SociablePeak

A SOCIAL MEDIA WEB PLATFORM FOR PERSONALIZED CONNECTIONS AND CONTENT THROUGH GRAPH TRAVERSAL AND FILTERING

ABSTRACT

In the evolving landscape of digital communication, existing social media platforms often lack features that prioritize personalization, user engagement, and administrative control. This project addresses that gap by developing SociablePeak, a modern social media web application tailored to meet the dynamic needs of users, businesses, and administrators. The platform was constructed in Laravel with MySQL implementing the MVC architectural pattern and the Incremental Software Development Model that guaranteed the development of modules that would be scalable in nature. The most notable features are post scheduling, AES 256 secured messaging, tracking profile views, suggestion of friends based on graph traversal algorithms, and delivery of ads based on content-based filtering. Also, a caption-creating service based on AI was built with Gemini API to encourage the creative idea of users. Tracing of the stability and successful platform capability to offer a feature-rich user experience was proven through wide-ranging unit testing and system tests. The outcomes reveal that SociablePeak is able to fill the void caused by older applications, delivering smart engagement, safe communication and analytical knowledge. As SociablePeak continues to get improved with mobile responsiveness, cloud deployment, and greater AI capabilities, the potential of sociably peak as a next-generation social media solution also improves significantly.

Keywords: *Social Media Platform, Laravel, Graph Traversal, Content-Based Filtering, AES Encryption, Post Scheduling, AI Caption Generator, Profile Analytics, MVC Architecture, Web Application, SociablePeak, User Engagement.*

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AES-256	Advanced Encryption Standard
AI	Artificial Intelligence
API	Application Programming Interface
BFS	Breadth-First Search
CASE	Computer-Aided Software Engineering
CBC	Cipher Block Chaining
CI/CD	Continuous Integration/Continuous Deployment
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IV	Initialization Vector
JSON	JavaScript Object Notation
MVC	Model-View-Controller
MySQL	My Structured Query Language
NLP	Natural Language Processing
PHP	Hypertext Preprocessor
SNS	Social Network Sites
UAT	User Acceptance Testing
UI	User Interface
XAMPP	Cross-platform, Apache, MySQL, PHP , Perl
2FA	Two-Factor Authentication

CHAPTER 1: INTRODUCTION

1.1. Introduction

In the modern digital age, social media has become an essential part of individuality all around the world. Social media has completely changed the way we communicate, connect, and consume information in the modern digital age [1] .With billions of users worldwide, these platforms help people communicate and consume media on a daily basis. The increasing number of users and the amount of time spent on these platforms demonstrate their significance and impact in the current digital era. As the digital landscape evolves, there is a continuous demand for innovative features that meet the diverse user needs and enhance the overall social media experience.

SociablePeak is developed to meet the demand of users and aim to deliver a smarter, more balanced social media experience. Unlike traditional platforms, SociablePeak introduces features which were developed to modern digital habits. One such feature is post scheduling, which allows users to plan and automate their content uploads which is particularly useful for those managing busy routines.

What sets SociablePeak apart is t carefully combines analytics-driven insights, content sharing, and real-time interaction into an easy-to-use interface. The platform supports secure chatting, profile management, post analytics, profile view tracking, friend suggestions based on mutual connections (using graph traversal), and personalized content recommendations (via content-based filtering). For businesses, it offers advertising and audience engagement tools, while administrators are provided with robust moderation and reporting systems.

SociablePeak is not just another social media platform, it is a whole ecosystem created to improve awareness and interaction, not just another social media site. SociablePeak wants to reshape the norms of contemporary social networking by emphasizing usability, personalization, and digital health. This will make it more effective, secure, and ultimately more significant for administrators, users, and businesses.

1.2. Problem Statement

In today's digital age, social media plays an important role in our day-to-day communication and social interactions. This platform has transformed the way we connect, share and engage with people. However, despite their heavy influence, many social media have various problems due to which they fall short to provide efficient user experience.

- Unable to automatically schedule posts.
- Minimal Insights into profile engagement.
- Lack of tools to analyze user behavior, track growth for admin.
- Inadequate support for businesses to track engagement on advertisements.

The proposed solution for this problem is to develop SociablePeak, a user-friendly web application that solves the issues by using various algorithms and developing various features. The platform will provide a user-friendly interface, allowing users to control their overall experience, and enhance user engagement strategy.

1.3. Objectives

The Objectives of this project are:

- To develop a user-friendly social media platform with modern features.
- To implement Post Scheduling that allows users to automate posting their content.
- To enable profile view tracking.
- To provide understanding of user engagement
- To personalized user experience through filtering
- To provide administrators with tools to monitor user and business activities

1.4. Scope and Limitation

The scope of SociablePeak project is to develop user-friendly web applications that enhance social media experiences through advanced features that are developed to meet the user needs and preference. The scope of this project includes:

- Designing and implementing features such as user and businesses registration and login, profile management, post creation.
- Integration of post scheduling that allows users to automate their post.

- Develop real time chat with encryption.
- Admin functionalities to observe user activities.
- Content-based filtering to personalized content to user.
- Use of graph traversal to generate friend suggestions.
- Provide user engagement analysis.

While developing the project, there were some limitations. The limitations are as follows:

- The platform doesn't support video content.
- The platform doesn't support external file sharing in chat.
- The mobile responsiveness is out of the current project scope.

1.5. Development Methodology

For the development of SociablePeak, The Incremental Software Development Model was used. This methodology was selected due to its iterative nature, which allows for systematic development, testing, and refinement of features in manageable phases.

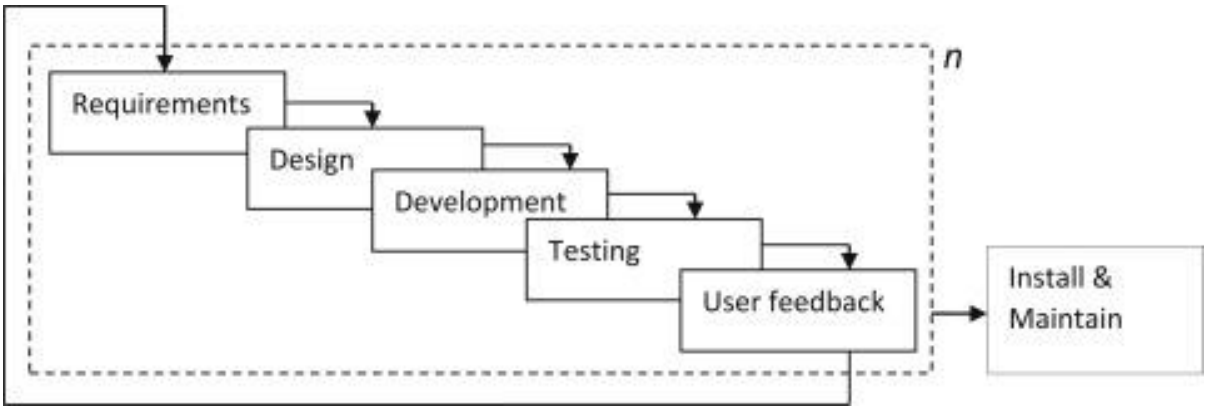

Figure 1: Incremental Model

Figure 1 represents Incremental model illustrating the system development in smaller incremental functional units where each increment is designed, developed, tested allowing new functional units to be added progressively.

1.6. Report Organization

Chapter 1: Introduction

This chapter introduces the SociablePeak project, a social media platform designed to provide a more personalized, secure and conscious digital social experience. The chapter outlines the motivation behind the project, including the need for features such as post scheduling, real-time chat, profile view tracking, and business advertisement management. The problem statement, project objectives, scope, and methodology used for development are discussed. The chapter also provides an overview of how the rest of the report is structured.

Chapter 2: Background Study and Literature Review

This chapter provides theories and relevant studies that inform the development of SociablePeak. It covers key concepts in social media platforms, digital communication, and user analytics. The literature review explores the application of graph theory for friend suggestions, content-based filtering for personalized ads, and encryption methods for securing user messages. It highlights the gaps in current social platforms that SociablePeak aims to address, such as the lack of scheduling, limited analytics, and poor personalization.

Chapter 3: System Analysis

This chapter provides a detailed analysis of system requirements and feasibility. Technical, operational, economic, and schedule feasibility assessments are conducted to evaluate the practicality of developing and deploying SociablePeak. The chapter also includes an object-oriented system analysis with diagrams such as class, sequence, and activity diagrams to model system behavior. Functional components like user interactions, ad engagement, and admin monitoring are explored in depth.

Chapter 4: System Design

This chapter describes the design architecture of SociablePeak, including refined class and component diagrams. It details the structure of major components such as the post management system, chat module, friend network, profile analytics, and ad system. The design of backend components, front-end views, and user experience flows are illustrated. The database schema

is explained, highlighting data handling for posts, messages, ads, users, and reports. Deployment and hosting considerations are also included.

Chapter 5: System Implementation and Testing

This chapter focuses on how the platform was developed using Laravel, MySQL, and supporting technologies. It outlines the implementation of key features like AES-encrypted messaging, graph-based friend suggestions, content-based ad filtering, and role-based dashboards. The chapter discusses challenges faced during development, such as data flow coordination between modules and UI responsiveness. Testing strategies for performance, functionality, and security are outlined, along with results for key modules like chat, scheduled posts, and profile tracking.

Chapter 6: Conclusion and Future Recommendations

This chapter summarizes the accomplishments and learnings from the SociablePeak project. It reflects on how the platform bridges the gap between typical social interaction and user-conscious networking. The chapter provides recommendations for future work, including mobile app development, integration of AI-driven content suggestions, advanced moderation tools, and expansion of business analytics features. It also highlights the potential for scaling the platform and improving algorithm accuracy through more refined data models.

CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

2.1. Background Study

Social media has become an essential element of digital interaction in today's world. Social media has influenced communication, entertainment, market and information sharing processes. With billions of active users globally, Facebook was the most popular social media platform in 2019. YouTube, Instagram, and WeChat followed, with over a billion users. Tumblr and TikTok came next, with over half a billion users have transformed how individuals and businesses interact [2]. The evolution of social networking sites has emphasized the importance of user engagement, personalization, and real-time connectivity, shaping the development of new platforms like SociablePeak.

Several ideas were explored for the design of SociablePeak. One of the key principles is user centric design which helps to enhance the user experience. Another core area was privacy and authentication where techniques like bcrypt hashing, and two-factor authentication are used to protect the user data. Additionally, content scheduling is a modern function which helps users to maintain their online presence.

From a technical point of view, SociablePeak uses concepts like graph traversal for friends recommendation and content-based filtering for personalized content. Graph traversal algorithms enable the identification of mutual connections, enhancing the social networking aspect, while filtering algorithms tailor content according to user behavior and preferences, increasing relevance and engagement.

Overall, this project is created by integrating various techniques for user interface design, data security, analysis of behavior and intelligent content delivery to build a robust, user friendly and mindful social media platform that meets the modern digital needs. This project meets the gaps between traditional social media by offering advanced features.

2.2. Literature Review

Social media has transformed from simple communication to complex systems that deeply influence many factors of daily life, including both personal and professional interactions. This transformation has been influenced by advanced technology and increasing user engagement, which have increased the chances of innovation to meet the user demands.

The foundational study "Social Network Sites: Definition, History, and Scholarship" by Boyd and Ellison provides a detailed historical analysis of social network sites (SNSs), charting their progression from early platforms like SixDegrees to current giants like Facebook [3]. This evolution informs the design of SociablePeak, which builds upon established social networking practices while incorporating modern enhancements such as friend suggestion through graph traversal and real-time analytics.

A study by the Pew Research Center ("Social Networking Usage: 2005-2015") documents the dramatic increase in social media usage, rising from 7% to 65% among American adults over a decade [4]. This trend underscores the widespread dependence on social media, affirming the need for platforms like SociablePeak that not only support basic social interactions but also provide deeper engagement through scheduled posts, encrypted messaging, and content customization tailored to modern user expectations.

Social media encompasses a broad range of Internet-based and mobile services that allow users to engage in online exchanges, contribute user-generated content, and participate in digital communities [5]. This evolution is driven by technological advances and an increase in user engagement, emphasizing the need for platforms to innovate continually to cater to the dynamic user demands.

Graph theory has proven to be a powerful analytical framework in understanding and modeling social media networks, where users are represented as nodes and their interactions such as friendships, messages, likes, or comments form the edges. As discussed in the paper "Application of Graph Theory in Social Media", these relationships allow platforms to analyze connectivity, influence, and community structure using metrics like centrality, clustering, and

path length [6]. Concepts such as interpersonal ties, triadic closure, and bridges help determine the strength and type of relationships among users, which are essential for building features like intelligent friend suggestions and user engagement analysis. In the development of SociablePeak, graph theory has been applied specifically to enhance the friend recommendation system through graph traversal algorithms based on mutual connections. By leveraging these graph-based techniques, the platform aims to create more meaningful and relevant connections between users while improving overall social engagement.

Content-based filtering has become a prominent recommendation technique in social media platforms, enabling personalized content delivery based on individual user preferences. This approach analyzes user activity such as likes, clicks, and interactions to construct a user profile that reflects specific interests [7]. The system then recommends content with similar characteristics, ensuring relevance and improving engagement. Unlike collaborative filtering, content-based filtering does not depend on the behavior of other users, making it suitable for niche recommendations and privacy-conscious applications. Despite limitations in handling novel or diverse interests, its integration with feedback loops and natural language processing enables adaptive learning over time. In the context of SociablePeak, this technique is applied to deliver personalized advertisements and post suggestions, helping users discover content that aligns with their activity while also increasing satisfaction and platform retention.

2.3. Current System

There are various popular social media platforms accessible today that provide varying features for online interactions, content exchange, and networking. While these platforms have great functionalities, they lack some personalized, analytical, and well-being-related features that modern users crave. Some of the current systems relevant to the features provided by SociablePeak are discussed below:

2.3.1 Facebook

Facebook is one of the most widely used social networking platforms that allows users to connect with friends, share posts, send messages, and join communities. SociablePeak is highly

influenced by Facebook and has various similar feature like post creation, friend requests and suggestions, group and private messaging, business pages and ad posting.

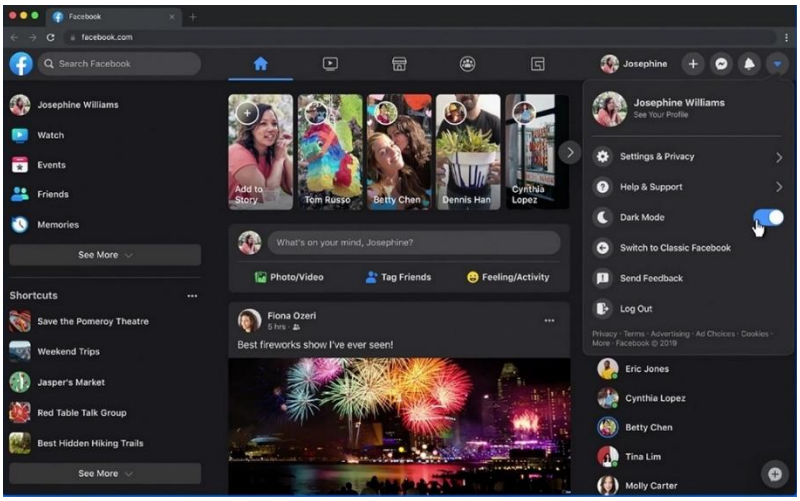

Figure 2 [8]: Facebook

2.3.2 Instagram

Instagram is a visual-first platform owned by Meta that allows users to share images/videos, follow others, and interact through likes and comments. SociablePeak has many similar feature like Instagram like Image-based post sharing, business advertising, explore different user interest content, and direct and group message.

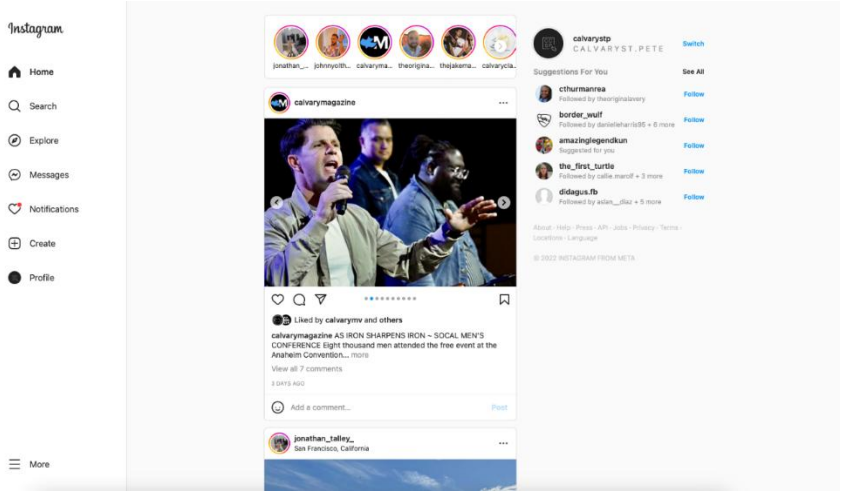

Figure 3 [9]: Instagram

2.4. The problem with Current System

Despite the widespread use and advancements in social media platforms, several critical gaps remain in current systems that impact user experience and engagement. A major limitation is the lack of post scheduling functionality, which restricts users from efficiently planning and managing their content sharing, especially for those with busy routines. Additionally, many platforms fail to offer intelligent friend suggestion systems based on mutual connections, which can reduce the chances of meaningful interactions. Another shortcoming is the absence of profile view tracking, preventing users from gaining insight into who is engaging with their profiles. Furthermore, the limited representation of business advertisements and insufficient analytics tools for tracking ad engagement reduce the value for business users. The lack of detailed interaction analysis also leaves users and administrators without a clear understanding of engagement trends. By addressing these deficiencies, our project aims to create a smarter, more insightful, and user-conscious social media experience.

CHAPTER 3: SYSTEM ANALYSIS

3.1. Requirement Analysis

The requirement analysis for SociablePeak began with identifying the limitations of existing social media platforms and understanding the evolving needs of users, businesses, and administrators. To gather meaningful requirements, several techniques were used, including literature reviews, observation of existing platforms (like Facebook and Instagram), informal interviews with frequent social media users, and feedback from peers and faculty members.

During this process, a key communication gap was identified between what users expect and what existing platforms offer. Users expressed frustration over the lack of control and insights into their activity such as not being able to schedule posts, track who views their profiles, or receive relevant content recommendations. Similarly, businesses desired more effective tools for tracking advertisement performance and engaging with their audience. Administrators lacked robust analytics tools to monitor overall engagement and user behavior.

These insights highlighted the need for a more user-conscious, feature-rich platform that emphasizes personalization, transparency, and usability. Based on the collected requirements and gap analysis, the following core features were defined and integrated into SociablePeak:

- Users shall be able to schedule posts to be published at a future date and time.
- The platform shall support real-time one-on-one and group messaging, including message encryption.
- Users shall be able to post and share content such as text and images.
- Users shall receive intelligent friend suggestions based on graph traversal algorithms analyzing mutual connections.
- The platform shall provide profile view tracking, displaying both total views and viewer details.
- Users shall be able to like and comment on posts and ads.
- Business accounts shall be able to post advertisements and monitor engagement metrics such as likes and comments.

- The platform shall include user interaction analytics for both users and admins to evaluate engagement trends and user growth.

This requirement gathering process ensured that the platform was not just designed based on assumptions, but was informed by actual user expectations, existing system analysis, and modern digital behavior patterns.

3.1.1. Functional Requirement

The functional requirements of this project are as follows:

- Web applications shall allow users, businesses and administrators to perform the authentication process to access their respective functionalities securely.
- Users, business and administrators shall have access to engagement analytics
- Users and Business shall be able to upload posts immediately or schedule them for later.
- Users and businesses shall be able to like and comment on posts and ads
- Users shall be able to send messages to other users or create group chat.
- Users shall be able to report to another user.
- Users shall be able to search for a user and see the profile of other users.
- Users shall be able to send friend requests and accept or reject the request received.
- Users shall be able to get friends suggestions.
- Users shall be able to see various ads.
- Admin shall be able to view reports and block user.

These functional requirements serve as the core functionalities of the " SociablePeak: A Social Media Web Platform for personalized connections and content through graph traversal and filtering ". These requirements aim to enhance the user experience.

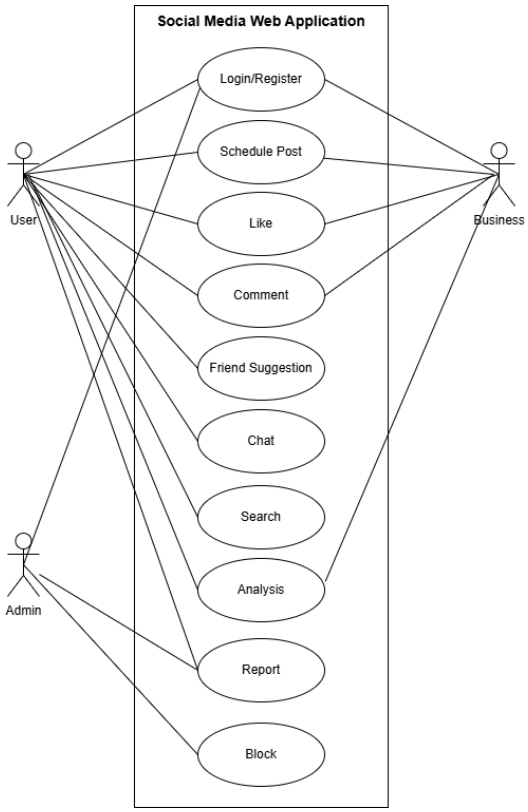

Figure 4: Use-Case Diagram for Social Media Web Application

3.1.2. Non-Functional Requirement

Non-functional requirements are essential aspects of any software application, focusing on how the system behaves rather than what it does. In the context of the " SociablePeak: A Social Media Web Platform for personalized connections and content through graph traversal and filtering " the non-functional requirements are:

- Application must be user friendly.
- Application must be able to prevent unauthorized access.
- Application must represent error free data.
- Application must follow consistent design patterns.

3.2. Feasibility Analysis

3.2.1. Technical Feasibility

The SociablePeak platforms are developed using Laravel PHP framework, which provides a robust backend structure and seamless integration with MySQL for data management. Frontend components were built using Blade templating along with HTML, CSS, and JavaScript. The system also integrates JavaScript libraries such as jQuery for UI interactions and uses technologies like bcrypt for secure password hashing and Laravel Echo for real-time messaging. The project was deployed and tested on a local Apache server (XAMPP), which is compatible with minimal hardware. Thus, all required software and hardware were readily available, making the project technically feasible.

3.2.2. Operational Feasibility

Operationally, SociablePeak fits well into existing user habits on social platforms. It introduces features such as post scheduling, encrypted messaging, profile view tracking, and friend suggestions using familiar social interaction flows, ensuring minimal user learning curve. The addition of business account modules and admin dashboards aligns with how social media is used for promotion and moderation today. Admins can monitor user growth, reported content, and engagement trends, ensuring the system supports both user activity and platform governance effectively.

3.2.3. Economic Feasibility

The project development was completed using open-source tools and technologies, minimizing financial costs. Laravel, MySQL, and XAMPP are free to use, and no additional costs were incurred for licensing. Development was carried out by existing team members with no need for paid external resources. Hosting and deployment are also cost-effective when using shared or cloud-based web hosting solutions. Overall, the project was economically feasible with no significant expenditure.

3.2.4. Schedule Feasibility

The development of SociablePeak followed the incremental development model and was completed within the estimated academic project timeline. Major features like user

authentication, posting, chat, analytics, and admin modules were implemented in stages over a period of 12–14 weeks. Weekly goals and milestones were met through structured planning, making the project feasible within the allocated schedule. The project timeline is visually represented using a Gantt chart, which outlines the task distribution and duration across the development period. It began with planning and analysis in early November 2024, followed by critical modules like email integration, two-factor authentication, chat systems, profile view tracking, ad management, and grammar checking. The final stages included integration, testing, and deployment, extending into June 2025. Each feature was developed with clear milestones, and the Gantt chart demonstrates that development was consistent and strategically paced, confirming that the schedule was well-organized and realistically achievable.

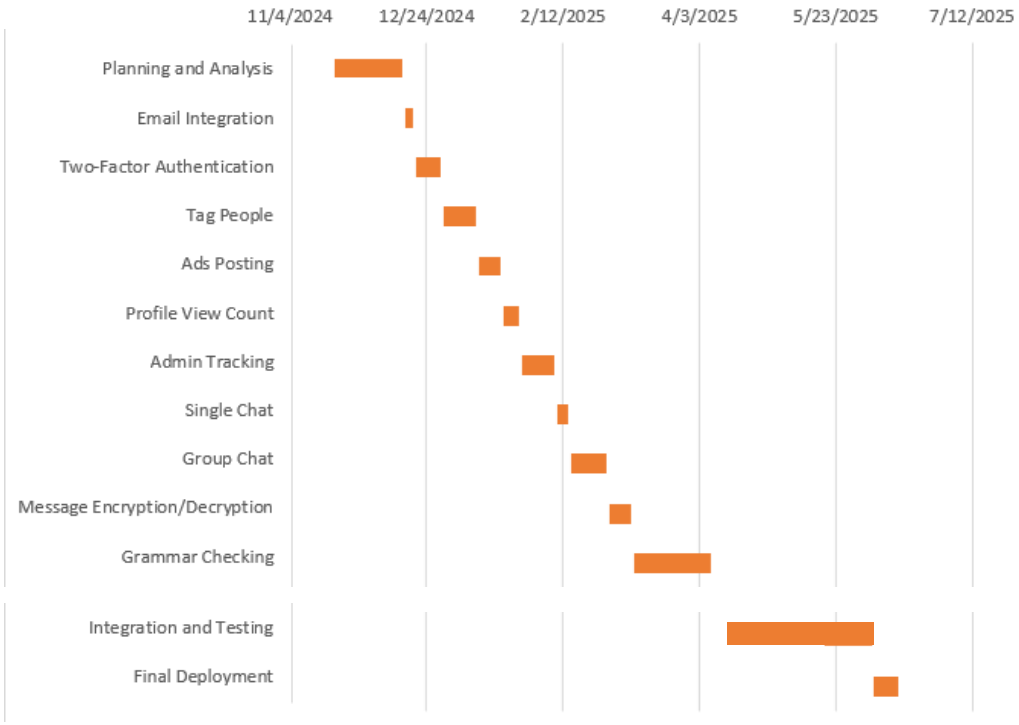

Figure 5: Gantt Chart of SociablePeak

3.3. Analysis

Since SociablePeak was developed using an Object-Oriented Approach with the Laravel framework, the analysis of the system was carried out through object, dynamic, and process modelling techniques. These models were essential for visualizing system structure, behavior, and workflows, ensuring organized and scalable development.

3.3.1 Class Diagram

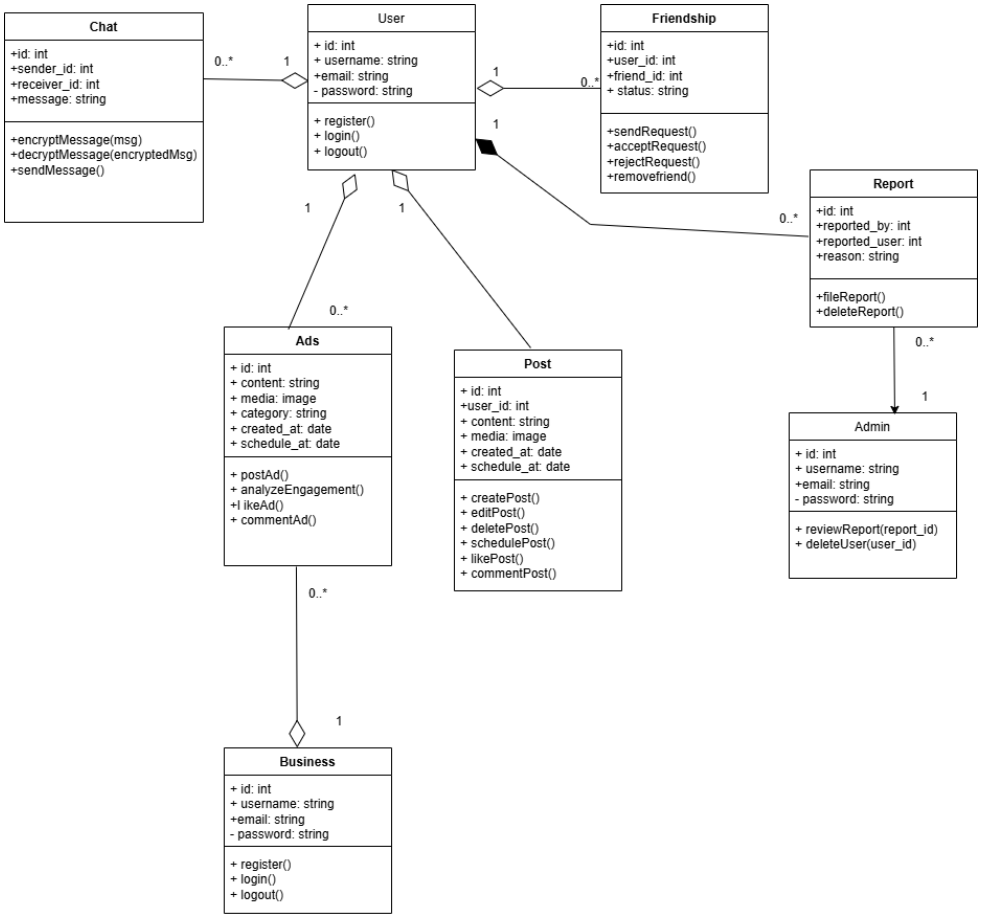

Figure 6: Class Diagram for SociablePeak

Figure 6 models are a class diagram where user can create posts with comments and likes. Chat manages messages between Users. Friendship tracks relationships. Reports document user issues which will be resolved by the admin and businesses can add ads.

3.3.2 Activity Diagram

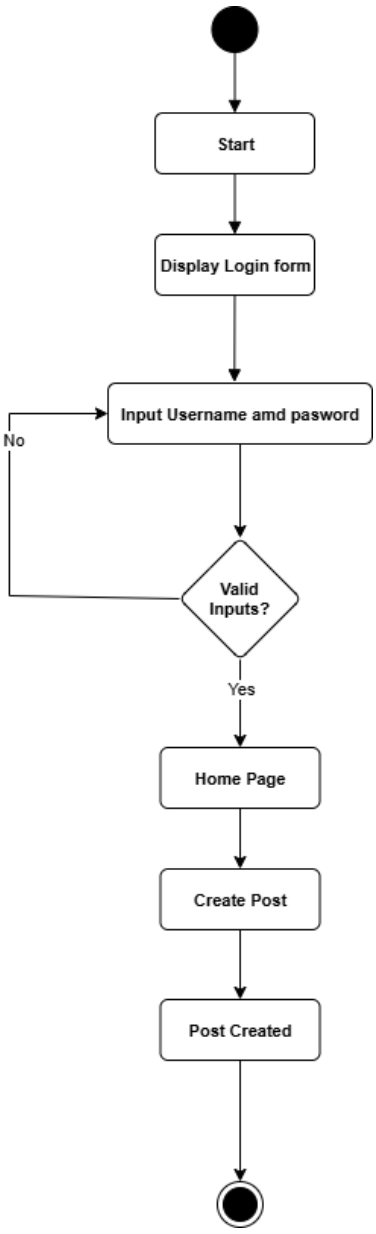

Figure 7: Activity Diagram for SociablePeak

Figure 7 represents activity diagram that shows the process of logging in and creating a post. It starts with displaying a login form, then checks email and password. If valid, it proceeds to the home page and allows post creation with a caption. Invalid login info loops back to input.

3.3.3 State Diagram

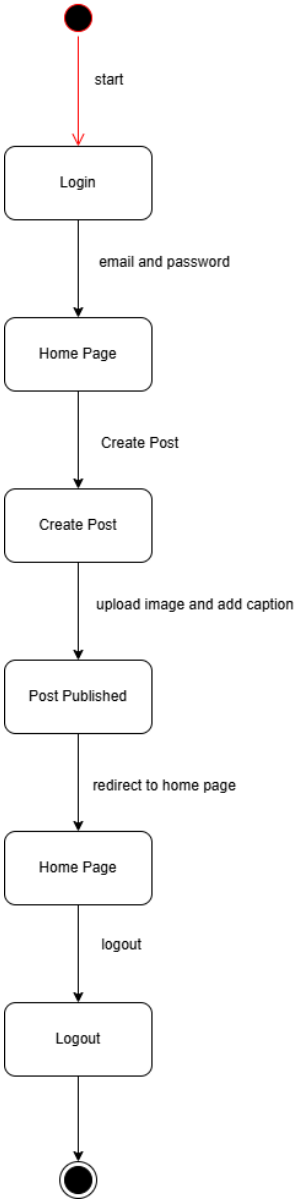

Figure 8: State Diagram for SociablePeak

Figure 8 represents State diagram that shows a user's journey through a system. It starts with "Login," transitions to "Home Page," then "Create Post" (uploading and captioning), resulting in "Post Published." It redirects back to "Home Page" before finally "Logout," ending the session. It depicts the sequence of states and transitions triggered by user actions

3.3.4 Sequence Diagram

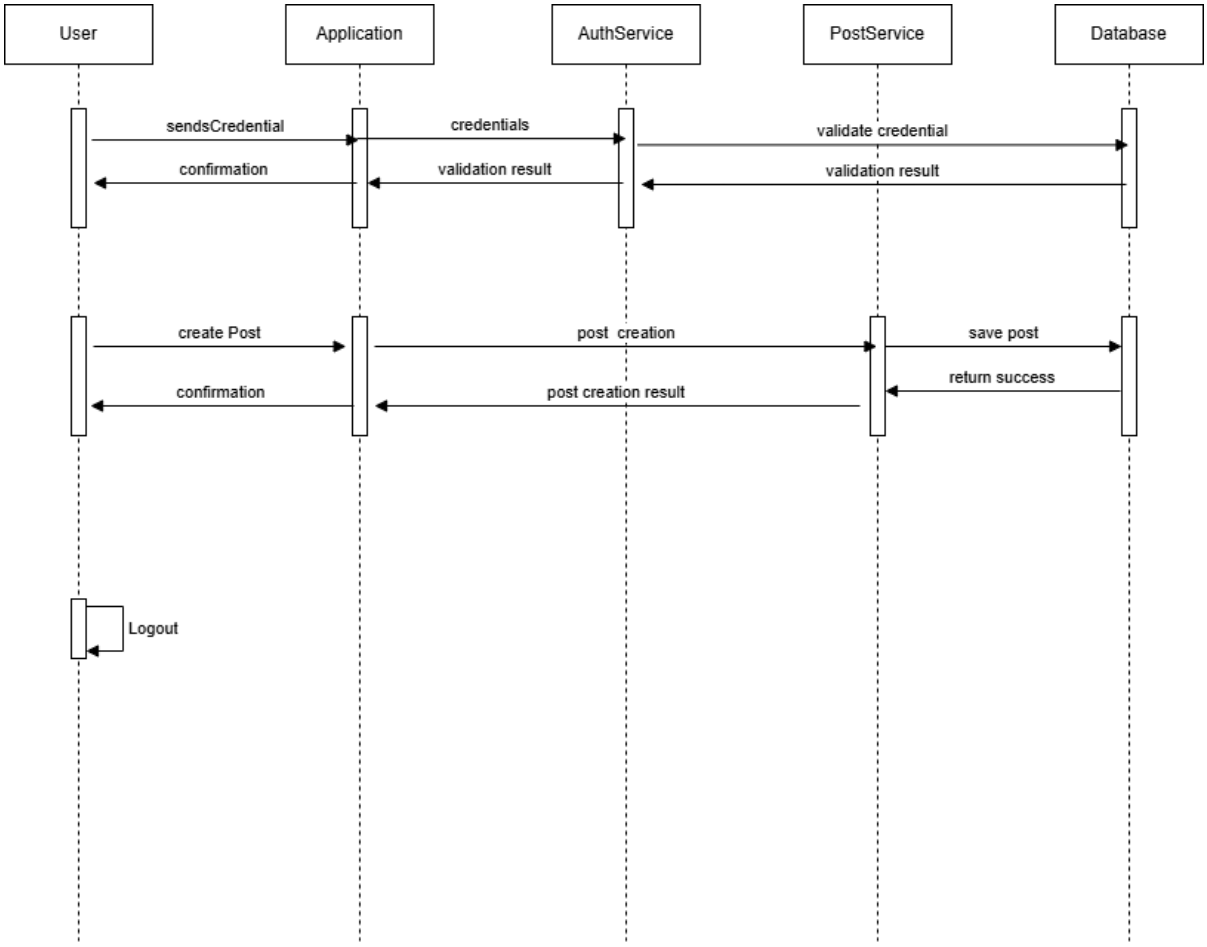

Figure 9: Sequence Diagram of SociablePeak

Figure 9 represents a sequence diagram that shows a user logging in, creating a post, and logging out. The user sends credentials to the application, which validates them via the AuthService and Database. Upon success, the user creates a post, which the application saves via the PostService and Database. Finally, the user logs out. It illustrates the interaction flow between user, application, services, and database.

CHAPTER 4: SYSTEM DESIGN

4.1. Design

4.1.1 Refinement of Class Diagram

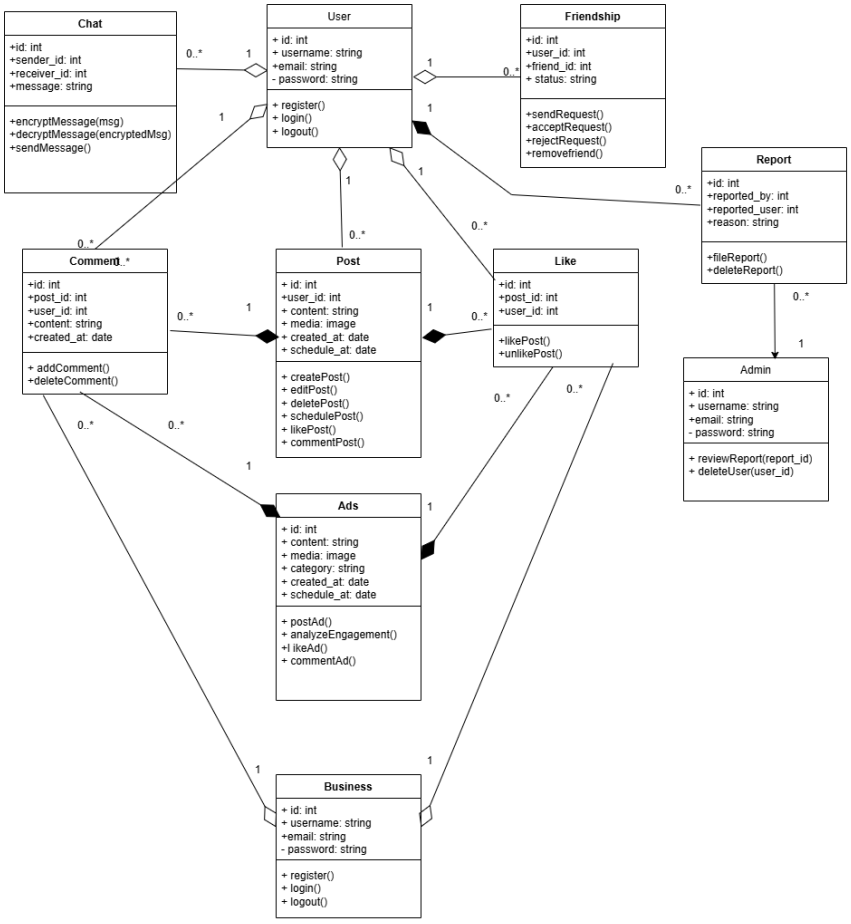

Figure 10: Refined Class Diagram of Social Media Web Application

Figure 10 models a class diagram where user can create posts with comments and likes. Chat manages messages between Users. Friendship tracks relationships. Reports document user issues which will be resolved by the admin and businesses can add ads which contain likes and comments.

4.1.2 Refinement of Activity Diagram

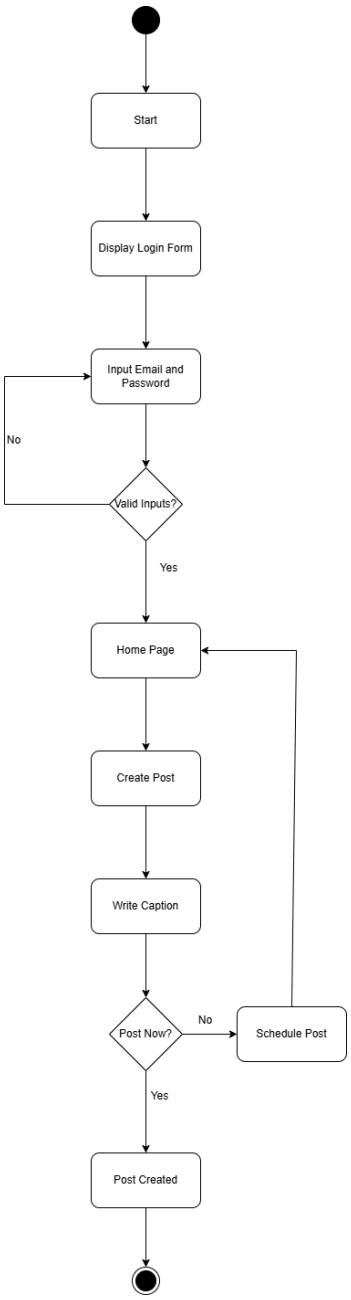

Figure 11: Refined Activity Diagram of SociablePeak

Figure 11 represents an activity diagram that shows the process of logging in and creating a post. It starts with displaying a login form, then checks email and password. If valid, it proceeds to the home page and allows post creation with a caption. Users can choose to post immediately or schedule it. Invalid login info loops back to input.

4.1.3 Refined Sequence Diagram

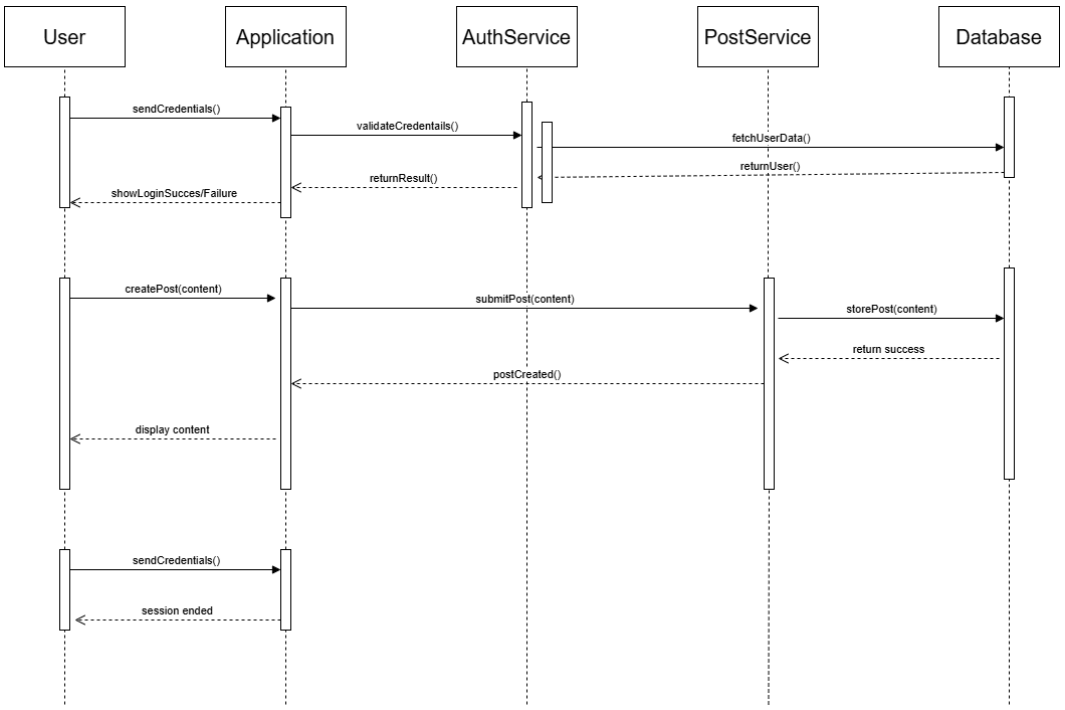

Figure 12: Refined Sequence Diagram of SociablePeak

Figure 12 represents sequence diagram which show the interaction flow for user login, post creation, and logout in the SociablePeak platform.

4.1.4 Component Diagram

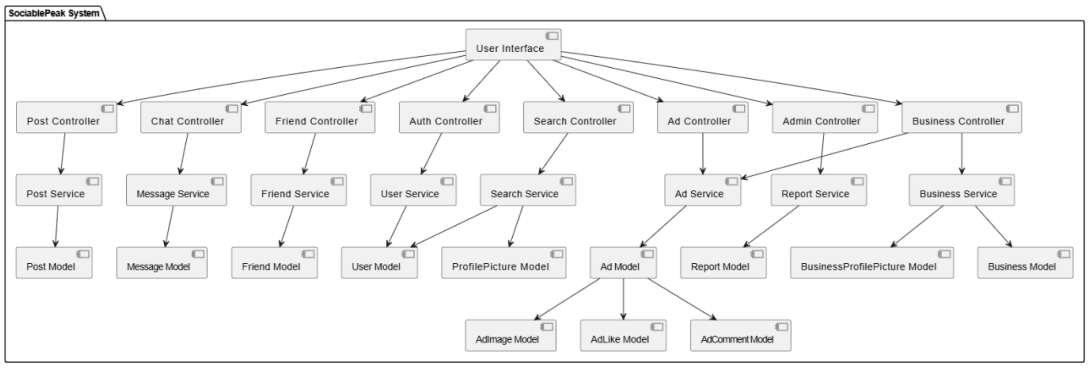

Figure 13: Component Diagram of SociablePeak

Figure 13 represents the Deployment Diagram which shows the interaction between client, Laravel-based web server, MySQL database, email service, and file storage in the SociablePeak platform.

4.1.5 Deployment Diagram

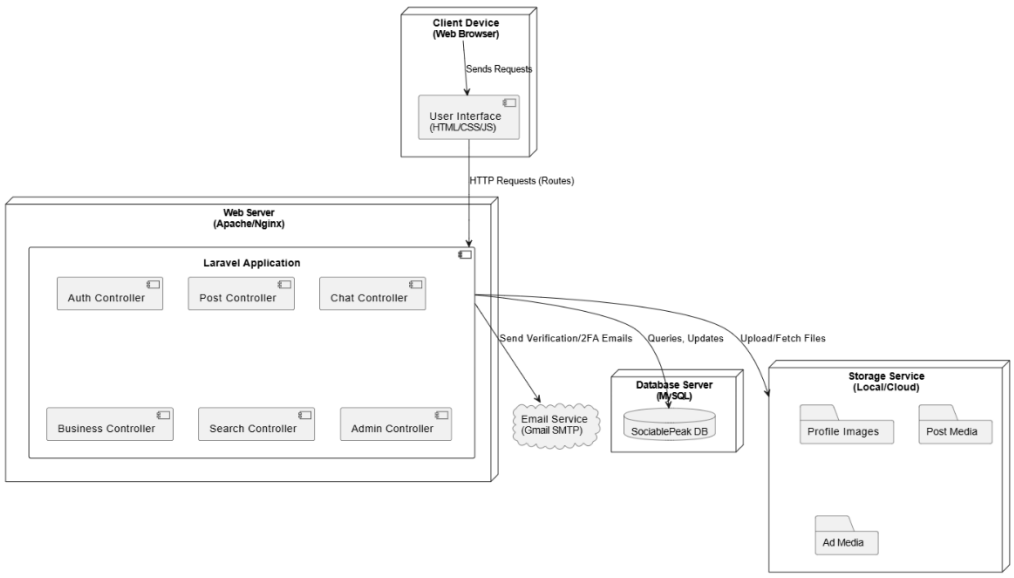

Figure 14: Deployment Diagram of SociablePeak

Figure 14 represents Component diagram illustrating the modular architecture of the SociablePeak platform, showing interactions between the user interface, controllers, services, and underlying data models, including business and advertisement components.

4.2. Algorithm Details

During the development of SociablePeak, multiple algorithms were implemented to enhance user experience, personalization, and intelligent interaction. Below are the core algorithms integrated into the system:

a. AES-256-CBC (Advanced Encryption Standard - Cipher Block Chaining)

The AES-256-CBC algorithm is used to encrypt and decrypt messages in both one-to-one and group chat systems. AES is a symmetric block cipher and is widely recognized for its reliability and high level of security. Here each block of plaintext is XORed with the previous ciphertext block before being encrypted, improving security against pattern recognition. Each message is encrypted with a unique Initialization Vector (IV), which is stored along with the ciphertext.

The openssl_encrypt() and openssl_decrypt() functions are used in Laravel to handle encryption and decryption.

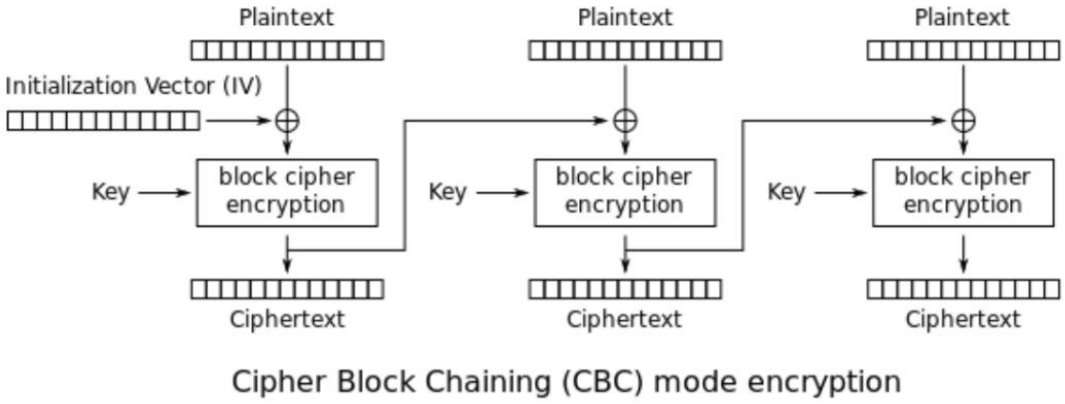

Figure 15: Cipher Block Chaining mode Encryption

b. Graph Traversal Algorithm

Graph Traversal Algorithm is used to suggest new friends based on mutual connections.

This algorithm is inspired by breadth-first search (BFS) principles, where the system identifies the first-level connections (direct friends) of the current user then finds the second-level connections (friends of friends). After that it Filters out users who are already friends with the current user or the user themselves and suggests these second-level users as potential friends.

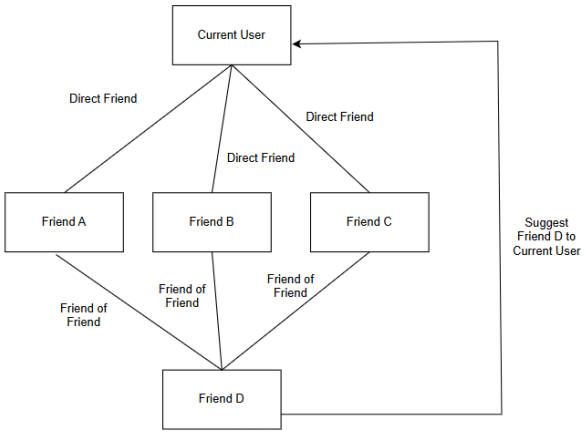

Figure 16: Graph Traversal

c. Content-Based filtering

Content Based filtering is used to recommend personalized advertisements to users based on their previous interactions with ad categories. This recommendation algorithm works by tracking the categories of ads a user has liked in the past. It builds a user preference profile based on these interactions and recommends ads that belong to the same or similar categories but which the user hasn't seen or interacted with yet.

d.Bcrypt Hashing Algorithm

Bcrypt Hashing Algorithm is used to securely store passwords for both users and business accounts during registration and login processes. Bcrypt is a one-way password hashing algorithm designed for secure password storage. It offers automatic salting, adaptive cost factor and irreversibility. In Laravel, this is seamlessly implemented using Laravel's built-in bcrypt() function or Hash::make(). It ensures that even if the database is compromised, passwords remain safe from reverse-engineering.

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1. Implementation

The implementation phase of SociablePeak was focused on converting system designs into modular, scalable code using the Laravel framework. The application was structured using the MVC (Model-View-Controller) architecture, with a clear separation of business logic, data handling, and user interface layers.

Laravel's built-in features such as routing, middleware, Eloquent ORM, task scheduling, and authentication were extensively utilized. Alongside standard implementations, certain modules required complex logic, such as:

- Friend Suggestion System using mutual connections (graph-like traversal without graph data structures)
- Post Scheduling System using Laravel Scheduler and time-based conditions
- Caption Generator Chatbot integrating third-party Gemini API with prompt engineering and fallback handling
- Profile View Tracker ensuring duplicate/self views were filtered and accurately recorded

These modules required careful query design, performance considerations, and error handling. Each was iteratively developed and tested to ensure reliability across different user scenarios. the technical knowledge base of the group towards firmer and loosely coupled code.

5.1.1. Tools Used

The development of SociablePeak required a combination of frontend, backend, database, design, and testing tools to ensure a seamless, secure, and feature-rich user experience. The following tools and platforms were utilized across various phases of the system development lifecycle:

Programming Language

- PHP
- JavaScript

Backend Framework

- Laravel

Frontend Technology

- Blade templating Engine
- HTML
- CSS
- Bootstrap
- JavaScript

Database Platform

- MySQL

CASE Tools

- Draw.io
- PlantUML

API

- Gemini API

Testing Tools

- Laravel Artisan Test Commands
- PHPUnit

Development Environment

- VS Code
- XAMPP
- Postman

Version Control

- Github

5.1.2. Implementation Details of Modules

This section outlines the major modules used in SociablePeak system, highlighting their purpose, design logic, and the classes or functions responsible for their functionality. Each module plays an important role to overall functionality and user experience of the platform.

1. **User Authentication and Two Factor Verification**

   The user authentication module was built using Laravel's built-in authentication system i.e Auth . It includes user registration, login, and password change functionality. For enhanced security, two-factor authentication was introduced during the password change process. When a user initiates a password change, a verification code is sent to their email using a custom Mailable class. The system waits for user input and verifies the code before allowing the password update.

   Code Snippets

   ```
   $userCode = rand(100000, 999999);
   Mail::to($user->email)->send(new SendCode($userCode));
   ```

   The system temporarily stores this code in the session and compares it when the user submits it through the verification form.

2. **Post Management**

   Users can post content either immediately or schedule it for future publishing. Scheduled posts are stored with a scheduled_at timestamp. A Laravel scheduled task runs at regular intervals to check and publish due posts and updates their status to "published. This scheduling mechanism ensures that user posts go live automatically without requiring manual action.

   Code Snippets

   ```
   if (!empty($set_time)) {
      $post->set_time = $set_time;
      $post->status = 0; // Scheduled
      $post->save();
   }
   ```

3. **Friend System and Suggestion Logic**

   The platform includes a comprehensive friend system that allows users to send, accept, reject, and delete friend requests. A pivot table tracks the relationship status between users. The suggestion feature identifies potential friends using mutual connection logic. Internally, it compares friend relationships using Laravel's many-to-many pivot tables. It

identifies users who share common friends but are not yet connected, simulating a second-degree graph traversal without external libraries.

Code Snippets

```
$myFriends = Friend::where(function($query) use ($userId) {
    $query->where('user_id', $userId)
          ->orWhere('friend_id', $userId);
})
->where('status', '!=', 'pending')
->get()
->map(function($friend) use ($userId) {
    return $friend->user_id == $userId ? $friend->friend_id : $friend->user_id;
})
->toArray();

$friendsOfMyFriends = Friend::where(function($query) use ($myFriends) {
    $query->whereIn('user_id', $myFriends)
          ->orWhereIn('friend_id', $myFriends);
})
->where('status', '!=', 'pending')
->get()
->map(function($friend) use ($myFriends) {
    return in_array($friend->user_id, $myFriends) ? $friend->friend_id : $friend->user_id;
})
->unique()
->toArray();
```

4. **Chat Functionality with Encryption**

Individual and Group chats were implemented to allow users to communicate securely. Each group is uniquely identified and has associated members. Messages in group chats and individual chats are encrypted before being stored in the database using Laravel's

encryption methods. When a user retrieves a message, it is decrypted in real-time to ensure that only authorized members can read the content.

Code Snippets

```
protected function encryptMessage($message) {
    $encryptMethod = "AES-256-CBC";
    $secretKey = config('app.encryption_key');
    $iv = openssl_random_pseudo_bytes(openssl_cipher_iv_length($encryptMethod));
    return openssl_encrypt($message, $encryptMethod, $secretKey, 0, $iv) . "::" . bin2hex($iv);
}

protected function decryptMessage($encryptedMessage) {
    $encryptMethod = "AES-256-CBC";
    $secretKey = config('app.encryption_key');
    list($data, $iv) = explode("::", $encryptedMessage);
    return openssl_decrypt($data, $encryptMethod, $secretKey, 0, hex2bin($iv));
}
```

Both individual and group messages are stored with timestamps, and a combined chat list is created by sorting the messages based on the last interaction time.

5. **Profile View Tracking**

A separate module tracks how many times a user's profile has been viewed and by whom. This feature uses a dedicated table to log each view, excluding repeated views from the same user within a short time frame. The system calculates total views and displays a list of recent viewers, offering users insight into their profile engagement.

Code Snippets

```
if ($currentUser->id != $viewedUser->id && !$existingView) {
    ProfileView::create([
        'viewer_id' => $currentUser->id,
        'viewed_id' => $viewedUser->id
    ]);
```

}

This helps track profile visibility and engagement metrics.

6. **Caption Generator Chatbot**

To enhance post creativity, a chatbot was integrated that generates captions based on user input keywords. The system sends user input to an external AI service (Gemini API), receives a generated caption, and displays it in the post interface. The integration is handled by a dedicated controller that manages API requests, response parsing, and error handling.

Code Snippets

```
$response = Http::post($url, [
   'contents' => [
      [
         'parts' => [
            ['text' => "Generate a caption: {$message}"]
         ]
      ]
   ]
]);

$caption = $response->json()['candidates'][0]['content']['parts'][0]['text'] ?? 'Default caption';
```

7. **Ads Management Module**

This module enables business users to create and publish advertisements. Ads include images, titles, descriptions, and category and are displayed to users across the platform. Other users can like and comment on these ads. Each interaction is tracked, and engagement metrics are calculated for business analytics.

Code Snippets

```
$ad = Ad::create([
   'user_id' => Auth::id(),
   'title' => $request->title,
```

```
    'description' => $request->description,
    'category' => $request->category,
]);
```

Interactions like likes and comments are stored in separate tables and counted for analytics.

8. Content-Based Ad Filtering

To personalize user experience, a basic content-based filtering system was implemented. It matches ad categories or tags with user interests inferred from past interactions (e.g., like posts or viewed ads). Using a simple tag-matching algorithm, the system displays more relevant ads to each user, improving ad effectiveness.

Code Snippets

```
$likedCategories = Ad::whereHas('adLikes', function($query) use ($userId) {
    $query->where('user_id', $userId);
})
->pluck('category')
->unique();

$similarAds = Ad::whereIn('category', $likedCategories)
->whereNotIn('id', $user->adLikes()->pluck('ad_id')) // Exclude already liked ads
->inRandomOrder()
->get();

$randomAds = Ad::inRandomOrder() ->get();
$ads = $similarAds->merge($randomAds)->unique('id');
```

9. Admin Dashboard Module

Admins have access to a dashboard that displays total users, reported cases, and user growth trends. Admins can view user details, delete reported accounts, and analyze activity metrics. The dashboard pulls data from various models and presents a summarized overview in visual and tabular formats.

Code Snippets

```
$totalUsers = User::count();
$totalReports = Report::count();

$userGrowth = User::selectRaw('count(*) as count, MONTH(created_at) as month')
    ->whereYear('created_at', Carbon::now()->year)
    ->groupBy('month')
    ->pluck('count', 'month');

$userGrowthLabels = $userGrowth->keys()->map(function ($month) {
    return Carbon::create()->month($month)->format('F');
});

$userGrowthData = $userGrowth->values();
```

The values $userGrowthLabels and $userGrowthData are passed to Chart.js in the frontend to visualize monthly user registrations for admin analysis.

Each of these modules was implemented following Laravel's Model-View-Controller (MVC) architectural pattern, which promotes modularity, reusability, and maintainability by clearly separating the application's logic, user interface, and data handling components [11]. This structured approach provides a robust foundation for building a scalable and feature-rich social media platform.

5.2. Testing

For the SociablePeak platform, both unit testing and system testing were performed to verify individual components and their integration. The testing process aimed to detect bugs, verify business logic, and ensure data integrity across modules such as registration, posting, messaging, and admin controls.

5.2.1. Test Cases for Unit Testing

Table 1: Unit Test for User Registration and Login

Test Case ID	Description	Input	Expected Output	Actual Output	Status
UT-UL-01	Register with valid data	Name, Email, Password	User registered	User account created and redirected to dashboard	Pass
UT-UL-02	Register with existing email	Duplicate email	Error: Email exists	System showed error "Email already exists"	Pass
UT-UL-03	Login with correct credentials	Email and Password	Redirect to dashboard	User successfully logged in and redirected to dashboard	Pass
UT-UL-04	Login with incorrect password	Email and wrong password	Error: Invalid login	System displayed "Invalid credentials" message	Pass

Table 2: Unit Test for Post Scheduling and Uploading

Test Case ID	Description	Input	Expected Output	Actual Output	Status
UT-PS-01	Schedule a valid post	Image and Future Date	Post scheduled successfully	Post appeared in scheduled posts with correct date and time	Pass

UT-PS-02	Upload a valid image without scheduling	Image only	Post uploaded immediately	Post appeared instantly in the user's timeline	Pass
UT-PS-03	Upload with missing image	No image and Future Date	Error: Image or Description required	System displayed validation message “Image or Description required”	Pass
UT-PS-04	Schedule a post with past date	Image and Past Date	Error: Invalid scheduling date	Post was scheduled despite past date	Fail

Table 3: Unit Test for Friend Request Management

Test Case ID	Description	Input	Expected Output	Actual Output	Status
UT-FR-01	Send friend request	Valid target user	Request sent	Friend request sent and shown as pending in target profile	Pass
UT-FR-02	Accept friend request	Valid request ID	Request accepted	Friend request status changed to accepted and user added to friend list	Pass
UT-FR-03	Reject friend request	Valid request ID	Request rejected	Friend request removed from pending list and marked as rejected	Pass
UT-FR-04	Send friend request to self	User tries to send request to own account	Error: Cannot send friend request to yourself	Request sent to self without error	Fail

Table 4: Unit test for Email and 2FA Verification

Test Case ID	Description	Input	Expected Output	Actual Output	Status
UT-EM-01	Send verification email	Valid email	Email sent	Verification code sent to the user's email address	Pass
UT-EM-02	Verify correct 2FA code	Valid code	Verification success	User successfully verified and redirected to secure page	Pass
UT-EM-03	Verify expired 2FA code	Expired code	Error: Code expired	System accepted expired code without error	Fail
UT-EM-04	Verify incorrect code	Invalid code	Error message shown	System showed error "Invalid verification code	Pass

Table 5: Unit test for Chat with Encryption

Test Case ID	Description	Input	Expected Output	Actual Output	Status
UT-CH-01	Send encrypted message	Message and key	Message stored encrypted	Encrypted message saved in database and sent	Pass
UT-CH-02	Decrypt received message	Encrypted message and key	Original message shown	Message decrypted and displayed in chat window	Pass
UT-CH-03	Send message to group	Group ID and message	Message delivered to all	All group members received the message in real-time	Pass

5.2.2. Test Cases for System Testing

Table 6: System Test for User Registration and Login

Test Case ID	Description	Input	Expected Output	Actual Output	Status
ST-UL-01	Register new user	Valid registration form data	User created and redirected to dashboard	Account created and redirected successfully	Pass
ST-UL-02	Login with correct credentials	Email + password	Redirect to dashboard	Dashboard loaded with user info	Pass
ST-UL-03	Login with invalid password	Email + wrong password	Error message shown	"Invalid login credentials" displayed	Pass

Table 7: System Test for Post Scheduling and Uploading

Test Case ID	Description	Input	Expected Output	Actual Output	Status
ST-PS-01	Schedule a post and verify it publishes on correct date	Image + future date	Post appears on user timeline on scheduled date	Post appeared at scheduled time on timeline	Pass
ST-PS-02	Try uploading a post without an image	Only caption or date	Error shown and post not created	Error shown: "Image is required"	Pass
ST-PS-03	Verify that user can view scheduled posts list	Logged-in user	Scheduled posts displayed with date and time	List of scheduled posts displayed correctly	Pass

Table 8: System Test for Friend Request System

Test Case ID	Description	Input	Expected Output	Actual Output	Status
ST-FR-01	Send a friend request and check	Logged-in user clicks "Add Friend"	Request status	Request sent and shows as pending	Pass

	pending status		changes to pending		
ST-FR-02	Accept a request and verify mutual connection	Accept pending request	Users appear in each other's friend list	Mutual friendship established	Pass
ST-FR-03	Reject a request	Reject button clicked	Request disappears from pending list	Request removed from pending section	Pass

Table 9: System Test for Email Verification and 2FA

Test Case ID	Description	Input	Expected Output	Actual Output	Status
ST-EM-01	Register and check email verification	Valid email	Verification code sent	Email with code received	Pass
ST-EM-02	Enter correct code	Matching code	Access granted	Redirected after successful verification	Pass
ST-EM-03	Enter incorrect code	Wrong input	Error shown	Error: "Invalid code"	Pass

5.3. Result Analysis

The development and testing of SociablePeak prove that the system works efficiently presenting its required functions. Due to the performance of the unit and system testing, the major features of the application, including user registration, post scheduling, friend communication, group messaging with AES encryption, advertisement activities, and AI-powered caption generation , were tested comprehensively. The majority of critical functions were known to go through all the predetermined test conditions and ascertained functionality by logic correctness and seamless frontend hardware mismatching with a backend.

The testing process was designed to reflect real user behavior. As an example, the postings were scheduled and then the schedule observed to ensure whether it was posted at the exact time stipulated. The chatbot feature was also experimented on by typing different keywords so that it could come up with suitable captions utilizing the embedded AI API. This aligns with the growing trend of using AI in social media to enhance personalization and engagement, where intelligent systems generate content based on user input, preferences, and context [12]. Encryption of conversations within group chat module was verified by testing the privacy of the messages through ensuring the data integrity of conversations only the group members are capable of hearing. These results increased our confidence in the functional stability of the system.

Although the system behaved well when used at normal usage conditions, a few testing shortcomings were encountered. Load testing, security penetration testing, and automated regression testing were not done due to limitations of time. Consequently, the behavior of the system when it is faced with a wide user base, or malicious attacks, has not been confirmed. More tests can be carried out such as:

- Stress testing to examine system behavior with simultaneous user activity,
- User acceptance testing (UAT) with diverse users to gather usability feedback are recommended.

The tools and technologies on which the entire project was based also contributed greatly to effective implementation and testing. For instance:

- Draw.io was utilized for system specification and design through ER diagrams, data flow diagrams, and use-case models
- Laravel (PHP framework) served as the core backend technology, offering features like routing, task scheduling, authentication scaffolding, and middleware-based logic separation.
- MySQL handled structured data efficiently across users, posts, interactions, and logs.
- HTML, CSS, Bootstrap, and JavaScript were employed to create a responsive and interactive user interface.
- For testing, PHPUnit and Laravel's built-in testing features were used to validate methods, database interactions, and user flows.

The choice of using Laravel as the Framework proved effective due to its string tools and clear MVC structure. Although advanced feature like integrating AI or building filter required extra effort and learning due to limited documentation and API restriction.

In terms of strengths, the system offered:

- Real-time functionalities such as group chat and scheduled posting.
- A modular design with reusable components,
- Integration of modern technologies like AI-based caption generation.

However, the system also faced challenges such as:

- Time-consuming API integration and testing.
- Lack of advanced NLP features like tone detection and grammar correction.
- Limited real-user testing and scalability validation.

In spite of these issues, the project manages to prove the concept of a feature-laden social platform implementable with the help of open-source tools and modern development approach. Not only has the experience produced better technical skills, but it has proved to give a good understanding of planning, integration, testing and iteration. The findings validate the fact that SociablePeak is a stable prototype that can be improved and scaled up in the future.

CHAPTER 6: CONCLUSION AND FUTURE RECOMMENDATION

6.1. Conclusion

SociablePeak project was launched with a goal of development of an all-finned social platform that is concerned with user interaction, intelligent content analyses and support along with secure connection. The main functions in the system are the user registration and log in, scheduling the posts, encrypted group messaging, suggestion of friends depending on mutual connections, statistics on views of the profile, and modules of interaction with ads. All these features were executed via current development resources including Laravel to execute logics, MySQL to store information and Bootstrap to make the frontend responsive. In the development process using a modular approach and Model-View-Controller architecture (MVC) was used to allow transparency, reusability as well as scalability.

Another boost to the project occurred with adding artificial intelligence with the Gemini API by Google. This enabled it to introduce a chatbot that assists the users to create captions of their posts with the assistance of keywords. This trait, powered by AI, brought an artistic twist to the platform, letting the users demonstrate themselves in a more easy and productive way. The experience of incorporating this outside API showed that the contemporary technology of AI could be easily integrated into the web application to enhance the experience and personalization of the users.

System and unit testing was done thoroughly and centred on key user functions as well as backend tasks. Highlights such as scheduled posting were checked on the basis of time, friend requests and chat were checked on the basis of continuity of usage. During basic use of the platform, all the test cases successfully passed and this implies that the platform has high confidence levels in its functional reliability. Even though the schedule of the project did not allow running advanced performance testing, the core system was stable and consistent during the processes of manual testing.

Overall, the project has turned out to be a worthwhile effort on complete stack web advancement and smart framework assembly. It not only accomplished its practical objectives but it provided us with practical experience with AI APIs, a secure means of communication, and user-centric design. SociablePeak can serve as a sturdy basis of the future development and shows how effective applications can be created with the help of open-source frameworks and modern AI abilities. It can be further refined and then become a practical social platform that can work in the real world.

6.2. Future Recommendation

Although SociablePeak successfully implements a wide range of features for a social media platform, several areas offer room for enhancement and optimization in future work. Lack of automated performance testing was among the main limitations. Such tests as a stress test, load balancing as well as real-time traffic handling were not conducted due to the limits of time and resources. These are critical when it comes to knowing how the system will perform when subjected to a lot of users accessing similar or not, even when the platform is designed to support thousands of users. In the future, tools like Apache JMeter or Laravel Dusk might be included with the intent of imitating the work of multiple users and testing the system at its limits.

The other improvement point concerns responsiveness in mobile devices and across platforms. The current version is optimized to work on a desktop, and, though Bootstrap includes a barebones responsiveness, the experience on smaller devices can be improved. The next release may have a completely responsive design or a separate mobile app built on such frameworks as Flutter or React Native. This would provide uniform and immaculate interaction experience by the users at various devices.

Though the implementation of the integration of a Gemini API made it possible to generate captions with the help of AI, it is possible to extend its application. One can experiment with more sophisticated AI features like tone detection, content moderation, or even automatic responses of the chatbots in the future versions. Such improvements would not only increase user experience but also bring into the system a brain that would come up with safety and

engagement in regard. Moreover, the capability of natural language processing (NLP) might be added to enhance the correction of the grammar or make more specific suggestions of the content.

The latest version of the SociablePeak leverages local storage, and the development environments. To be ready to be deployed in the real world, the system must be transferred to a scalable cloud computing provider such as Google Cloud, AWS or Azure. This would enhance supply as well as security and elasticity. The usage of Continuous Integration/Continuous Deployment (CI/CD) pipelines to automatize the testing and deployment can also be applied, which also makes the system more effective and reliable.

Finally, although SociablePeak has already achieved the intended functionality, this set of recommendations makes it clear how the system can be developed to become more capable, perform better, and make its users even more satisfied over a long period.

REFERENCES

[1] S.Larson, "Priordata," Priordata, 15 April 2025. [Online]. Available: https://prioridata.com/data/social-media-usage/. [Accessed 14 May 2025].

[2] E. Oritiz-Ospina, "Our World in Data," 18 september 2019. [Online]. Available: https://ourworldindata.org/rise-of-social-media. [Accessed 14 May 2025].

[3] B. a. Ellison, "Journal of Computer-Mediated Communication," vol. 13, pp. 210-230, 2008.

[4] A. Perrin, "Social Networking Usage: 2005-2015 ," Pew Research Center, 2015.

[5] M. Dewing, "Social Media: An Introduction," Library of Parliament, 2012.

[6] T. D. S. M. A. N. Anwesha Chakraborty, "Application of Graph Theory in Social Media," *International Journal of Computer Sciences and Engineering ,* vol. 6, no. 10, 2018.

[7] P. J. W. Siti Zaiton Mohd Hashim, "Content-based filtering algorithm in social media," *Wasit Journal of Computer and Mathematic Science,* vol. 2, no. 1, 2023.

[8] "Facebook," Meta , [Online]. Available: https://www.facebook.com/. [Accessed 25 06 2025].

[9] "Instagram," Meta, [Online]. Available: https://www.instagram.com/. [Accessed 25 06 2025].

[10] "Google AI for Developers," Google, [Online]. Available: https://ai.google.dev/gemini-api/docs. [Accessed 24 06 2025].

[11] F. Hanif, "Medium," [Online]. Available: https://fkrihnif.medium.com/understanding-the-mvc-architecture-in-laravel-a-comprehensive-guide-8f620cc139b6. [Accessed 24 06 2025].

[12] T. J. A. A. A.-M. a. S. M. M. Matthew N. O. Sadiku, "Artificial Intelligence in Social Media," *International Journal of Scientific Advances (IJSCIA),* vol. 2, no. 1, 2021.

APPENDIX I

This appendix contains supplementary materials that support the main content of the SociablePeak project report. These materials include system screenshots, sample API interactions, database schema, and additional information that may help the reader better understand the design, implementation, and functionality of the system. These elements are not included in the main chapters to maintain clarity and flow but are documented here for reference and completeness.

Appendix I.A: System User Interface Screenshots

a. Login Page

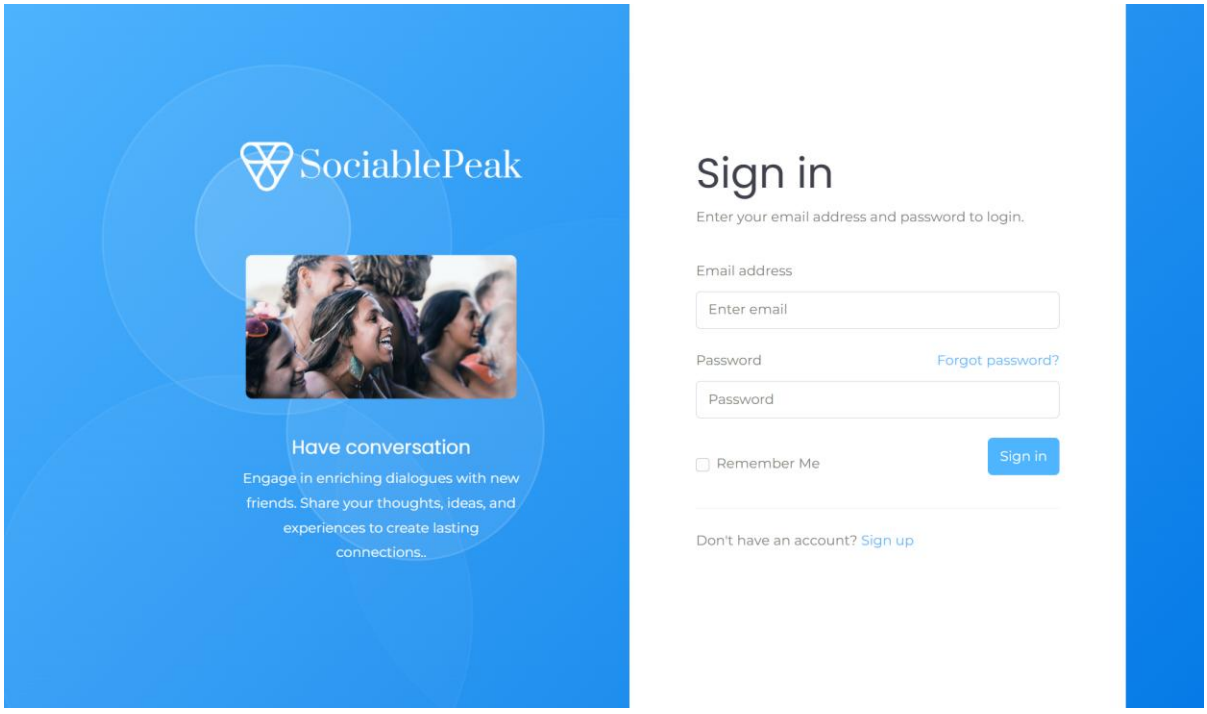

Figure 17: Login Page UI

b. Home Page

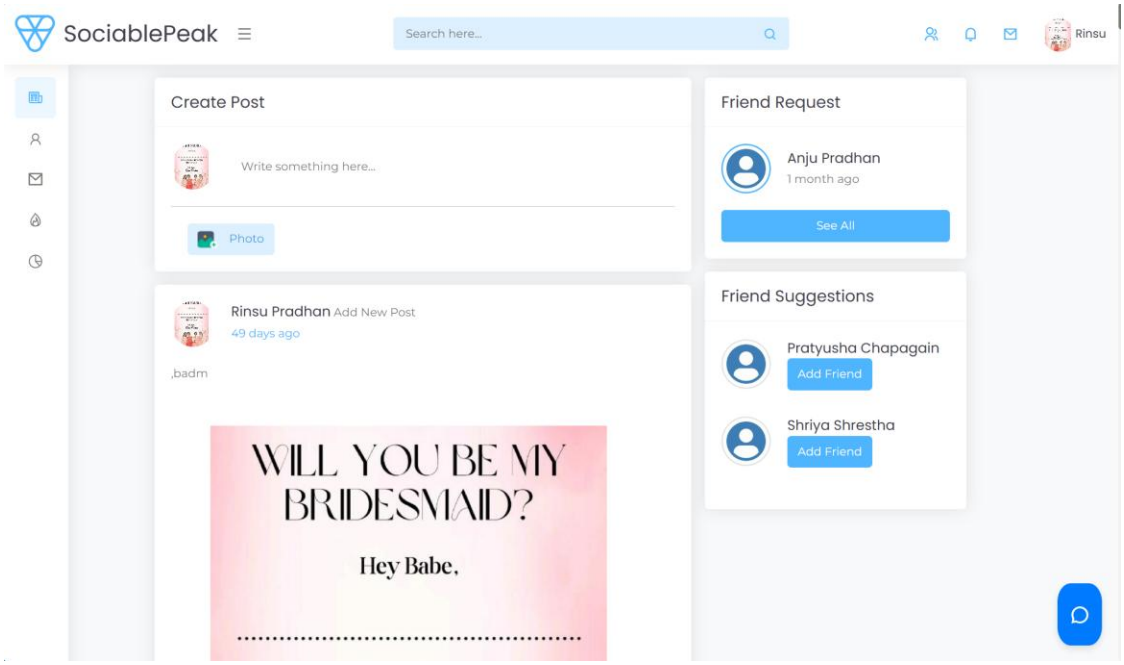

Figure 18: Home Page UI

c. Caption generator Chatbot

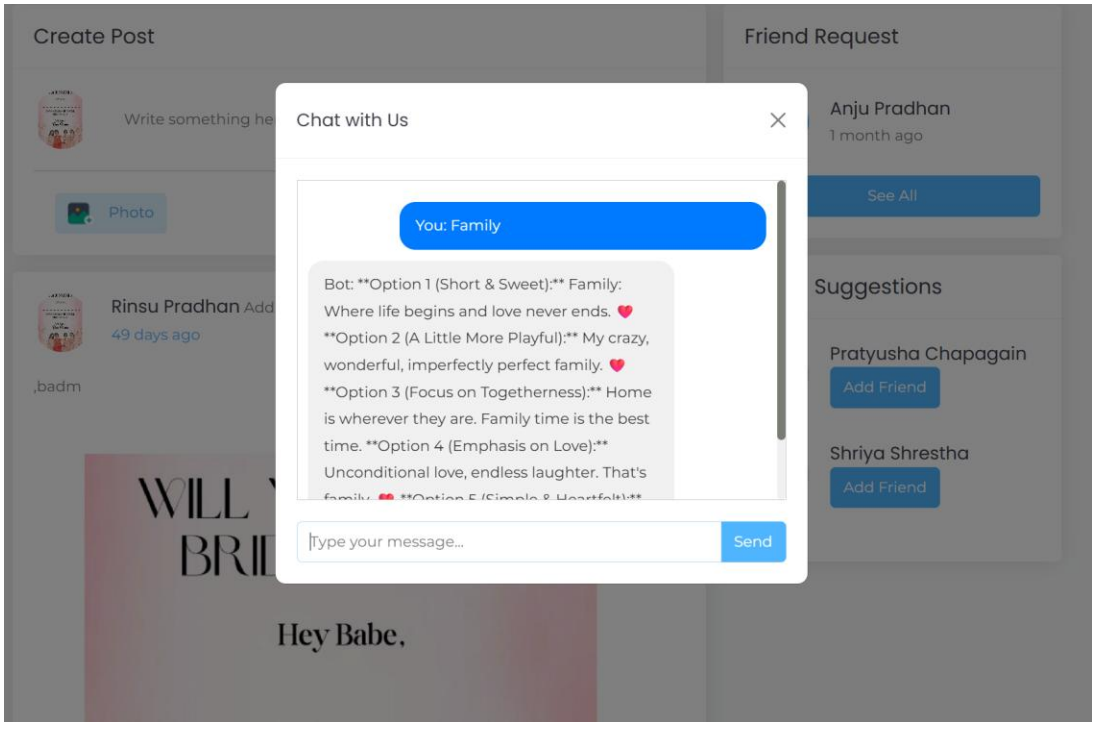

Figure 19: Chatbot UI

Appendix I.B: Gemini API Request and Response

The generateCaption feature in SociablePeak enables users to create AI-powered Instagram captions. It sends a user-provided description to the Gemini API and returns a creative caption. The integration uses Laravel's HTTP client and handles both success and failure responses.

- Input: User-entered text (e.g., "Picnic with friends at the lake").
- Process: Sends a POST request to Gemini API using Laravel.
- Output: AI-generated caption, under 200 characters.
- Fallback: If API fails, a backup caption is generated locally.

Sample API Request

Endpoint:

POST https://generativelanguage.googleapis.com/v1beta/models/gemini-1.5-flash-latest:generateContent?key=YOUR_API_KEY

Request Body:

```
{
  "contents": [
    {
      "parts": [
        {
          "text": "Generate a creative and catchy Instagram caption for this description: Picnic with friends at the lake. Keep it under 200 characters."
        }
      ]
    }
  ]
```

}

Sample API Response


```
{
  "candidates": [
    {
      "content": {
        "parts": [
          {
            "text": "Lakeside laughs, sunshine vibes 🌞🧺 #PicnicPerfect #WeekendChill"
          }
        ]
      }
    }
  ]
}
```


Error Handling:

If the API request fails or the key is missing, a default caption such as “Enjoying beautiful moments outdoors!” is returned.

Appendix I.C: Friend Suggestion

The following function getFriendSuggestions provides friend suggestions for a user by recommending friends of their existing friends who are not yet directly connected to the user. This uses a simple graph traversal logic based on friend relationships stored in the database.

Code

```
private function getFriendSuggestions($userId, $friendIds)
{
 $userId = Auth::id();
   // Step 1: Get the user's friends whose status is not 'pending'
   $myFriends = Friend::where(function($query) use ($userId) {
                    $query->where('user_id', $userId)
                        ->orWhere('friend_id', $userId);
                })
                ->where('status', '!=', 'pending')
                ->get()
                ->map(function($friend) use ($userId) {
                    return $friend->user_id == $userId ? $friend->friend_id : $friend->user_id;
                })
                ->toArray();
   // Example output: [4, 8, 2]

   // Step 2: Retrieve friends of the user's friends
   $friendsOfMyFriends = Friend::where(function($query) use ($myFriends) {
      $query->whereIn('user_id', $myFriends)
          ->orWhereIn('friend_id', $myFriends);
   })
   ->where('status', '!=', 'pending')
   ->get()
   ->map(function($friend) use ($myFriends) {
      return in_array($friend->user_id, $myFriends) ? $friend->friend_id : $friend->user_id;
```

```
    })
    ->unique()
    ->toArray();
    // Example output: [3, 2, 1, 9, 10]

    // Step 3: Filter out friends already connected and the user themselves
    $myFriendsOfFriendsWhoAreNotMyFriends = array_diff($friendsOfMyFriends,
$myFriends, [$userId]);
    // Example difference: [3, 9, 10]

    // Step 4: Fetch suggested friends with profile pictures, limit to 5
    return User::whereIn('id', $myFriendsOfFriendsWhoAreNotMyFriends)
            ->with('profilePicture')
            ->take(5)
            ->get();
}
```

Step 1: Retrieves the current user's friends from the Friend table where the friendship status is not pending, ensuring only confirmed friends are considered.

Step 2: Collects the friends of these friends (second-level connections), again filtering out any pending relationships.

Step 3: Removes any users who are already friends with the current user and excludes the user themselves.

Step 4: Returns up to five user profiles from this filtered list, including their profile pictures, as friend suggestions.