UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Higher Polytechnic School of Gandia

Sights - A social networking platform for sharing special moments

End of Degree Project

Bachelor's Degree in Interactive Technologies

AUTHOR: Hernández López, Fabio

Tutor: Palanca Cámara, Javier

ACADEMIC YEAR: 2022/2023

Resumen

Este proyecto representa la conceptualización y ejecución de una plataforma de redes sociales con el propósito de simplificar y enriquecer la experiencia de compartir momentos significativos con una audiencia global. Sights, el nombre de esta innovadora red social, está diseñada para ofrecer una amplia gama de funcionalidades destinadas a satisfacer las necesidades y deseos de sus usuarios.

En el corazón de esta iniciativa se encuentra la intención de proporcionar a los usuarios una plataforma donde puedan cargar y compartir contenido nuevo de manera sencilla. Además, Sights permite la creación de conexiones y la expansión de círculos sociales al permitir a los usuarios agregar amigos a su lista de contactos, lo que fomenta una mayor interacción y compromiso entre los miembros de la comunidad. También ofrece la posibilidad de interactuar con los perfiles de otros usuarios, fomentando así la comunicación y la interacción entre individuos.

Para lograr la implementación efectiva de esta visión, se ha optado por utilizar una arquitectura MERN full-stack, que consta de MongoDB como base de datos, Express.js como framework de servidor, React.js como biblioteca de interfaz de usuario y Node.js como entorno de tiempo de ejecución del servidor. Esta elección de tecnologías de vanguardia permite una experiencia de usuario fluida y altamente receptiva, junto con la capacidad de escalar eficazmente para acomodar un crecimiento significativo en el número de usuarios y la carga de trabajo de la plataforma.

Además, como parte de la estrategia de Sights para ofrecer una experiencia personalizada y atractiva a sus usuarios, se ha incorporado el algoritmo de correlación de Pearson. Este algoritmo se ha seleccionado con el fin de proporcionar recomendaciones de contenido altamente relevantes a los usuarios, basándose en sus preferencias y comportamientos previos. Esto contribuye a enriquecer aún más la experiencia del usuario al ofrecer contenido que es más propenso a captar su interés y mantenerlos comprometidos con la plataforma.

Palabras clave: red social, página web, seguidores, publicaciones, perfil.

Abstract

This project represents the conceptualization and execution of a social media platform with the purpose of simplifying and enriching the experience of sharing meaningful moments with a global audience. "Sights," the name of this innovative social network, is designed to offer a wide range of features aimed at satisfying the needs and desires of its users.

At the heart of this initiative is the intention to provide users with a platform where they can easily upload and share new content. Additionally, Sights allows the creation of connections and the expansion of social circles by allowing users to add friends to their contact list, encouraging greater interaction and engagement among community members. It also offers the possibility to interact with the profiles of other users, thus fostering communication and interaction among individuals.

To achieve the effective implementation of this vision, the choice has been made to use a full-stack MERN architecture, consisting of MongoDB as the database, Express.js as the server framework, React.js as the user interface library, and Node.js as the server runtime environment. This selection of cutting-edge technologies enables a smooth and highly responsive user experience, along with the ability to scale effectively to accommodate significant growth in the number of users and platform workload.

Furthermore, as part of Sights' strategy to offer a personalized and engaging experience to its users, the Pearson correlation algorithm has been incorporated. This algorithm has been chosen to provide highly relevant content recommendations to users based on their preferences and previous behaviors. This further enriches the user experience by offering content that is more likely to capture their interest and keep them engaged with the platform.

Key words: Social media, webpage, followers, posts, likes, profile.

Contents

List of Figures

CHAPTER 1
Introduction

1.1 Introduction

Social media apps have been one of the biggest creations of the 21st century. In the early 2000s, the concept of internet was much different from the one we have nowadays. It was categorised as Web 1.0 [1], meaning you could only view the information provided by pages on a social level (Read-Only Web). A couple of years later this concept evolved thanks to upgraded serves and increase in speed connection, becoming Web 2.0.

A new way of communication was growing and users started to interact more and more with the web page's content, up to a point some interactions were made exclusively between users. This lead to the creation of different platforms, such as Facebook, where the user could upload and share their own content with other people in the world, known as social media.

The definition of social media[2] can be described as a group of websites and applications that focus on communication, interaction, content-sharing and collaboration. This powerful tool can be used for personal use, when interacting with friends or family; and also for business purposes, to promote and advertise products and analyse the customer behaviour.

As a result, the purpose of this project is to recreate a social media web application where you can create your own profile, share your experiences and interact with other user's content all inside the app. The goal is to achieve a good user experience based on a simple web design yet functional, that makes content interaction as simple as possible so all age groups are able enjoy it.

Throughout this document is reflected the whole process of this project's creation. From the early study phases to the validation tests and conclusion.

1.2 Motivation

As mentioned in the introduction, social media apps are such a powerful tool nowadays that can be used with many purposes. It is considered a reliable source of marketing and data analysis as well as the key of user communication.

For this reason, I decided to go in depth on this revolutionary way of social interaction and create a prototype of a social media app following the current concept.

To achieve this, I focused more on simplifying the interface and functionalities so any age group can get the most out of it.

Furthermore, this is a great opportunity to showcase the knowledge acquired in the Interactive Technologies degree; combining coding and UX/UI design skills learned in the different courses throughout the degree.

1.3 Project goals

The main purpose of this project is to develop not only a web application but also a social media environment where any user is able to share best life moments the easiest way possible. Posting new content and interacting with others through likes and comments.

As a result, in order to achieve the main goal the following goals need to be accomplished:

- Analyse the state of art of the social media world as well as the web development, taking in consideration different implementation methods.
- Study the most successful social media platforms to include some of their functionalities in our app.
- Obtain information about the social media world, checking on the different applications and features.
- Study the strengths and weaknesses of the main social media platforms as well as their direct competitors.
- Ensure the web page security requiring to log in in order to access the app.
- Research the most used social media algorithms and implement one in the project.
- Test the work functionalities through validation tests.

1.4 Methodology

This project will carry out SCRUM methodology for its development. By following this, a minimal viable product will be available for the client at all times.

Furthermore, a control version tool has been used with the purpose of taking control of all changes and being able to take a step back if needed. This allows us to have a better tracking of the direction of the project.

During this project creation, an specific order has been has been followed. First, a study of the state of art was necessary to analyse the technological side of the project. Afterwards, the final project is developed based on the previous design and gets checked through validation tests to make sure everything has been done according to the plan.

1.5 Document structure

Chapter I: Introduction

In this first chapter anything related to the introduction can be found. For instance, a brief description about the project's content, the motivation that led me to develop it and the goals that I intend to achieve on the process.

Chapter II: State of art

During this second chapter the technological side of this project will be theoretically analysed. Therefore all different development options will be taken in consideration both on the web creation and the algorithm design. Furthermore, the most successful social media apps nowadays and the most effective algorithms will be compared and broken down.

Chapter III: Requirements analysis and design

In the third chapter it is mentioned all the requirements needed to achieve the desired project version. Therefore the methodology follow will be discussed as well as all app designs, including not only the database and roadmap but also each one of the components forming the webpage interface.

Chapter IV: Project development

The fourth chapter explains the creation process of this project.The web page development will be discussed covering up its functionalities and implementation in both the client and the server side. It also includes a brief explanation of the chosen algorithm for this project, the design and implementation.

Chapter V: Conclusions

In this sixth chapter takes place the final thoughts involving the project. As a result, it showcases the goals achieved and the improving proposals of our project.

Appendix A: User Manual

Finally, this seventh chapter contains the user manual as well as the references to all cites and documents used during the creation process of this project.

CHAPTER 2
State of the art

In this chapter a compounded study of the most successful social media apps as well as the most used recommendation algorithms will take place. As a result, all different tools for its development will be evaluated in order to proceed with the one that fits best our project idea.

2.1 Social media

Social media refers to a collection of internet-based platforms, applications, and websites that facilitate the creation, sharing, and exchange of user-generated content, ideas, information, and multimedia in a virtual social environment. These platforms allow individuals, groups, and organizations to connect, communicate, and interact with each other, often in real-time or near-real-time, regardless of their physical location. Social media encompasses various formats, including text, images, videos, links, and other multimedia content [2].

Key features of social media include user profiles, friend or follower connections, the ability to post and share content, as well as mechanisms for engagement such as likes, comments, shares, retweets, and more. The term "social" highlights the emphasis on interactions and relationships among users within the digital space [5].

Social media platforms vary in their specific focus and functionalities, catering to different interests, such as social networking (e.g., Facebook, LinkedIn), microblogging (e.g., Twitter, Tumblr), photo sharing (e.g., Instagram, Pinterest), video sharing (e.g., YouTube, TikTok), professional networking (e.g., LinkedIn), and messaging (e.g., WhatsApp, Facebook Messenger)[4].

Overall, social media has become a significant part of modern communication and culture, shaping how people connect, express themselves, share information, and engage with a global audience.

2.1.1. Advantages of social media

Social media offers numerous advantages and benefits, both for individuals and businesses. Here are some key advantages of using social media [6]:

- Global Connectivity: Social media allows people from around the world to connect and interact, breaking down geographical barriers and fostering cross-cultural communication.

- Information Sharing: Social media platforms serve as a quick and efficient way to share news, information, and updates on a wide range of topics, from current events to personal interests.
- Networking and Collaboration: Social media is a powerful tool for building professional networks and collaborations. It's especially valuable for job seekers, freelancers, entrepreneurs, and businesses seeking partnerships.
- Education and Learning: Social media platforms often host communities and groups dedicated to education and learning. People can access a wealth of information, tutorials, and resources to enhance their knowledge and skills.
- Promotion and Marketing: For businesses, social media provides a cost-effective way to reach a wide audience and promote products or services. Targeted advertising and analytics help tailor marketing efforts for specific demographics.
- Entertainment and Discovery: Users can discover new music, movies, books, and other forms of entertainment through social media recommendations and shares.
- Democratic Expression: Social media can empower individuals to voice their opinions, participate in discussions, and engage in political discourse, fostering democratic values.
- Market Research: Businesses can gather valuable insights about consumer preferences, behaviors, and trends through social media interactions and data analysis.

2.1.2. Disadvantages of social media

While social media offers numerous advantages, it also comes with a range of disadvantages and potential drawbacks. Here are some of the key disadvantages of social media [7]:

- Privacy Concerns: Social media platforms often collect personal data, and inadequate privacy settings can lead to unauthorised access, identity theft, or misuse of personal information by third parties.
- Misinformation: Due to the speed at which information spreads on social media, false or misleading content can become viral before it's properly verified, leading to widespread misunderstanding and confusion.
- Cyberbullying: The anonymity provided by social media can enhance individuals to engage in hurtful behaviour, leading to cyberbullying and online harassment, which can have severe emotional and psychological impacts on victims.
- Addiction: The constant stream of notifications, updates, and the fear of missing out can lead to addictive behaviour, where individuals find it challenging to disconnect and prioritise offline activities.
- Mental Health Impact: The curated nature of social media content can lead to social comparison, where individuals perceive others' lives as more ideal, potentially leading to feelings of inadequacy, low self-esteem, and even depression.
- Reduced Face-to-Face Interaction: Spending excessive time on social media can reduce real-world interactions, leading to a decline in meaningful in-person relationships and communication skills.

- Digital Footprint: Information posted online, even if deleted later, can leave a lasting trace. Inappropriate or unprofessional content can affect one's reputation, job prospects, and personal relationships.
- Echo Chambers: Social media algorithms often show users content that aligns with their existing beliefs, limiting exposure to diverse viewpoints and contributing to polarization and a lack of understanding of opposing perspectives.

Understanding these disadvantages can help users navigate social media more responsibly and make informed decisions about their online presence and engagement.

2.2 Social media apps

In order to have a better understanding of how present social media is in our lives, lets have a look at the most successful social media apps of our generation and what made them escalate to the point they are nowadays[4].

2.2.1. Facebook

Facebook is a prominent social media platform founded by Mark Zuckerberg in 2004. It enables users to create profiles, connect with friends, and share content. Its success stems from its early innovation, user-friendly interface, and global reach. It gained popularity due to its exclusivity on college campuses initially and later opened to the public, fostering a sense of connection. Facebook's News Feed, introduced in 2006, revolutionized content distribution by curating posts based on user interactions. Its acquisition of Instagram and WhatsApp expanded its influence [8].

Facebook's success can be attributed to its user-centered design, consistent updates, integration of multimedia content, and the introduction of features like reactions and events. The platform has faced criticism for privacy concerns, misinformation, and its impact on mental health. Unlike other social media apps, Facebook's historical focus on personal connections and content sharing sets it apart from platforms like Twitter (focused on microblogging) and LinkedIn (focused on professional networking). However, its evolving ecosystem aims to offer a wide array of features to cater to various user needs.

2.2.2. Instagram

Instagram, launched in 2010, is a visual-centric social media platform allowing users to share photos and short videos. Its success can be attributed to its emphasis on visual storytelling, aesthetic appeal, and user engagement. Instagram's use of filters and easy editing tools made amateur photography more appealing, fostering creativity and self-expression. It gained rapid popularity due to its mobile-first approach and integration with other platforms like Facebook [9].

The platform's success lies in its simplicity, focusing primarily on images and later incorporating Stories, IGTV, and Reels for more diverse content. Its visual nature sets it apart from text-heavy platforms like Twitter, and its focus on lifestyle and imagery makes it distinct from LinkedIn, which focuses on professional connections. Instagram's strong community engagement, influencer culture, and visual branding opportunities have made it a favourite for individuals, brands, and artists seeking to showcase their visual identity and connect with audiences in a visually compelling way.

2.2.3. Tiktok

TikTok, launched in 2016, is a short-form video platform that enables users to create and share engaging, often humorous, 15 to 60-second videos. Its success stems from its innovative content format, algorithm-driven feed, and the ability for users to easily create and participate in viral challenges. TikTok's algorithm curates content based on user preferences, encouraging discovery and engagement [10].

TikTok's success is largely attributed to its user-generated content culture, fostering a sense of authenticity and creativity. Its emphasis on entertainment and short videos distinguishes it from platforms like YouTube, which often features longer content. Unlike Instagram, TikTok's algorithm-driven nature means that even users with small followings can experience widespread visibility. The platform's immersive and easily digestible content format sets it apart from text-heavy platforms like Twitter, creating a unique space for users to engage through short video snippets and participate in global trends and memes.

2.2.4. Youtube

YouTube, established in 2005, is a video-sharing platform allowing users to upload, view, and share videos. Its success can be attributed to its democratization of video content creation, enabling individuals and organizations to showcase diverse content, from educational tutorials to entertainment. YouTube's user-friendly interface, monetization options, and recommendation algorithm that suggests relevant videos have contributed to its popularity [11].

YouTube's success lies in its vast content library, hosting a wide array of video genres. Unlike text-based platforms like Twitter, YouTube focuses exclusively on video content, setting it apart from platforms like Instagram that also incorporate images and Stories. Its longevity and influence have led to the rise of content creators who have built massive followings and careers on the platform. Additionally, YouTube's role as a platform for long-form content differentiates it from short-form platforms like TikTok. The platform's extensive search capabilities and diverse content make it a go-to source for both entertainment and information.

Nevertheless, with Youtube's last update short vertical video format is now available as well. This inspiration comes from TikTok and Instagram since it is obvious the future of content creation is driving towards this format.

2.2.5. LinkedIn

LinkedIn, established in 2002, is a professional networking platform designed for career development, connecting professionals, job seekers, and businesses. Its success is attributed to its focus on professional connections, networking, and career opportunities. LinkedIn's features, such as user profiles highlighting work experience and endorsements, have facilitated meaningful connections in the professional world [12].

LinkedIn's success is driven by its niche focus on professional networking, making it distinct from platforms like Facebook and Instagram, which emphasize personal and social connections. It provides a platform for users to showcase their resumes, skills, and accomplishments, which is different from content-sharing platforms like TikTok or YouTube. LinkedIn's emphasis on business relationships and industry-related content sets it apart from platforms like Twitter, where a broader range of topics is discussed. Its

job listings and company pages also make it a hub for recruitment and business networking, catering specifically to professional growth and development.

2.3 System algorithms in social media apps

Big part of social media apps' success comes from algorithm-driven strategies. Every user has some kind of profile where an specific type of content is displayed on his feed. This content varies depending on what the user likes, follows or comments as well as what his/her followers do inside the app to some extend. The content shown changes dynamically as the behaviour of the user does overtime [13].

This is an excellent mandatory functionality to have in a social media app in order to retain the user the most amount of time possible. Therefore lets discuss some of the most successful algorithms present nowadays [14].

2.3.1. Types of algorithms in social media apps

- **Feed Ranking Algorithm**: This algorithm determines the order in which content appears in a user's feed. It considers factors like user engagement history, content relevance, recency, and sometimes even the relationships between users.
- **Relevance Algorithm**: This algorithm analyzes user interactions and preferences to show content that aligns with their interests. It considers past likes, comments, shares, and clicks to predict what users are likely to engage with.
- **Recommendation Algorithm**: Often used in video platforms, this algorithm suggests additional content based on a user's current viewing habits, aiming to keep users engaged and increase their time on the platform.
- **Explore or Discover Algorithm**: Found in platforms like Instagram, this algorithm suggests new content or profiles to users, exposing them to posts from accounts they may not follow but might find interesting.
- **Hashtag Algorithm**: Used in platforms that support hashtags, this algorithm helps categorize and organize content. It can show users the most popular or relevant posts for a specific hashtag.
- **Search Algorithm**: Used in search functions, this algorithm ranks search results based on factors like relevance, popularity, and user engagement.
- **Ad Targeting Algorithm**: For advertising purposes, social media platforms use this algorithm to show ads to users based on their demographics, interests, and online behavior.
- **Engagement Algorithm**: Used to calculate metrics like likes, shares, and comments, this algorithm helps determine the popularity of a piece of content.
- **Sentiment Analysis Algorithm**: This algorithm gauges the sentiment of comments or posts, classifying them as positive, negative, or neutral. It's often used to assess brand reputation and public sentiment.
- **Personalization Algorithm**: Employed across various features, this algorithm tailors the user experience by showing content, ads, and recommendations that match individual preferences and behavior.

- **Friend or Connection Suggestion Algorithm**: Helps users find and connect with people they may know based on mutual friends, location, workplace, and other factors.
- **Time Decay Algorithm**: This algorithm considers the recency of content, giving priority to newer posts to ensure users see the most up-to-date information.

These algorithms continuously evolve based on user behavior and platform goals. Their aim is to enhance user engagement, provide a personalized experience, and optimize the user's time spent on the platform. Ideally every app should have a combination of different types of algorithms in order to get the best metrics out of each user and maximize its potential.

2.3.2. Recommendation algorithms

In our case, we are going to focus on recommendation algorithms to build the motor of our app. Lets discuss more in depth how can they be applied to the social media environment and the most frequent used ones.

Recommendation systems in social media platforms rely on various techniques, including collaborative filtering (based on user behaviors and preferences), content-based filtering (analyzing the content of posts), hybrid approaches (combining multiple methods), and machine learning algorithms that leverage user data patterns to make predictions [15].

These systems aim to create a personalized and engaging user experience, keeping users active on the platform by offering them content and interactions that are relevant to their individual tastes and preferences. However, ethical considerations regarding data privacy, transparency, and potential biases are essential when implementing and using recommendation systems in social media.

KNN algorithm

The k-Nearest Neighbours (KNN) algorithm is a machine learning technique used for classification and regression tasks. In the context of recommendation systems, KNN is used as a collaborative filtering method to suggest items (or users) to a target user based on the preferences of similar users. It works by identifying the k-nearest neighbours to the target user, then recommending items that these neighbours have liked or interacted with. Here's how the KNN algorithm functions [16]:

1. **Data Collection**: Collect user interaction data, such as item ratings, likes, or clicks, and create a matrix where rows represent users and columns represent items.
2. **User Similarity**: Calculate the similarity between users based on their interactions. Common similarity metrics include cosine similarity.
3. **Neighbour Selection**: Identify the k users with the highest similarity to the target user. These users are the "nearest neighbours."
4. **Item Recommendation**: Recommend items that the nearest neighbours have interacted with and that the target user has not yet interacted with.

An example of an app using this algorithm is a movie recommendation system called "FilmFlix" that employs the KNN algorithm to provide movie suggestions to users based on their preferences and similarities to other users:

1. **Data Collection**: "FilmFlix" [17] collects user ratings and reviews for various movies. Each user rates movies on a scale from 1 to 5 stars.

2. **User Similarity**: The app calculates the similarity between users based on their movie ratings. One common metric for similarity is the cosine similarity, which measures the cosine of the angle between two users' rating vectors.

3. **Neighbour Selection**: For a given user, "FilmFlix" identifies the k users with the highest similarity scores (nearest neighbours) based on their movie ratings.

4. **Movie Recommendation**: The app recommends movies that the nearest neighbours have highly rated but that the target user has not yet seen or rated.

Example Scenario:

Suppose User A and User B are both users of the "FilmFlix" app. Their movie ratings are as follows:

```
User A: The Matrix (4), Inception (5), Jurassic Park (3), Avatar (4)
User B: The Matrix (3), Inception (4), Jurassic Park (4), Avatar (3)
```

Let's say "FilmFlix" employs the KNN algorithm with k = 1 (nearest neighbor). Since User A and User B have rated three out of four movies similarly, they are considered close neighbors.

If User A has not yet seen or rated "Jurassic Park" and User B highly rated it, the KNN algorithm might suggest "Jurassic Park" to User A based on the strong similarity between their ratings.

In this example, "FilmFlix" uses the KNN algorithm to recommend movies to users based on their movie preferences and the preferences of their nearest neighbours. KNN is effective for identifying users with similar tastes and making personalized recommendations, but its accuracy can vary based on the choice of k and the sparsity of data.

KNN-based recommendation systems are effective for identifying user preferences based on their similarities with others. However, they can face challenges when dealing with sparse data or cold-start problems (new users with limited interaction history). Enhancements like weighting neighbours based on similarity strength and using matrix factorization techniques can improve the accuracy of KNN-based recommendations.

Pearson correlation algorithm

The Pearson correlation algorithm is a statistical method used to measure the linear relationship between two variables. In the context of recommendation systems, it's used to determine the similarity between users or items based on their interactions with different items. The Pearson correlation coefficient quantifies the strength and direction of the linear relationship, indicating how well the two variables move together [18].

$$\rho = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.1}$$

Here's how the Pearson correlation algorithm works:

1. **Data Preparation**: Collect user interaction data, such as ratings or likes for different items, and create a matrix where rows represent users and columns represent items.

2. **Centering Data**: Subtract the mean rating or interaction score from each user's ratings to account for individual rating tendencies.

3. **Calculate Pearson Correlation**: Calculate the Pearson correlation coefficient (Equation 2.1) between two users (or items) by comparing their centered ratings. A positive coefficient indicates a positive linear relationship, while a negative coefficient indicates a negative one.

4. **Neighbour Selection**: Identify users (or items) with high correlation coefficients as potential neighbours. These users have similar interaction patterns.

5. **Item Recommendation**: Recommend items that neighbours have interacted with but that the target user has not yet interacted with.

An example of an app using this algorithm is Last.fm [19], a music discovery platform that uses the Pearson correlation algorithm to recommend music to users based on their listening habits and the preferences of similar users. Users on Last.fm can scrobble (track) the songs they listen to across various music streaming services:

1. **Data Collection**: Last.fm collects data on users' scrobbled songs, including song titles, artists, and listening frequency.

2. **Calculate Pearson Correlation**: The app calculates the Pearson correlation coefficient between users based on their scrobbled songs. Users who have similar patterns of listening to songs are considered to have a higher correlation.

3. **Neighbour Selection**: Last.fm selects users with high positive correlation coefficients as neighbours, implying they have similar music preferences.

4. **Song Recommendation**: The app recommends songs and artists that neighbors with similar listening habits have enjoyed but that the target user has not yet listened to.

Example Scenario:

Suppose User A and User B are both Last.fm users. Based on their scrobbled songs, the Pearson correlation algorithm finds that User A and User B have a high correlation coefficient, indicating similar music preferences.

If User A has been listening to songs by Artist X and User B has been listening to similar songs by the same artist, Last.fm might recommend songs by Artist X to User A that they haven't yet discovered.

In this example, Last.fm effectively uses the Pearson correlation algorithm to suggest new songs and artists to users based on the musical preferences of similar users, enhancing the music discovery experience.

Pearson correlation-based recommendation systems can be useful for identifying users or items that align well with a target user's preferences. However, they may face challenges when dealing with sparse data or when interactions are highly skewed. Regularization techniques and incorporating additional information can enhance the accuracy of recommendations.

2.3.3. Algorithm selection

Once the most used algorithms have been analysed, it is time to decide which one is going to be implemented in our proyect.

After discussing it with my tutor, we think the Pearson correlation algorithm adjusts best to our needs and our app concept, therefore it will be the one utilized to recommend content based on its measurements.

2.4 Web development tools

In this section we will study the different alternatives in order to achieve the ideal development of the app. Full-stack and single back-end and front-end options will be taken into account to determine the best fit.

2.4.1. Laravel

Laravel is a popular open-source PHP web application framework used for building web applications. It provides tools and features to streamline development, including routing, templating, ORM, and authentication. Laravel follows the MVC (Model-View-Controller) architectural pattern and emphasizes code readability and maintainability.

However, building a social media app involves various complexities beyond technical aspects, including scalability, real-time updates, data privacy, and user experience. While Laravel provides a solid base, it's essential to consider other technologies, such as real-time communication frameworks (like WebSocket), database choices, and frontend technologies for a comprehensive social media app development.

Since PHP usage is decreasing as years pass by, we will try to avoid this option as there are plenty of more modern and common frameworks available at this time [21].

2.4.2. Django

Django is a high-level open-source Python web framework that enables developers to build web applications quickly and efficiently. It follows the Model-View-Controller (MVC) architectural pattern (referred to as Model-View-Template in Django) and provides a comprehensive set of tools for handling various aspects of web development, including URL routing, template rendering, database management, and user authentication.

It follows more or less the same concept as Laravel but using Python language instead. It also allows the option of using it just for back-end purposes, which will be considered as an option as well.

Some of the most known features are its admin panel, which offers a solid content management; use of DRY (Don't Repeat Yourself) coding principles and encourage efficient development. It also provides simplification of HTML forms as well as serialization of JSON or XML data among others [22].

2.4.3. Node.js

Node.js is an open-source runtime environment that allows developers to build server-side applications using JavaScript. It is built on Chrome's V8 JavaScript engine, making it efficient and well-suited for building scalable and real-time applications. Node.js uses an event-driven, non-blocking I/O model, which makes it particularly suitable for handling asynchronous operations and concurrent connections. When building a social media app, Node.js can offer the following [23]:

- Real-time Capabilities: Node.js excels in handling real-time features, like instant messaging, notifications, and live updates, which are crucial for a dynamic social media app.
- Scalability: Its non-blocking architecture makes it highly scalable, allowing the app to handle a large number of concurrent connections efficiently.
- JavaScript Everywhere: Using JavaScript both on the frontend and backend (Node.js) can lead to a more consistent and streamlined development process.
- Vibrant Ecosystem: Node.js has a vast ecosystem of libraries and frameworks, like Express.js for building APIs and Socket.io for real-time communication.
- Performance: Node.js is known for its speed and efficiency, making it a suitable choice for handling the high concurrency demands of a social media app.

Express.js

Express.js is a fast, minimalistic, and flexible web application framework for Node.js. It provides developers with a set of tools and utilities to build web applications and APIs efficiently. These are some of its key features:

- Routing: Express simplifies the definition of routes and URL endpoints, allowing developers to map specific HTTP methods (GET, POST, PUT, DELETE, etc.) to corresponding actions and responses. This makes it easy to create APIs and handle different types of requests.
- Middleware: Express's middleware system enables developers to insert functions that process requests and responses in a sequential manner. Middleware functions can perform tasks like authentication, logging, data parsing, and error handling. This modular approach enhances code organization and reusability.
- Extensibility: Express can be extended with third-party packages, known as middleware, which add functionality to the application. These middleware components can be easily integrated to enhance the application's capabilities, such as adding authentication, security, or form validation.

In summary, Express.js is a versatile and widely-used framework that streamlines the development of web applications and APIs in the Node.js environment. Its routing, middleware, and extensibility features provide developers with the tools to efficiently create robust and scalable backend systems.

However, while Node.js and Express.js are a powerful choice for building the backend of a social media app, it's important to consider other technologies and frameworks for handling frontend development, databases, and complex features like user interactions and data privacy [24].

2.4.4. React

React.js is an open-source JavaScript library developed by Facebook. It is used for building user interfaces, particularly for single-page applications and dynamic web applications. React focuses on creating reusable UI components that update efficiently when the underlying data changes, making it highly suitable for building interactive and responsive interfaces. Some of its main features include:

- Component-Based Architecture: React's component-based approach allows developers to create modular UI elements that can be reused throughout the app, which is beneficial for building consistent and maintainable user interfaces in a social media app.
- Virtual DOM: React uses a virtual DOM to efficiently update only the necessary parts of the actual DOM when data changes. This results in improved performance and smoother user experiences, especially important for real-time updates in a social media app.
- Declarative Syntax: React's declarative syntax makes it easier to describe how the UI should look based on the app's state. This is particularly helpful for handling the dynamic content and interactions required in a social media app.
- Efficient Updates: React's reconciliation algorithm optimizes the rendering process, ensuring that only the necessary changes are made to the UI. This is crucial for handling a large number of posts, comments, and interactions in a social media app.
- Ecosystem and Libraries: React has a rich ecosystem of libraries and tools, such as Redux for state management and react-router for routing, that can enhance the development of a feature-rich social media app.
- Responsive UI: React's ability to handle UI updates efficiently supports the creation of a responsive user interface, crucial for delivering a seamless experience across various devices.

All of this in combination with a powerful back-end framework can result in a really appealing option for any web development project [25].

2.4.5. MongoDB

MongoDB is a popular open-source NoSQL database management system that uses a document-oriented approach to store and manage data. Unlike traditional relational databases, MongoDB stores data in flexible, JSON-like documents, making it well-suited for handling large volumes of unstructured or semi-structured data:

- Document-Oriented Storage: it stores data in flexible, JSON-like documents instead of traditional rows and columns. This allows developers to store and retrieve complex, semi-structured, or unstructured data easily, making it suitable for applications with diverse data types like social media apps.
- Scalability and Flexibility: it is designed for horizontal scalability, allowing you to distribute data across multiple servers to handle increasing loads. It offers sharding, which splits data across clusters, and replication, which ensures data availability and fault tolerance. This scalability is essential for managing the high volume of user-generated content and interactions in a social media app.
- Rich Query Language and Indexing: it supports powerful queries and indexing to efficiently retrieve data. Its query language enables filtering, sorting, and aggregation of data, which is crucial for retrieving specific posts, comments, and user interactions in a social media app. MongoDB's indexing capabilities improve query performance by allowing faster data retrieval based on specific fields.

This system needed to be mention since it merges with Node.js quite well, using Javascript as language in both technologies. It is also a good option to consider due to its good performance with low complexity applications such as the one that will be developed [26].

2.4.6. Web tools selection

After reviewing some of the most used tools nowadays, the choosen option for the follow up of our proyect will be MERN (MongoDB, Express, React and Node).

During my time at university Javascript has been by far the most used language, therefore is the one I am most familiar with and have more knowledge of. Moreover, the decision of combining technologies that share the same language will make it easier overall to come up with a well-structured app.

On one hand, it is true Node and Express have been used at university so I carry a decent knowledge base; but on the other hand MongoDB and React are two tools I have never utilized so I decided to take the challenge and implement them in this app. This is an excellent opportunity to expand my knowledge and get more familiar with the most recent technologies.

CHAPTER 3
Project requirements and design

This chapter showcases the followed path of this project's creation. By this we define the methodology used, the designs and the project requirements to achieve.

3.1 Methodology

3.1.1. Scrum agile Methodology

Scrum has been the methodology chosen to follow since it has been the one used throughout all four years of this degree in all projects.It allow us to work with a minimal viable product, something with enough functionalities to satisfy our initial clients.

It consists on a simple concept since all tasks are identified at all times, including its person assigned, goal, estimated time of accomplishment and the expected outcome. It is important to mention the continuous follow up due to the numerous meetings and check outs so possible mistakes can be detected and fixed.

Furthermore, the project is constantly adapting to the client requests so we can modify the final project features at any given moment during its development. This allows us to keep our client satisfied, integrate him in the creation process and let him choose what tasks should be done in each sprint and check out the results accomplished after each one of them.

Now that all reasons of choosing Scrum methodology are covered up, it is time to analyse the procedure followed on this project.

Firstly, a list of requirements and goals to achieve by the end of the project is precised to track the proper development of the functionalities as well as the different designs and other parameters. After this initial step is done, we can proceed with the "Sprints", which are fixed time periods where the creation of a product takes place.

At the beginning of each sprint we come up with the planing of each one of the tasks to accomplish. Afterwards, it is important to distribute those tasks between the development team members. Throughout each Sprint short meetings take place to catch up and review everyone's work, to ensure the direction taken is the correct one and to solve possible doubts that may come up in the process.

Finally, every time a Sprint ends comes the "Sprint Review" day, where every in the team showcases the work done to the client. This is useful for the team to analyse the results and to take in consideration possible client's comments and improvements [27].

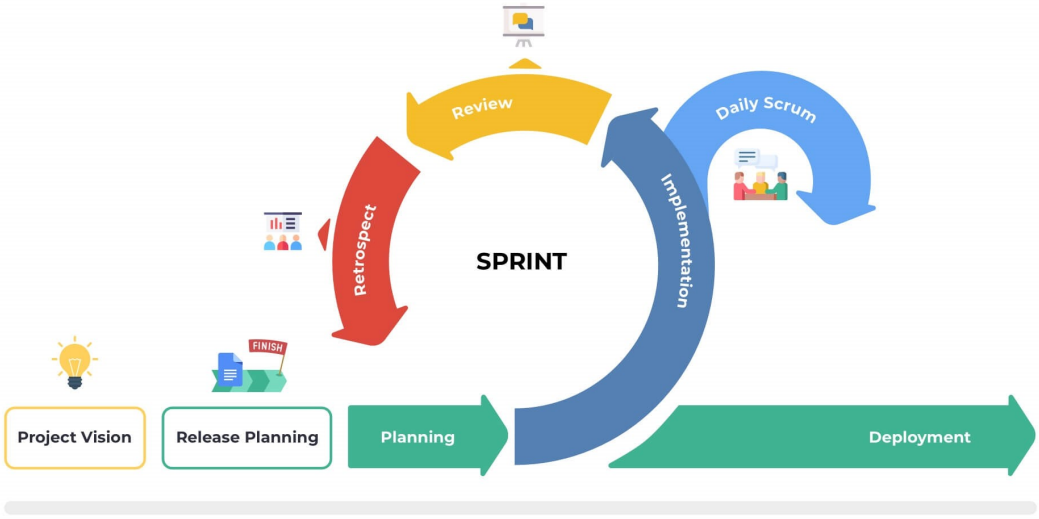

Figure 3.1: Scrum methodology steps

In my case each Sprint lasted 2 weeks and my client was represented by my tutor Javier Palanca. Furthermore, no daily meetings took place since I am developing this project by myself and the tasks assignment has always been done to myself as well.

3.1.2. Version control system

Version control systems allow us to make sure the project is versioned at all times in order to track the changes made in each Sprint. By this, going back to a previous code version is always possible.

In this case I decided to use Git in combination with GitHub Desktop, storing all changes in the main branch.

3.1.3. Roadmap

In this section, the whole creation process is explained on detail.

Firstly, a study of the state of art where the technological situation is analysed. Here we include a comparison of the most successful social media apps as well as their development side. The use of recommendation algorithms was key for their growth, therefore it is necessary to mention the most used nowadays to determine which ones would fit in our project.

Secondly, the design part comes in to settle what is next to be programmed. It is crucial this part is done properly in order to anticipate and fix possible errors that may come up later. In this section we will design the visual part through logos and mock ups as well as the database and its methods.

Thirdly, we implement the designs on our final project. By doing this we select from all functionalities the ones that adjust to our project the most to set the identity of our website. We will include authorisation and authentication to ensure security for our users. Once all the previous mentioned is completed, we proceed with the algorithm implementation to finalise our app.

Finally, the proper tests will be run in order to check everything was done the way it was planned to. There are validation tests for both the functionalities and API.

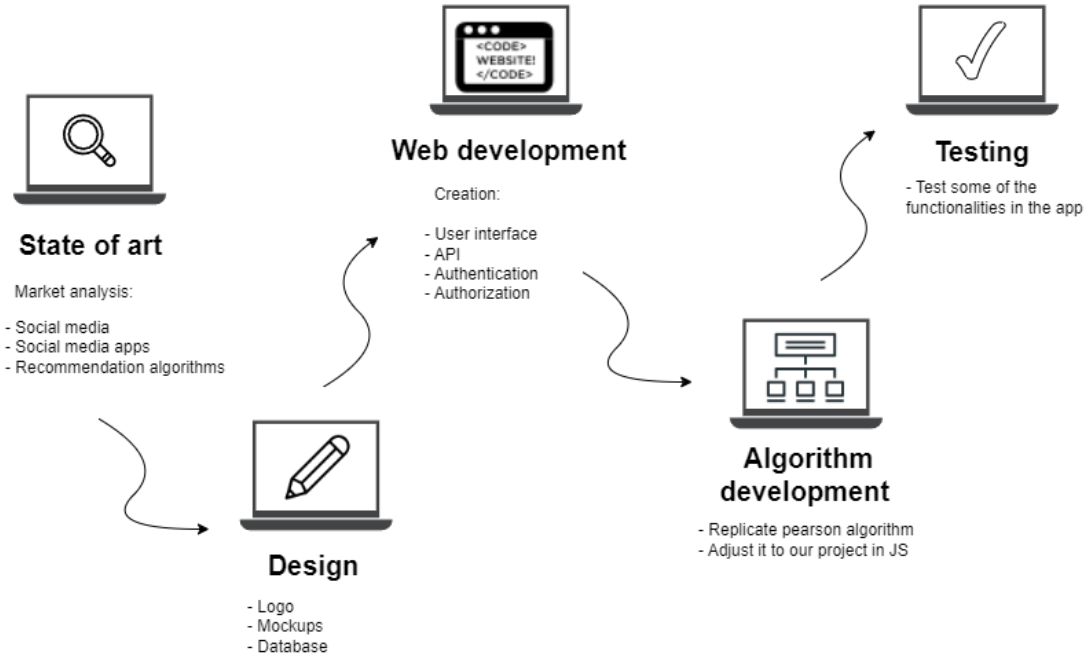

Figure 3.2: Roadmap of the project

3.2 Requirements analysis

The main purpose of the project is to create a website with a social media app where any user can interact with content in multiple ways, thanks to the implementation of a recommendation algorithm. In order to achieve we showcase the following requirements.

Web development wise, the main goal is to facilitate the users different ways of interaction, including liking and commenting on posts. Those likes and comments will be stored in a database so they can be used later on for the recommendation algorithm. In this project we insists on going for the most simple design to ensure we reach wide target.

Another topic to be mentioned is users must be able to upload their own posts. For this request we will create a simple API REST to display all posts as well as creating new ones. Nevertheless, the user needs to be registered in the web page in order to access to all these functionalities. This personal profile will display their personal information as well as followers and profile views.

It is important that the user is able to navigate to followers and non followers profiles so any content in this app can be accessed at any given time.

The order in which the content is displayed in the website is crucial for an optimal result in retaining and engaging as much as possible with users. Therefore even though we want to keep it simple with the recommendation algorithm, it needs to efficient with the posts displaying and use data from all users in the app.

This recommendation algorithm needs to adapt to the change of user behaviour and actions throughout time. Apart from these, the web page will have very little restrictions so users can feel comfortable with its usage.

With that being said, hereby the tasks that should be accomplished by the end of the project:

- **Login and registering:** The user should be able to log in inside the app if the account has been previously created. Otherwise via the register form the user can create one with the possibility of including a profile picture.
- **Creating a new post:** Once logged in, the user should be able to create a new post with the possibility of adding a picture on it.
- **Adding and removing friends:** The user should be able to interact with the icon next to each user's to follow or unfollow him/her. Seeing this change below the ad section and in the current user's profile.
- **Profile info:** The user should be able to see his/her information when entering the profile not only from the current user but also from the rest, showing each user's posts depending on the profile.
- **Likes and comments:** The user should be able to interact with every post in the app by adding or removing likes, as well as visualising the comment section of each post.
- **Dark mode:** The user should be able to choose between light and dark mode and the interface should adapt by changing the colours accordingly.
- **Content displayed:** The user should be able to visualise the content in the correct order based on his followers and post interactions. This order will be determined by the algorithm choosen.
- **Responsiveness:** The user should be able to navigate through the app with no interface issues, no matter the screen size of the device used, from mobile phone to desktop view.

3.3 Design

3.3.1. Project architecture

As explained previously, to elaborate the web side we will implement the MERN full-stack. It is based on a Client-Server architecture, where React.js will be used on the Client side and Express.js will be used on the Server side as a bridge between the client and the server, among Node.js. Finally MongoDB will be used for the database. Below we have a summary picture of the project's architecture:

3.3.2. Database

MongoDB has been the tool chosen for the database due to its easy and simple implementation. It is divided in different tables as shown in the diagram below (Figure 3.4).

On one hand we have the user model, which includes the id, name, last name, friends (list of users), email, password, profile picture, location, occupation, profile views and impressions. With all this fields we look for a general view of the user to display later on the app. For each one of these users there is a friends list, the people that the user follows, which contains the basic info of each one of them to show later on the user feed.

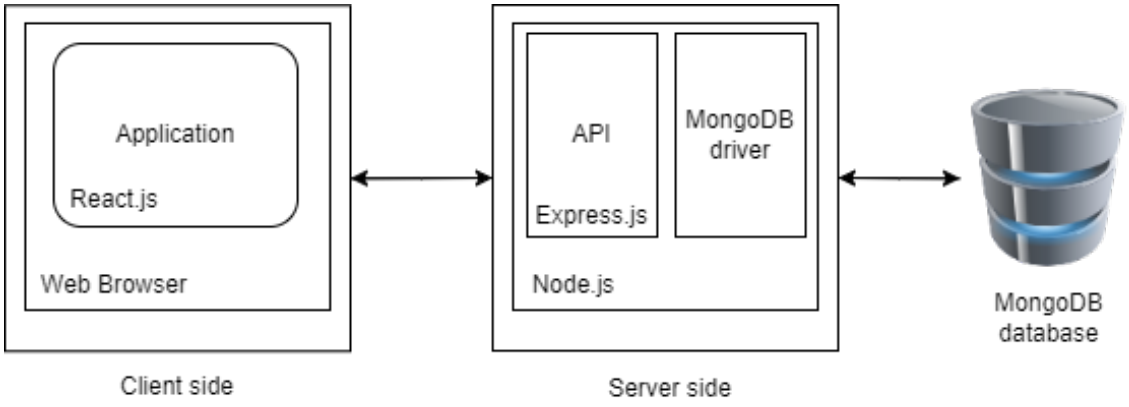

Figure 3.3: Project architecture

On the other hand we have the post model, which includes the id of the post as well as the one from the user posting it, the user's name and last name, the location of the

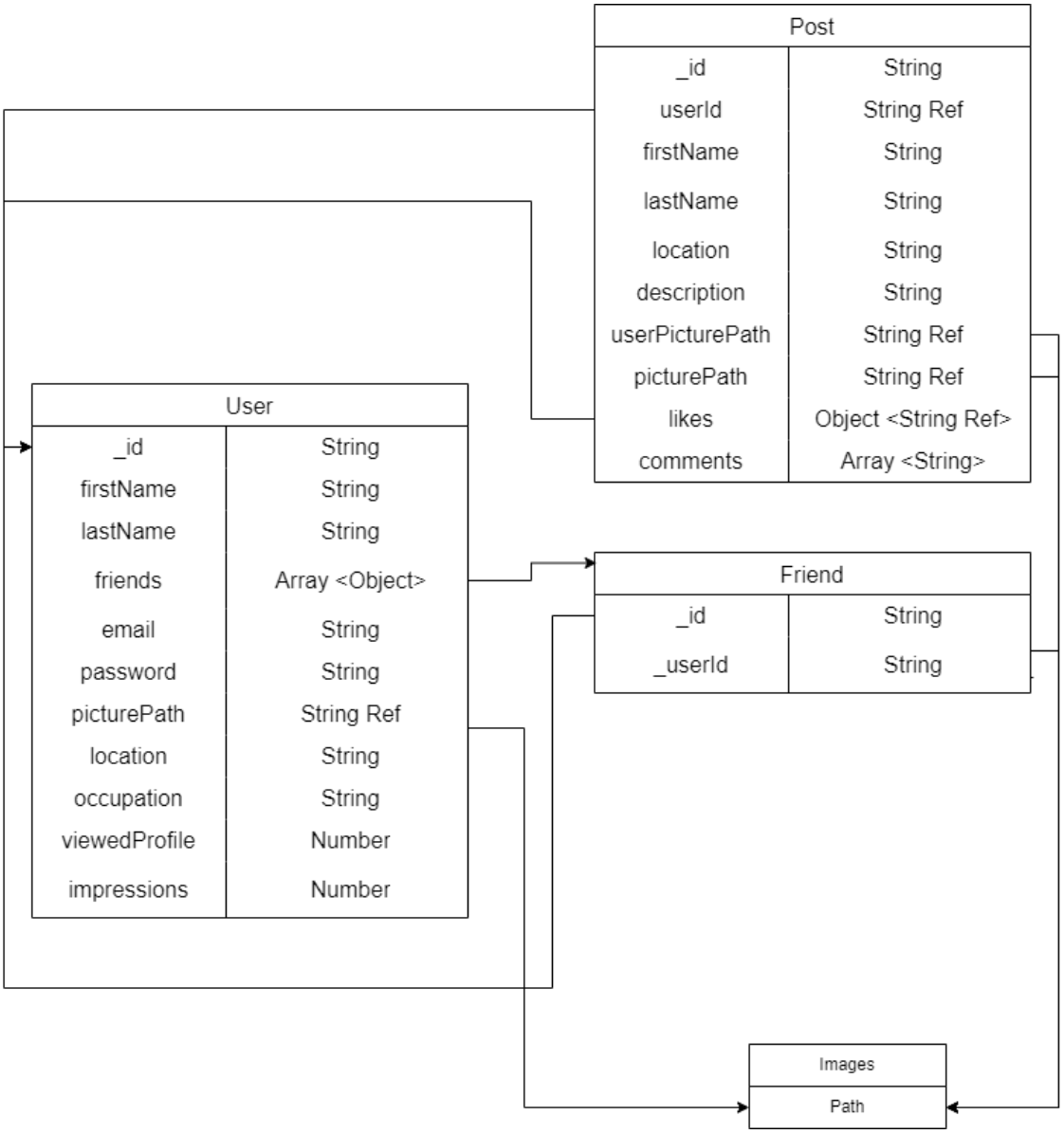

Figure 3.4: Database design

post, a description, likes, comments and a picture path in case the user decides to upload it.

It is important to clarify that the likes in the post model are defined as Map, a data structure used to hold key-value pairs of any datatype. By doing this we can get the id of every user and pair it with a boolean to determine not only the amount of likes of every post but also who likes it. This was necessary in order to properly develop the algorithm later on.

3.3.3. Mock ups and logo

Before we start designing the interface it is important to decide on the colour palette to be used throughout the whole project. Since many social media apps are based on blue colours, I decided to move towards a scale of greens, since it is frequently associated with nature, safety, and reliability; but also keeping a neutral blue present for smaller details (Figure 3.5).

Figure 3.5: Colour palette

Once the final version of the colour palette is established, it is time to define the web page logo (Figure 3.6). Since both light and dark mode have been implemented in the website, the lighter green has been the one chosen for the logo since it adapts to both environments quite well.

Figure 3.6: Logo

Now that the colours and logo have been set, the format of the web page is next. I decided to prioritize the desktop view since I consider it more adjustable to the needs we want to cover with this app (Figure 3.7).

To start, every user that wants to access the app has to go through the login/register page(Figure 3.8). Based on the picture, all it is needed is to fill in with basic information,with the possibility of adding a profile picture. If the user has already an account created the link at the end can take him/her back to the login form and enter with email and password.

Once the login process is made, the user access to the main page (Figure 3.9). On top we have the navigation bar component, including the most common features on a social

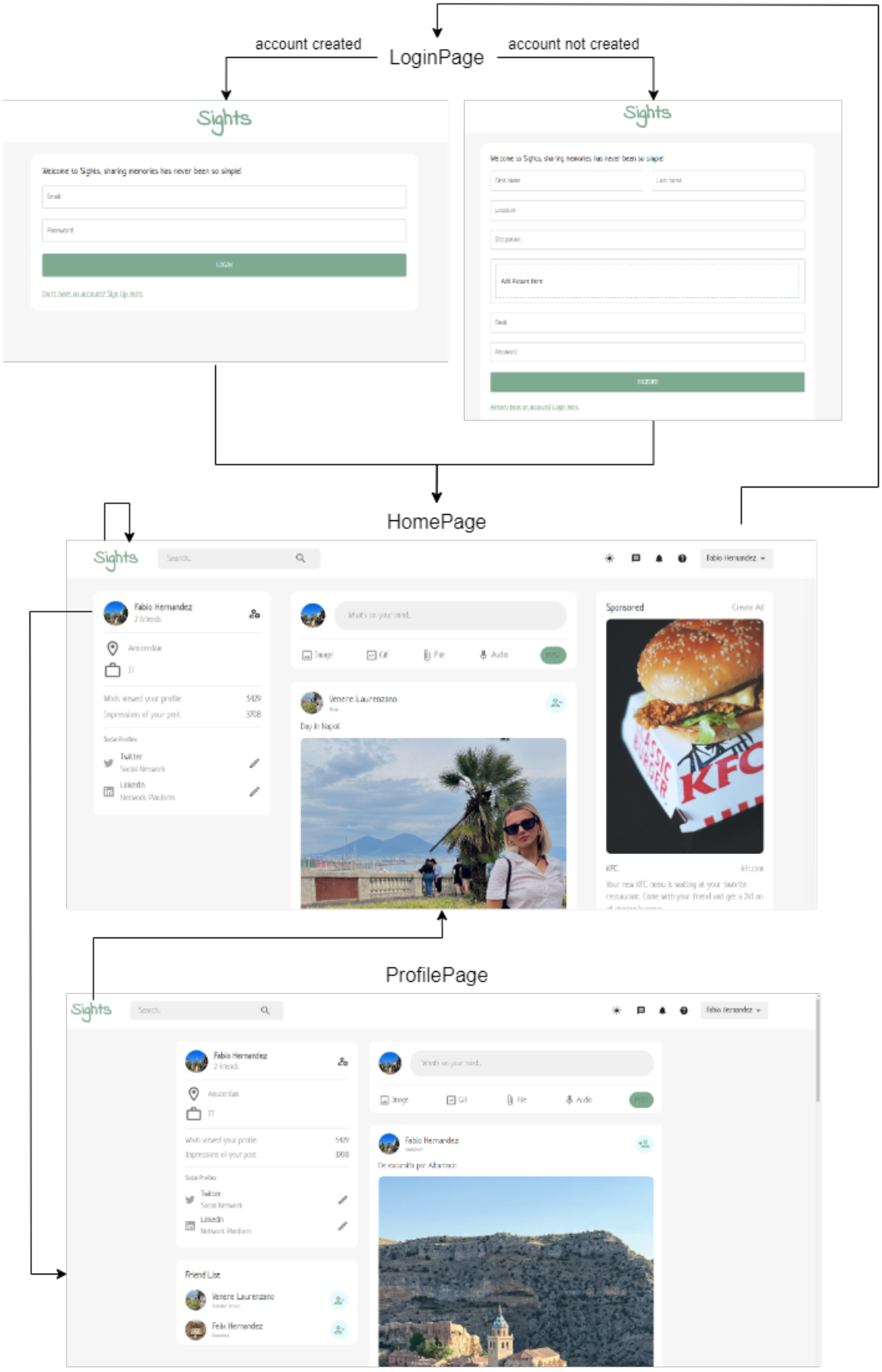

Figure 3.7: Flow diagram

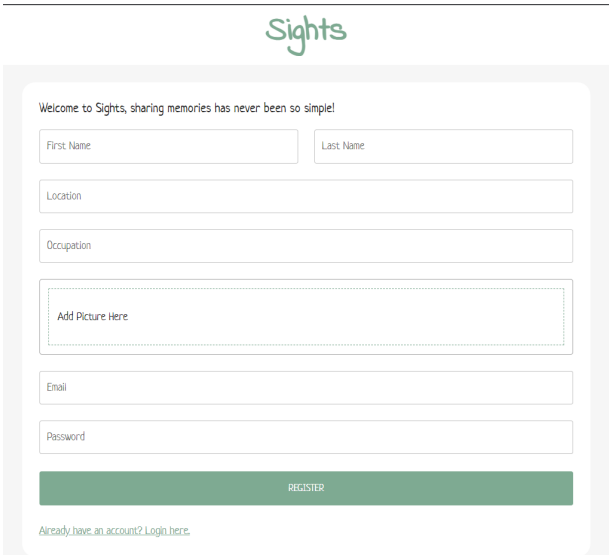

Figure 3.8: Login/register page

media app; and from left to right we see the profile component, displaying the info of the user logged in; the new post form, to create a new post; the posts feed, where we can check and interact with other users' content; and finally the sponsored ads and the friends list of the user.

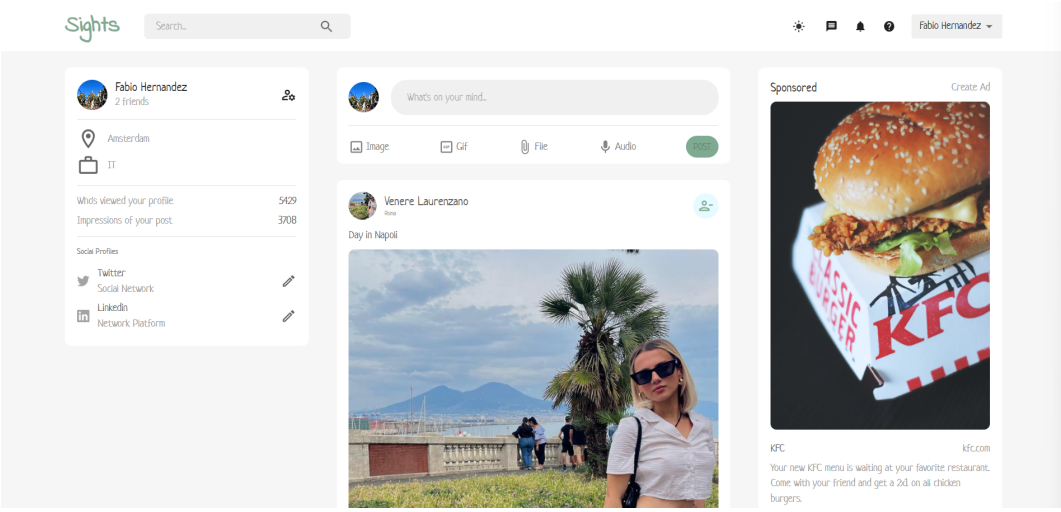

Figure 3.9: Home page

Despite being a desktop app mainly, the mobile view has also been implemented successfully (Figure 3.10), transforming the navigation bar into a expanded menu and displaying the components vertically one after the other.

Lets focus now on each component of the UI. Thanks to React every part of this interface has been designed and develop so it could be reuse or escalated. The navigation bar (Figure 3.11) for instance contains some of the most used functionalities in any social media app. From left to right we can see the logo of the web page and the search bar. On the right side we see the light/dark mode, as well as the chats, notification and contact page. Finally we have the user's name and the option to log out.

Placing ourselves in the home page (Figure 3.9), on the left the profile of the user logged in is displayed. If we click on the name or picture it will navigate to that user's profile (Figure 3.12). As shown in the image below, the user has access to his/her infor-

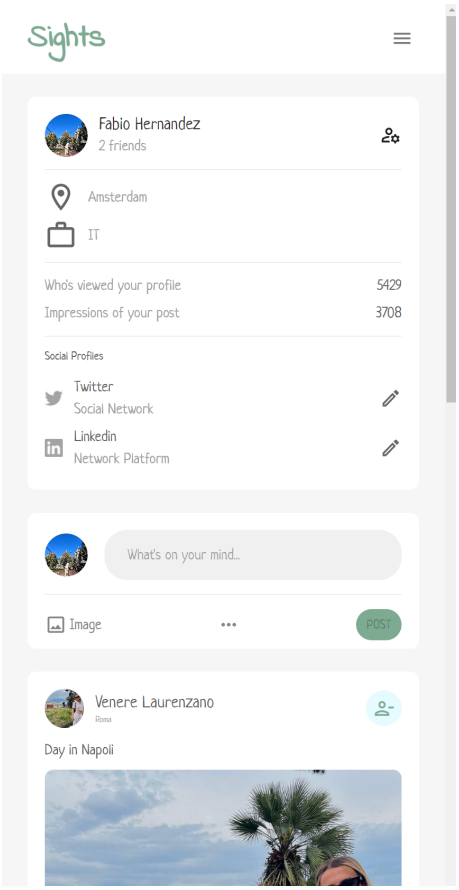

Figure 3.10: Mobile view

mation, friends list and content posted. Editing info is not available on this prototype's phase.

Going back to the main page, we have the new post form on top (Figure 3.13). The user has the possibility of typing any kind of message or caption as well as adding a picture to it. The post button will remain disabled until it detects there is something to be posted. Adding GIFs, files or audios is not available on this prototype's phase.

Scrolling down we find the posts feed (Figure 3.14). The user can interact with all content posted from other users in the app. That includes liking, viewing comments, following or unfollowing users and visiting other users' profiles. Commenting on posts is not available on this prototype's phase, only viewing them. The order the content is displayed on is decided by the recommendation algorithm implemented.

Last but not least we have the sponsor ads and the user's friend list on the right (Figure 3.15). As many social media apps nowadays include ads, Sights will incorporate some as well. In order to get some attention from the user on this right side of the screen, the friend list has been placed in between. The user can check who follows and remove any user from the list. This section of the app is not displayed on the mobile version.

Figure 3.11: Navigation bar

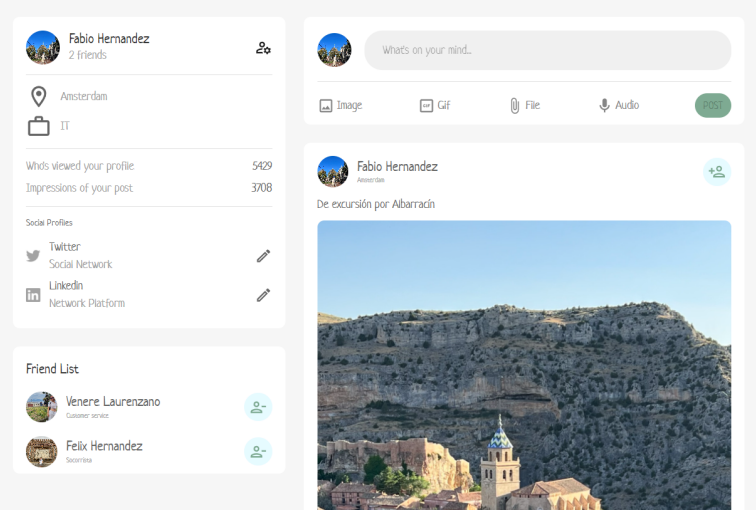

Figure 3.12: Profile page

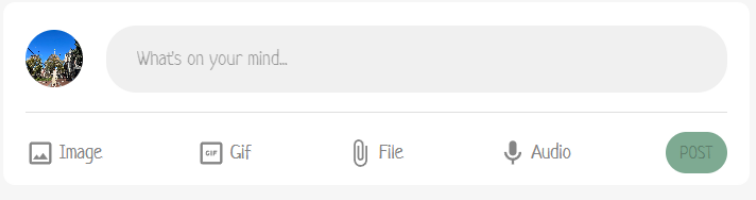

Figure 3.13: New post component

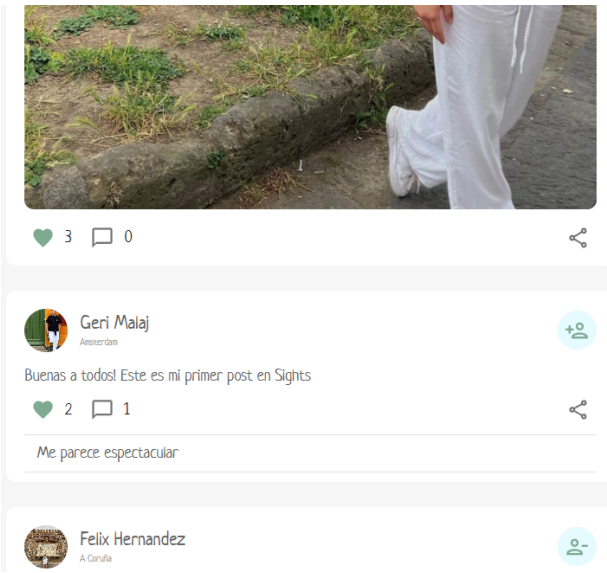

Figure 3.14: Posts feed

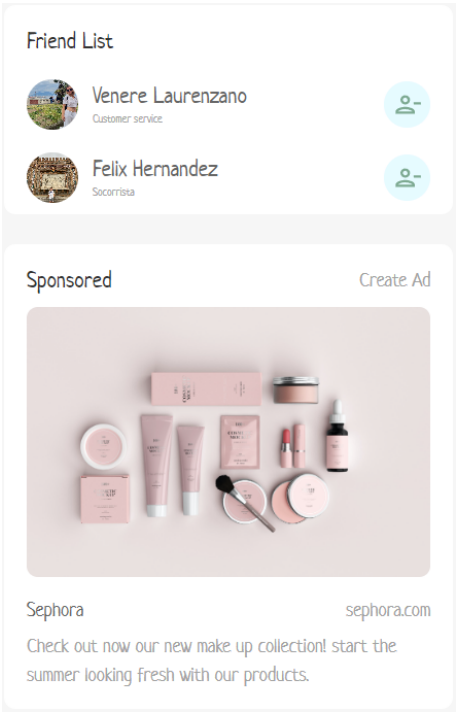

Figure 3.15: Ads and friend list section

CHAPTER 4
Web implementation

As previously explained, during the development phase we achieve a minimal viable product. Therefore a progressive creation has been done based on small improvements of each component. As a result, a general view of the project is built and right after all functionalities designed get implemented accordingly.

4.1 Front-end

In this section the implementation of all the mock ups will be done according the different screen sizes. Since React is being used, it is quite easy to adapt the content to any display format.

As React uses Javascript exclusively, there is no need to include HTML in the project. As for the styles we used a CSS template to define the colours and the font, storaged in variables and export them so they can be used in other files.

The implementation with React allows to dynamically update the interface on any change without reloading it, which highly improves the user experience. Also the usage of external libraries like Material UI [28], to simplify the styles of the components; Redux [29], to pass data around the app components; ReactRouter [30], to allow navigation without refreshing the page; and Formik [31], which facilitates the creation of forms and the fields' validation.

Navigating more in depth into the front-end, it has been divided into reusable components as explained in the design chapter. These components are displayed in multiple parts of the app. The navigation bar itself is a good example, appearing in both profile page and home page. Inside of it, the user can return to the home page by clicking on the app logo, enabling or disabling the dark mode and logging out from the app.

The main purpose of the profile component is exclusively to display the information of the user. Inside of the profile page the user is also able to create a new post as well as checking the ones posted by that user.

Lastly the post component allows the user to interact not only by liking or seeing the comments but also following or unfollowing the user. The user is also able to navigate to the post author profile by clicking on the username on top of the post.

Overall the main focus is based on a simple interface that involves any age target on a easy going experience.

4.2 Back-end

As explained in the project architecture, we will use Node.js and Express to implement the server side. The fact of using the same programming language as the front-end facilitates the integrity of the app. The usage of Express.js adds improved features such middleware and routing and it is really easy to connect with a MongoDB database.

Also some external libraries were imported such as JWT [32], to authenticate the user data; Multer and GridFs [33], to upload files and storage them efficiently; and yup, which in combination with formik validates the input values through regular expressions.

Navigating more in depth into the back-end, the index contains all the project configuration: path urls, file storage, app routes and database connection.

For the app routing we used Express, allowing the app to respond to different HTTP requests for various URLs or endpoints. By that, the app routes has been defined and associated with specific actions or functions to execute when a request regarding a post or a user is being made. Moreover, it is also used as a middleware to perform the login task.

Focusing on the logging, JWT has been used to properly hash and encrypt the user password to verify the authenticity of the user (Figure 4.1)

Logic wise, there are functions to get the current user, the user that he/she follows and modify the followers list. regarding the posts, there are methods to create a post, display them in the feed, get the posts from an specific user and liking a post.

Overall the main focus was to achieve a simple back-end structure, providing user authentication as well as an efficient API to manages the HTTPs requests.

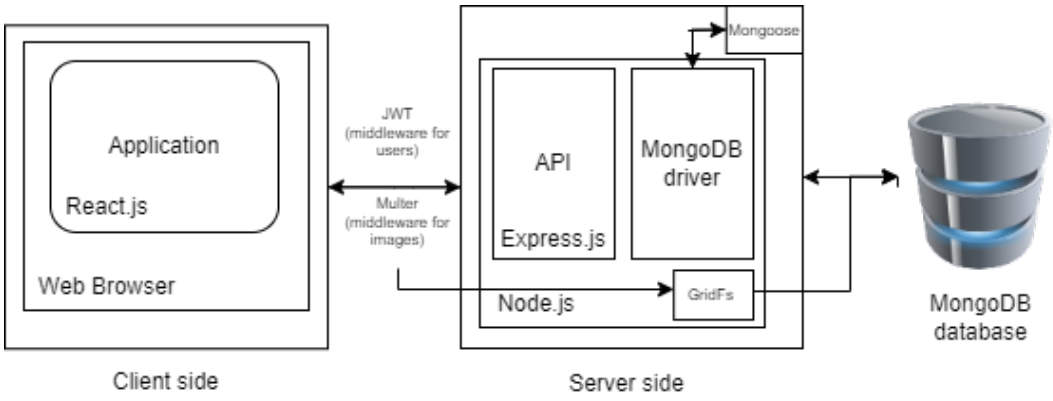

Figure 4.1: Back-end architecture

4.3 Data sample

MongoDB has been the tool chosen to implement the database design since there are no relations at all and the schemas designed are quite simple. ODMs [34], like Mongoose for Node.js, provide developers with an abstraction layer that allows them to work with MongoDB using the familiar paradigms of object-oriented programming, such as defining classes and mapping them to database documents. ODMs help simplify tasks like data validation, querying, and schema management when working with MongoDB.

In this project two ODMs have been established as shown in the design chapter 3.4). The user one allows us to determine all the fields a user in the app should have as well

as extra conditions each one of the should follow. To point out the friends field, each user has an array of users that are followed in the app, obtaining the id as a friend and the own user id.

The post one specifies how each post in the app is structured. Among all the fields contained, it is good to mention that specifying the likes field as an object allows us to have a reference of which user likes the post through a boolean condition.

4.4 Tests

With the purpose of testing and verifying the work done in this project has been achieved successfully, various validating tests have been carried out. It is important to verify a functionality when added to properly track the correct development before jumping into the next one. For that reason, some functional tests have been established using Jest [35], a quite powerful tool estandarized in the Javascript world to elaborate unit tests.

4.4.1. Login functional test

The firsts' one purpose is to verify that the app recognises the user signed in. After mocking the user model, we call the `getUser` function using Jest's describe function. It allows us to define two individual test cases, the first one being through `User.findById` method returning the user declared as mock example; and the second one returning an error in case the user cannot be returned.

In summary, this Jest test suite is used to thoroughly test the behavior of the getUser function under different conditions (success and error scenarios) by mocking the User module and simulating HTTP responses using mock objects

4.4.2. Display posts functional test

Secondly, there is one to check if the app is displaying all posts in the feed. After mocking the post model, we call the `getFeedPosts` function using Jest's describe function. It allows us to define two individual test cases, the first one being through `Post.find` method returning the two posts declared as mock example; and the second one returning an error in case the posts cannot be returned.

In summary, this Jest test suite is used to thoroughly test the behavior of the getFeed-Posts function under different conditions (success and error scenarios) by mocking the Post module and simulating HTTP responses using mock objects.

4.4.3. Like functional test

Lastly, there is another test to check if the boolean of liking a post in the app works accordingly. After mocking the post model, we call the `likePost` function using Jest's describe function. It allows us to define two individual test cases, the first one being tin a scenario where the like status of a post is toggled successfully, taking into account an id and a likes property with a map. This mock post will be returned when `Post.findById` is called. The second one returning an error in case the likes in the post cannot be traced.

In summary, this Jest test suite is used to thoroughly test the behavior of the likePost function under different conditions (success and error scenarios) by mocking the Post module and simulating HTTP responses using mock objects.

4.5 Pearson algorithm implementation

In this section we will discuss the implementation of the chosen algorithm in our app. In the state of art chapter we selected the Pearson correlation algorithm since it fits best with the goals proposed.

Since we are building a prototype, it was more convenient to aim for a more simple yet efficient algorithm. As discussed previously we will carry out an algorithm that suggests users based on Pearson's correlation value. On the image below the code is displayed, followed by a brief explanation of it [20].

The code is divided in two functions: `Pearson()` and `recommendPosts()`.

Regarding `Pearson` function it calculates and returns the Pearson's correlation value based on the number of likes of the posts of the user logged in and the number of likes of the posts of the rest of users (Figure 2.1).

Regarding `recommendPosts` function, it takes two arguments: userPosts and followers, representing the posts made by different users and the followers of the current user, respectively. We first get the total posts number and then get the sum of likes of each post. After we calculate the sum square and the product following the equation, we conclude by defining the numerator and denominator resulting in the correlation value.

To start, the `recommendPosts` function uses a loop to iterate through each user's posts in the userPosts object. It excludes the current user's posts by checking if the key (user ID) is not equal to userId. For each user's posts, it calculates the number of likes for the current user's posts (likesUser) and the likes for the posts of the other user (likesOtherUser). This is done using the Object.keys(p.likes).length expression, which counts the number of likes in each post.

The Pearson correlation coefficient (correlation) is calculated using the Pearson function. This coefficient measures the similarity of likes between the current user and the other user. A positive correlation indicates similar liking patterns. If the correlation is positive (greater than 0), the code checks if the other user is a follower of the current user. If they are a follower, a weight of 1.5 is assigned to the correlation. Otherwise, a weight of 1 is used. This step implies that followers' correlations have more influence on recommendations.

Then, for each post by the other users, the correlation is assigned based on the calculated correlation and weight. The post's correlation value is stored in the correlation property within the post object. Afterwards, they are added to the recommendations array, each with their assigned correlation value.

If the correlation is not positive (less than or equal to 0), it means there is no significant similarity in liking patterns. In this case, the posts from the other user are still added to the recommendations array, but with a correlation value of 0.

Overall, the algorithm is attempting to generate recommendations for posts based on the correlation of liking patterns between the current user and other users, considering the user's followers' influence. It uses a weighted correlation to prioritize posts from followers with a higher correlation and assigns a correlation value to each recommended post.

Lets proceed with some examples to have a better understanding:

- **Case 1:** for case 1, we have the posts A, B, C, and D, and the users 1, 2, 3, and 4.

  User 1 has liked A and B, user 2 has liked B and D, user 3 has liked B and C, and user 4 has liked D (for example).

```
const Pearson = (LikesA, LikesB) => {
  const NumberLikes = LikesA.length
  const SumA = LikesA.reduce((acc, n) => n + acc), SumB = LikesB.reduce((acc, n) => n + acc)
  const SumSquareA = LikesA.reduce((acc, n) => acc + Math.pow(n, 2), 0), SumSquareB = LikesB.reduce((acc, n) => acc + Math.pow(n, 2), 0)
  const SumProduct = LikesA.reduce((acc, n, i) => acc += LikesB[i] == undefined ? 0 : n * LikesB[i], 0)

  const Numerator = (SumProduct - (SumA * SumB / NumberLikes)), Denominator = Math.sqrt((SumSquareA - Math.pow(SumA, 2) / NumberLikes) * (SumSquareB - Math.pow(SumB, 2) / NumberLikes))
  return !Denominator ? 0 : Numerator / Denominator
}

const recommendPosts = (userPosts, followers) => {
  const recommendations = []
  for (const [key, value] of Object.entries(userPosts)) {
    if (key !== userId) {
      const likesUser = userPosts[userId].map(p => Object.keys(p.likes).length)
      const likesOtherUser = value.map(p => Object.keys(p.likes).length)
      const correlation = Pearson(likesUser, likesOtherUser)

      if (correlation > 0) {
        const weight = followers.filter(f => f._id == key).length ? 1.5 : 1
        value.forEach(e => {
          e.correlation = weight * correlation
          recommendations.push(e)
        });
      } else {
        value.forEach(e => {
          e.correlation = 0
          recommendations.push(e)
        });
      }
    }
  }
  recommendations.sort((a,b) => b.correlation - a.correlation )
  return recommendations
}
```

Figure 4.2: Pearson correlation algorithm in code

With this data, you calculate the Pearson correlation between user A and the other three users. To calculate the Pearson correlation coefficient between user A and the other three users (2, 3, and 4) based on likes on posts, we first need to represent their like data in arrays. Then, we will use the 'Pearson' function to calculate the correlation. Here are the calculations:

Like Data:

```
- User 1: [1, 1, 0, 0] (A, B, C, D)
- User 2: [0, 1, 1, 0] (A, B, C, D)
- User 3: [0, 0, 1, 0] (A, B, C, D)
- User 4: [0, 1, 0, 1] (A, B, C, D)
```

1. Pearson(User 1, User 2):

```
- LikesA: [1, 1, 0, 0]
- LikesB: [0, 1, 1, 0]
```

Calculating the Pearson correlation coefficient:

- $Numerator = (1*0+1*1+0*1+0*0) - ((1+1)*(0+1)/4) = 1-(2/4) = 1-0.5 = 0.5$
- $Denominator = \sqrt{((1^2+1^2+0^2+0^2) - ((1+1)^2/4)) * ((0^2+1^2+1^2+0^2) - ((0+1)^2/4))} = \sqrt{(2-1)*(2-0.25)} = \sqrt{1*1.75} = \sqrt{1.75}$

$PearsonCorrelationCoefficient = Numerator/Denominator = 0.5/\sqrt{1.75} \approx 0.37796$

2. Pearson(User 1, User 3):

```
- LikesA: [1, 1, 0, 0]
- LikesC: [0, 0, 1, 0]
```

Calculating the Pearson correlation coefficient:

- $Numerator = (1*0+1*0+0*1+0*0) - ((1+1)*(0+0)/4) = 0-(2/4) = 0-0.5 = -0.5$
- $Denominator = \sqrt{((1^2+1^2+0^2+0^2) - ((1+1)^2/4)) * ((0^2+0^2+1^2+0^2) - ((0+0)^2/4))} = \sqrt{(2-1)*(1-0)} = \sqrt{1*1} = 1$

$PearsonCorrelationCoefficient = Numerator/Denominator = -0.5/1 = -0.5$

3. Pearson(User 1, User 4):

```
- LikesA: [1, 1, 0, 0]
- LikesD: [0, 1, 0, 1]
```

Calculating the Pearson correlation coefficient:

- $Numerator = (1*0+1*1+0*0+0*1) - ((1+1)*(0+1)/4) = 1-(2/4) = 1-0.5 = 0.5$
- $Denominator = \sqrt{((1^2+1^2+0^2+0^2) - ((1+1)^2/4)) * ((0^2+1^2+0^2+1^2) - ((0+1)^2/4))} = \sqrt{(2-1)*(2-0.25)} = \sqrt{1*1.75} = \sqrt{1.75}$

$PearsonCorrelationCoefficient = Numerator / Denominator = 0.5 / \sqrt{1.75} \approx 0.37796$

These are the Pearson correlation coefficients between user 1 and users 2, 3, and 4 based on their like data on posts. Each coefficient represents the similarity in their like patterns, where a positive value indicates a positive correlation and a negative value indicates a negative correlation. In this case, user 1 has the same value with user 2 and 4, therefore any of both posts will be prioritised on user's 1 feed.

- **Case 2:** For case 2, we will use the same example but the user 1 follows user 2. Therefore the final correlation value is multiplied by the weight: $0.37796 * 1.5 = 0.56694$.

  Now the correlation value for with user 2 is 0.56694 and with user 4 is 0.37796. The post from user 2 will be displayed to user 1 before the one from user 4.

CHAPTER 5
Conclusions

In this chapter the project results will be enumerated, taking into account which goals have been completed during the development process and which areas will be improved as future work.

As shown throughout the document, all goals proposed in the introduction have been successfully accomplished.

Moreover, both the social media app as well as the recommendation algorithm have been carried out smoothly, which were the primary objectives to accomplish. Thanks to this, everyone will get recommended posts within the app based on their shared interests, allowing them to engage with others by commenting on or liking posts.

Furthermore, part of his accomplishment can be attributed to the analysis of current technologies, including some that were new to us. This is why we conducted a study of the current state of the art, with a focus on the most common social media apps. We analyzed their success and also compared the various algorithms they employed to maximize user engagement.

Security was also a critical component that needed to be implemented. This is why the user must have a registered account in order to enter the app. This is also needed to properly track the behaviour of each interaction and get the most out of the algorithm.

Finally, this was an excellent opportunity to showcase the acquired knowledge throughout the degree and even to learn new tools and technologies. Some of the courses implemented in this project are Programming 2, Web development project and UI/UX design. In fact, even new skills such MongoDB and React were learnt as well as some external libraries such Formik.

5.1 Future work

In this section we will discuss the elements that were left pending, the ones planned as future work and the ones that have margin of improvement. By doing this, we not only enhance our service but also expand the range of functionalities and tools within our project.

As we all know, creating a social media app gives you endless possible functionalities to implement and scale it up as much as desired. For this reason I had to be cautious and decided to include only the must needed features to achieve a simple yet efficient and easy-going interface. Functionalities such profile editing or chatting between users are next in the list to be implemented and improve even more the user engagement of the app; same with sharing different files formats such video or audios. Also, it is a future

work to add the functionality of searching users inside the app in order to improve the app-flow and maximise the engagement.

Regarding the algorithm, it has been designed to recommend posts but it could even be scaled up to prioritise taking comments and shares into account as well. Moreover, the implementation of additional algorithms such as feed ranking or ad-targeting algorithms will allow us to have a better understanding of what each user wants and provide the most matching content for them.

5.2 Sustainable Development Goals

The Sustainable Development Goals (SDGs), also known as Global Goals, were adopted by the United Nations in 2015 as a universal call to end poverty, protect the planet, and ensure that by 2030, all people enjoy peace and prosperity.

The 17 SDGs are interconnected: they recognize that action in one area will affect outcomes in other areas and that development must balance social, economic, and environmental sustainability. Countries have committed to prioritizing the progress of the most disadvantaged.

The SDGs are designed to eradicate poverty, hunger, AIDS, and discrimination against women and girls.

Creativity, knowledge, technology, and financial resources from all of society are needed to achieve the SDGs in all contexts.

In this project we aimed for number 10, trying to battle the market and commerce inequality reinforcing the power of social media to allow any user in the world to access to this app and enjoy the content and share their experience as any other person in the planet.

APPENDIX A
User manual

The project can be found on the following GitHub link: https://github.com/fbs11/sights. Inside, you will find both the client and the server side.

If the web app wants to be executed, these steps need to be followed. In all of them the user needs to open the command prompt on his operating system and be located the projects root.

In the first command prompt we enter the server folder by typing `"cd server"`. Once inside the server folder, we type `"nodemon start"` and the server should be executed successfully.

In the second command prompt we enter the client folder by typing `"cd client"`. Once inside the client folder, we type `"npm start"` and the webpage should open automatically on your predefined browser. In case it does not, please open the following link to access. http://127.0.0.1:3000/

If this is the first time in the app you will need to register by creating a new account, otherwise please log in with your email and password. Once inside, feel free to explore around, post your content and interact with others.

Bibliography

[1] Web 1.0 vs Web 2.0. Consulted on 24/06/2023. https://history-computer.com/web-1-0-vs-web-2-0-full-comparison.

[2] Definition of social media. Consulted on 24/06/2023. https://www.techtarget.com/whatis/definition/social-media.

[3] History of social media. Consulted on 24/06/2023. https://online.maryville.edu/blog/evolution-social-media.

[4] Social media app examples Consulted on 24/06/2023. https://www.searchenginejournal.com/social-media/biggest-social-media-sites/.

[5] Social media features. Consulted on 24/08/2023. https://sproutsocial.com/insights/social-media-features/.

[6] Advantages of social media. Consulted on 24/08/2023. https://helpfulprofessor.com/social-media-pros-and-cons/.

[7] Disadvantages of social media. Consulted on 24/08/2023. https://helpfulprofessor.com/social-media-pros-and-cons/.

[8] Facebook's success. Consulted on 25/08/2023. https://www.forbes.com/sites/gilpress/2018/04/08/why-facebook-triumphed-over-all-other-social-networks/?sh=2b71c55c6e91.

[9] Instagram's success. Consulted on 25/08/2023. https://www.linkedin.com/pulse/snapshot-success-story-instagram-stefan-tudor-murariu.

[10] Tiktok's success. Consulted on 25/08/2023. https://www.forbes.com/sites/tomtaulli/2020/01/31/tiktok-why-the-enormous-success/?sh=7ed652fa65d1.

[11] Youtube's success. Consulted on 25/08/2023. https://nofilmschool.com/YouTube-created-year.

[12] LinkedIn's success. Consulted on 25/08/2023. https://learn.marsdd.com/article/how-linkedin-succeeded-as-the-best-social-media-site-for-business-networking/.

[13] System algorithms in social media apps. Consulted on 26/08/2023. https://digitalmarketinginstitute.com/blog/how-do-social-media-algorithms-work.

[14] Types of system algorithms in social media apps. Consulted on 26/08/2023. https://sproutsocial.com/insights/social-media-algorithms/.

[15] Recommendation algorithms. Consulted on 26/06/2023. https://knightcolumbia.org/content/understanding-social-media-recommendation-algorithms.

[16] KNN algorithm. Consulted on 26/08/2023. https://www.geeksforgeeks.org/k-nearest-neighbours/.

[17] KNN algorithm example. Consulted on 26/08/2023. https://filmflix-watch.vercel.app/.

[18] Pearson algorithm. Consulted on 26/08/2023. https://www.geeksforgeeks.org/pearson-correlation-coefficient/.

[19] Pearson algorithms example. Consulted on 26/08/2023. https://www.last.fm/.

[20] Jesús Bobadilla, Francisco Serradilla y Jesus Bernal, *A new collaborative filtering metric that improves the behavior of recommender systems*, *Knowledge-Based Systems*, Vol. 23, No. 6, pp. 520-528, 2010, Elsevier.

[21] Laravel applied to a social media app. Consulted on 26/06/2023. https://builtin.com/software-engineering-perspectives/laravel.

[22] Django applied to a social media app. Consulted on 26/08/2023. https://www.simplilearn.com/tutorials/django-tutorial/what-is-django-python.

[23] Node.js applied to a social media app. Consulted on 26/08/2023. https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs.

[24] Express.js applied to a social media app. Consulted on 26/08/2023. https://www.codecademy.com/article/what-is-express-js.

[25] React.js applied to a social media app. Consulted on 26/08/2023. https://blog.hubspot.com/website/react-js.

[26] MongoDB applied to a social media app. Consulted on 26/08/2023. https://www.techtarget.com/searchdatamanagement/definition/MongoDB.

[27] Scrum methodology. Consulted on 26/08/2023. https://www.scrum.org/resources/what-scrum-module.

[28] Material UI documentation. Consulted on 26/08/2023. https://mui.com/.

[29] Redux documentation. Consulted on 26/08/2023. https://react-redux.js.org/.

[30] React router documentation. Consulted on 26/08/2023. https://reactrouter.com/.

[31] Formik documentation. Consulted on 26/08/2023. https://formik.org/.

[32] JWT documentation. Consulted on 26/08/2023. https://jwt.io/.

[33] Multer documentation. Consulted on 26/08/2023. https://www.npmjs.com/@types/multer-gridfs-storage.

[34] Object Data Modeling. Consulted on 26/08/2023. https://link.springer.com/article/10.1007/BF03037506.

[35] Jest documentation. Consulted on 26/08/2023. https://jestjs.io/.