat en objectius a curt termini, ajustant constantment el meu enfocament i direcció basant-me en els resultats immediats i l'aprenentatge continu.

Aquest enfocament ad-hoc, encara que no té l'estructura formal de les metodologies àgils, té els seus propis avantatges. M'ha permès adaptar ràpidament les meves estratègies i mètodes de treball als problemes trobats, també m'ha donat la llibertat d'explorar solucions i enfocaments diferents sense estar lligat a un pla predefinit. Tot i això, reconec que aquesta flexibilitat ha tingut el cost de no tenir una visió a llarg termini tan clara i estructurada com la que ofereix Agile.

En conclusió, encara que inicialment havia planejat seguir la metodologia Agile, la naturalesa del meu projecte i les circumstàncies personals em van portar a adoptar un enfocament més flexible i adaptatiu. Aquesta experiència ha estat valuosa, ja que m'ha ensenyat la importància de l'adaptabilitat i la capacitat de respondre als canvis.

5. Anàlisis sistema

L'objectiu d'aquest apartat d'anàlisis és, tal com indica el seu nom, analitzar el sistema que vull desenvolupar.

Definim l'abast de la plataforma, el seu marc regulador i requisits d'usuari, a més de la definició dels casos d'ús.

5.1 Abast del sistema

El sistema consisteix en la creació d'una plataforma web de comerç electrònic perquè els clients puguin comprar roba de manera online. L'usuari podrà accedir a la plataforma per crear-se un compte, realitzar una cerca dels productes disponibles al catàleg i efectuar una compra en cas de voler obtenir algun d'ells. La plataforma a més, ha de contemplar la gestió d'usuaris, productes i comandes pels usuaris amb permís d'administrador, aquests poden també donar permisos d'administrador a un altre usuari en cas de ser necessari.

5.2 Marc regulador

La realització d'aquest projecte ha de seguir una sèrie d'estàndard i normes. En aquest apartat s'especificaran quins estàndard i normes són els que s'han de tenir en compte a l'hora de realitzar el projecte per aconseguir un producte de qualitat.
Seguirem l'estàndard descrit a la Llei Orgànica de Protecció de dades (LOPD), ja que la plataforma fa una recopilació de dades personals dels usuaris.

Tot i que el tractament de dades és mínim i l'objectiu d'aquest treball no és la comercialització, hem de seguir les normes dictades per la llei. En el cas que en un futur es vulgui continuar el projecte i comercialitzar-lo, s'hauran de tenir en compte les legislacions descrites per la Unió Europea en matèria de protecció de dades.

Però com la finalitat d'aquest projecte no es troba la comercialització final de la plataforma, no és necessari afegir una política de privacitat, cosa que haurà de canviar en el cas de voler continuar amb el projecte un cop acabat el treball.

A pesar d'això, la web tindrà una sèrie de restriccions generals que s'hauran de complir. A continuació les definirem.

5.2.1 Restriccions de seguretat

- Les contrasenyes es guarden a la base de dades xifrades.
- Les dades personals d'un usuari només seran accessibles pel mateix usuari o pels usuaris administradors.
- En cap cas es mostrarà la contrasenya d'un usuari.
- L'única funcionalitat que podrà realitzar un usuari que no estigui loguejat és la navegació per la web.

5.2.2 Restriccions de compatibilitat

- La web ha de ser accessible des de qualsevol navegador.
- L'idioma de la web serà únicament Espanyol.

5.2.3 Restriccions de la base de dades

- La gestió de la base de dades es farà a través de Firebase.
- Les contrasenyes s'han de guardar xifrades a la base de dades.
- Únicament es crearan col·leccions especificades durant el disseny.

5.2.4 Restriccions de navegació

- Per accedir a la web serà necessari una connexió a internet estable.
- Les diferents interfícies d'usuari han de ser simples i intuïtives perquè el client obtingui una bona experiència d'ús.
- La web estarà disponible les 24 hores al dia

5.3 Identificació usuaris

En aquest apartat s'identificaran els usuaris que fan ús de la plataforma.

- **Visitant:** Un visitant és un usuari que navega per la web sense haver-se registrat i loguejat. Aquest no tindrà cap benefici excepte el de poder fer un ull a la web i observar el catàleg de productes.

- **Client:** Un client es tracta d'un usuari prèviament registrat i loguejat, aquest pot navegar per la web de la mateixa manera que un visitant, però a més, obtindrà els beneficis de compra.

- **Administrador:** Un usuari administrador és un usuari especial, un treballador de l'empresa, aquest obté permisos per fer unes funcionalitats per tractar el stock o modificar altres aspectes de la web.

5.4 Casos d'ús

En aquest apartat s'ha realitzat una explicació detallada de tots els casos d'ús que són necessaris per un correcte funcionament de la plataforma.

Per cada cas d'ús hi trobem:

- **Nom**: Nom amb el que identifiquem el cas d'ús.
- **Actor**: Indica el tipus d'usuari que utilitza la web.
- **Objectiu**: Finalitat que busca realitzar l'actor
- **Precondicions**: Condicions que s'han d'haver complert anteriorment per realitzar l'operació.
- **Postcondicions**:Indica l'estat en que es queda el sistema un cop realitzada l'operació.
- **Flux**: Descriu les fases que forman el cas d'ús
- **Cas d'ús alternatius**: Descriu situacions que es poden donar en cas de no realitzar correctament els passos indicats.

C-01 Registre

- **Nom**: Registre
- **Actor**:
- **Objectiu**: Registrar-se a l'aplicació
- **Precondicions**: -
- **Postcondicions**: L'usuari te un compte creat
- **Flux:**
    - L'usuari introdueix el nom, correu electrònic i contrasenya al formulari de registre.
    - Es redirigeix a l'usuari a la pàgina de login.
- **Casos Alternatius:**
    - L'usuari no introdueix les dades correctament al formulari.
        - L'usuari introdueix un correu electrònic ja registrat.
        - L'usuari no introdueix un format correcte de correu electrònic.
        - L'usuari no introdueix un format correcte de contrasenya.

C-02 Login

- **Nom:** Accés a la plataforma web
- **Actor**: Visitant

- **Objectiu**: Accedir a la web
- **Precondicions**: L'usuari ha d'haver-se registrat
- **Postcondicions**: L'usuari accedeix a la web
- **Flux:**
    - L'usuari introdueix el correu electrònic i contrasenya amb els que s'ha registrat.
    - El sistema comprova que les dades siguin correctes.
    - Es redirigeix a l'usuari a la pàgina principal.
- **Casos Alternatius:**
    - L'usuari no introdueix les seves dades correctament al formulari.
        - L'usuari introdueix un correu electrònic no registrat.
        - L'usuari no introdueix la contrasenya correcte.

C-03 Login com administrador

- **Nom:** Accés a la plataforma web amb els beneficis d'administrador
- **Actor**: Administrador
- **Objectiu**: Accedir les funcionalitats d'administrador
- **Precondicions**: L'usuari ha de tenir un compte amb permisos d'administrador
- **Postcondicions**: L'usuari accedeix a la web amb les funcionalitats d'administrador visibles
- **Flux:**
    - L'usuari introdueix el seu correu electrònic i contrasenya.
    - El sistema comprova que les dades siguin correctes.
    - Al llistat d'opcions de la barra de navegació apareix un nou camp Administrador.
- **Casos Alternatius:**
    - L'usuari no introdueix les seves dades correctament al formulari.
        - L'usuari introdueix un correu electrònic no registrat.
        - L'usuari no introdueix la contrasenya correcte.

C-04 Tancar sessió

- **Nom:** Tancar la sessió
- **Actor**: Client
- **Objectiu**: Tancar la sessió iniciada
- **Precondicions**: L'usuari ha d'estar loguejat
- **Postcondicions**: L'usuari surt de la web
- **Flux:**
    - L'usuari selecciona l'opció de tancar sessió de la barra de navegació.
    - Es redirigeix a l'usuari a la pàgina d'inici de sessió..
- **Casos Alternatius:**
    - L'usuari tanca la web sense haver tancat sessió.

C-05 Veure dades d'un usuari

- **Nom:** Veure dades d'un usuari
- **Actor**: Client
- **Objectiu**: Veure les seves dades asociades d'un usuari
- **Precondicions**: -
- **Postcondicions**: L'usuari ha pogut veure les seves dades
- **Flux:**
    - L'usuari selecciona l'opció perfil a la barra de navegació.
    - Es redirigeix a l'usuari a la vista de perfil.
    - L'usuari pot observar les seves dades.
- **Casos Alternatius:**
    - -

C-06 Actualitzar el compte

- **Nom:** Gestionar i actualitzar les dades del compte
- **Actor**: Client
- **Objectiu**: Modificar els camps del perfil de l'usuari
- **Precondicions**: L'usuari ha d'estar loguejat
- **Postcondicions**: L'usuari actualitza les dades del seu compte
- **Flux:**
    - L'usuari accedeix al seu perfil a través de la barra de navegació.
    - Es redirigeix l'usuari a la pàgina mostrant les seves dades.
    - L'usuari pot omplir o modificar els camps que vulgui.
    - S'actualitza les dades de l'usuari.
- **Casos Alternatius:**
    - L'usuari omple els camps pero no guarda els canvis.

C-07 Veure un producte

- **Nom:** Veure un producte
- **Actor**: Visitant
- **Objectiu**: Observar els detalls d'un producte
- **Precondicions**: -
- **Postcondicions**: L'usuari ha pogut observar els detalls d'un producte
- **Flux:**
    - L'usuari selecciona un producte del catàleg.
    - Es redirigeix a l'usuari a la vista del producte.
    - L'usuari pot observar els detalls del producte.
- **Casos Alternatius:**
    - -

C-08 Veure el catàleg

- **Nom:** Veure el catàleg
- **Actor**: Visitant
- **Objectiu**: Observar el catàleg sencer de la tenda
- **Precondicions**: -
- **Postcondicions**: L'usuari ha pogut veure tots els productes de la tenda
- **Flux:**
    - L'usuari navega a la vista del càtaleg.
    - Es redirigeix a l'usuari a la vista del catàleg.
    - L'usuari pot observar els productes.
- **Casos Alternatius:**
    - -

C-09 Gestió usuaris

- **Nom:** Modificar el rol d'un usuari
- **Actor**: Administrador
- **Objectiu**: Gestionar usuaris - modificar el rol
- **Precondicions**: L'usuari administrador ha d'estar loguejat
- **Postcondicions**: El rol d'un usuari s'ha modificat
- **Flux:**
    - L'usuari selecciona l'opció Administrador a la barra de navegació.
    - Es redirigeix a administrador a la vista Administrador.
    - Introdueix l'identificador de l'usuari a modificar.
    - Modifica el rol del usuari
- **Casos Alternatius:**
    - El sistema no ha pogut modificar el usuari, es mostra missatge d'error.

C-10 Gestió productes

- **Nom:** Modificar dades d'un producte
- **Actor**: Administrador
- **Objectiu**: Modificar les dades d'un producte
- **Precondicions**: L'usuari administrador ha d'estar loguejat
- **Postcondicions**: S'ha modificat les dades del producte
- **Flux:**
    - L'usuari selecciona l'opció Administrador a la barra de navegació.
    - Es redirigeix a administrador a la vista Administrador.
    - L'administrador busca el producte.
    - Modifica el stock del producte.
- **Casos Alternatius:**

    - El sistema no ha pogut modificar el producte, es mostra missatge d'error.

C-11 Gestió comandes

- **Nom:** Modificar estat d'una comanda
- **Actor**: Administrador
- **Objectiu**: Modificar l'estat d'una comanda
- **Precondicions**: L'usuari administrador ha d'estar loguejat
- **Postcondicions**: S'ha modificat l'estat d'una comanda
- **Flux:**
    - L'usuari selecciona l'opció Administrador a la barra de navegació.
    - Es redirigeix a administrador a la vista Administrador.
    - L'administrador busca una comanda.
    - Modifica el seu estat.
- **Casos Alternatius:**
    - El sistema no ha pogut modificar la comanda, es mostra missatge d'error.

C-12 Veure la cistella de compra

- **Nom:** Veure la cistella de compra
- **Actor**: Client
- **Objectiu**: Veure la cistella de compra
- **Precondicions**: L'usuari ha d'estar loguejat i ha d'haver afegit un producte a la cistella
- **Postcondicions**: S'ha visualitzat la cistella
- **Flux:**
    - L'usuari selecciona la icona de la cistella a la barra de navegació.
    - S'obre el panel lateral amb la cistella.
    - L'usuari pot visualitzar la cistella.
- **Casos Alternatius:**
    - El sistema no ha pogut modificar la comanda, es mostra missatge d'error.

C-13 Afegir un producte a la cistella de compra

- **Nom:** Afegir producte a la cistella
- **Actor**: Client
- **Objectiu**: Afegir un producte a la cistella de compra
- **Precondicions**: L'usuari ha d'estar loguejat
- **Postcondicions**: S'ha afegit un nou producte a la cistella
- **Flux:**

    - L’usuari selecciona un producte del catàleg.
    - Es redirigeix a l’usuari a la vista del producte.
    - L’usuari selecciona una talla disponible del producte.
    - L’usuari afegeix el producte a la cistella.
- **Casos Alternatius:**
    - El sistema no ha pogut modificar la comanda, es mostra missatge d’error.

C-14 Eliminar un producte de la cistella de compra

- **Nom:** Eliminar un producte de la cistella
- **Actor**: Client
- **Objectiu**: Eliminar un producte de la cistella de compra
- **Precondicions**: L’usuari ha d’estar loguejat i tenir productes a la cistella
- **Postcondicions**: S'ha eliminat un producte de la cistella
- **Flux:**
    - L’usuari obre la cistella de la compra.
    - L’usuari elimina el producte que desitja.
- **Casos Alternatius:**
    - El sistema no ha pogut eliminar el producte, es mostra missatge d’error.

C-15 Finalitzar una compra

- **Nom:** Finalitzar una compra
- **Actor**: Client
- **Objectiu**: Omplir les dades restants i efectuar la compra
- **Precondicions**: L’usuari ha d’estar loguejat i tenir productes a la cistella
- **Postcondicions**: S'ha efectuat una comanda
- **Flux:**
    - L’usuari obre la cistella de la compra.
    - L’usuari selecciona finalitzar la compra.
    - Es redirigeix a l’usuari a la vista de finalitzar comanda.
    - L’usuari omple el formulari de dades d’entrega.
    - L’usuari fa el pagament.
- **Casos Alternatius:**
    - L’usuari no introdueix direcció d’entrega o no omple tots els camps.
    - El pagament no ha sigut efectiu.

C-16 Revisar comandes

- **Nom:** Reviasr comandes
- **Actor**: Client

- **Objectiu**: Veure l'historial de comandes i el seus detalls
- **Precondicions**: L'usuari ha d'estar loguejat i haver efectuat una comanda
- **Postcondicions**: L'usuari ha pogut veure el seu historial de comandes
- **Flux:**
    - L'usuari selecciona Comandes a través de la barra de navegació.
    - Es redirigeix a l'usuari a la vista historial de comandes.
    - L'usuari pot observar les diferents comandes efectuades junt amb el seu estat.
- **Casos Alternatius:**
    - L'usuari no ha realitzat mai una comanda, es mostrarà un missatge informatiu.

C-17 Modificar contrasenya

- **Nom:** Modificar contrasenya
- **Actor**: Client
- **Objectiu**: Modificar la contrasenya per iniciar sessió
- **Precondicions**: L'usuari ha de tenir un usuari creat
- **Postcondicions**: L'usuari pot iniciar sessió amb la nova contrasenya
- **Flux:**
    - L'usuari selecciona l'opció *¿Has olvidado tu contraseña?* a la pantalla de login.
    - El sistema envia un correu a l'usuari amb les indicacions per modificar la contrasenya.
    - L'usuari modifica la contrasenya.
- **Casos Alternatius:**
    - -

C-18 Filtrar productes

- **Nom:** Filtrar productes
- **Actor**: Visitant
- **Objectiu**:Filtrar els productes del catàleg
- **Precondicions**: -
- **Postcondicions**: L'usuari pot veure els productes que compleixen els seus requisits
- **Flux:**
    - L'usuari navega a la vista del càtaleg.
    - Es redirigeix a l'usuari a la vista del catàleg.
    - L'usuari selecciona una opció del panel de filtres.
    - L'usuari pot observar els productes filtrats.
- **Casos Alternatius:**

-

Per acabar, adjunto una imatge d'un diagrama amb tots els casos dús.:

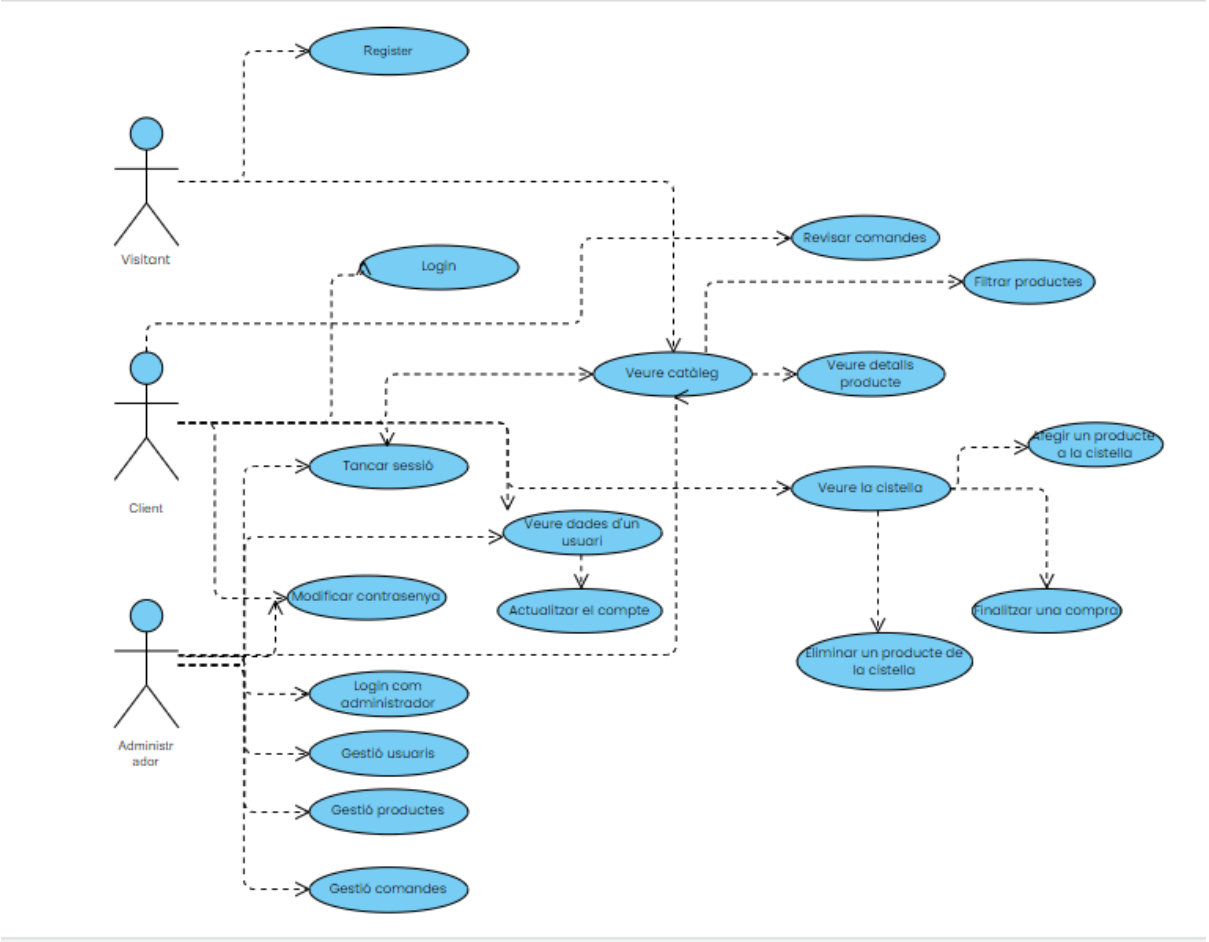


Diagrama de casos dús

5.5 Requisits de software

En aquest apartat es defineixen els requisits necessaris per poder portar a terme el desenvolupament del projecte.

5.5.1 Funcionals

Els requisits funcionals són els que indiquen quines funcionalitats ha de poder realitzar el sistema.

R-01

Nombre: Frontend i Backend
Descripció: La plataforma ha d'estar formada per una part client (frontend) i una part servidor (backend).

R-02

Nombre: Accés a la plataforma
Descripció: Algunes funcionalitats a la plataforma estaran limitades, per poder utilitzar-les s'haurà d'estar dins d'una sessió.

R-03

Nombre: Tipus d'usuari

Descripció: Podem classificar els usuaris segons clients o administradors en funció del seu rol.

R-04

Nombre: Administrador

Descripció: Només pot accedir al panel d'Administració un usuari Administrador.

R-05

Nombre: Client

Descripció: Un client té els permisos que li permet l'administrador.

R-06

Nombre: Registrat

Descripció: Un cop registrat, un usuari té un perfil creat al sistema.

R-07

Nombre: Actualitzar dades

Descripció: Un client podrà actualitzar les seves dades del perfil.

R-08

Nombre: Gestionar cistella

Descripció: Un client podrà gestionar la seva cistella de la compra afegint o eliminant productes.

R-09

Nombre: Pàgina principal client

Descripció: Un usuari que hagi iniciat sessió accedirà a la vista principal i podrà buscar productes, veure la cistella de la compra, el seu perfil i tancar sessió.

R-10

Nombre: Pàgina administrador

Descripció: Un administrador que hagi iniciat sessió podrà efectuar les mateixes funcions que un visitant a més d'obrir el panel d'administrador.

R-11

Nombre: Gestionar comanda

Descripció: Un administrador podrà gestionar una comanda actualitzant el seu estat.

R-12

Nombre: Gestionar permisos

Descripció: Un administrador podrà donar permisos a un altre usuari.

R-13

Nombre: Gestionar productes

Descripció: Un administrador podrà modificar les dades d'un producte com poden ser les talles disponibles.

R-14

Nombre: Tancar sessió

Descripció: Un usuari podrà tancar sessió per abandonar la plataforma.

R-15

Nombre: Pàgina producte

Descripció: A la pàgina individual d'un producte es podrà veure la informació detallada d'aquest.

R-16

Nombre: Afegir producte

Descripció: A la pàgina individual d'un producte es podrà afegir aquest a la cistella de la compra.

R-17

Nombre: Pàgina perfil

Descripció: A la pàgina de perfil es mostra la informació de l'usuari.

R-18

Nombre: Modificar perfil

Descripció: A la pàgina de perfil es mostra un formulari on es podrà modificar la informació d'aquest.

R-19

Nombre: Pàgina comandes

Descripció: A la pàgina de comandes es mostra un llistat amb l'historial de compres de l'usuari.

R-20

Nombre: Filtrar productes

Descripció: A la pàgina de catàleg de productes s'ofereix l'opció de filtrar-los segons la necessitat de l'usuari.

R-21

Nombre: Pàgina productes

Descripció: A la pàgina de productes es podrà veure tot el catàleg de la web.

R-22

Nombre: Revisar cistella

Descripció: En el panel de la cistella es podran veure els productes afegits.

R-23

Nombre: Pàgina checkout

Descripció: A la pàgina on finalitzar la comanda.

5.5.2 D'operacions

Els requisits d'operació són els que defineixen les eines que s'utilitzaran per a garantir la prestació del sistema.

R-24

Nombre: Tipus sol·licituds

Descripció: El sistema rebrà peticions de l'usuari mitjançant sol·licituds GET, POST, PUT y DELETE.

R-25

Nombre: Operacions base de dades

Descripció: Per qualsevol operació que realitzi un usuari, es guardaran les dades a la base de dades per poder mostrar-les més endavant.

R-26

Nombre: Gestió base de dades

Descripció: Tota actualització de dades es guardarà en una base de dades gestionada per Firebase.

5.5.3 D'interfície

Els requisits d'interfície són els que can relacionats a la usabilitat i el disseny de les interfícies que es mostraran als usuaris.

R-27

Nombre: Compatibilitat

Descripció: La web serà compatible únicament amb navegadors web.

R-28

Nombre: Navegació web

Descripció: La navegació de la web permetrà sempre tornar enrere o navegar a la pàgina principal.

5.5.4 De seguretat

Els requisits de seguretat defineixen com s'assegurarà el sistema en contra d'amenaces externes, en concret amb relació a la confidencialitat de les dades dels usuaris.

R-29

Nombre: Visualització contrasenyes

Descripció: Les contrasenyes no seran visibles en cap moment.

R-30

Nombre: Usuari ùnic

Descripció: No poden existir dos usuaris amb el mateix correu electrònic.

R-31

Nombre: Xifrat contrasenyes

Descripció: Les contrasenyes es guarden xifrades a la base de dades.

R-32

Nombre: Inici de sessió d’un usuari

Descripció: Els usuaris han d'introduir el seu correu electrònic i contrasenya per poder iniciar sessió. Aquestes dades es validen amb la base de dades per comprovar que són correctes.

6. Backend

En aquest apartat, detallem el procés de desenvolupament del backend del nostre projecte. Aquest desenvolupament s'ha centrat en les necessitats específiques dels casos d'ús establerts, assegurant que cada funcionalitat de l'aplicació sigui suportada adequadament per la nostra infraestructura de servidor.

Un element clau en aquest procés és la creació i gestió d'endpoints. Un endpoint és, bàsicament, un punt final d'una comunicació en una xarxa. En el context d'una aplicació web, un endpoint és una URL específica en el servidor web on l'aplicació client pot sol·licitar dades o enviar dades per ser processades. Cada endpoint està dissenyat per manejar diferents tipus de sol·licituds HTTP, que són els mètodes estandarditzats per comunicar-se amb un servidor web.

Dins del nostre projecte, hem implementat 4 tipus principals de sol·licituds HTTP, cadascuna amb un propòsit específic:

GET

El mètode GET s'utilitza per sol·licitar les dades d'un recurs específic en un servidor web. Es tracta d'una sol·licitud només de lectura, i serveix per recuperar la informació del servidor sense fer capa modificació del recurs sol·licitat.

POST

El mètode POST s'utilitza per enviar dades al servidor pel seu processament o emmagatzemen. S'acostuma a fer servir per enviar formularis o realitzar accions que modifiquen l'estat del servidor o crear nou recursos.

DELETE

El mètode DELETE s'utilitza per sol·licitar al servidor que s'elimini un recurs específic. En cas que el servidor admeti aquesta operació i l'usuari tingui permís, el servidor eliminarà el recurs sol·licitat.

PUT

El mètode PUT s'utilitza per actualitzar o reemplaçar un recurs existent al servidor. Requereix que el client proporcioni la nova representació del recurs, que s'utilitzarà per modificar el recurs existent.

C-01 Registre

Perquè un usuari pugui registrar-se a l'aplicació tenim l'endpoint POST user, aquest recull les dades de l'usuari que li passem i crea un nou usuari a la base de dades.

```python
def post(self):

    parser = reqparse.RequestParser()
    parser.add_argument(
        "Nombre", type=str, required=True, help="This field cannot be left blank"
    )
    parser.add_argument(
        "Apellidos", type=str, required=True, help="This field cannot be left blank"
    )
    parser.add_argument(
        "Correo", type=str, required=True, help="This field cannot be left blank"
    )
    parser.add_argument(
        "Contraseña",
        type=str,
        required=True,
        help="This field cannot be left blank",
    )
    parser.add_argument(
        "Rol", type=str, required=False
    )
    parser.add_argument("Pais", type=str, required=False)
    parser.add_argument("Calle", type=str, required=False)
    parser.add_argument("Piso", type=str, required=False)
    parser.add_argument("CodigoPostal", type=str, required=False)
    parser.add_argument("Ciudad", type=str, required=False)
    parser.add_argument("Provincia", type=str, required=False)



    data = parser.parse_args()
    if data["Nombre"] == "":
        return {"message": "name cannot be left blank"}, 400
    if data["Apellidos"] == "":
        return {"message": "surname cannot be left blank"}, 400
    if data["Correo"] == "":
        return {"message": "mail cannot be left blank"}, 400
    if data["Contraseña"] == "":
        return {"message": "password cannot be left blank"}, 400

    user = User(
        nombre=data["Nombre"],
        apellidos=data["Apellidos"],
        correo=data["Correo"],
        contraseña=data["Contraseña"],
        rol=data.get("Rol", ""),
        pais=data.get("Pais", ""),
        calle=data.get("Calle", ""),
        piso=data.get("Piso", ""),
        codigo_postal=data.get("CodigoPostal", ""),
        ciudad=data.get("Ciudad", ""),

    )

    try:
        user_record = auth.get_user_by_email(data["Correo"])
        if user_record:
            return {
                "message": "Ya existe un usuario con ese correo electrónico"
            }, 400
    except auth.UserNotFoundError:
        pass

    try:
        user.add_user()
        return {
            "message": "El usuario {} ha sido añadido".format(data["Nombre"])
        }, 200

    except Exception as e:
        return {"message": "Error al crear el usuario: " + str(e)}, 500
```

Sol·licitud POST user

Podem observar que el mètode recull molts paràmetres, però pel registre només fem servir les marcades com a *required* = *True*. La resta de paràmetres es guarden a la base de dades com a cadenes de caràcters buides, i els farem servir més endavant.

Un cop recollits els paràmetres es comprova que no existeix un usuari a la base de dades amb el mateix correu electrònic.
En aquest cas, es crea l'usuari mitjançant la funció add_user:

```
def add_user(self):
    auth.create_user(email=self.correo, password=self.contraseña, uid=self.correo)
    doc_ref = db.collection("users").document(self.correo)
    try:
        doc_ref.set(
            {
                "Nombre": self.nombre,
                "Apellidos": self.apellidos,
                "Correo": self.correo,
            }
        )
        return 0

    except:
        return -1
```

Funció add_user de la classe User

A la base de dades es crea la instància de l'usuari tant a Firebase Authentication, que guarda l'identificador de l'usuari (en el nostre cas el correu electrònic) i la contrasenya, sobre la qual no tenim visibilitat, com a la Firestore Database, on guardem la resta de dades de l'usuari.

C-02 Login i C-03 Login com administrador

Per poder iniciar sessió a l'aplicació, s'utilitza l'endpoint POST login:

```python
class Login(Resource):
    def post(self):
        parser = reqparse.RequestParser()
        parser.add_argument("username", type=str, required=True)
        parser.add_argument("password", type=str, required=True)
        data = parser.parse_args()

        try:
            token_response = sign_in_with_email_and_password(
                data["username"], data["password"]
            )
            if token_response.get("error"):
                error_message = token_response["error"]["message"]
                if "EMAIL_NOT_FOUND" in error_message:
                    return {"message": "Usuario no existe"}, 400
                elif "INVALID_PASSWORD" in error_message:
                    return {"message": "Contraseña incorrecta"}, 400
                else:
                    return {"message": "Error al iniciar sesión"}, 400
            else:
                return {"idToken": token_response["idToken"]}, 200
        except Exception as e:
            return {"message": str(e)}, 500
```

Sol·licitud POST login

Aquest rep un identificador d'usuari (el correu electrònic) i una contrasenya, que s'envien com a paràmetre a la funció de Firebase sign_in_with_email_and_password per comprovar que corresponen a un usuari registrat a la base de dades.

En cas que l'identificador i la contrasenya siguin correctes, es retorna un token d'inici de sessió, que ens servirà per identificar si un usuari segueix loguejat.

Si l'intent de login és incorrecte, mira d'identificar si es tracta d'un error conegut per retornar un missatge més específic.

El login per un administrador es fa de la mateixa manera que per un client.

C-05 Veure dades d'un usuari

Per obtenir totes les dades relacionades a un usuari de la base de dades, hem de cridar l'endpoint GET user.

```python
def get(self):
    token_header = request.headers.get("Authorization", None)
    if not token_header:
        return {"message": "No se proporcionó token"}, 401

    token = token_header.split(" ")[1]

    try:
        decoded_token = auth.verify_id_token(token)
    except exceptions.FirebaseError as e:
        return {"message": "Token inválido o expirado"}, 401

    correo = decoded_token.get("email")
    if not correo:
        return {"message": "No se pudo obtener el correo del token"}, 400

    us = db.collection("users").document(correo).get().to_dict()
    if not us:
        return {"message": "Usuario no encontrado"}, 404

    return us
```

Sol·licitud GET user

S'ha de proporcionar el token de sessió de l'usuari, ja que només un usuari pot rebre les seves dades guardades.
Amb el token s'aconsegueix el correu de l'usuari, que serveix com a identificador per obtenir les dades de l'usuari a la base de dades.

C-06 Actualitzar el compte

De cara a actualitzar les dades d'un usuari a la base de dades, es pot fer amb l'endpoint PUT user, que pertany a la mateixa classe que el endpoint POST user que hem vist al C-01.

```python
def put(self):
    parser = reqparse.RequestParser()
    parser.add_argument("Nombre", type=str, required=False)
    parser.add_argument("Apellidos", type=str, required=False)
    parser.add_argument("Telefono", type=str, required=False)
    parser.add_argument("Pais", type=str, required=False)
    parser.add_argument("Provincia", type=str, required=False)
    parser.add_argument("Calle", type=str, required=False)
    parser.add_argument("Ciudad", type=str, required=False)
    parser.add_argument("CodigoPostal", type=str, required=False)
    parser.add_argument("Piso", type=str, required=False)


    data = parser.parse_args()

    token_header = request.headers.get("Authorization", None)
    if not token_header:
        return {"message": "No se proporcionó token"}, 401

    token = token_header.split(" ")[1]  #

    try:
        decoded_token = auth.verify_id_token(token)
        correo_usuario = decoded_token.get("email")
        if not correo_usuario:
            return {
                "message": "No se pudo obtener el correo del usuario del token"
            }, 400
    except exceptions.FirebaseError as e:
        return {"message": "Token inválido o expirado"}, 401

    user_ref = db.collection("users").document(correo_usuario)
    user = user_ref.get()
    if not user.exists:
        return {"message": "Usuario no encontrado"}, 404

    update_data = {}
    if data.get("Nombre"):
        update_data["Nombre"] = data["Nombre"]
    if data.get("Apellidos"):
        update_data["Apellidos"] = data["Apellidos"]
    if data.get("Telefono"):
        update_data["Telefono"] = data["Telefono"]
    if data.get("Pais"):
        update_data["Pais"] = data["Pais"]
    if data.get("Provincia"):
        update_data["Provincia"] = data["Provincia"]
    if data.get("CodigoPostal"):
        update_data["CodigoPostal"] = data["CodigoPostal"]
    if data.get("Ciudad"):
        update_data["Ciudad"] = data["Ciudad"]
    if data.get("Calle"):
        update_data["Calle"] = data["Calle"]
    if data.get("Piso"):
        update_data["Piso"] = data["Piso"]

    user_ref.update(update_data)
    return {"message": "Datos del usuario actualizados"}, 200
```

Sol·licitud PUT user

El PUT user rep tant els paràmetres que es volen modificar, com el token de sessió de l'usuari que s'obté en completar el login (C-02).

El primer que fa és extreure el correu de l'usuari a partir del token, així es comprova que ningú excepte el mateix usuari pot modificar les seves dades.

Un cop tenim el correu de l'usuari, es revisa quins són els paràmetres que es volen modificar i se sobreescriuen a la base de dades.

C-07 Veure un producte i C-08 Veure el catàleg

Per poder veure el catàleg de productes o els detalls d'un dels seus productes, hem d'utilitzar el mateix endpoint, i aquest és el GET products.

```python
def get(self, id=None):
    if id:
        product = Product.find_by_id(id)
        if product:
            return product.json()
        return {"message": "Product not found"}, 404
    else:
        all_products_query = db.collection("products").stream()
        products_list = [doc.to_dict() for doc in all_products_query]
        return {"products": products_list} if products_list else {
            "message": "No products found"
        }, 404
```

Sol·licitud GET product

Per veure el catàleg sencer simplement s'ha de cridar l'endpoint, sense pasar cap parametre. I aquest retornara tots els productes del catàleg guardats a la base de dades amb els seus detalls.

Si es vol obtenir simplement els detalls d'un producte en concret, al cridar l'endpoint se li ha de pasar el id del producte.

```python
@classmethod
def find_by_id(cls, id):
    product = db.collection("products").document(id).get()
    if product.exists:
        return cls(**product.to_dict())
    return None
```

Funció find_by_id de la classe Product

C-09 Gestió usuaris, C-10 Gestió productes i C-11 Gestió comandes

Per gestionar usuaris i poder modificar el seu rol, hem d'utilitzar l'endpoint adminUser.

Aquest endpoint ha estat exclusivament creat perquè un administrador pugui veure les dades d'un usuari, i modificar-les.

```python
def get(self):
    parser = reqparse.RequestParser()
    parser.add_argument("CorreoUsuario", type=str, required=True,location='args', help="Email del usuario requerido")
    data = parser.parse_args()
    correo_consulta = data['CorreoUsuario']

    token_header = request.headers.get("Authorization", None)
    if not token_header:
        return {"message": "No se proporcionó token"}, 401

    token = token_header.split(" ")[1]

    try:
        decoded_token = auth.verify_id_token(token)
        correo_admin = decoded_token.get("email")
        if not correo_admin:
            return {"message": "No se pudo obtener el correo del token"}, 400

        user_ref = db.collection("users").document(correo_admin)
        user = user_ref.get().to_dict()
        if not user or user.get('Rol') != 'Admin':
            return {"message": "Acceso denegado. Se requiere rol de Administrador"}, 403

        user_data = db.collection("users").document(correo_consulta).get().to_dict()
        if not user_data:
            return {"message": "Usuario no encontrado"}, 404

        return user_data, 200

    except exceptions.FirebaseError as e:
        return {"message": "Token inválido o expirado"}, 401
```

Sol·licitud GET adminUser

A la sol·licitud GET, es comprova que l'usuari sigui administrador, això es fa obtenint l'identificador de l'usuari a partir del seu token, un cop tenim l'identificador es revisa que aquest tingui rol Administrador.

En cas de ser administrador es retornen les dades de l'usuari passat per paràmetre.

També tenim la sol·licitud PUT que permet a un administrador modificar les dades associades a un usuari.
De la mateixa forma que abans, es comprova que qui fa la sol·licitud és un administrador, en cas afirmatiu sobreescriu les dades de l'usuari a la base de dades amb les enviades per paràmetre.
En cas de ser administrador es retornen les dades de l'usuari passat per paràmetre.

```python
def put(self):
    parser = reqparse.RequestParser()
    parser.add_argument("CorreoUsuario", type=str, required=True, help="Email del usuario requerido")
    parser.add_argument("Nombre", type=str, required=False)
    parser.add_argument("Apellidos", type=str, required=False)
    parser.add_argument("Telefono", type=str, required=False)
    parser.add_argument("Pais", type=str, required=False)
    parser.add_argument("Provincia", type=str, required=False)
    parser.add_argument("Calle", type=str, required=False)
    parser.add_argument("Ciudad", type=str, required=False)
    parser.add_argument("CodigoPostal", type=str, required=False)
    parser.add_argument("Piso", type=str, required=False)

    data = parser.parse_args()
    correo_consulta = data['CorreoUsuario']

    token_header = request.headers.get("Authorization", None)
    if not token_header:
        return {"message": "No se proporcionó token"}, 401

    token = token_header.split(" ")[1]

    try:
        decoded_token = auth.verify_id_token(token)
        correo_admin = decoded_token.get("email")
        if not correo_admin:
            return {"message": "No se pudo obtener el correo del token"}, 400

        admin_ref = db.collection("users").document(correo_admin)
        admin = admin_ref.get().to_dict()
        if not admin or admin.get('Rol') != 'Admin':
            return {"message": "Acceso denegado. Se requiere rol de Administrador"}, 403

        user_ref = db.collection("users").document(correo_consulta)
        user = user_ref.get()
        if not user.exists:
            return {"message": "Usuario no encontrado"}, 404

        update_data = {k: v for k, v in data.items() if v is not None and k != 'CorreoUsuario'}
        user_ref.update(update_data)

        return {"message": "Datos del usuario actualizados correctamente"}, 200

    except exceptions.FirebaseError as e:
        return {"message": "Token inválido o expirado"}, 401
```

Sol·licitud POST adminUser

C-12 Veure la cistella de la compra

Per veure la cistella de la compra, el backend ens ha de retornar la llista d'items de la cistella, per això hem de fer un GET shoppingcart.

```python
def get(self):
    token_header = request.headers.get("Authorization", None)
    if not token_header:
        return {"message": "Token no proporcionado"}, 401

    token = token_header.split(" ")[1]

    try:
        decoded_token = auth.verify_id_token(token)
        user_id = decoded_token.get("email")
        if not user_id:
            return {"message": "No se pudo obtener el ID del usuario"}, 400
    except Exception as e:
        return {"message": f"Error al decodificar el token: {str(e)}"}, 401

    cart = Cart.find_by_id(user_id)
    if cart:
        return cart.json()
    return {"message": "Carrito no encontrado"}, 404
```

Sol·licitud GET shoppingcart

Se li ha de proporcionar el token d'usuari, amb el qual extrèiem el correu de l'usuari.

Un cop tenim el correu, podem buscar la cistella de la compra associada a l'usuari.

```python
@classmethod
def find_by_id(cls, user_id):
    doc = db.collection("user_cart").document(user_id).get()
    if doc.exists:
        return cls(**doc.to_dict())
    return None
```

Funció find_by_id de la classe Cart

En cas d'existir una cistella asociada a l'usuari, retorna una llista dels productes de la cistella.

C-13 Afegir un producte a la cistella de compra

Per poder afegir un producte a la cistella de compra d'un usuari, tenim l'endpoint POST shoppingcart.

```python
def post(self):
    token_header = request.headers.get("Authorization", None)
    if not token_header:
        return {"message": "Token no proporcionado"}, 401

    token = token_header.split(" ")[1]

    try:
        decoded_token = auth.verify_id_token(token)
        user_id = decoded_token.get("email")
        if not user_id:
            return {"message": "No se pudo obtener el ID del usuario"}, 400
    except Exception as e:
        return {"message": f"Error al decodificar el token: {str(e)}"}, 401

    data = ShoppingCart.parser.parse_args()
    new_items = data["items"]

    existing_cart = Cart.find_by_id(user_id)
    if existing_cart:
        for item in new_items:
            existing_cart.items.append(item)
        existing_cart.update_cart()
        return existing_cart.json(), 200
    else:
        cart = Cart(user_id=user_id, items=new_items)
        cart.save_to_db()
        return cart.json(), 201
```

Sol·licitud POST shoppingcart

De la mateixa manera que s'ha explicat anteriorment, s'extreu del token de sessió de l'usuari el seu correu, que equival al seu identificador.

Amb aquest identificador es comprova si l'usuari ja té una cistella de la compra creada a la base de dades, és a dir, si l'usuari ja ha afegit algun producte abans.

```python
@classmethod
def find_by_id(cls, user_id):
    doc = db.collection("user_cart").document(user_id).get()
    if doc.exists:
        return cls(**doc.to_dict())
    return None
```

Funció find_by_id de la classe Cart

Si es detecta que ja existeix una cistella de la compra relacionada amb l'usuari, s'afegeix a l'array de productes de la cistella el nou item.
En cas que no es detecti una cistella existent relacionada amb l'usuari, es crea a la base de dades la instància de la cistella amb el producte.

```python
def save_to_db(self):
    doc_ref = db.collection("user_cart").document(self.user_id)
    doc_ref.set(self.json())
```

Funció save_to_db de la classe Cart

C-14 Eliminar un producte de la cistella de la compra

Si el que es busca és eliminar un producte de la cistella de la compra, s'ha de cridar l'endpoint DELETE de la classe shoppingcart.

```python
def delete(self):
    token_header = request.headers.get("Authorization", None)
    if not token_header:
        return {"message": "Token no proporcionado"}, 401

    token = token_header.split(" ")[1]

    try:
        decoded_token = auth.verify_id_token(token)
        user_id = decoded_token.get("email")
        if not user_id:
            return {"message": "No se pudo obtener el ID del usuario"}, 400
    except Exception as e:
        return {"message": f"Error al decodificar el token: {str(e)}"}, 401

    product_id = request.args.get("product_id")

    existing_cart = Cart.find_by_id(user_id)
    if existing_cart:
        if product_id:
            existing_cart.items = [
                item
                for item in existing_cart.items
                if str(item["id"]) != product_id
            ]
        else:
            existing_cart.items = []

        existing_cart.update_cart()
        return {"message": "Producto(s) eliminado(s)"}, 200
    else:
        return {"message": "Carrito no encontrado"}, 404
```

Sol·licitud DELETE shoppingcart

El primer que fa és extreure el token de sessió de l'usuari i obtenir el seu correu, amb aquest correu s'obté la cistella de la compra de l'usuari i busca si a la cistella existeix un item amb el mateix id que ha arribat per paràmetre.

Un cop detecta l'item que correspon a l'id enviat per paràmetre, s'elimina de la cistella i s'actualitza a la base de dades.

```python
def update_cart(self):
    try:
        doc_ref = db.collection("user_cart").document(self.user_id)
        doc_ref.update(self.json())
        return True
    except Exception as e:
        return False
```

Funció update_cart de la classe Cart

C-15 Finalitzar una compra

En cas de voler finalitzar una compra, s'ha de cridar el endpoint POST order.

```python
def post(self):
    parser = reqparse.RequestParser()
    parser.add_argument("items", type=dict, action="append", required=True)
    parser.add_argument("total_price", type=float, required=True)

    token_header = request.headers.get("Authorization")
    if not token_header:
        return {"message": "Token no proporcionado"}, 401

    token = token_header.split(" ")[1]
    try:
        decoded_token = auth.verify_id_token(token)
        user_id = decoded_token["email"]
    except Exception as e:
        return {"message": f"Error al verificar el token: {str(e)}"}, 403

    data = parser.parse_args()
    items = data["items"]
    total_price = data["total_price"]

    order_id = Order.create_order_id(user_id)
    order_date = datetime.now()
    new_order_data = {
        "user_id": user_id,
        "items": items,
        "total_price": total_price,
        "order_date": order_date.strftime("%Y-%m-%d"),
        "status": "En preparación",
        "order_id": order_id,
    }

    user_orders_ref = db.collection("user_orders").document(user_id)
    user_orders_doc = user_orders_ref.get()

    if user_orders_doc.exists:
        user_orders = user_orders_doc.to_dict().get("orders", [])
        user_orders.append(new_order_data)
        user_orders_ref.update({"orders": user_orders})
    else:
        user_orders_ref.set({"orders": [new_order_data]})

    return new_order_data, 201
```

Sol·licitud POST orders

Igual que en algun dels endpoints vistos, s'ha de proporcionar el token d'inici de sessió de l'usuari i així s'obté el seu correu.

Amb aquest correu podem crear una nova comanda associada a l'usuari, a la qual inclou un array de productes, el preu total de la comanda, un id de la comanda, la data de quan s'ha efectuat aquesta i un estat inicial.

Es comprova si és la primera comanda de l'usuari o ja n'ha fet alguna prèviament. Si és la seva primera comanda es crea a la base de dades el llistat de comandes de

l'usuari amb la comanda inclosa, en cas que ja existeixi el llistat relacionat a l'usuari aquesta s'afegeix.

C-16 Revisar comandes

Per tal de poder obtenir el llistat de comandes que ha fet un usuari, hem d'utilitzar l'endpoint GET orders.

```python
def get(self):
    token_header = request.headers.get("Authorization", None)
    if not token_header:
        return {"message": "Token no proporcionado"}, 401

    token = token_header.split(" ")[1]

    try:
        decoded_token = auth.verify_id_token(token)
        user_id = decoded_token.get("email")
        if not user_id:
            return {"message": "No se pudo obtener el ID del usuario"}, 400
    except Exception as e:
        return {"message": f"Error al decodificar el token: {str(e)}"}, 401
    orders = Order.find_by_user_id(user_id)
    if orders:
        return {"orders": [order.json() for order in orders]}, 200
    return {"message": "No orders found for this user"}, 404
```

Sol·licitud GET orders

Simplement, se li ha de proporcionar el token de sessió de l'usuari, i amb aquest extreure el seu identificador.

Amb aquest identificador es busca a la base de dades una col·lecció de comandes associada a l'identificador de l'usuari.

```python
@classmethod
def find_by_user_id(cls, user_id: str):

    user_orders_ref = db.collection("user_orders").document(user_id)
    user_orders_doc = user_orders_ref.get()

    if user_orders_doc.exists:
        orders_data = user_orders_doc.to_dict().get("orders", [])
        return [cls(**order) for order in orders_data]

    return []
```

Funcií find_by_user_id de la classe Order

Si es troba aquesta col·lecció retorna el llistat de comandes del seu interior juntament amb les dades associades a cada una.
En cas contrari retorna un missatge d'error.

C-17 Modificar contrasenya

Per modificar la contrasenya d'un usuari guardada a Firebase Authentication, hem de cridar l'endpoint POST reset.

```python
def post(self):
    user_id = request.json.get("user_id")

    try:
        auth.generate_password_reset_link(user_id)

        return {
            "message": "Correo electrónico de restablecimiento de contraseña enviado."
        }, 200
    except auth.EmailNotFoundError:
        return {"message": "Usuario no encontrado."}, 404
    except Exception as e:
        return {
            "message": "Error al enviar el correo electrónico de restablecimiento de contraseña.",
            "error": str(e),
        }, 500
```

Sol·licitud POST reset

L'endpoint recull l'user_id enviat, que equival al correu electrònic de l'usuari i utilitza la funció generate_password_reset_link propia de Firebase Authentication que fa arribar un correu a l'usuari amb els passos a seguir per modificar la contrasenya.

7. Frontend

En aquest apartat del treball, explorarem la implementació del frontend de la nostra aplicació web. El frontend, o interfície d'usuari, és el component de l'aplicació amb el qual els usuaris interactuen directament. Hem dissenyat aquesta part tenint en compte la facilitat d'ús i la coherència visual, a més de la seva integració amb el backend per a la funcionalitat completa dels casos d'ús definits.

C-01 Registre

Pantalla de Login

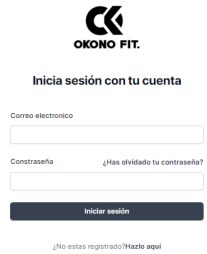
Perquè un usuari es pugui registrar, ho ha de fer a través d'aquesta pantalla.
Hi trobem un formulari en el qual l'usuari ha d'introduir el seu nom, cognoms, correu electrònic i contrasenya.
Un cop les dades estan introduïdes, per procedir al registre es fa una sol·licitud a l'endpoint POST user del backend.
En cas d'efectuar-se correctament l'usuari navegarà automàticament a la pantalla de login.

La trucada pot fallar per error de connexió o per usuari ja existent a la base de dades, en qualsevol dels dos casos es mostra un missatge indicant l'error.

Des d'aquesta vista l'usuari pot navegar a la pàgina principal a través de la icona de la part superior esquerra o navegar a la pàgina de login a través del botó: *Inicia sesión*.

En cas que el format del correu electrònic o la contrasenya no siguin correctes, es mostra un missatge d'error.

C-02 Login i C-03 Login com administrador



Pantalla de Registre

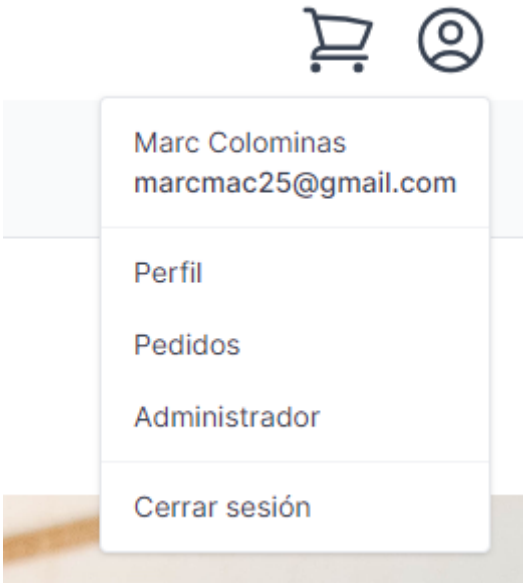
De cara a proporcionar a l'usuari l'interfície perquè l'usuari es pugui loguejar, s'ha creat aquesta pantalla.
Per iniciar sessió l'usuari ha d'introduir el correu electrònic i la seva contrasenya amb els que prèviament s'ha d'haver registrat.

Un cop les dades introduïdes, en clicar iniciar sessió s'executa una trucada a l'endpoint POST login del backend, el qual ens retorna un token de sessió sempre que les dades siguin correctes. La trucada pot fallar per 3 errors diferents, que s'indiquen per pantalla en cas que succeeixin: usuari amb el correu electrònic introduït inexistent, contrasenya incorrecta o error de connexió amb el servidor.

Un cop rebut el token de sessió, el guardem tant al context de l'aplicació com al localStorage, ja que el necessitem guardar mentre l'usuari sigui connectat a la web.

A més, des d'aquesta vista l'usuari pot navegar a la pàgina principal a través de la icona de la part superior esquerra o navegar a la pàgina de registre a través del botó: ¿No estas registrado?.

C-04 Tancar sessió

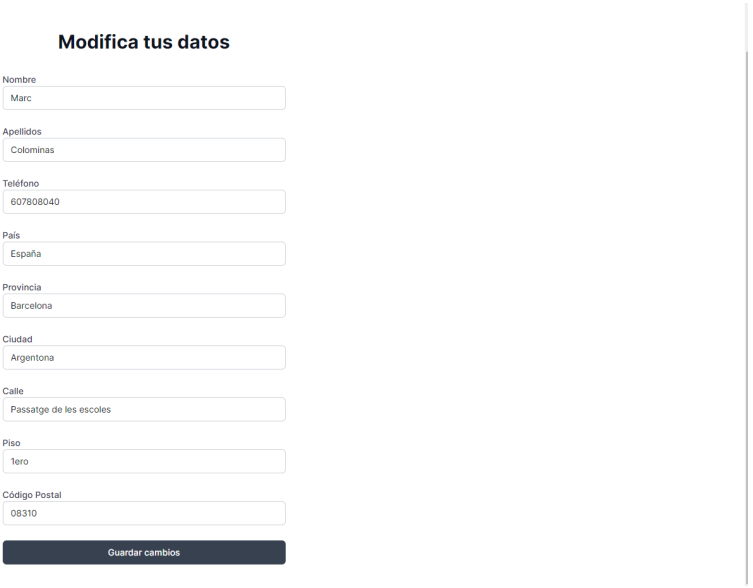


Desplegable icona usuari a la barra de navegació

Per tancar la sessió a la web, l'usuari ha d'obrir el desplegable de la barra de navegació. Un cop obert ha de seleccionar l'opció de Cerrar sesión.

En seleccionar-la, s'eliminen totes les dades del context de l'aplicació cosa, del localStorage i redirigeix a l'usuari a la pantalla de login.

C-05 Veure dades del compte i C-06 Actualitzar el compte



Pantalla de perfil

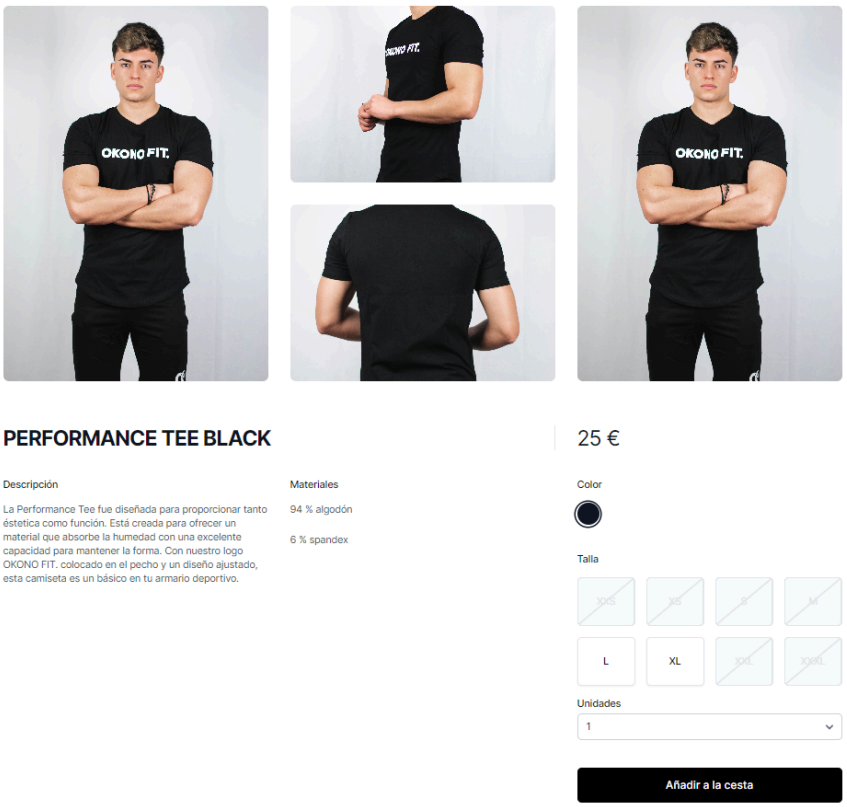
Per proporcionar a l'usuari l'opció de visualitzar i modificar les dades associades al seu compte, tenim aquesta pantalla. S'accedeix a ella a través de la barra de navegació, i només entrar es fa un GET user per recuperar les dades de l'usuari.
Un cop es tenen les dades de l'usuari, s'omplen els camps del formulari amb les dades obtingudes.

El primer cop que un usuari accedeix a aquesta pantalla, només visualitzarà el Nom i Cognoms omplerts automàticament, ja que són les úniques dades que es proporcionen al moment del registre.

Un cop hagi omplert la resta de dades (no és obligatori omplir tots els camps), en fer clic a guardar els canvis s'efectuarà el PUT user.

C-07 Veure un producte i C-13 Afegir un producte a la cistella

Per veure els detalls d'un producte hem d'accedir a la vista individual d'aquest.



Pantalla de un producte

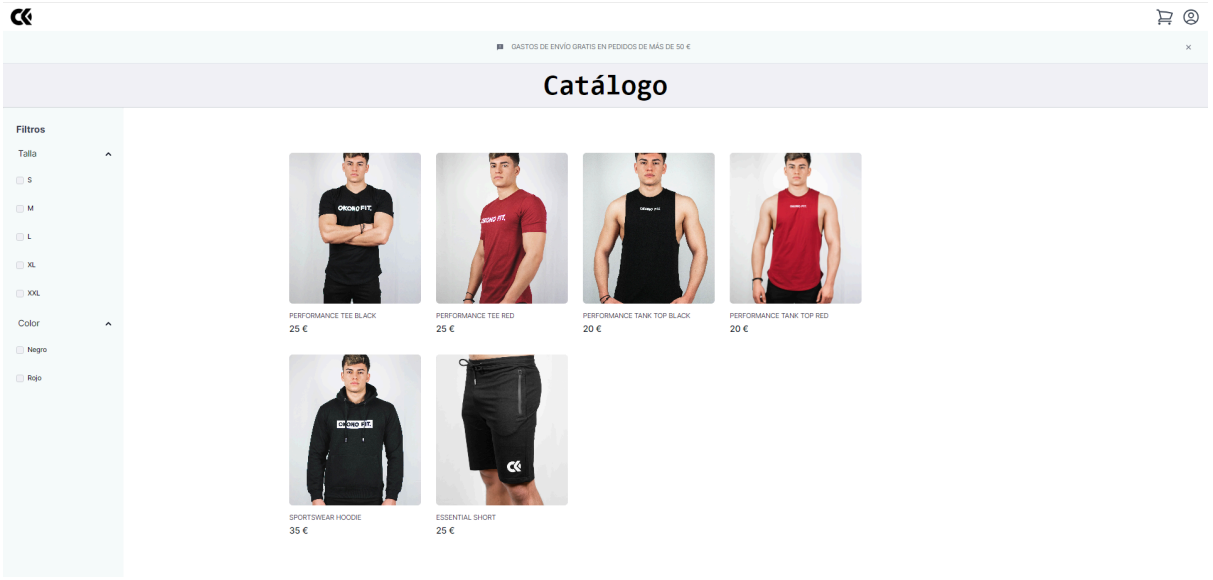
A l'entrar a la pantalla es fa un GET product passant l'id del producte. Amb les dades obtingudes es pinta la pantalla, on es pot observar diverses fotos del producte, una descripció, materials...

A més, hi trobem un component que ens mostra les talles disponibles del producte i hem de seleccionar una. Ja que si no se selecciona cap, no deixarà afegir un producte a la cistella.
També hi veiem un segon component on hem d'indicar el nombre d'unitats del producte que es volen afegir.

Un cop hem seleccionat les dues coses, podem afegir un nou producte a la cistella clicant el botó corresponent. En fer-ho es crida el POST shoppingcart per actualitzar la cistella a la base de dades.

C-08 Veure el catàleg i C-18 FIltrar productes

Un cop accedim a la pantalla del catàleg, es realitza un GET product, sense indicar cap id que ens retorna una llista dels productes del catàleg.



Pantalla del catàleg

En aquesta pàgina hi trobem tots els productes de la web, i els podem filtrar segons la necessitat que tinguem.

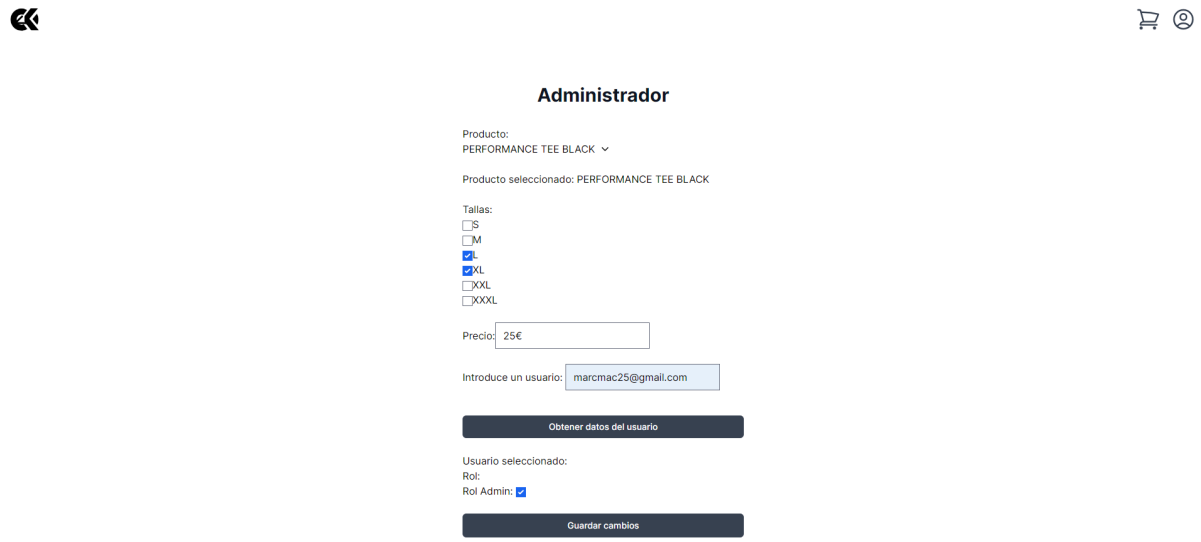
En clicar un dels possibles filtres, s'obre un llistat de checkbox amb les diferents opcions de filtratge. Aquest llistat és multiselecció, és a dir, podem seleccionar més d'una opció de filtratge com pot ser talla L i XL per veure tots els productes que compleixin una de les dues opcions. En desseleccionar tots els filtres torna a aparèixer el catàleg sencer.

En seleccionar un producte del catàleg navegarà la pàgina individual del producte.

C-09 Gestió usuaris i C-10 Gestió productes

Un cop el login és completat i s'accedeix a la vista principal de la web, es fa un GET user que ens retorna les dades de l'usuari, entre elles el seu rol, que es guarda al context.

Si es tracta d'un usuari administrador, en el desplegable de la barra de navegació apareix l'opció Administrador, que redirigeix a l'usuari a la següent vista.



Pantalla d'administrador

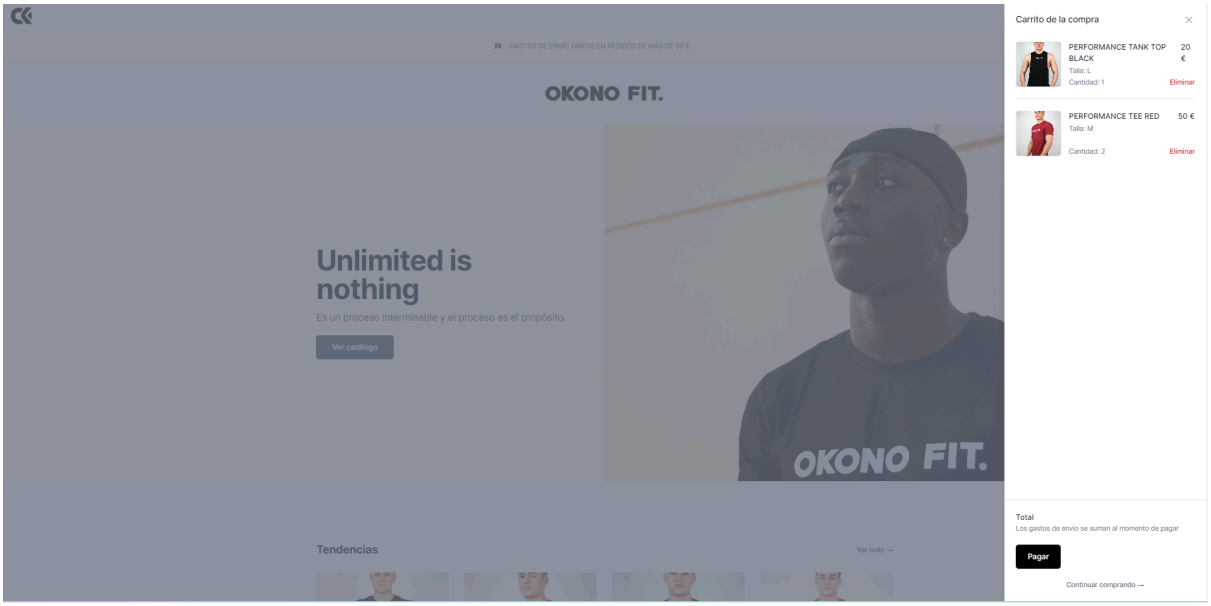
En aquesta vista l'administrador pot fer dues funcions diferents:

- Modificar un producte: Es mostra un desplegable amb tots els productes del catàleg, i en cas de seleccionar un d'ells. Es fa un GET product per obtenir el catàleg de producte, aquest es mostra en un desplegable.
- En seleccionar un producte del desplegable es mostren les talles disponibles i el seu preu, amb l'opció de modificar totes dues.
  Un cop modificades, l'administrador pot guardar els canvis, cosa que farà u PUT product per actualitzar les dades del producte a la base de dades.

- Modificar rol usuari: Hi trobem un camp input on l'administrador pot introduir l'identificador d'un usuari per aconseguir les seves dades fent un GET adminUser.
  Un cop ha rebut les dades de l'usuari és mostra un checkobx indicant si l'usuari té el rol d'administrador.
  En cas de modificar l'estat del checkbox, l'administrador pot guardar els canvis amb el botó proporcionat i així s'efectua el PUT adminUser.

C-12 Veure la cistella i C-14 Eliminar un producte de la cistella

Perpoder visualitzar la cistella de la compra, l'usuari ha de premer la icona de la cistella de la barra de navegació. Un cop fet, s'obre un panel lateral amb la cistella.

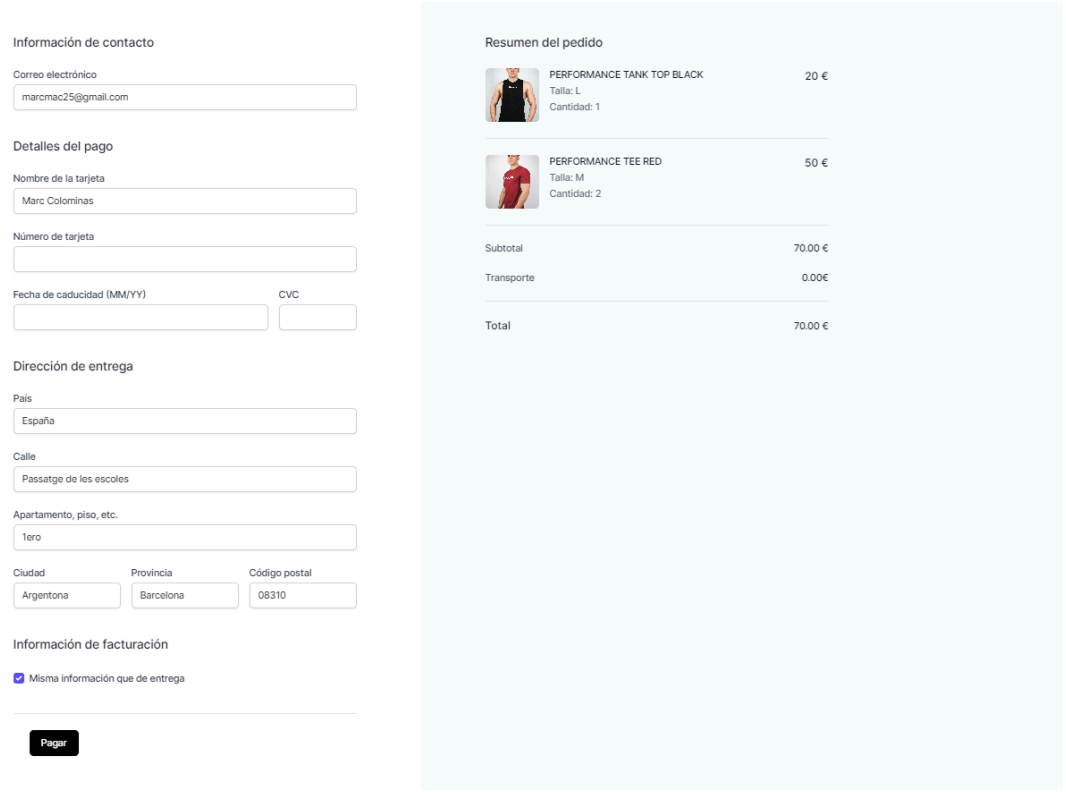


Panel lateral cistella de la compra

En obrir el panel s'efectua el GET shoppingcart que ens retorna el llistat de productes de la cistella associada a l'usuari. Aquests es mostren al component lateral juntament amb un botó eliminar per cada item de la llista.
En prémer el botó eliminar es fa un DELETE shoppingcart que elimina el producte de la cistella.

C-15 Finalitzar una compra

Per finalitzar una compra, l'usuari ha d'accedir a la pantalla de checkout, a la qual s'accedeix seleccionant Pagar a la cistella de compra.



Pantalla de finalitzar compra

Només accedir a la pantalla es fa un GET user i GET shoppingcart per obtenir tant les dades de l'usuari, com les dades de la cistella de compra.

La pantalla està dividida en dos:

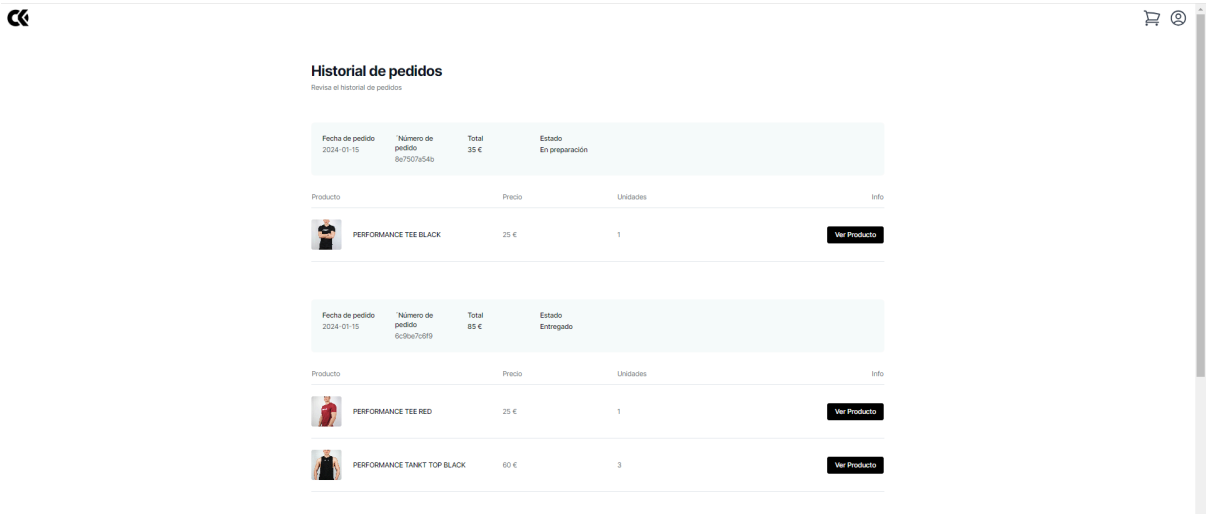
- En una part tenim un formulari que s'omple automàticament en cas que s'hagin introduït abans les dades actualitzant el perfil de l'usuari, exceptuant les dades de la targeta que s'han d'introduir sempre en aquesta pantalla.
-
- A l'altra banda de la pantalla, hi trobem la informació dels productes que es tenien a la cistella, i el preu total de la comanda.

Un cop tenim tots els camps omplerts, es pot acabar la comanda amb el botó pagar, en fer-ho s'efectuarà el POST order.

En cas que el format de les dades associades als detalls del pagament, es mostra un missatge d'error.

C-16 Revisar comandes

Per revisar les comandes que un usuari ha fet anteriorment o que té en procés, l'usuari ha d'accedir a la vista de comandes a través de l'opció *Pedidos* a la barra de navegació.

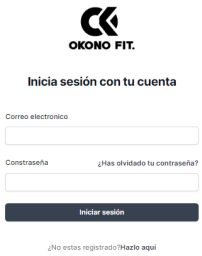


Pantalla d'historial de comandes

En accedir a aquesta pantalla es fa un GET orders, que ens proporciona la llista de comandes de l'usuari i les seves dades.
Un cop les tenim, es mostren, juntament amb la informació corresponent a cada una d'elles.

C-17 Modificar contrasenya

Per modificar la contrasenya, un usuari ha de seleccionar l'opció ¿Has olvidado tu contraseña?, en la pantalla de login.



Pantalla de Login

Llavors es realitza una crida a POST reset, que envia un email a l'usuari amb els passos per modificar la contrasenya.

8. Publicació de la web

Un cop tenim tant el backend com el frontend de l'aplicació desenvolupat i funcionant, és hora de publicar l'aplicació, cosa que es pot fer mitjançant plataformes de hosting.

Cada plataforma ofereix diferents característiques, a continuació recullo algunes de les opcions més populars.

8.1 Frontend

- **Netlify**: És una opció molt popular per a aplicacions frontend basades en JavaScript. Ofereix desplegaments automàtics des de repositoris de GitHub, GitLab i Bitbucket. És fàcil de fer servir i té una generosa oferta gratuïta.

- **Vercel**: Similar a Netlify, Vercel està optimitzat per a aplicacions JavaScript i és particularment amigable amb React. També ofereix desplegaments automàtics des dels principals dipòsits de codi.

- **GitHub Pages**: Si el teu projecte està allotjat a GitHub, pots fer servir GitHub Pages per allotjar el teu frontend de forma gratuïta. És ideal per a projectes més petits o llocs web estàtics.

- **Amazon S3 + Amazon CloudFront**: Si prefereixes una solució més manual i tens coneixements d'AWS, pots utilitzar S3 per emmagatzemar els fitxers i CloudFront per al lliurament de contingut.

Per desplegar el nostre frontend, ho hem fet amb Vercel. A la qual li hem associat el repositori de GItHub, indicant quina es la carpeta on es troba el frontend. D'aquesta manera cada cop que realitzem un nou push a la branca main del repositori, es desplega un nou deployment i actualitza l'aplicació.

URL per accedir a la web: https://okonofit.vercel.app/

8.2 Backend

- **Heroku**: És una de les plataformes més senzilles per desplegar aplicacions web de Flask. Ofereix un nivell gratuït i és força fàcil d'utilitzar, però pot tenir limitacions en termes de rendiment en els seus plans gratuïts.

- **Google Cloud Platform (GCP):** Google App Engine o Google Cloud Run són bones opcions. Són més escalables i tenen una capa gratuïta, però poden ser més complexes de configurar en comparació de Heroku.

- **Amazon Web Services (AWS):** AWS ofereix diversos serveis com EC2, Elastic Beanstalk, o Lambda, que poden ser utilitzats per allotjar aplicacions Flask. Ofereixen una gran escalabilitat i control, però tenen una corba d'aprenentatge més pronunciada.

- **Microsoft** Azure**:** Azure App Service és una altra opció per allotjar aplicacions Flask. Ofereix una integració fàcil amb altres eines i serveis de Microsoft.

- **DigitalOcean**: Ofereix Droplets, que són servidors virtuals privats, i App Platform, un servei més automatitzat similar a Heroku. És una opció econòmica i senzilla per a petites aplicacions.

Finalment, hem desplegat el backend amb Heroku. És fàcil d'utilitzar i ens permet clonar un repositori en el qual a través de git podem anar pujant els canvis i automàticament llança un nou deploy amb els canvis efectuats.

9. Proves

Un cop acabat tot el desenvolupament, s'han de realitzar un conjunt de proves per comprovar i garantir el funcionament de la web. Així com revisar que els requisits funcionals descrits es compleixen.

9.1 Especificació de les proves

Prova-01

Descripció: Accedir amb un usuari registrat
Precondició: -
Passos a seguir:

1. Accedir a la pantalla de login.
2. Omplir el formulari de login.
3. Apretar botó *Iniciar sesión.*

Postcondició: L'usuari és redirigir a la pàgina principal, i té accés a la icona d'usuari de la barra de navegació.
Validació: Correcte
Requisits associats: R-01,R-02, R-03, R-09, R-05, R-29

Prova-02

Descripció: Accedir amb un usuari administrador
Precondició: -
Passos a seguir:

1. Accedir a la pantalla de login.
2. Omplir el formulari de login amb les dades d'un usuari administrador.
3. Apretar botó *Iniciar sesión.*

Postcondició: L'usuari és redirigir a la pàgina principal, i té accés a la icona d'usuari de la barra de navegació. Aquesta mostra l'opció *Administrador*.
Validació: Correcte
Requisits associats: R-01,R-02, R-03, R-04, R-10, R-29

Prova-03

Descripció: Registrar un nou usuari
Precondició: -
Passos a seguir:

1. Accedir a la pantalla de login.
2. Seleccionar el botó *¿No estas registrado?Hazlo aquí*

3. Omplir el formulari amb un nom, cognoms, correu i contrasenya.
4. Apretar botó *Registrarse*

Postcondició: L'usuari ha sigut creat i es redirigeix a la pàgina de login.
Validació: Correcte
Requisits associats: R-01, R-06, R-27

Prova-04

Descripció: Tancar sessió d'un usuari
Precondició: Tenir una sessió iniciada
Passos a seguir:

1. Des de qualsevol pàgina de la web, seleccionar la icona d'usuari de la barra de navegació.
2. Seleccionar l'opció *Cerrar sesión.*

Postcondició: La sessió de l'usuari s'ha tancat i es redirigeix a la pàgina de login.
Validació: Correcte
Requisits associats: R-02, R-09, R-14

Prova-05

Descripció: Accedir a la pàgina principal
Precondició: -
Passos a seguir:

1. Seleccionar la icona de la part superior esquerra.

Postcondició: Es redirigir a l'usuari a la pàgina principal de la web.
Validació: Correcte
Requisits associats: R-09

Prova-06

Descripció: Accedir a la pàgina de productes
Precondició: -
Passos a seguir:

1. Seleccionar *Ver catálogo* a la pàgina principal.

Postcondició: Es redirigir a l'usuari a la pàgina on es mostra tot el catàleg de la web.
Validació: Correcte
Requisits associats: R-21

Prova-07

Descripció: Accedir a la pàgina de individual d'un producte
Precondició: -

Passos a seguir:

1. Seleccionar *Ver catálogo* a la pàgina principal.
2. Selecciona un producte del catàleg.

Postcondició: Es redirigir a l'usuari a la pàgina individual d'un producte.
Validació: Correcte
Requisits associats: R-15

Prova-08

Descripció: Afegir un producte a la cistella
Precondició: Tenir una sessió iniciada
Passos a seguir:

1. Seleccionar *Ver catálogo* a la pàgina principal.
2. Selecciona un producte del catàleg.
3. Seleccionar una talla del producte.
4. Clicar botó *Añadir a la cesta.*

Postcondició: S'ha afegit un nou producte a la cistella de compra.
Validació: Correcte
Requisits associats: R-02, R-08, R-15, R-16, R-21

Prova-09

Descripció: Accedir a la pàgina de perfil de l'usuari
Precondició: Tenir una sessió iniciada
Passos a seguir:

1. Seleccionar l'opció *Perfil* de la barra de navegació.

Postcondició: S'ha redirigit a l'usuari a la pàgina de perfil.
Validació: Correcte
Requisits associats: R-02, R-17

Prova-10

Descripció: Actualitzar perfil d'un usuari
Precondició: Tenir una sessió iniciada
Passos a seguir:

1. Seleccionar l'opció *Perfil* de la barra de navegació.
2. Omple el formulari
3. Selecciona el botó *Guardar cambios.*

Postcondició: S'ha actualitzat el perfil de l'usuari i redirigit a la pàgina principal.
Validació: Correcte
Requisits associats: R-02, R-07, R-17, R-18

Prova-11

Descripció: Veure historial de comandes
Precondició: Tenir una sessió iniciada
Passos a seguir:

1. Seleccionar l'opció *Pedidos* de la barra de navegació.

Postcondició: S'haredirigit a l'usuari a la pàgina de comandes, on es mostren, en cas de que existeixin, les comandes realitzades per l'usuari..
Validació: Correcte
Requisits associats: R-02, R-19

Prova-12

Descripció: Obrir la cistella de la compra

Precondició: Tenir una sessió iniciada

Passos a seguir:

1. Seleccionar la icona de la cistella de la barra de navegació.

Postcondició: S'ha obert el panel que ensenya la cistella.

Validació: Correcte

Requisits associats: R-02, R-07, R-17, R-18, R-22

Prova-13

Descripció: Eliminar un producte de la cistella
Precondició: Tenir una sessió iniciada i un producte a la cistella
Passos a seguir:

1. Seleccionar la icona de la cistella de la barra de navegació.
2. Clicar botó *Eliminar* del costat d'un producte de la cistella.

Postcondició: S'ha eliminat un producte de la cistella de compra.
Validació: Correcte
Requisits associats: R-02, R-08, R-15, R-21

Prova-14

Descripció: Formalitzar una comanda
Precondició: Tenir una sessió iniciada i mínim un producte a la cistella
Passos a seguir:

1. Seleccionar la icona de la cistella de la barra de navegació.
2. Clicar botó *Pagar* del footer de la cistella.
3. Omplir formulari de dades i pagament.
4. Clicar botó *Pagar.*

Postcondició: S'ha creat una nova comanda i redirigir a l'usuari a pàgina de Historial de comandes.
Validació: Correcte
Requisits associats: R-02, R-19, R-23

Prova-15

Descripció: Accedir al panel d'administrador
Precondició: Tenir una sessió iniciada amb un usuari administrador
Passos a seguir:
1. Seleccionar l'opció *Administrador* de la barra de navegació..

Postcondició: S'ha credirigit a l'usuari a la pàgina d'administrador.
Validació: Correcte
Requisits associats: R-02, R-03, R-04, R-10, R-12

Prova-16

Descripció: Editar un producte
Precondició: Tenir una sessió iniciada amb un usuari administrador
Passos a seguir:
1. Seleccionar l'opció *Administrador* de la barra de navegació
2. Seleccionar un producte del desplegable de productes.
3. Seleccionar una nova talla disponible o modificar preu.
4. Apretar botó *Guardar cambios.*

Postcondició: S'ha modificat un producte.
Validació: Correcte
Requisits associats: R-02, R-03, R-04, R-10, R-13

Prova-17

Descripció: Editar el rol d'un usuari
Precondició: Tenir una sessió iniciada amb un usuari administrador
Passos a seguir:
1. Seleccionar l'opció *Administrador* de la barra de navegació
2. Introduir l'identificador d'un usuari.
3. Seleccionar el checkobox rol administrador.
4. Apretar botó *Guardar cambios.*

Postcondició: S'ha modificat un usuari i ara es administrador.
Validació: Correcte
Requisits associats: R-02, R-03, R-04, R-10, R-13

Prova-18

Descripció: Editar una comanda
Precondició: Tenir una sessió iniciada amb un usuari administrador
Passos a seguir:

1. Seleccionar l'opció *Administrador* de la barra de navegació
2. Introduir l'identificador d'una comanda.
3. Introduir el nou estat de la comanda.
4. Apretar botó *Guardar cambios.*

Postcondició: S'ha modificat l'estat d'una comanda.
Validació: Incorrecte
Requisits associats: R-02, R-03, R-04, R-10, R-11

Prova-19

Descripció: Filtrar el catàleg de productes
Precondició: -
Passos a seguir:

1. Seleccionar *Ver catálogo* a la pàgina principal.
2. Seleccionar una talla, a l'apartat de filtres.

Postcondició: Es mostren només els productes que tenen la talla disponible.
Validació: Correcte
Requisits associats: R-21, R-20

Totes les proves definides han sigut completades amb èxit exceptuant la d'editar una comanda per part de l'administrador. Això es deu al fet que per falta de temps no s'ha acabat de completar aquesta funcionalitat. Tot i tenir el backend preparat, faltar desenvolupar aquest cas d'ús per part del frontend.

Amb aquestes proves s'ha comprovat que tots els requisits funcionals descrits estan implementas, exceptuan el R-11 que corresponda gestionar una comanda.

10. Plans futurs

Des d'un inici s'ha plantejat aquest treball per desenvolupar una primera versió de la web, que ha de complir uns requisits mínims, però sense estar preparada per sortir al públic.

En aquest apartat nombraré els següents passos a seguir en cas de voler continuar amb el projecte.

10.1 Nous casos d'ús

A continuació definire nous casos d'ús que crec que s'haurien de desenvolupar abans de sortir al mercat.

C-19 Modificar cistella

- **Nom:** Modificar cistella
- **Actor**: Client
- **Objectiu**: Modificar les unitats d'un producte de la cistella
- **Precondicions**: L'usuari ha de haver iniciat sessió i tenir un producte a la cistella
- **Postcondicions**: S'ha modificat la cistella
- **Flux:**
    - L'usuari obre la cistella.
    - L'usuari modifica les unitats d'un producte a la cistella.

C-19 Afegir preferits

- **Nom:** Afegir preferits
- **Actor**: Client
- **Objectiu**: Afegir un producte a preferits
- **Precondicions**: L'usuari ha de haver iniciat sessió
- **Postcondicions**: S'ha afegit un producte a la llista de preferits
- **Flux:**
    - L'usuari accedeix a la pàgina individual d'un producte.
    - L'usuari selecciona el botó per afegir el producte a preferits.

C-20 Eliminar preferits

- **Nom:** Eliminar preferits
- **Actor**: Client
- **Objectiu**: Eliminar un producte a preferits

- **Precondicions**: L'usuari ha de haver iniciat sessió i tenir un producte a preferits
- **Postcondicions**: S'ha eliminat un producte de la llista de preferits
- **Flux:**
    - L'usuari accedeix a la pàgina individual d'un producte.
    - L'usuari selecciona el botó per eliminar el producte de preferits.

C-21 Login amb compte de google

- **Nom:** Login google
- **Actor**: Client
- **Objectiu**: Iniciar sessió amb un compte de google
- **Precondicions**: -
- **Postcondicions**: L'usuari ha accedit a la pàgina principal
- **Flux:**
    - L'usuari selecciona l'opció *Iniciar sessió amb Google* a la pantalla de login.

C-22 Crear tiquet de suport

- **Nom:** Crear tiquet de suport
- **Actor**: Client
- **Objectiu**: Crear un tiquet de suport en cas de trobar-se qualsevol problema
- **Precondicions**: L'usuari ha d'haver iniciat sessió
- **Postcondicions**: L'usuari ha enviat un tiquet

C-22 Gestionar tiquet de suport

- **Nom:** Gestionartiquet de suport
- **Actor**: Administradpr
- **Objectiu**: Gestionar un tiquet de suport per contestar-lo o modifcar el seu estat
- **Precondicions**: L'administrador ha d'haver iniciat sessió
- **Postcondicions**: L'adminsitrador ha modificat el tiquet

10.2 Millores

En aquest apartat explicare quines són les principals millores que li falten a la web.

Web exclusiva administradors

Tot i que en el projecte actual existeix una vista administrador que apareix en funció del rol de l'usuari emmagatzemat a la base de dades, de cara a un futur l'ideal seria tenir una plataforma exclusiva pels administradors i que no compartís espai amb la web actual.
Es tractaria d'una web de gestió a la qual només el personal de l'empresa tindria accés.
A més, dins dels administradors hi hauria d'haver rols específics. El mànager és el que ha de tenir accés a tots els beneficis d'administrador i qui gestionaria els rols de la resta. Però una persona que treballa a atenció al client no tindria permisos per fer a un altre usuari administrador o modificar l'estoc o el preu d'un producte.

Integrar pagament

En la web desenvolupada, no es comprova que les dades introduïdes siguin reals, i no es cobra cap pago per realitzar una comanda.
Això òbviament hauria de canviar, per fer-ho es podria utilitzar Stripe, que ofereix una àmplia gamma de serveis de pagaments en línia, i eines per a negocis. L'objectiu de Stripe és facilitar les transaccions en línia i encaixa perfectament amb el que busquem.

Gestió d’errors

Tenir una bona gestió d'errors és primordial a l'hora de fer una web, ja que ajuda a millorar l'experiència de l'usuari, en poder proporcionar-li una explicació del problema i com solucionar-lo en lloc de deixar la pàgina penjada inclús quan succeeixen errors inesperats.
També pot ajudar a prevenir vulnerabilitats de seguretat i a millorar la confiança de l'usuari.

Tot i que s’ha fet una petita gestió d’errors durant el desenvolupament de la web, encara és molt millorable.

UX

En cas de voler continuar endavant amb el projecte, considero necessari involucrar una persona d'UX, el disseny d'una web no és tasca senzilla i que una persona qualificada estigui present en el disseny milloraria molt la pàgina tant en l'àmbit visual com en experiència d'usuari.

10.3 Cost del projecte

Fins ara la feina feta no ha tingut un cost real en tractar-se d'un projecte educatiu.
Per tant, el que faré és una estimació aproximada del cost que tindria la continuació del projecte.

Cost del personal

Fa referència als costos relacionats amb els recursos humans involucrats en el projecte.
En aquest cas, plantejaré que per desenvolupar un projecte com aquest és necessari formar un equip de 4 persones.

- **Project manager:** S'encarrega de la planificació del projecte, coordinació de l'equip, gestió de recursos, de complir uns estàndards de qualitat, etc
  .
- **Analista**: Recopila i analitza els requisits de negoci, actua com a comunicació entre l'equip tècnic i de negoci, participa en les proves per assegurar-se que es compleixen els requisits establerts, etc.

- **Dissenyador (UX):** Crea el disseny de la web, dissenya i millora l'experiència d'usuari, col·labora amb el programador per assegurar-se que s'implementa el disseny correctament, etc.

- **Programador**: Implementa el disseny de la interfície d'usuari, construeix la lògica del backend, base de dades i la seva corresponent integració, proporciona manteniment i actualitzacions de la web, etc.

:

Un cop definits els membres de l'equip, cal definir un cost aproximat mensual de cada un:

- Project manager: Entre 2.500€ i 4.000€
- Analista: Entre 2.000€ i 3.500€
- Dissenyador: Entre 1.800€ i 3.000€
- Programador: Entre 2.200€ i 3.800€

Cost total de personal mensual: Entre 8.500€ i 14.300€.

Aproximadament ens quedem amb 11.400€/mes.

Cost del software utilizat

Fins aquest punt del projecte, no s'han utilitzat softwares de pagament.

- VisualStudio Code: 0€
- Firebase: 0€
- Postman: 0€
- Heroku: 0€
- Vercel: 0€

Cal tenir en compte, que els casos de Firebase, Heroku o Vercel, tot i tenir plans gratuïts, si el projecte creix i comencen a tenir més tràfic probablement es necessitaria passar a un pla de pagament.

Cost de hardware

Per realitzar aquest treball no s'ha hagut d'adquirir cap hardware al utilitzar el meu ordinador personal, però en cas de formar un equip hauriem d'adquirir 4 ordinadors com a eines de treball.

Per tant, podem comptabilitzar un cost de 700€ per equip de treball a proporcionar a cada treballador.

L'equip de treball estaria format per un ordinador, un ratoli i cascos.

Cost de hardware total = 700€*4 = 2.800€

11. Conclusions

L'objectiu d'aquest treball de fi de grau consistia en el desenvolupament d'un comerç electrònic que permetés als usuaris comprar productes.
A continuació, descriure les conclusions extretes una vegada acabat el projecte.

El primer punt a destacar és que s'ha complert el principal objectiu establert, i s'ha desenvolupat una primera versió d'una web per una marca de roba, complint els casos d'ús establerts i les funcionalitats requerides. Així i tot, considero que amb una millor organització, tenint en compte les hores dedicades, es podria haver desenvolupat algun dels futurs millores dins d'aquest mateix projecte.

Estic satisfet amb l'elecció de les tecnologies amb les quals he desenvolupat el projecte, ja que s'han adaptat a les necessitats que tenia i el que buscava des d'un inici.

Sobre l'organització del projecte, és el punt negatiu que trec, en haver de canviar la metodologia inicialment plantejada perquè no m'he organitzat correctament i no ser capaç d'executar-la.
M'ha sigut bastant complicat compaginar el projecte amb la feina i la vida social, i m'he quedat bloquejat en certs punts on per moltes hores que li dediques no era capaç d'avançar suficient.

Això no obstant, considero la metodologia Agile molt útil i pràctica, però més enfocada en projectes amb equips més grans que en projectes individuals, i sobretot amb els rols ben definits.

Un dels punts amb el que he estat estancat és amb el disseny visual de la pàgina, he fet i refet visualment algunes vistes al no quedar satisfet amb els resultats obtinguts, és per això, que en l'apartat de plans futurs plantejo la incorporació de personal d'UX, qualificat per fer aquestes funcions.

Per altre banda, des del moment en que vaig començar el projecte la marca de roba s'ha estancat i m'he trobat amb un acatàleg molt curt i poca varietat de productes amb els que "jugar".

Finalment, a escala personal la realització d'aquest treball m'ha permès aprendre i millorar en noves tecnologies i eines. He profunditzat el meu coneixement en React i Flask i assentat les bases per continuar desenvolupant en aquests frameworks en futures feines en el món laboral.

12 .Bibliografia

Vue.js - Official Guide
Descripción: Guía oficial de Vue.js, que ofrece una completa introducción a este popular framework de JavaScript.
URL: https://vuejs.org/guide/introduction.html

Tokio School - ¿Qué es Angular?
Descripción: Explicación detallada y accesible sobre Angular, un framework avanzado para desarrollo web.
URL: https://www.tokioschool.com/noticias/que-es-angular/

SDI - Tipos de Comercio Electrónico
Descripción: Análisis de los diferentes modelos y tipos de comercio electrónico y sus características.
URL: https://www.sdi.es/actualidad/tipos-de-comercio-electronico/

Amazon - ¿Qué es el E-commerce?
Descripción: Un artículo de Amazon que ofrece una perspectiva sobre lo que es el comercio electrónico.
URL: https://sell.amazon.com/es/learn/what-is-ecommerce

Clavei - Ventajas de un E-commerce
Descripción: Blog que discute las ventajas del comercio electrónico frente a los comercios tradicionales.
URL:
https://www.clavei.es/blog/ventajas-de-un-ecommerce-frente-a-los-comercios-tradicionales/

Gadae - 5 Formas de Crear Tu Tienda Online
Descripción: Consejos prácticos y estrategias para crear una tienda en línea exitosa.
URL: https://www.gadae.com/blog/5-formas-de-crear-tu-tienda-online/

Qindel Group - ¿Conoces React y sus beneficios?
Descripción: Artículo que proporciona una visión general de React y sus beneficios en el desarrollo web.
URL:
https://www.qindel.com/conoces-react-y-sus-beneficios-si-no-es-asi-te-lo-explicamos/

BOE - Regulación de Comercio Electrónico en España
Descripción: Texto consolidado del Boletín Oficial del Estado sobre la

regulación del comercio electrónico en España.
URL:
https://www.boe.es/buscar/pdf/2018/BOE-A-2018-16673-consolidado.pdf

Flowbite-React - UI Components
Descripción: Biblioteca de componentes UI para React, facilitando el desarrollo de interfaces de usuario.
URL: https://www.flowbite-react.com/

Tailwind UI - Componentes de Interfaz de Usuario
Descripción: Galería de componentes de UI diseñados con Tailwind CSS, útiles para el desarrollo de frontends modernos.
URL: https://tailwindui.com/components/preview

Next.js - Framework de React
Descripción: Documentación oficial de Next.js, un framework de React para producción.
URL: https://nextjs.org/

Flask-RESTful - Documentación
Descripción: Guía y documentación oficial de Flask-RESTful, una extensión para la creación de APIs en Flask.
URL: https://flask-restful.readthedocs.io/en/latest/

Firebase - Documentación Oficial
Descripción: Documentación completa de Firebase, plataforma de desarrollo de aplicaciones móviles y web.
URL: https://firebase.google.com/docs?hl=es-419

BBVA - Metodología Agile
Descripción: Artículo de BBVA sobre la metodología Agile y cómo ha revolucionado las formas de trabajo.
URL:
https://www.bbva.com/es/innovacion/metodologia-agile-la-revolucion-las-formas-trabajo/

HubSpot - Metodología Agile en Marketing
Descripción: Explicación de cómo se aplica la metodología Agile en el ámbito del marketing.
URL: https://blog.hubspot.es/marketing/metodologia-agile

Innovadeluxe - Ventajas y Desventajas del E-commerce
Descripción: Análisis de las ventajas y desventajas del comercio electrónico.

URL:
https://www.innovadeluxe.com/ventajas-y-desventajas-del-comercio-electronico/

Bambu Mobile - Tecnologías para Desarrollo Web
Descripción: Descripción de las tecnologías más actuales utilizadas en el desarrollo web.
URL:
https://www.bambu-mobile.com/tecnologias-mas-actuales-para-desarrollo-web/

Inabaweb - Bases de Datos Populares en Desarrollo Web
Descripción: Una revisión de las bases de datos más populares utilizadas en el desarrollo web.
URL:
https://www.inabaweb.com/bases-de-datos-mas-populares-en-el-desarrollo-web/

Redigital - Marco Normativo Aplicable a tu Website
Descripción: Guía sobre el marco normativo aplicable a sitios web, incluyendo aspectos legales y de cumplimiento.
URL: https://redigital.economistas.es/marco-normativo-aplicable-a-tu-web-site

PythonES - Solicitudes en Flask: GET y POST
Descripción: Explicación práctica de cómo manejar solicitudes GET y POST en Flask.
URL: https://pythones.net/solicitudes-en-flask-get-post/