HO CHI MINH UNIVERSITY OF TECHNOLOGY AND EDUCATION

FACULTY FOR INTERNATIONAL TRAINING

Project on Software Engineering

BUILDING A SOCIAL MEDIA WEBSITE THAT APPLIES ML TO BLOCK NUDE IMAGE

Lecturer: Associate Professor Dr. Hoang Van Dung

Group 01:

Nguyen Thanh Toan	20110012
Le Khanh Linh	20110377
Le Tan Loc	20110379

HCMC, December 2023

HO CHI MINH UNIVERSITY OF TECHNOLOGY AND EDUCATION

FACULTY FOR INTERNATIONAL TRAINING

Project on Software Engineering

BUILDING A SOCIAL MEDIA WEBSITE THAT APPLIES ML TO BLOCK NUDE IMAGE

Lecturer: Associate Professor Dr. Hoang Van Dung

Group 01:

Nguyen Thanh Toan	20110012
Le Khanh Linh	20110377
Le Tan Loc	20110379

HCMC, December 2023

ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO
www.fhq.hcmute.edu.vn

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc
----*----**

HCMC, November 2023

FEEDBACK FROM THE SUPERVISOR

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

HCMC, December 2023

Signature and full name of supervisor

ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO
www.fhq.hcmute.edu.vn

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc
----***----

HCMC, November 2023

FEEDBACK FROM THE OPPONENT

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

HCMC, December 2023

Signature and full name of opponent

ACKNOWLEDGMENT

First of all, we would like to express our sincere gratitude to the Faculty for International Training of Ho Chi Minh City University of Technology and Education for providing us with the opportunity to study, develop a deep knowledge base and carry out this project.

In addition, we would like to extend our deepest thanks to Mr. Hoang Van Dung. Throughout the long process of studying and completing the project, Mr. Dung has devotedly and enthusiastically guided us from the beginning to the end. This will be a great asset for us before stepping into a new life.

However, knowledge is limitless and with our limited abilities, we have tried our best to complete this project. Therefore, it is difficult to avoid mistakes. We hope to receive your constructive feedback so that we can learn from our experiences and improve our product in the best way possible.

Lastly, we would like to express our deepest gratitude to Mr. Hoang Van Dung and the esteemed faculty members of the Faculty for High Quality Training at Ho Chi Minh City University of Technology and Education. We wish you all good health. We sincerely thank you!

We sincerely thank you.

Table of Contents

List of Figures

List of Tables

CHAPTER I. INTRODUCTION

1.1. Project overview

1.1.1. Project name

Building a social media website that applies ML to block nude images

1.1.2. Project objective

The project develops a robust social media website that leverages machine learning algorithms to effectively detect and block nude images, thus creating a fundamental platform that prioritizes user safety and content appropriateness. The objective of the project is two-fold:

- To develop a foundational social media website that offers essential features and functionalities for users to connect, share content, and interact with each other. This includes creating user profiles, enabling posting and commenting, implementing a news feed, and facilitating social networking capabilities.
- To apply machine learning techniques to effectively detect and block nude or sexually explicit images from being uploaded and shared on the social media platform. By leveraging advanced image recognition algorithms and content filtering mechanisms, the project aims to create a safe and secure environment for users, preventing the dissemination of inappropriate or offensive content.

By achieving these objectives, the project aims to provide users with a user-friendly and engaging social media experience while prioritizing their safety and well-being. The integration of machine learning algorithms will enhance the platform's ability to automatically identify and block explicit content, ensuring that users can confidently usethe platform without encountering inappropriate or offensive material.

1.1.3. The urgency of the project

In the digital age, social networks have become an indispensable part of our daily lives. The topic " Building a social media website that applies ML to block nude images" - Building a social media website that applies ML to block nude images - is urgent for many important reasons.

- Enhance user experience: Using machine learning to process social media posting data can improve user experience.
- Detect harmful information: Social networks are an environment where harmful information, fake news and spam can spread quickly. Applying machine learning to post data can help detect and prevent fraud, fake news, and other

malicious activity. This ensures the social media environment is safe and trustworthy for users.

- Business Potential: Building a social networking site that applies machine learning to process posting data offers huge business potential. Providing relevant content and increasing social media engagement can attract users, ads, and revenue.

In summary, the topic Building a social media website that applies ML to block nude images" is urgent in improving user experience, automating the process of processing posting data, and detecting malicious information. and create more business potential.

1.1.4. Scientific and practical significance

Scientific significance:

- Research and development in the field of Machine Learning: This project contributes to the application and development of Machine Learning techniques in processing social media post data

- Research on security and privacy: Social networks are an environment that requires attention to security and privacy of users. This project can contribute to the research and development of measures to protect user data and ensure information security on social networks.

- Research on online social media: Building a social media website requires research on how people interact and communicate online, as well as the social, psychological and behavioral factors of users in online environment.

Practical significance: this project benefits users by enhancing experiences, personalizing content, ensuring information safe and creating opportunities to connect in a social network environment.

1.1.5. Research purposes

The purpose of the project is to build a new social network platform, optimize posting data processing, improve user experience, and contribute to research and development of technology related to intelligence application artificial intelligence and machine learning in this area.

1.1.6. Research subjects

Research the design and build of social media websites applying ML to process input images

1.1.7. Database

MongoDB

1.1.8. Front-end

React

1.1.9. Back-end

NodeJS, Express

1.1.10. AI integration

OpenCV, FastAPI

1.2. Survey the current situation

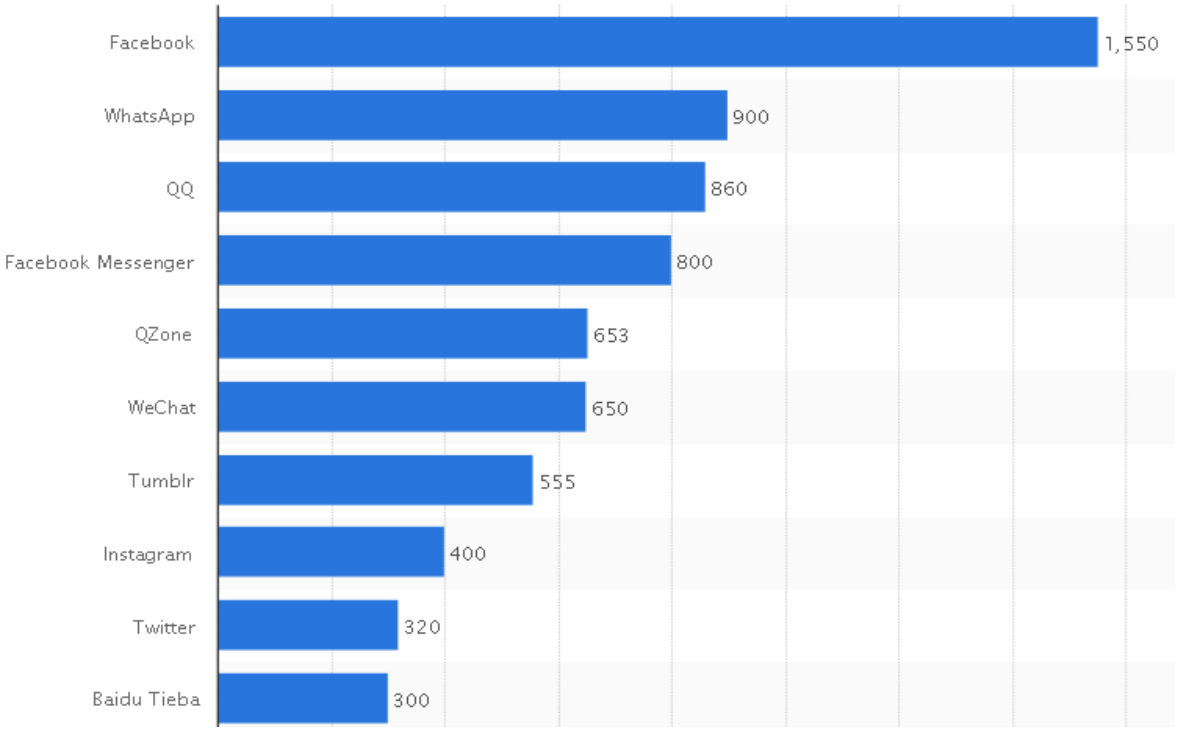


Figure 1: Top 10 largest social media website in the world today

1.2.1. Twitter

- Interface: Twitter interface includes Home page, explore page, Notification page, Message page, Lists page, Communities page, Profile page.

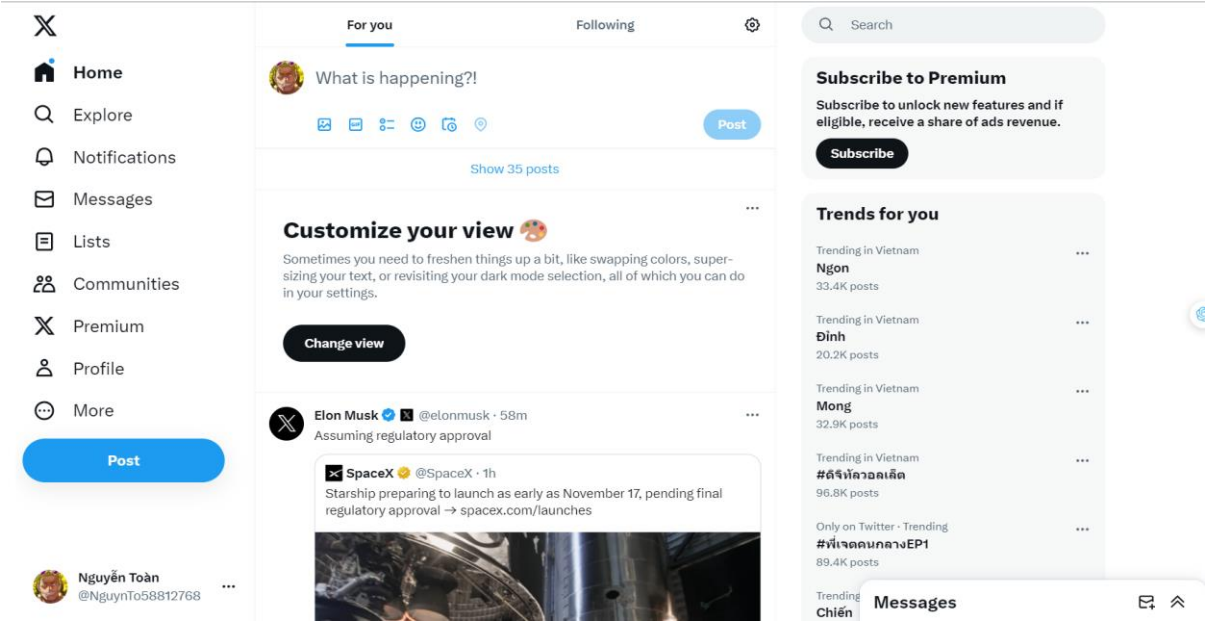


Figure 2: Home page Twitter

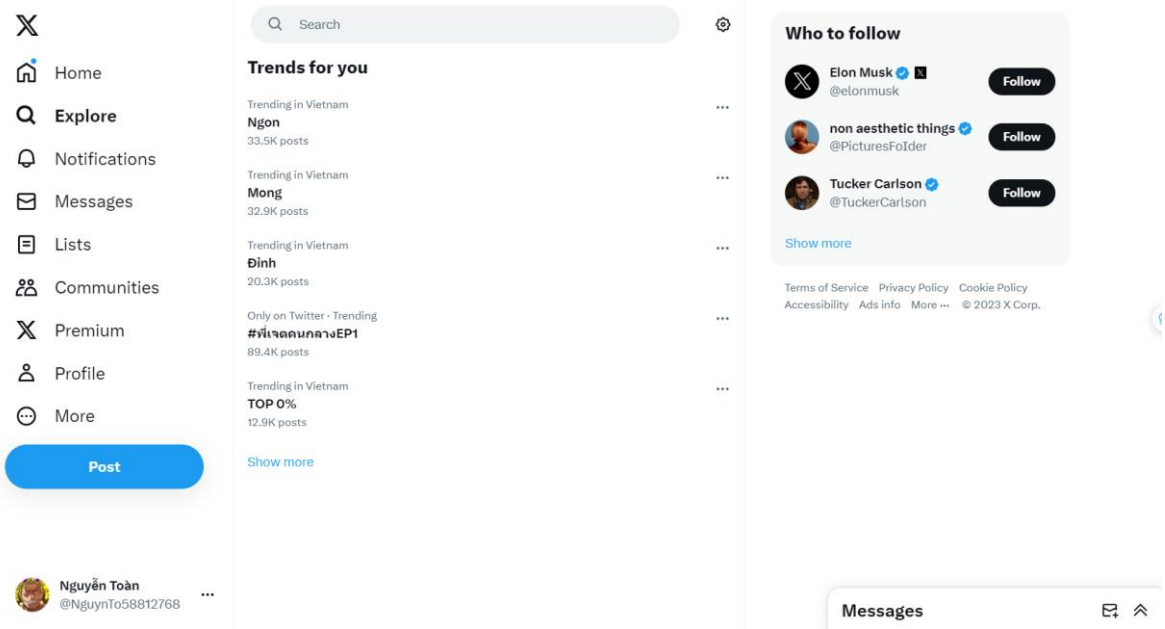


Figure 3: Explore page Twitter

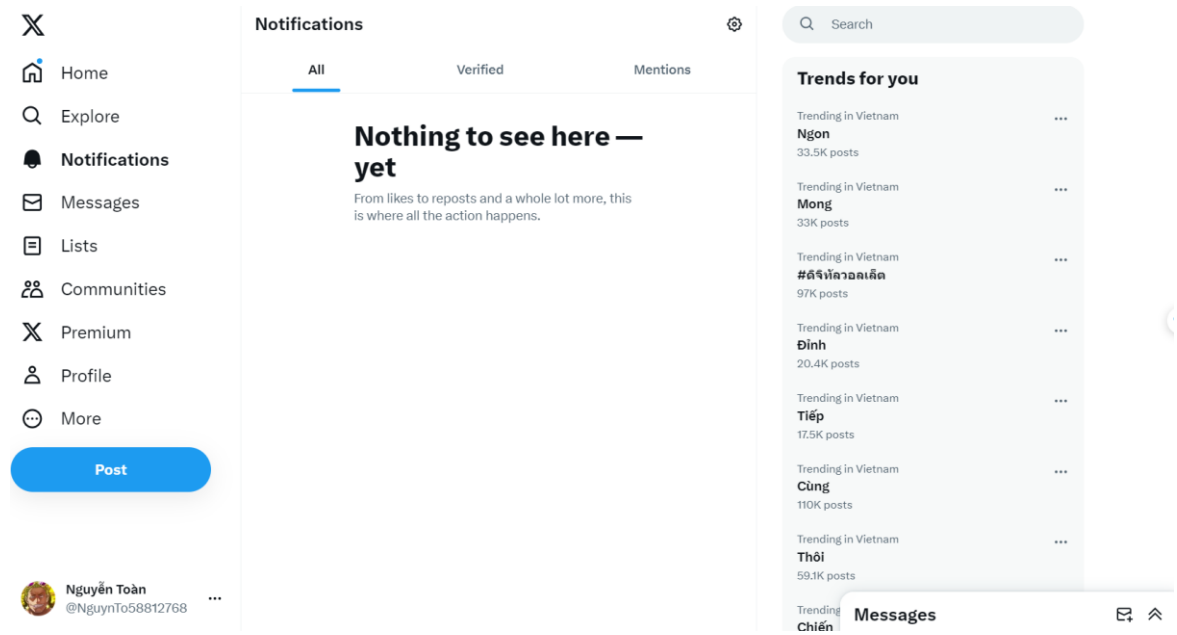


Figure 4: Notification page Twitter

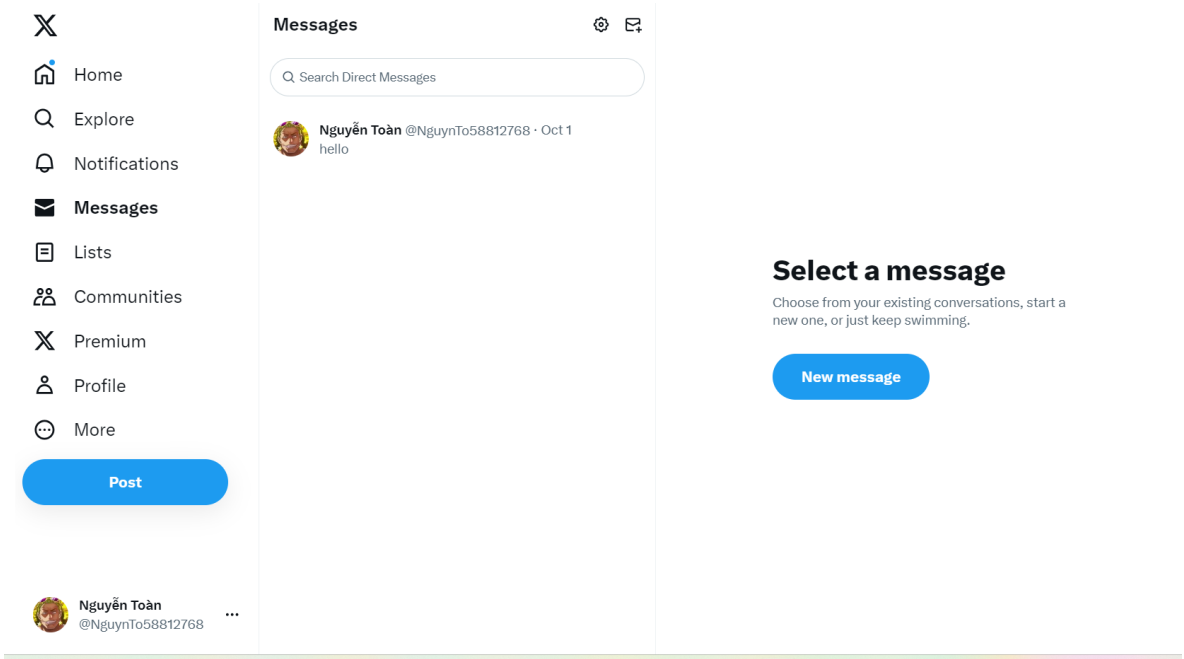


Figure 5: Message page Twitter

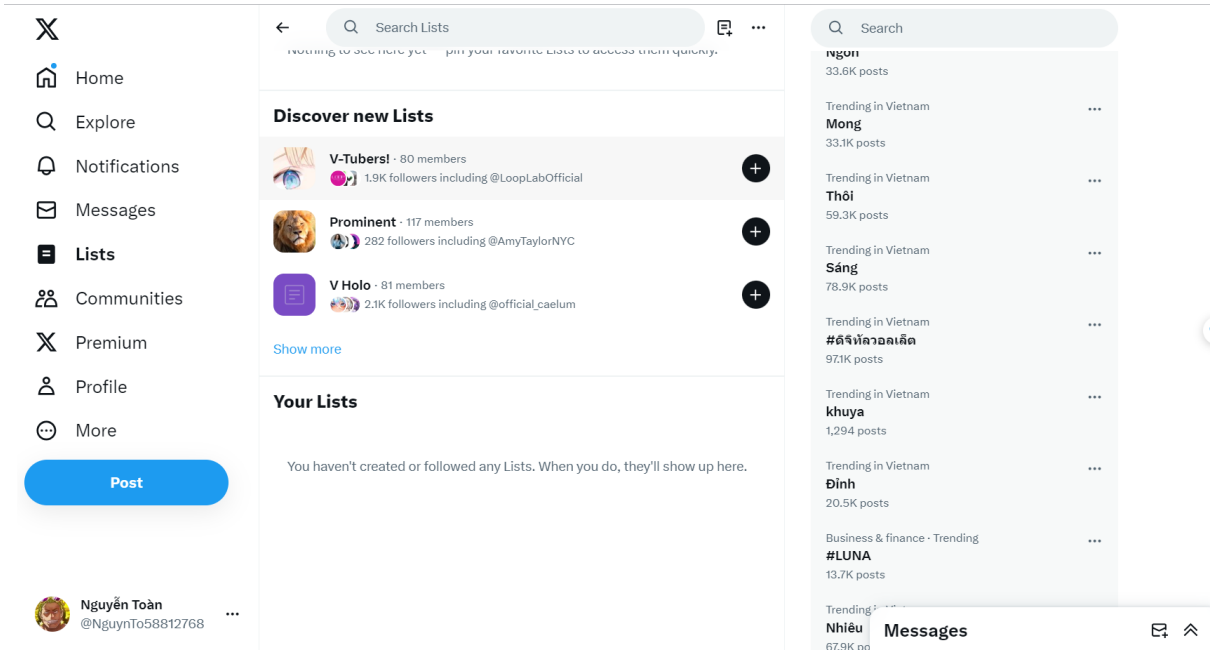


Figure 6: Lists page Twitter

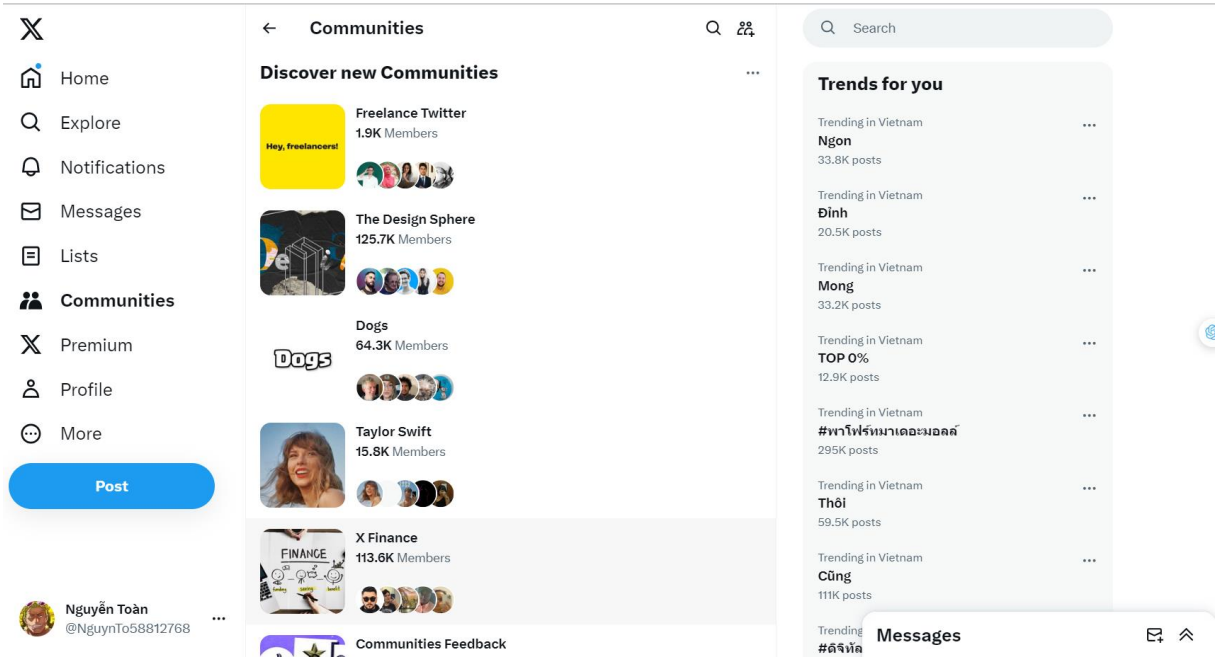


Figure 7: Communities page Twitter

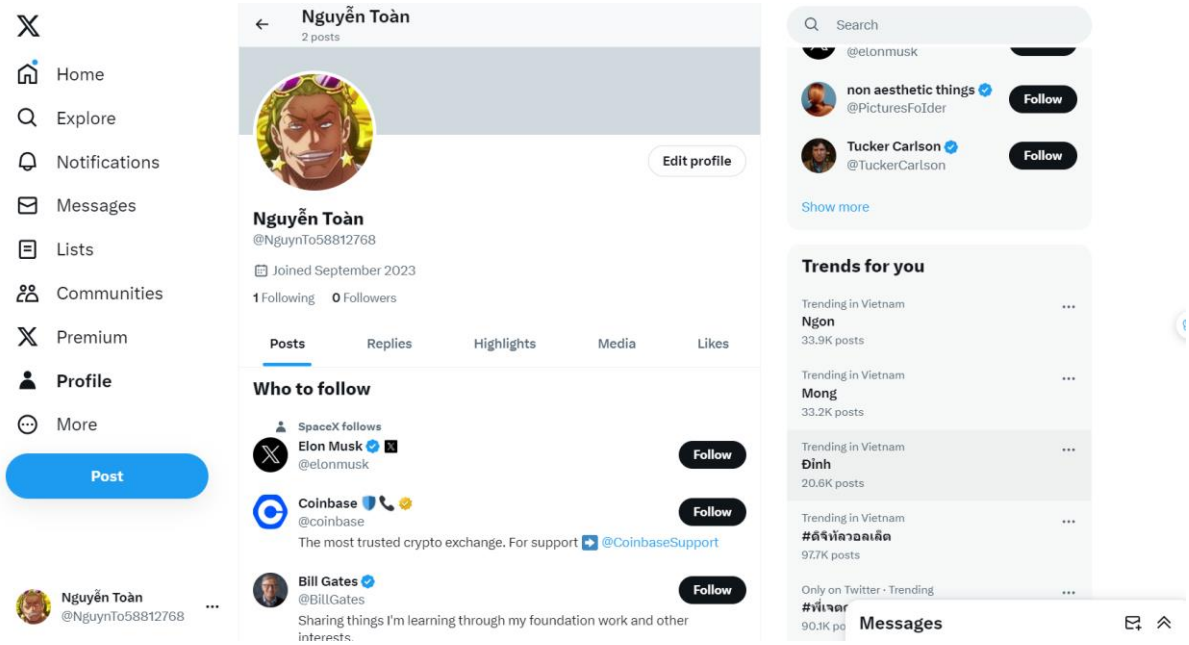


Figure 8: Profile page Twitter

- Characteristic:

- Microblogging: Twitter is a microblogging platform that allows users to write and share short posts called tweets, limited to a maximum of 280 characters. This helps users convey messages concisely and quickly.
- Public: Twitter is primarily used publicly, allowing users to view and interact with other users' tweets. This creates an environment of open and open communication.
- Following and followers: Users can follow other accounts on Twitter to see their tweets on the timeline. At the same time, users can also have their own followers, someone who follows their account.
- Hashtags: Twitter uses hashtags (hashtag #) to classify and group tweets related to the same topic. Hashtags allow users to search and follow specific content and topics.
- Social interaction: Twitter facilitates users' social interaction through activities such as liking, commenting, retweeting and replying to tweets. This creates an environment of diversity, excitement and discussion.
- Real-time information: With quick updates and a continuous timeline, Twitter provides real-time information about events, news, and trends happening around the world.

- Communication and social impact: Twitter has become an important tool for communication and social impact. It allows users to share messages, promote opinions, give information and participate in social debates and campaigns.

- Advantage:

- Speed and convenience: Twitter allows users to share messages concisely and quickly with a 280 characters limit allowing information to be conveyed quickly.
- Follow news sources and events: Twitter offers users the ability to follow accounts, hashtags, and lists related to specific fields, news sources, or events.
- Communication and social networking: Twitter provides a social communication environment, allowing users to connect, interact and interact with each other.
- Advertising and Marketing: Twitter is a powerful platform for advertising and marketing. Businesses and individuals can use Twitter to promote their products, services, events, and content
- Social influence and impact: Twitter has become an important tool for social influence and impact. Users can share their views, opinions and messages with a large crowd.
- Discovery and diversity: Twitter provides a discovery experience, allowing users to discover diverse content, ideas, information and perspectives from diverse sources around the world.

- Disadvantage:

- Limited expression: With a limit of 280 characters per tweet, users may have difficulty conveying opinions or messages in full and detailed terms. This can limit your ability to express complex ideas or detailed information.
- With discovery and following, Twitter can create emotional and trending dependence. Users may be affected by the pressure to receive likes, attention and approval from others, which can lead to imbalance and negative effects on mood and self-confidence.

1.2.2. Instagram

- Interface: The Instagram interface is simple, clean, and focused on images. There is a home screen, personal page, navigation menu and function buttons such as "Like", "Comment" and "Share". The explore page helps users discover new content.

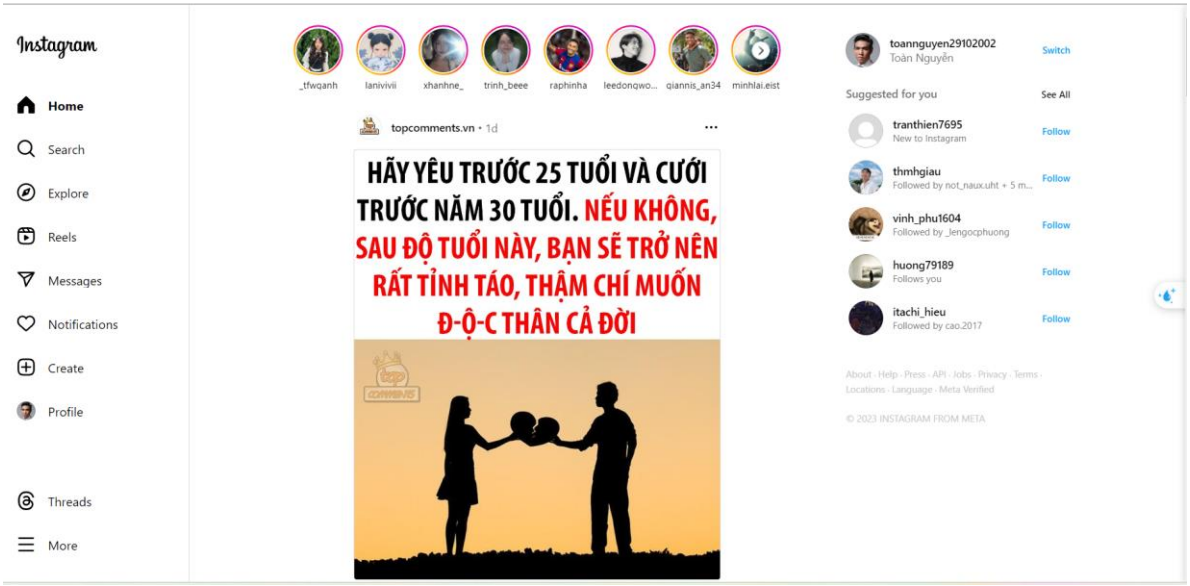


Figure 9: Home page of Instagram

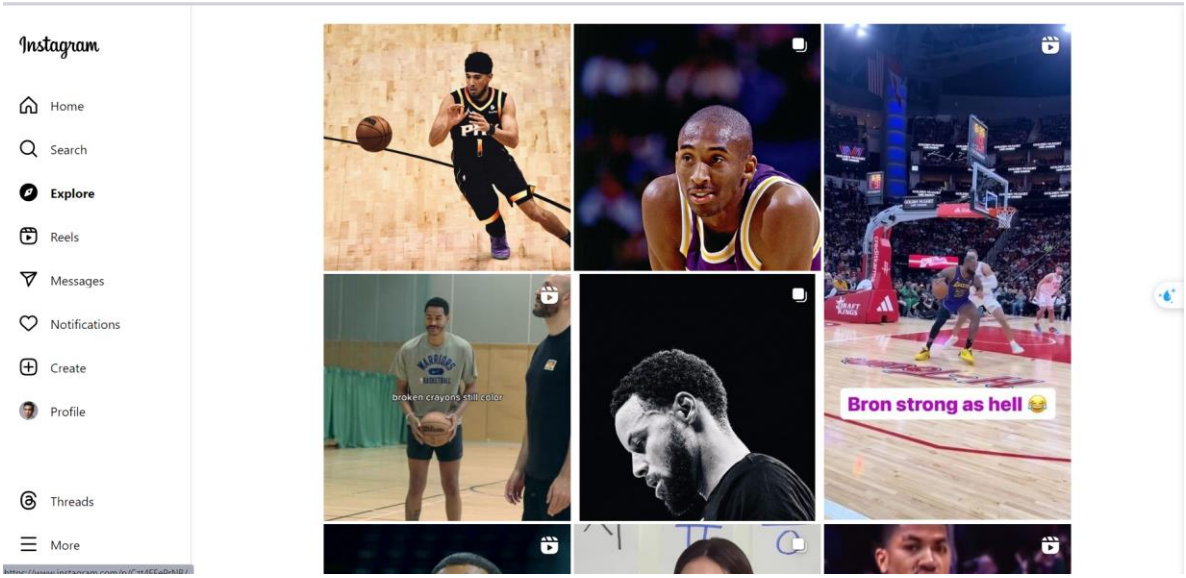


Figure 10: Explore page of Instagram

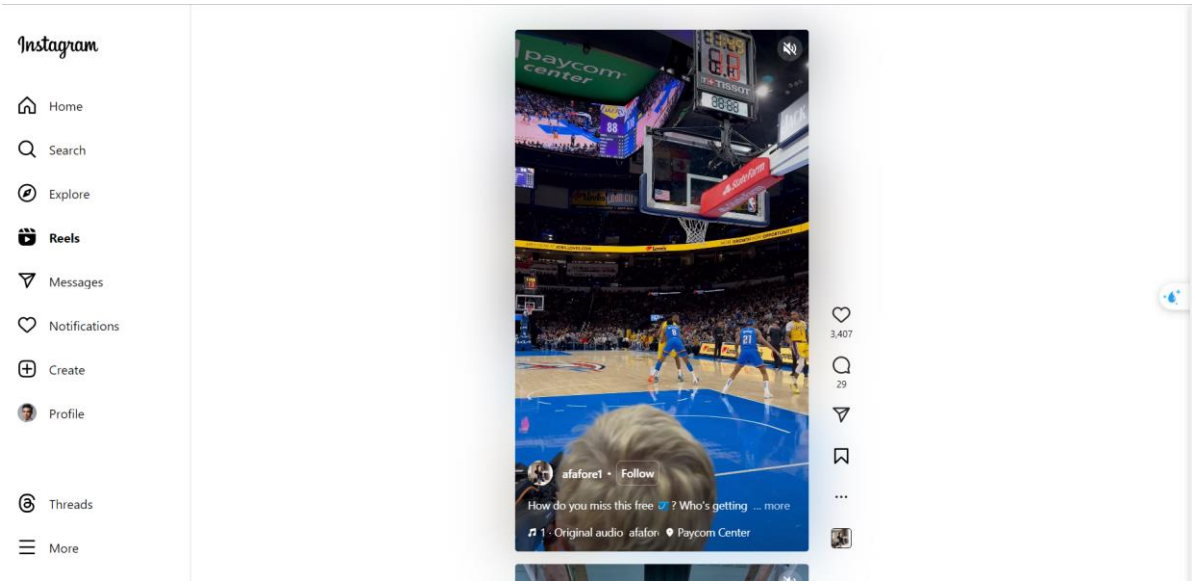


Figure 11: Reels page of Instagram

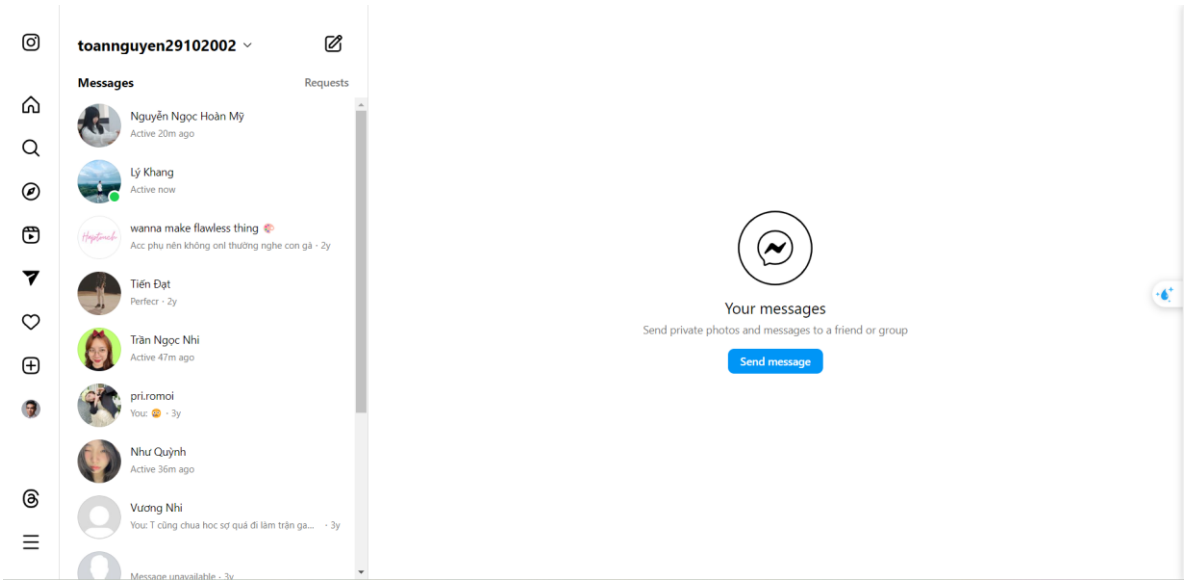


Figure 12: Message page of Instagram

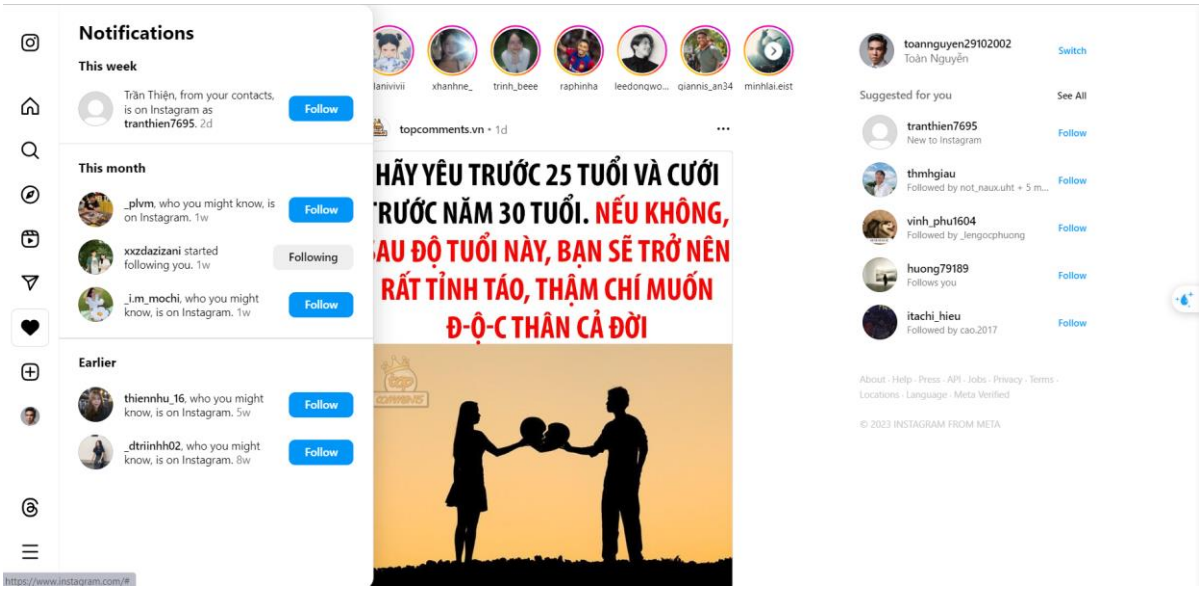


Figure 13: Notification page of Instagram

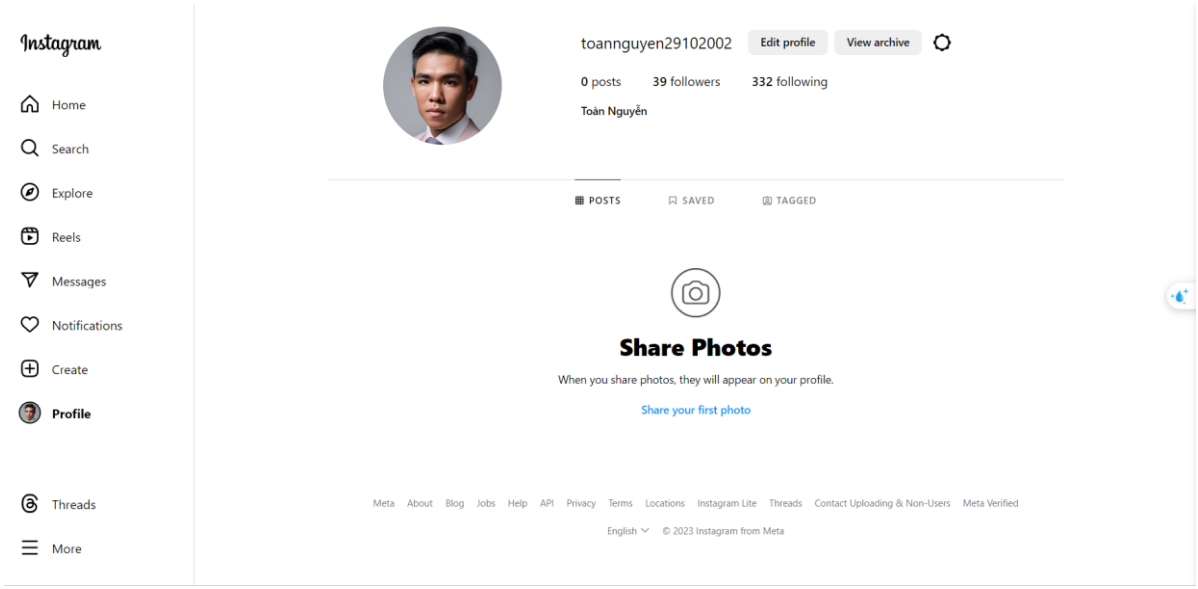


Figure 14: Profile page of Instagram

- Characteristics:

- Photos and videos are the focus: Instagram focuses on sharing photos and videos. Users can post photos and videos to their profile, and view visual content from others they follow. This creates an intuitive and engaging experience.
- Filters and editing: Instagram offers a wide range of filters and editing tools so users can create unique, interesting and beautiful images and videos. This allows users to customize their content and create their own style.

- Social networks and connections: Instagram is a social network, allowing users to connect with each other, follow people they are interested in, and interact through liking, commenting, and sharing content. This creates a diverse and active online community.
- Interactive communication: Instagram allows users to interact directly with each other through the "Stories" and "Direct Messages" features. Users can share short stories, images and videos for a limited time, and send private messages to each other.
- Content Discovery: Instagram offers a "Explore" page that allows users to discover new content, search accounts, and participate in trends. This helps users expand their understanding and interaction with the larger community.

- Advantages:

- Visual-focused platform: Instagram is primarily focused on visual content, such as photos and videos. This makes it a great platform for creative expression and sharing visually appealing content.
- Creative tools and filters: Instagram offers a wide range of creative tools, filters, and editing options that allow users to enhance their photos and videos. This enables users to showcase their creativity and create visually stunning content.
- Stories and live streaming: Instagram's Stories feature allows users to share ephemeral content that disappears after 24 hours. It provides a more casual and real-time way of sharing moments with followers. Additionally, Instagram offers live streaming capabilities, allowing users to engage with their audience in real-time.

- Disadvantages:

- Algorithmic feed: Instagram uses an algorithm to determine the content shown in users' feeds. This can sometimes result in users missing out on posts from accounts they follow, leading to a less chronological and potentially less personalized experience.
- Limited link sharing: Unlike some other social media platforms, Instagram has limitations on link sharing. Only one link can be included in a user's bio, making it challenging to share multiple links or drive traffic to external websites.
- Privacy concerns: As with any social media platform, privacy concerns exist on Instagram. Users need to be cautious about the information they share and the privacy settings they choose to protect their personal data.

- Influence on mental health: The highly curated nature of Instagram, along with the emphasis on likes, followers, and popularity, can contribute to feelings of anxiety, low self-esteem, and social comparison.

1.2.3. Facebook

- Interface: Facebook's interface is designed to be simple and easy to use. It has a top bar that provides quick access to important features like feed, profile, chat, notifications, and settings. Your News Feed shows posts, photos, and videos from friends, following pages, and groups. Personal pages allow users to customize their personal profiles and share personal information. Facebook also offers private and group messaging features through its chat site. However, the Facebook interface may change over time and over version of use, so a specific review of the current interface requires direct access to the Facebook website or app.

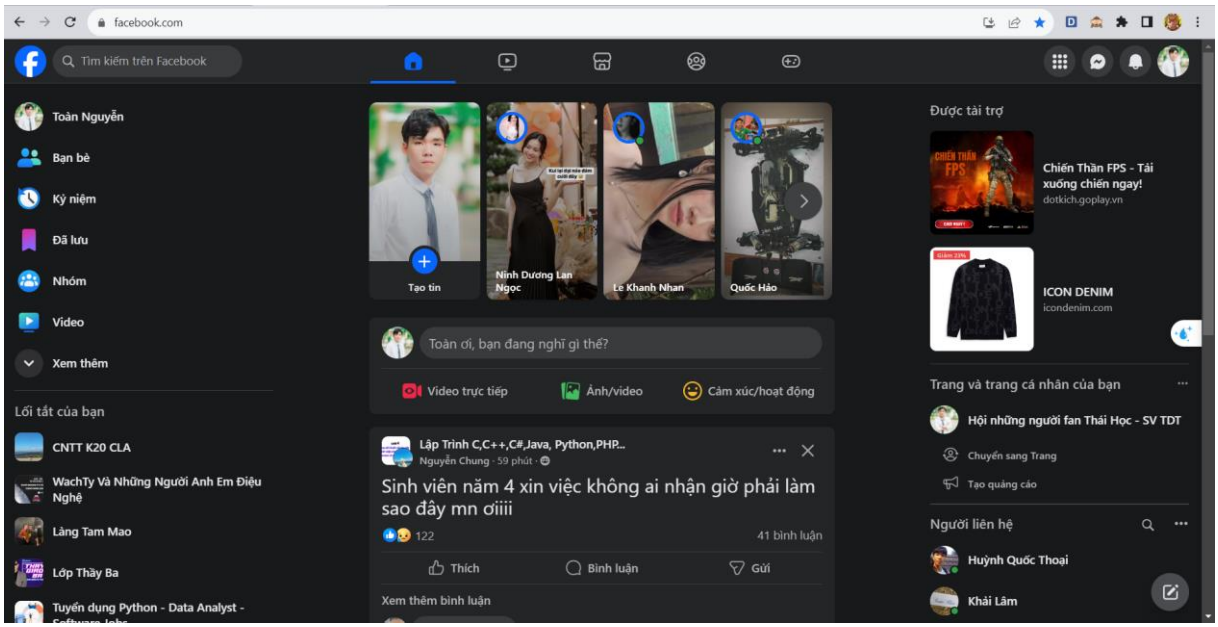


Figure 15: Home page Facebook

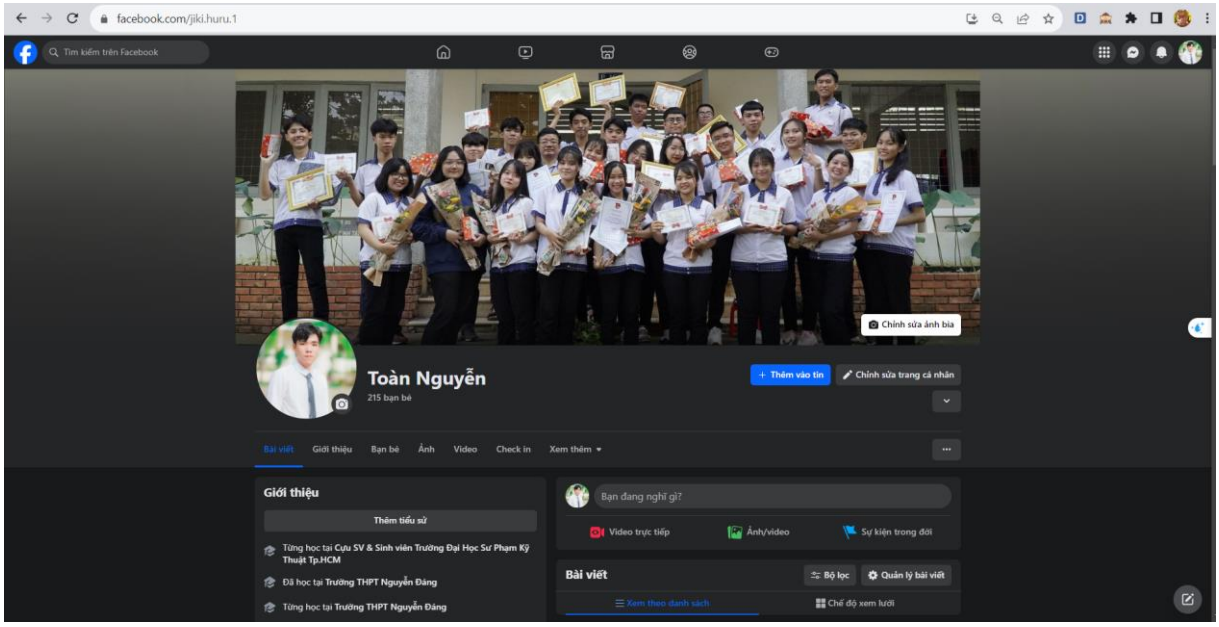


Figure 16: Profile page Facebook

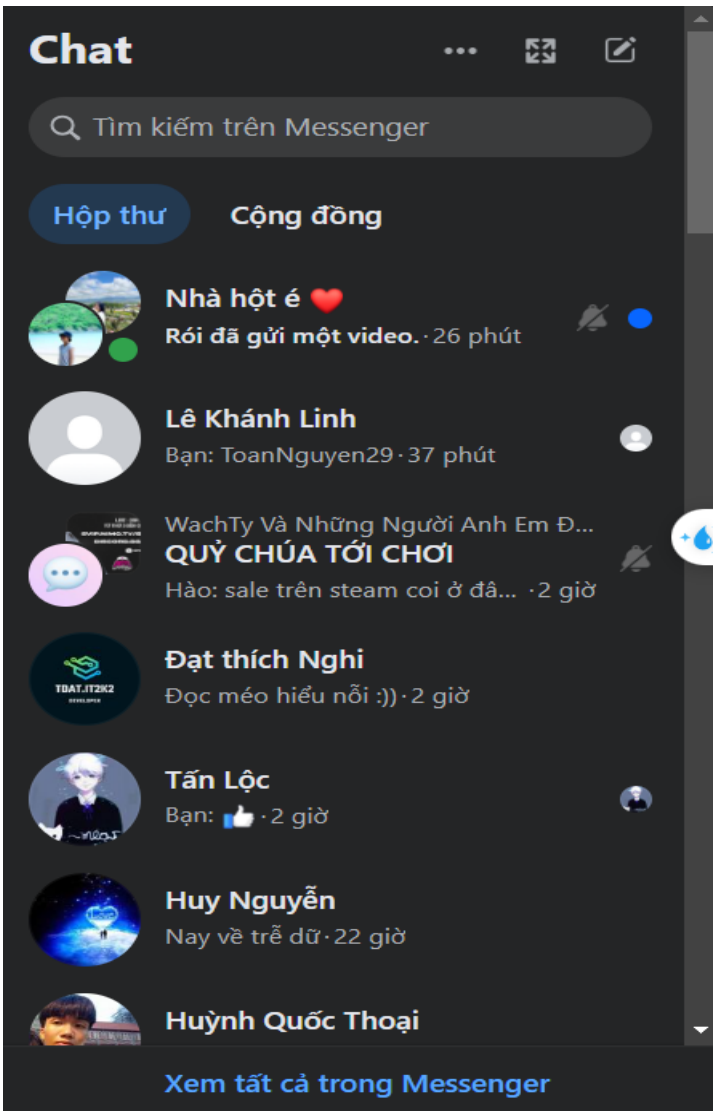


Figure 17: Chat list Facebook

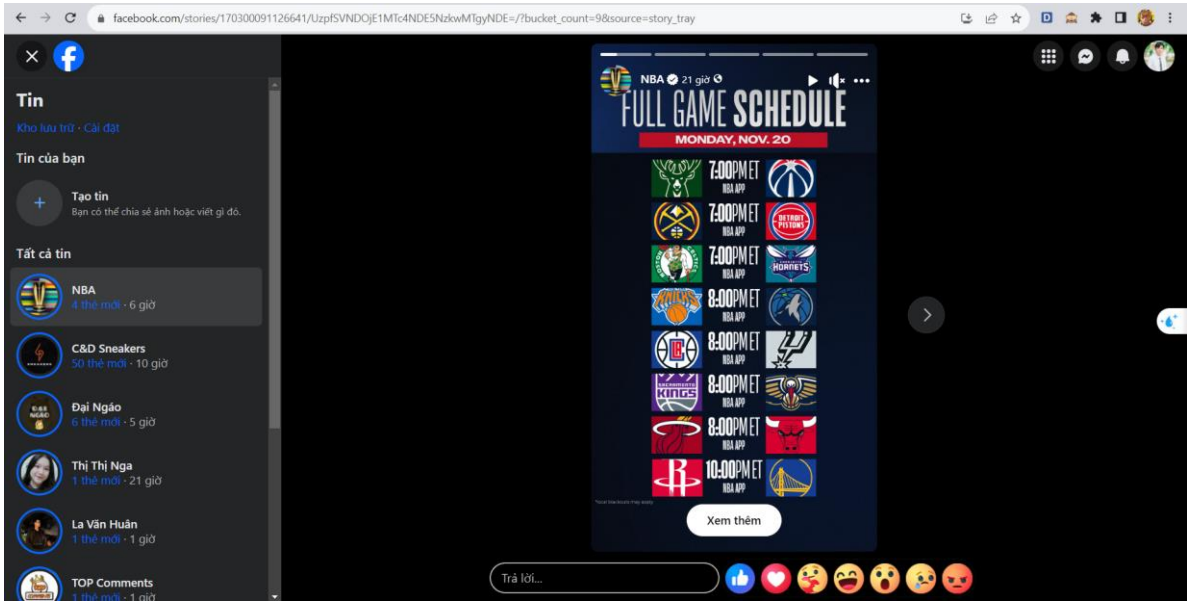


Figure 18: Story page Facebook

Figure 19: Post form Facebook

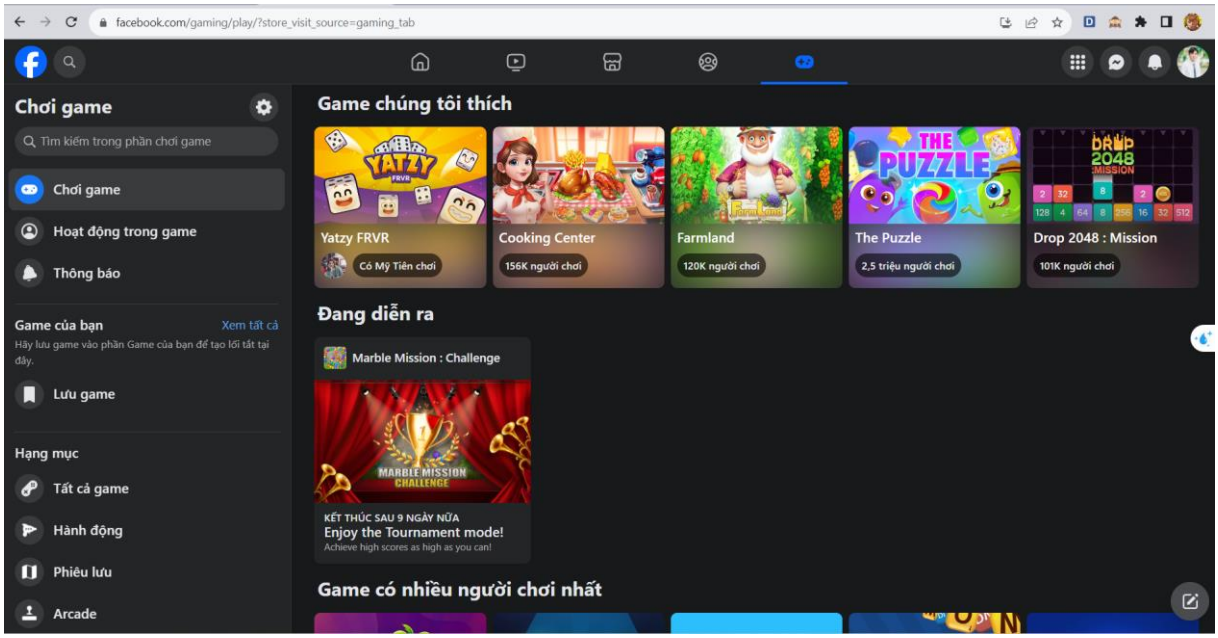


Figure 20: Game page Facebook

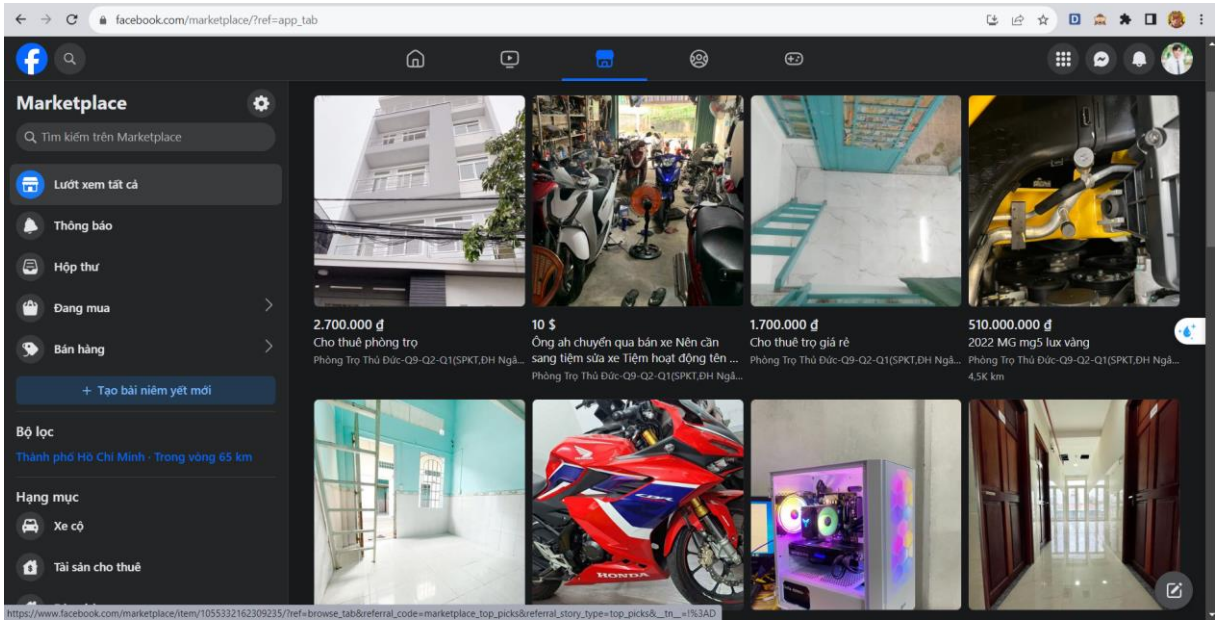


Figure 21: Shop page Facebook

- Characteristics:

- Social networking platform
- User profiles with customizable information and privacy settings
- News Feed displaying curated content
- Likes, comments, and shares for interaction
- Groups and Pages for communities and organizations
- Integrated messaging platform (Messengers)

- Event creation and management
- Advertising and business tools
- Privacy settings for user control
- Accessible through mobile and web applications.

- Advantage:

- User scale: Facebook has the largest number of users among social networking sites, this creates a large opportunity to connect and find acquaintances with many people.
- Variety of features: Facebook offers a variety of features such as news feeds, personal pages, groups, business pages and events. This creates a rich and flexible experience for users.
- Integrated messaging: Facebook integrates messaging features through Messenger, allowing users to send messages, make video calls and share files conveniently.
- Business utility: With advertising features and business pages, Facebook is a useful platform for businesses to reach potential customers, build their brand, and interact with customers.
- Communities and groups: Facebook is powerful in building communities and creating groups based on common interests. This allows users to find people with similar interests and participate in similar discussions and activities.

- Disadvantage:

- Privacy: Facebook has faced many controversies regarding privacy and use of user data.
- Negativity and misinformation: Facebook has become a place for the spread of misinformation, fake news and negative content.
- Abuse and dependence: Facebook use can become an abusive habit, leading to loss of time and concentration.
- Unhealthy environment: Facebook sometimes has an unhealthy online environment with the presence of personal bullying, insults and negative feedback.

1.3. Theoretical basis

1.3.1. React

React is a widely-used JavaScript library that revolutionizes the way user interfaces (UIs) are built in web applications. It offers a declarative and efficient approach to creating interactive UI components, making it a popular choice among

developers. This article provides an overview of React, highlighting its key features and benefits.

- Declarative and Component-Based Architecture: React follows a declarative programming paradigm, allowing developers to describe how the UI should look based on the application's state. This simplifies UI development by abstracting away the low-level DOM manipulation. React's component-based architecture enables the creation of reusable UI elements, making it easier to build complex UIs from smaller, self-contained components.
- Virtual DOM for Efficient Rendering: React utilizes a virtual DOM, a lightweight representation of the actual DOM, to efficiently track and update UI changes. When the application state changes, React intelligently updates only the necessary components, minimizing performance overhead. This approach significantly improves rendering speed and enhances the user experience.
- Unidirectional Data Flow: React follows a unidirectional data flow pattern, where data flows from parent components to child components through props. This ensures a clear and predictable data flow, making it easier to understand and debug the application. By maintaining a one-way data flow, React promotes better code organization and maintainability.
- Lifecycle Methods: React provides a set of lifecycle methods that allow developers to perform actions at specific stages of a component's lifecycle. These methods enable initialization, updating, and unmounting of components, providing fine-grained control over the UI behavior. Developers can leverage lifecycle methods to handle side effects, optimize performance, and integrate with external libraries.
- Thriving Ecosystem and Community Support: React has a vibrant ecosystem with a vast array of community-contributed libraries, tools, and frameworks. These resources extend React's capabilities and provide solutions for common UI development challenges. The strong community support ensures regular updates, bug fixes, and a wealth of learning resources, making React an attractive choice for developers.

React has revolutionized the way developers build dynamic user interfaces in web applications. Its declarative nature, component-based architecture, efficient rendering with the virtual DOM, unidirectional data flow, and robust lifecycle methods make it a powerful tool for UI development. With its thriving ecosystem and strong community support, React continues to evolve and empower developers to create engaging and responsive UIs.

1.3.2. NodeJS

Node.js is a powerful server-side JavaScript runtime environment that has gained significant popularity in recent years. This article provides an introduction to Node.js, highlighting its key features and advantages for building scalable and efficient server-side applications.

- JavaScript on the Server: Node.js allows developers to run JavaScript code on the server, extending the capabilities of JavaScript beyond the browser. This enables full-stack developers to use a single programming language for both frontend and backend development, streamlining the development process and promoting code reuse.
- Asynchronous and Non-Blocking I/O: One of the defining features of Node.js is its asynchronous and non-blocking I/O model. This means that Node.js can handle multiple concurrent requests without blocking the execution of other code. As a result, Node.js applications can handle high levels of concurrent traffic and deliver excellent performance, making it ideal for building scalable and real-time applications.
- Event-Driven Architecture: Node.js is built on an event-driven architecture, where actions are triggered by events and event handlers are used to respond to those events. This event-driven approach allows developers to write highly efficient and responsive code. It also enables the creation of real-time applications, such as chat applications or collaborative tools, where instant updates are crucial.
- Vast Ecosystem and NPM: Node.js has a vast ecosystem of open-source libraries and tools, thanks to its package manager called npm (Node Package Manager). With npm, developers can easily access and incorporate a wide range of pre-built modules into their applications, reducing development time and effort. The extensive npm registry offers thousands of packages for various functionalities, making Node.js a highly productive environment.
- Scalability and Performance: Node.js's lightweight and efficient architecture, coupled with its non-blocking I/O model, makes it highly scalable. It can handle a large number of concurrent connections with minimal resource consumption. This scalability, along with its fast response times, makes Node.js well-suited for building high-performance applications that need to handle heavy traffic loads.

Node.js has emerged as a powerful runtime environment for server-side JavaScript development. With its ability to handle asynchronous and non-blocking

I/O operations, event-driven architecture, vast ecosystem, and scalability, Node.js has become a preferred choice for building scalable and efficient server-side applications. Its unified JavaScript language across the frontend and backend, along with the extensive npm package manager, further enhances developer productivity. Node.js continues to evolve and revolutionize the way developers build server-side applications in the JavaScript ecosystem.

1.3.3. MongoDB

MongoDB is a popular NoSQL (Not Only SQL) database that offers a flexible and scalable solution for managing and storing data. This article provides an introduction to MongoDB, highlighting its key features and advantages for modern application development.

- Document-Oriented and Schema-less: MongoDB is a document-oriented database, which means it stores data in flexible, JSON-like documents. This allows developers to store and retrieve data in a way that closely resembles their application's data model. MongoDB's schema-less nature eliminates the need for predefined schemas, offering greater flexibility in adapting to evolving data structures.
- Scalability and High Performance: MongoDB is designed to scale horizontally, allowing applications to handle large volumes of data and high traffic loads. It supports automatic sharding, which distributes data across multiple servers, enabling seamless scalability. Additionally, MongoDB's efficient indexing and query optimization mechanisms contribute to high performance even with extensive data sets.
- Rich Querying and Indexing Capabilities: MongoDB provides a powerful query language that supports a wide range of querying operations, including complex aggregations and geospatial queries. It also supports indexing, allowing developers to optimize query performance by creating indexes on specific fields. These features enable efficient data retrieval and analysis.
- Replication and High Availability: MongoDB offers built-in replication, allowing data to be replicated across multiple servers for redundancy and high availability. Replication ensures that if one server fails, data can still be accessed from other replicas, providing fault tolerance and data durability. This feature is crucial for mission-critical applications that require continuous availability.
- Comprehensive Ecosystem and Community Support: MongoDB has a vibrant ecosystem with extensive community support. It provides official drivers for various programming languages, making it easy to integrate MongoDB into

different application stacks. Additionally, MongoDB offers a rich set of tools, libraries, and frameworks that enhance development productivity and ease of use.

MongoDB is a versatile and scalable NoSQL database that offers flexibility, high performance, and robust features for modern application development. Its document-oriented model, scalability options, rich querying capabilities, and replication support make it suitable for a wide range of use cases. With its comprehensive ecosystem and strong community support, MongoDB continues to empower developers to build data-driven applications efficiently and effectively.

1.3.4. Express

Express is a fast and minimalist web application framework for Node.js. It provides a simple and flexible way to build web applications and APIs. This article introduces Express, highlighting its key features and benefits for web development.

- Lightweight and Unopinionated: Express is known for its lightweight and unopinionated nature, allowing developers to have greater control over their application's structure and logic. It provides just the essential features needed for web development, without imposing rigid conventions or unnecessary abstractions. This flexibility makes Express suitable for a wide range of projects and enables developers to tailor their applications according to specific requirements.
- Routing and Middleware: Express offers a robust routing system that allows developers to define routes and handle HTTP requests efficiently. It supports various HTTP methods, such as GET, POST, PUT, and DELETE, enabling the creation of RESTful APIs. Additionally, Express's middleware architecture allows developers to define reusable functions that process requests and responses, providing a modular and extensible approach to handling application logic.
- Templating Engines and Views: Express integrates seamlessly with various templating engines, such as EJS, Pug (formerly Jade), and Handlebars. These templating engines enable developers to generate dynamic HTML views by combining data with pre-defined templates. Express simplifies the process of rendering views and passing data to them, making it easier to build server-side rendered web applications.
- Middleware Ecosystem: Express benefits from a rich ecosystem of middleware modules that extend its functionality. Developers can leverage these middleware

modules to add features like authentication, logging, error handling, and more to their applications. This modular approach promotes code reusability and accelerates development by leveraging existing solutions.

- Integration with Node.js: Express seamlessly integrates with Node.js, leveraging its asynchronous and non-blocking I/O capabilities. This allows developers to take advantage of Node.js's event-driven architecture and scalability features while building web applications. Express plays a crucial role in bridging the gap between the raw HTTP capabilities of Node.js and the higher-level functionality required for web development.

Express is a lightweight and flexible web application framework that empowers developers to build web applications and APIs with ease. Its simplicity, routing system, middleware architecture, templating engine support, and seamless integration with Node.js make it a popular choice for building scalable and efficient web applications. With its extensive middleware ecosystem and a large community of developers, Express continues to evolve and contribute to the vibrant Node.js ecosystem.

1.3.5. FastAPI

FastAPI is a fast, user-friendly, and high-performance Python framework for building web APIs. Built on the ASGI (Asynchronous Server Gateway Interface) platform, FastAPI combines the power of Python with an asynchronous architecture, allowing for concurrent handling of multiple requests without sacrificing performance.

- Speed and High Performance: FastAPI is designed to deliver exceptional speed and performance. By leveraging Python's type hints and modern asynchronous programming principles, FastAPI achieves high throughput and low latency. It utilizes the performance benefits of asynchronous programming to handle a large number of requests concurrently, making it ideal for highly demanding applications.
- Easy to Use and Developer-Friendly: FastAPI focuses on developer productivity and ease of use. It offers a straightforward and intuitive API design, making it simple to define endpoints and request/response models. FastAPI takes advantage of Python's static type checking to provide automatic data validation, serialization, and documentation generation, reducing boilerplate code and enhancing code maintainability.

- Asynchronous and Pythonic: FastAPI embraces Python's asynchronous programming capabilities, allowing you to write asynchronous code using Python's async and await syntax. This enables efficient handling of I/O-bound operations, such as accessing databases or making HTTP requests, without blocking the execution of other tasks. FastAPI's asynchronous nature contributes to its exceptional performance and scalability.
- Interactive Documentation and OpenAPI Support: FastAPI generates interactive and user-friendly API documentation automatically. With the integrated Swagger UI and OpenAPI (formerly known as Swagger) support, developers can explore and test API endpoints directly from the documentation. This feature simplifies API consumption and promotes collaboration between frontend and backend teams.
- Extensible and Integrates Well with Other Tools: FastAPI provides seamless integration with other Python libraries and tools. It supports the use of popular ORMs (Object-Relational Mappers) like SQLAlchemy and databases such as PostgreSQL and MongoDB. FastAPI can also be easily integrated with authentication systems, dependency injection frameworks, and asynchronous task queues, expanding its capabilities and adaptability.

FastAPI is a high-performance Python framework that empowers developers to build efficient and scalable web APIs. Its speed, ease of use, asynchronous nature, automatic documentation generation, and seamless integration with other Python tools make it a preferred choice for building robust and modern APIs. With its growing community support and active development, FastAPI continues to evolve, providing an excellent platform for building web applications in Python.

1.3.6. OpenCV

OpenCV (Open Source Computer Vision) is a powerful open-source library that provides a wide range of tools and algorithms for computer vision and image processing tasks. This article introduces OpenCV, highlighting its key features and advantages for developing computer vision applications.

- Comprehensive Image Processing Functionality: OpenCV offers a vast collection of algorithms and functions for image processing tasks, including image filtering, feature extraction, object detection, image segmentation, and more. These capabilities enable developers to manipulate and analyze images with ease, making OpenCV a valuable tool for a wide range of computer vision applications.

- Cross-Platform Compatibility: OpenCV is a cross-platform library that supports various operating systems, including Windows, macOS, Linux, and mobile platforms like Android and iOS. This allows developers to write code once and deploy it across different platforms, making it highly versatile and accessible.
- Real-Time Computer Vision: OpenCV is optimized for real-time computer vision applications, allowing for efficient processing of video streams and live camera feeds. It provides algorithms for real-time object detection, tracking, and gesture recognition, making it suitable for applications such as surveillance systems, augmented reality, robotics, and more.
- Machine Learning Integration: OpenCV seamlessly integrates with popular machine learning frameworks, such as TensorFlow and PyTorch, enabling developers to combine computer vision algorithms with advanced machine learning models. This integration empowers developers to build sophisticated applications that leverage the power of both computer vision and machine learning.
- Large Community and Active Development: OpenCV benefits from a large and active community of developers and researchers. The community contributes to the library by providing new algorithms, bug fixes, and tutorials, making it a valuable resource for learning and troubleshooting. The active development ensures that OpenCV remains up-to-date with the latest advancements in computer vision and image processing.

OpenCV is a versatile and powerful library for computer vision and image processing tasks. Its comprehensive functionality, cross-platform compatibility, real-time capabilities, machine learning integration, and strong community support make it a go-to choice for developers working on computer vision applications. OpenCV continues to evolve and play a significant role in the advancement of computer vision technology, enabling developers to build innovative and intelligent applications.

.

CHAPTER II. REQUIREMENT ANALYSIS

2.1. Functional requirements

2.1.1. User-Side Functional Requirements

Create a social networking site similar to ***Twitter*** and ***Instaqram***

- User registers an account

- User logs in with registered account

- User logs out of the system

- User reset and changes password via url sent via email.

- Users edit their personal information such as: name, phone number, avatar, cover photo, …

- Users use the system to create Anime-style avatars

- Users create, edit and delete their posts

- Users can view their own and other people's posts

- Users search for other users by name, email, etc.

- Users search post by content.

- Users can follow and unfollow other users

- Users like, share and comment on posts

- People create chat with others and send them messages

- Group creators chat with many other people and send them messages

- Receive notification

Table 1: Requirement description

No.	Functional name	Description
1	Registers	Users can register a new account on the website. Ask for basic information like name, email, and password.
2	Log in	Users can log in to their registered account. Requires a valid email and password to log in.

3	Log out	Users can log out of their current account.
4	Reset password	Users can request a password reset if they forget their current password. Send an email containing a link to reset your password.
5	Change password	Users can change the current password to a new password.
6	Edit personal information	Users can edit personal information such as name, avatar, description, etc.
7	Create Anime-style avatars	Users can create Anime-style avatars from input images. Apply machine learning techniques to process images and create avatars.
8	Create, edit and delete posts	Users can create new articles on the website. Ability to edit and delete previously created posts.
9	View posts	Users can view a list of articles available on the website. Show detailed information of each post including content, author, number of likes, etc.
10	Search user	Users can search for other users on the website. Provides the ability to search based on name, email, or other personal information.
11	Search post	Users can search for posts on the website. Provides search capabilities based on content, author, or other keywords.
12	Follow and unfollow	Users can follow other users on the website to receive notifications about their activities. Provides the ability to unfollow previously followed users.
13	Like	Users can like other people's posts to show interest and engagement. Displays the number of likes per post.
14	Share	Users can share other people's posts to spread important content.

15	Comment	Users can comment on other people's posts to exchange ideas or suggestions.
16	Manage chat	Users can create chats with other users on the website. Provides the ability to send and receive messages in a conversation between two users.
17	Manage group chat	Users can create chat groups with many other users on the website. Provides the ability to send and receive messages in group chats.
18	Receive and send message	Users can receive and send messages to each other in individual or group chats.
19	Receive notification	Users receive notifications about activities

2.1.2. System-Side Functional Requirements

The system must prevent images from being posted that do not conform to community standards, such as nude or too revealing images, etc.

2.2. Non-functional requirements

- Usability: The system should have a user-friendly interface with clear navigation and easy-to-understand instructions for each task.

- Reliability: The system should be highly reliable and available at all times to ensure that users can access it whenever they need to.

- Performance: The system should be able to handle large amounts of data and user requests without slowing down or crashing.

- Security: The system should be secure and protect user data from unauthorized access, theft, or loss.

- Scalability: The system should be designed to handle growth and increasing demand without compromising performance or usability.

- Maintainability: The system should be easy to maintain and update, with clear documentation and a modular design that allows for easy troubleshooting and debugging.

- Compatibility: The system should be compatible with different devices, operating systems, and web browsers to ensure that users can access it from any platform.

CHAPTER III. SYSTEM DESIGN

3.1. Use case diagram

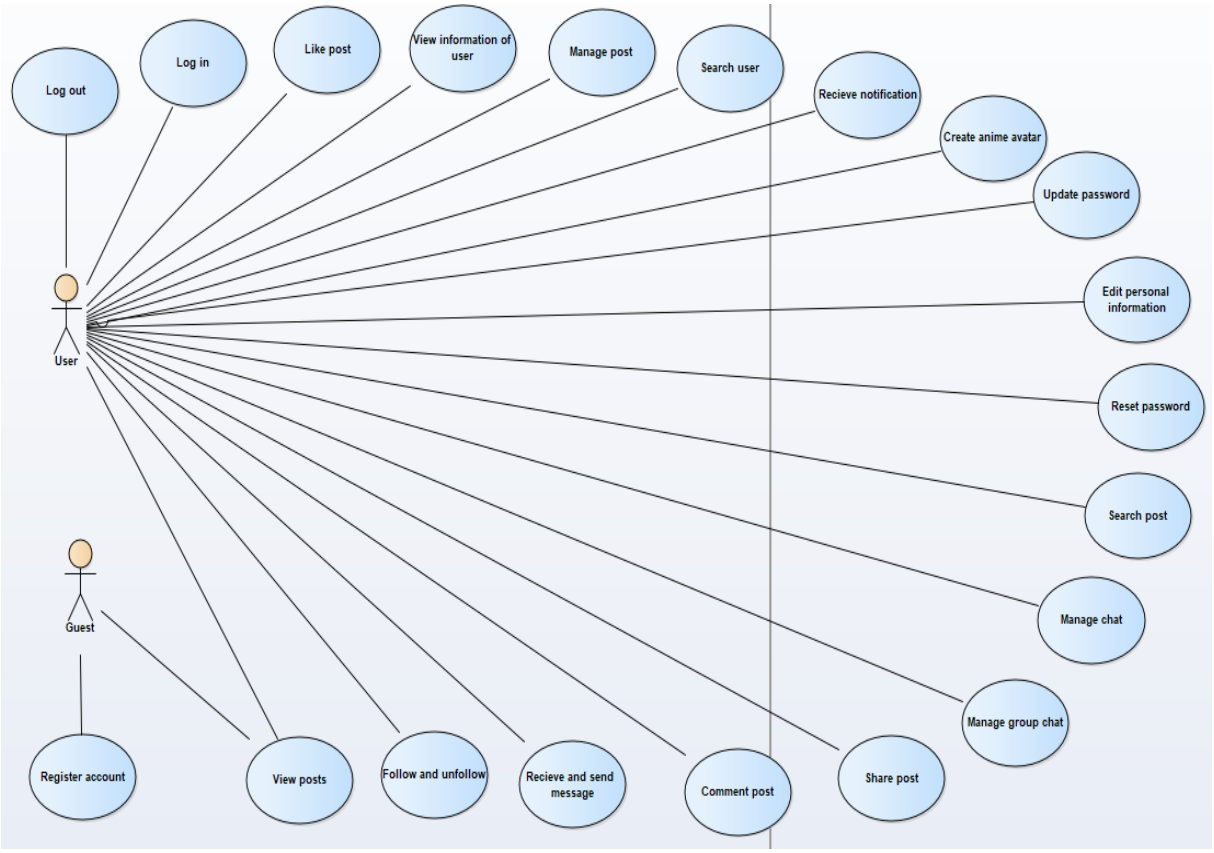


Figure 22: Use case

3.2. Use case description

3.2.1. Register account

Table 2: Use case description Register account

Use Case	**Register account**
Short Description:	Users create accounts to log in to the web and use user permissions
Actor:	Guest
Pre-Conditions:	
Post-Conditions:	The user successfully created an account

Main Flow: (1) User accesses the web (2) User selects account creation page (3) User fills in personal information, email, password and confirmation password. (4) Information authentication system (5) Notification of successful account creation and account saving to database
Alternate Flow(s)
Exception Flow(s): (4a) The system confirms registration failed due to invalid email or password, displays a message and re-registers.

3.2.2. Log in

Table 3: Use case description Log in

Use Case	**Log in**
Short Description:	Users create accounts to log in to the web and use user permissions
Actor:	User
Pre-Conditions:	
Post-Conditions:	The user successfully created an account
Main Flow: (1) User accesses the website (2) User presses the login button (3) The website displays the login section (4) User fills in email and password (5) Login information authentication system (6) The system determines the user's role (7) The system returns the user's home page	
Alternate Flow(s)	
Exception Flow(s): [5a] The login authentication system failed and displayed a message	

3.2.3. Log out

Table 4: Use case description Log out

Use Case	**Log out**
Short Description:	User logs out of the system
Actor:	User
Pre-Conditions:	The user has successfully logged in, the user is on the Website
Post-Conditions:	The user logs out of the system
Main Flow: (1) The user presses the button with his or her name on it (2) The system displays the toolbar (3) User accesses logout method (4) The system confirms and exits the user's login session	
Alternate Flow(s)	
Exception Flow(s):	

3.2.4. Update password

Table 5: Use case description Update password

Use Case	**Update password**
Short Description:	User changes password
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	User can change password
Main Flow: (1) The user selects the button with the username (2) User selects my profile (3) User enters old password, new password and password confirm (4) Click the save button (5) The system changes the password and notifies of successful password change	

Alternate Flow(s)
Exception Flow(s): (5a) The system reports incorrect password entry when the old password is incorrect or the new password is not entered Use case returns to step 3

3.2.5. Reset password

Table 6: Use case description Reset password

Use Case	**Reset password**
Short Description:	Forgot password
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	The user has a new password
Main Flow: (1) The user presses the forgot password button (2) User enters email and click send (3) The system sends a password reset link to the user via email (4) User accesses link in email (5) Enter new password and lose confirmation password (6) Click update (7) The system updates the user's password and redirects to the login page	
Alternate Flow(s)	
Exception Flow(s): (2a) The system cannot find the email the user entered and reports that the user does not exist Use case returns to step 2 (5a) The system notifies the user of entering an incorrect password Use case returns to step 5 (6a) User clicks update too late (more than 10 minutes) reset password failed	

3.2.6. Manage post

Table 7: Use case description Manage post

Use Case	**Manage post**
Short Description:	Create, update and delete post
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	The user create, update and delete post succefully
Main Flow: **1. Create post** 1.1. The user selects the "Create post" button on the home page 1.2. The system displays the "Create post" form 1.3. Enter the information to post 1.4. Click "Post" **2. Update post** 2.1. The user selects the "Update post" button on the top-right corner of post 2.2. The system displays the "Update post" form 2.3. Enter the information to post 2.4. Click "Update" **3. Delete post** 3.1. The user selects the "Delete post" button on the top-right corner of post 3.2. The system displays the "Deletion confirmation" form 3.3. Click "Delete"	
Alternate Flow(s): 1.4.a. Click "Cancel" to cancel create Post 2.4.a. Click "Cancel" to cancel update Post 3.3.a. Click "Cancel" to cancel delete Post	
Exception Flow(s):	

3.2.7. Like post

Table 8: Use case description Like post

Use Case	**Like post**
Short Description:	Like and unlike post

Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	The user like and unlike post successfully
Main Flow: 1. The user access “Home page” 2. The user views a post 3. Click "Like"	
Alternate Flow(s):	
Exception Flow(s):	

3.2.8. Comment post

Table 9: Use case description Comment post

Use Case	**Comment post**
Short Description:	Comment post
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	The user comment post successfully
Main Flow: 1. The user access “Home page” 2. The user views a post 3. Click "Comment" 4. The system show “Comment post” form 5. The user enter comment 6. The user click “Comment”	
Alternate Flow(s): 6.a. Click “Cancle” to cancel comment	
Exception Flow(s): 6.b. If user do not enter comment, the system will send message “please fill comment” Use case returns to step 5	

3.2.9. Search post

Table 10: Use case description Search post

Use Case	**Search post**
Short Description:	Users can search for posts on the website
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	The user serch and find post successfully
Main Flow: 1. The user access “Explore” page 2. The user select “Post” 3. User enter content of Post that you want to search to search bar 4. The user click “Search”	
Alternate Flow(s):	
Exception Flow(s): 4.a. If user do not enter content to searh, the system will send message “please fill content” Use case returns to step 3	

3.2.10. Share post

Table 11: Use case description Share post

Use Case	**Share post**
Short Description:	Users can share posts on the website
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	The user shares and unshares post successfully
Main Flow: 1. The user access “Home page” 2. The user views a post 3. Click "Share"	
Alternate Flow(s):	

Exception Flow(s):

3.2.11. View information of user

Table 12: Use case description View information of user

Use Case	**View information of user**
Short Description:	Users can view their own information and that of others
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	Users access personal pages to view their own information and that of others
Main Flow: **1. View information of themselves** 1.1. Click to avatar in top-right cornor 1.2. The system shows the profile page **2. View information of other people** 2.1. Click to avatar or name of other people 2.2. The system shows the profile other people page	
Alternate Flow(s):	
Exception Flow(s):	

3.2.12. Search user

Table 13: Use case description Search use

Use Case	**Search user**
Short Description:	Users can search for users on the website
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	The user serch and find users successfully

Main Flow: 1. The user access “Explore” page 2. The user select “User” 3. User enter name or email of User that you want to search to search bar 4. The user click “Search”
Alternate Flow(s):
Exception Flow(s): 4.a. If user do not enter content to searh, the system will send message “please fill content” Use case returns to step 3

3.2.13. Edit personal information

Table 14: Use case description Edit personal information

Use Case	**Edit personal information**
Short Description:	Users can edit information of themeselves
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	Users edit information of themeselves succefully
Main Flow: **1. Edit information** 1.1. Click to avatar in top-right cornor 1.2. The system shows the profile page 1.3. Click “Edit information” 1.4. The system displays “Edit information” form 1.5. The users edit their information 1.6. Click “Save” **2. Edit avatar and cover photo** 2.1. Click to avatar in top-right cornor 2.2. The system shows the profile page 2.3. Click on avatar or cover photo 2.4. The system displays “Edit” form 2.5. The users click on image 2.6. The users select image form their browser	

2.7. Click “Save”
Alternate Flow(s): 1.6.a. Click “Cancel” to cancel edit information 2.7.a. Click “Cancel” to cancel edit avatar and cover photo
Exception Flow(s): 1.6.b. If user do not enter all information require, the system will send message “please fill content” Use case returns to step 5 2.7.b. If user do not select image, the system will send message “please select image” Use case returns to step 6

3.2.14. Create avatar anime

Table 15: Use case description Create avatar anime

Use Case	**Create avatar anime**
Short Description:	Users can create anime image and set it as avatar
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	Users seleat an image then create anime image and set it as avatar
Main Flow: 1 Click on profile page 2. The system shows the profile page 3. Click on avatar or cover photo 4. The system displays “Edit” form 5. The users click on “Change anime image” 6. The users select image form their browser	

7. The output image show in this form 7. Click “Save” to update avatar
Alternate Flow(s): 8.a. Click “Cancel” to cancel
Exception Flow(s):

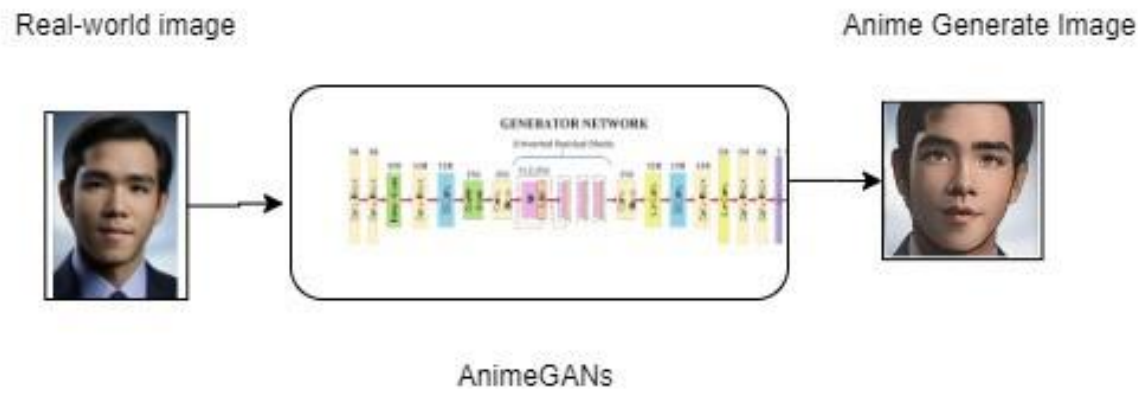


Figure 23: Inference flow of Create anime avatar

3.2.15. Receive and view notification

Table 16: Use case description Receive and view notification

Use Case	**Receive and view notification**
Short Description:	Users receive notifications about activities such as new posts, comments, likes, messages, etc.
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	Users can receive all notification to do not miss any event
Main Flow: 1 Click on Notification page 2. The system shows all notification of themselves 3. Click on anyone to view and mask as opened	
Alternate Flow(s):	

Exception Flow(s):

3.2.16. Manage chat

Table 17: Use case description Manage chat

Use Case	**Manage chat**
Short Description:	Users can view all their chat and access chats with other users on the website.
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	Users view all their chat and access chats with other users on the website successfully
Main Flow: 1. Click on Message page 2. The system shows all chats 3. Select a chat or search user 4. Click on anyone to access 5. The system show chat detail	
Alternate Flow(s):	
Exception Flow(s):	

3.2.17. Manage group chat

Table 18: Use case description Manage group chat

Use Case	**Manage chat**
Short Description:	Users can create group chat, reaname group chat Add, and remove members in group chat
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	Users create group chat, reaname group chat, add, and remove members in group chat successfully

Main Flow:

1. Create group chat

1.1. Click on Message page

2.2. The system shows all chats

1.3. Click “Create group chat”

1.4. Search user than click add to push them

1.5. Enter chat name

1.6. Click “Create”

2. Rename group chat

1.1. Click on Message page

2.2. Select a group chat

2.3. The system shows chat detail

2.4. Click on menu list of chat

2.5. Select rename

2.6. Enter new name

2.7. Click rename

3. Add new members

3.1. Click on Message page

3.2. Select a group chat

3.3. The system shows chat detail

3.4. Click on menu list of chat

3.5. Select Add member

3.6. The system shows an add member page

3.7. Search user to add

3.8. Select user

3.9. Click “Add member”

4. Remove members

4.1. Click on Message page

4.2. Select a group chat

4.3. The system shows chat detail

4.4. Click on menu list of chat

4.5. Select Remove member

4.6. The system shows a remove member page

4.7. Select user to remove

4.8. Click “Remove member”

Alternate Flow(s):

1.6.a. Click “Cancel” to cancel Create group chat 2.7.a. Click “Cancel” to cancel Rename group chat 3.9.a. Click “Cancel” to cancel Add member 4.8.a. Click “Cancel” to cancel Remove member
Exception Flow(s): 1.6.b. If you do not enter chat name, the system Message “Please Fill the name”. Use case returns to step 5 1.6.c. If you select less than or equal to 2 users, the system Message “More than 2 users are required to form a group chat”. Use case returns to step 6 2.7.b. If you do not enter new chat name, the system Message “Please Fill the name”. Use case returns to step 7 3.9.b. If you do not select user, the system Message “Select user to add”. Use case returns to step 8 2.7.b. If you do not select user, the system Message “Please select user to remove”. Use case returns to step 6

3.2.18. Follow user

Table 19: Use case description Follow user

Use Case	**Follow user**
Short Description:	Follow and unfollow user
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	The user follow and unfollow other user successfully
Main Flow: 1. The user access profile other user 2. The user click follow or unfollow	
Alternate Flow(s):	
Exception Flow(s):	

3.2.19. Receive and send message

Table 20: Use case description Recieve and send message

Use Case	**Receive and send message**
Short Description:	User can receive and send message
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	Users receive and send message successfully
Main Flow: **1. Send message** 1.1. The users access to Chat detail 1.2. The user enter message 1.3. Users click send **2. Receive message** 2.1. The users access to Chat detail 2.2. Users can view message from the others	
Alternate Flow(s):	
Exception Flow(s): 1.3.a. If users do not enter message, the system do not execute send	

3.2.20. View post

Table 21: Use case description View post

Use Case	**View post**
Short Description:	Guest and user can access home page to view post
Actor:	User
Pre-Conditions:	The user already has an account in the system and logged in successfully
Post-Conditions:	Guest and user access home page to view post succefully
Main Flow: 1. Access home page 2. View post	
Alternate Flow(s):	

Exception Flow(s):

3.3. Class diagram

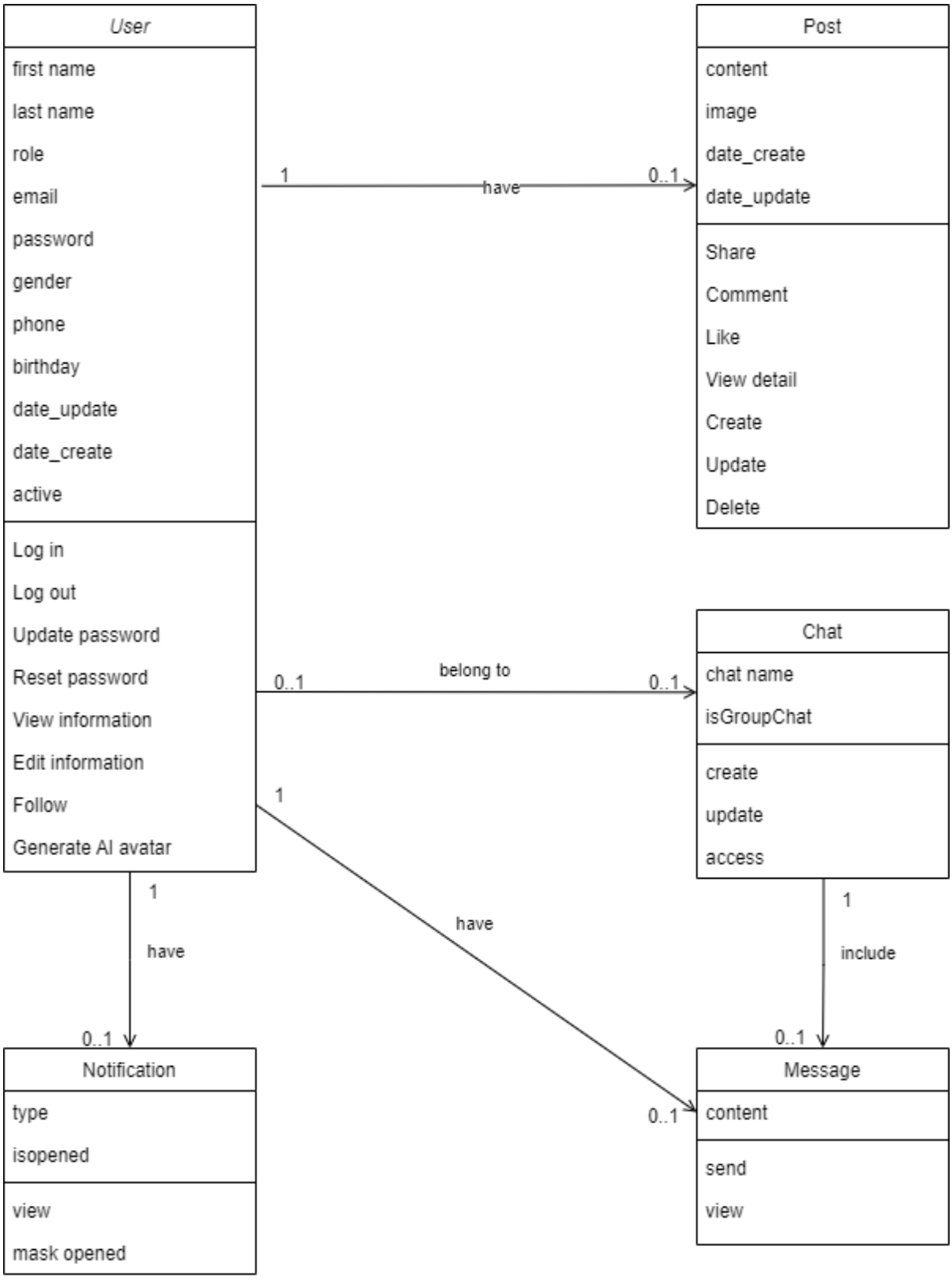


Figure 24: Class diagram

3.4. Sequence diagram

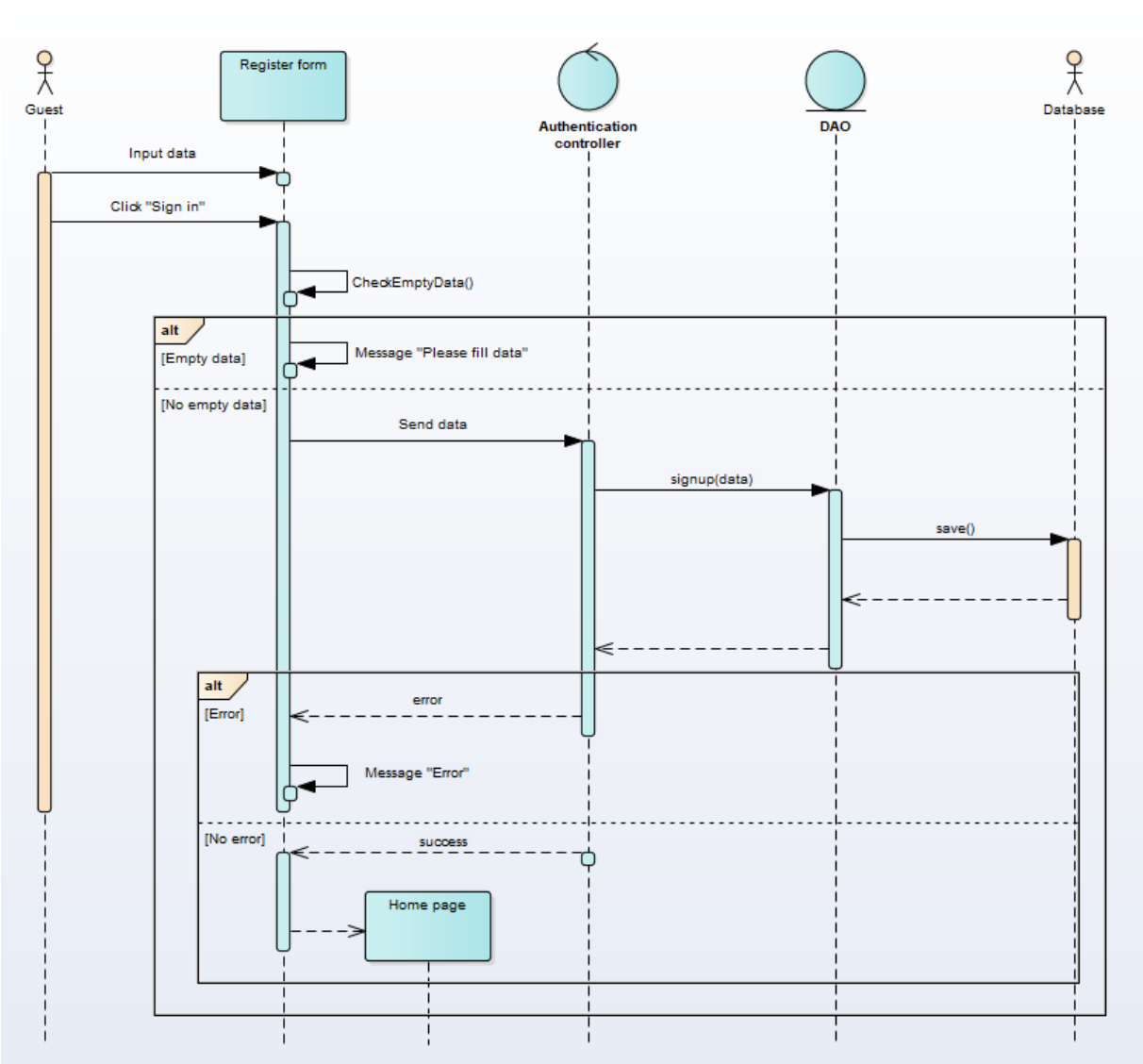


Figure 25: Sequence diagram Register account

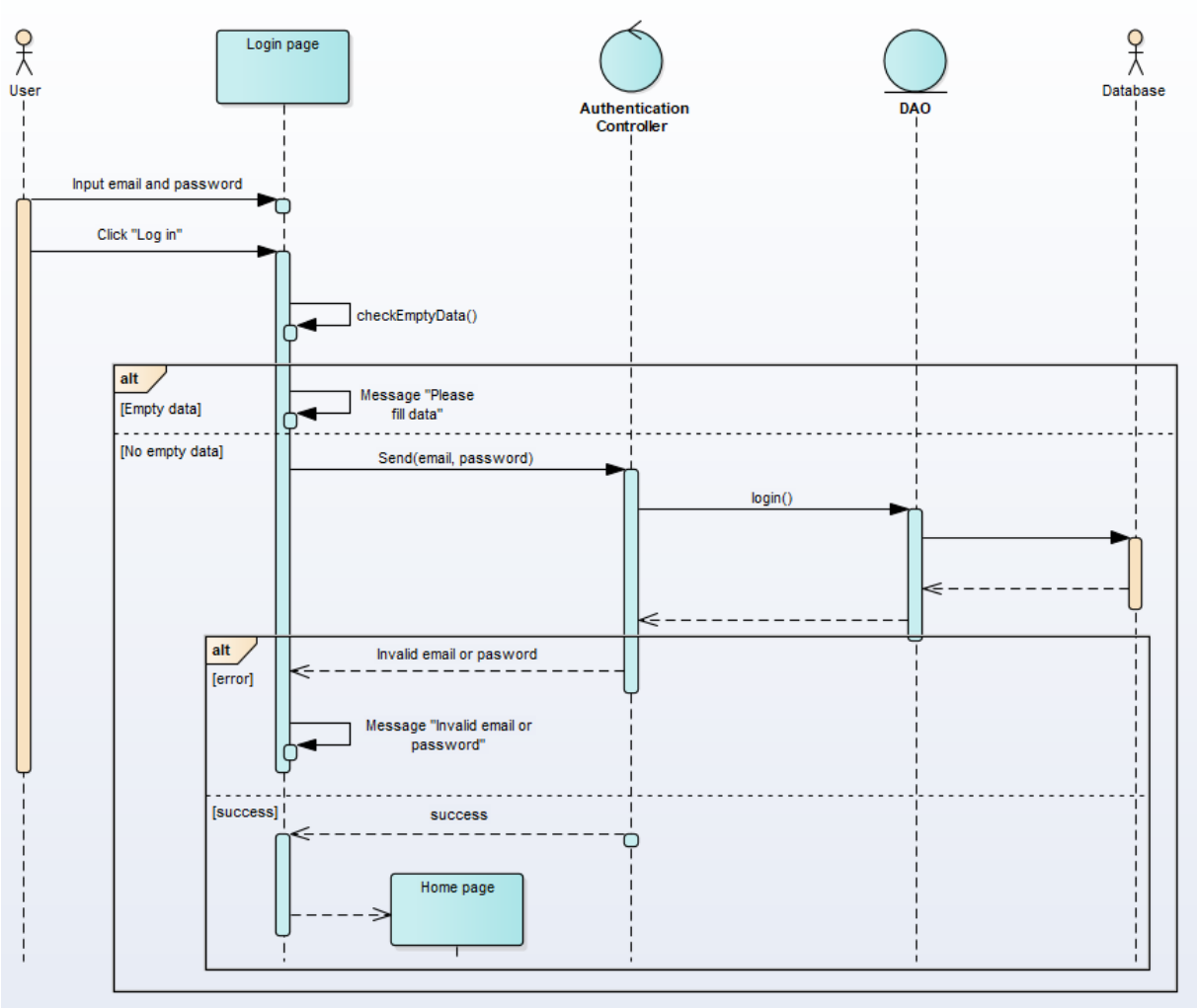


Figure 26: Sequence diagram Log in

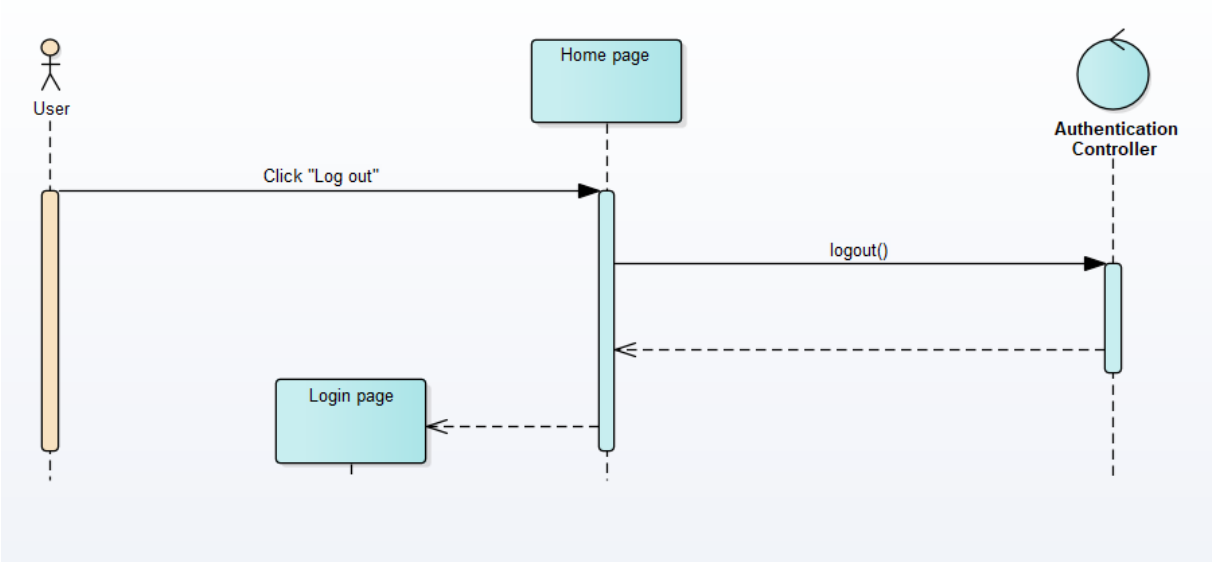


Figure 27: Sequence diagram Log out

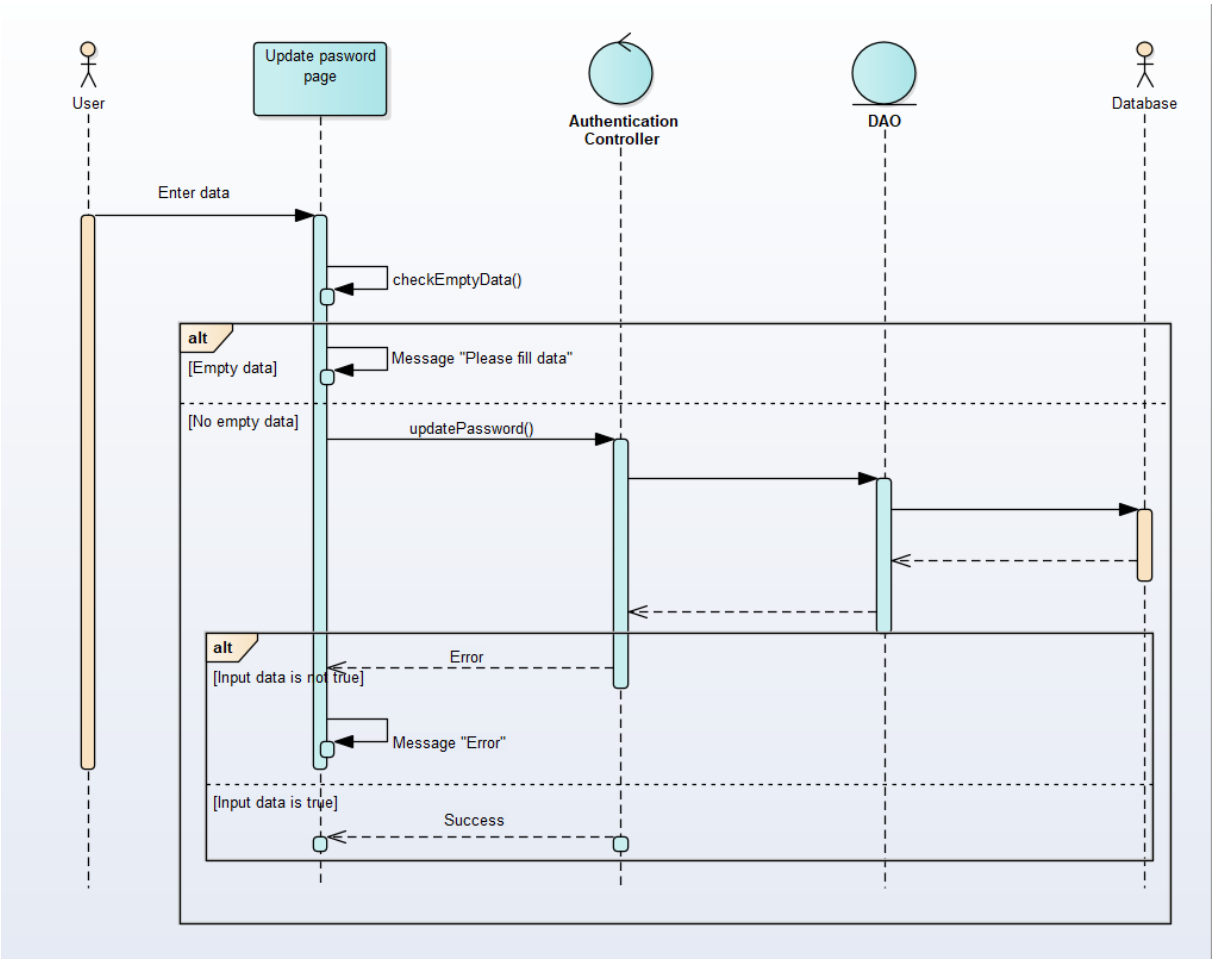
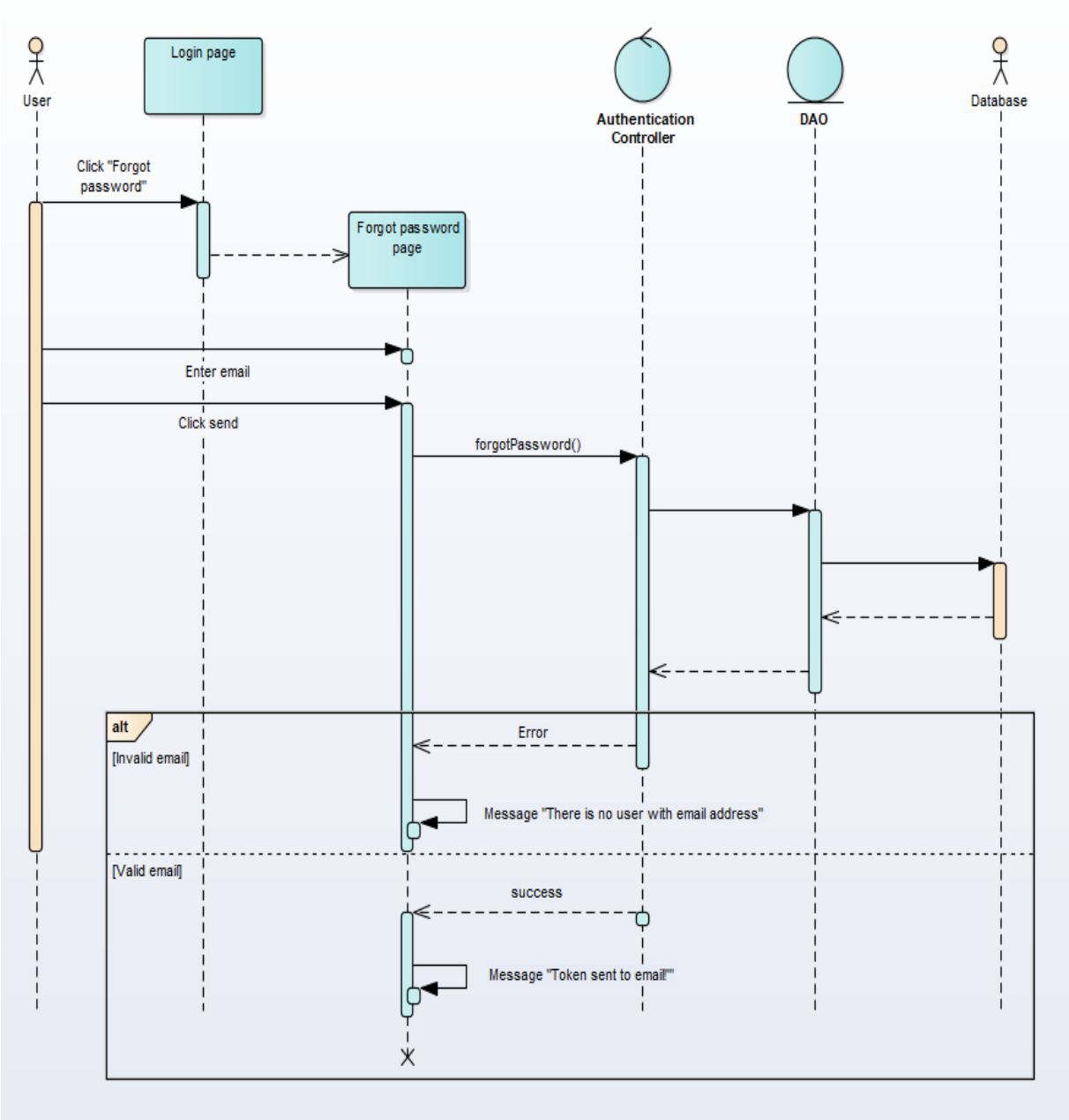


Figure 28: Sequence diagram Update password


User
Login page
Authentication Controller
DAO
Database
Click "Forgot password"
Forgot password page
Enter email
Click send
forgotPassword()
alt
[Invalid email]
Error
Message "There is no user with email address"
[Valid email]
success
Message "Token sent to email""

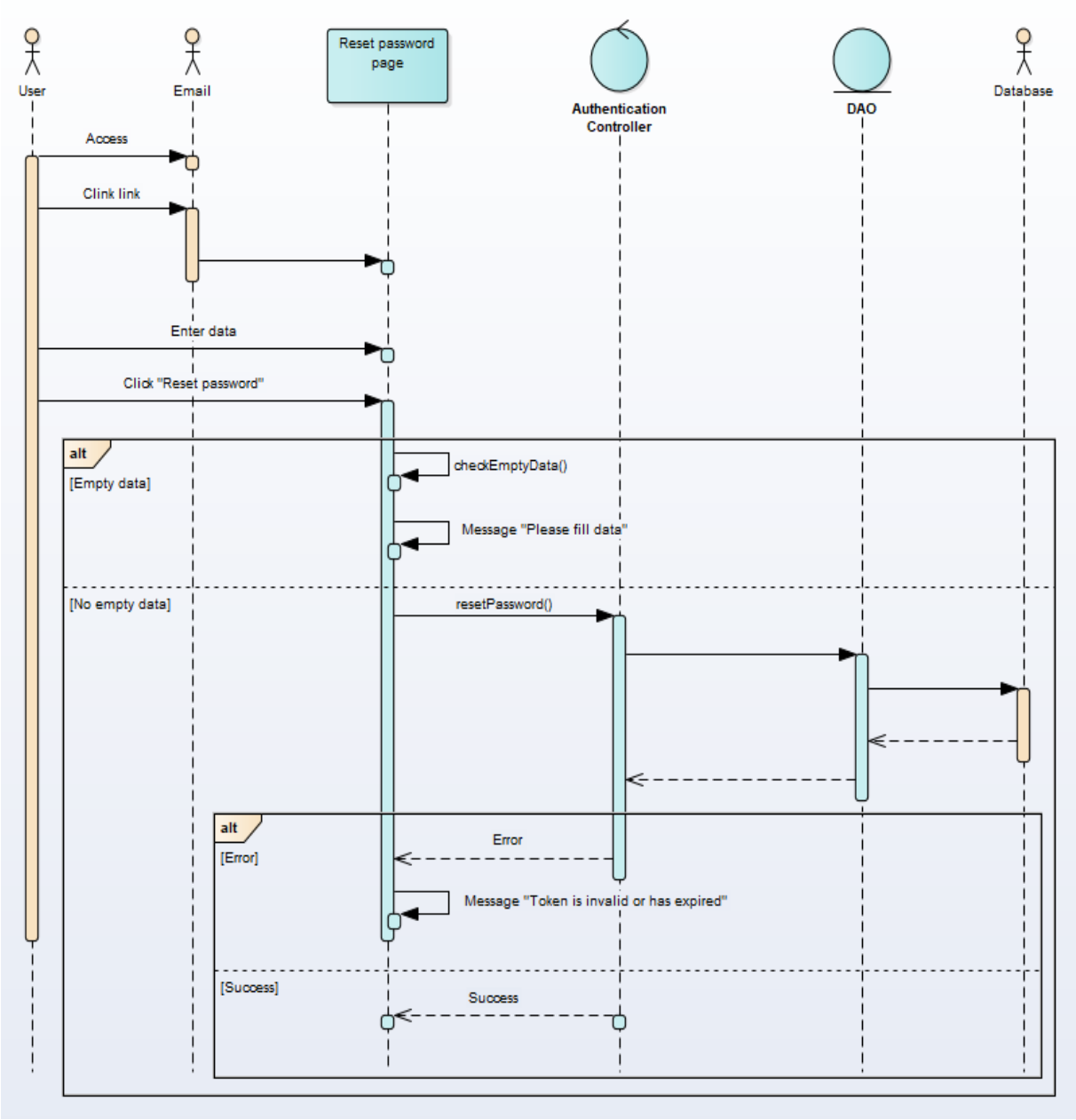
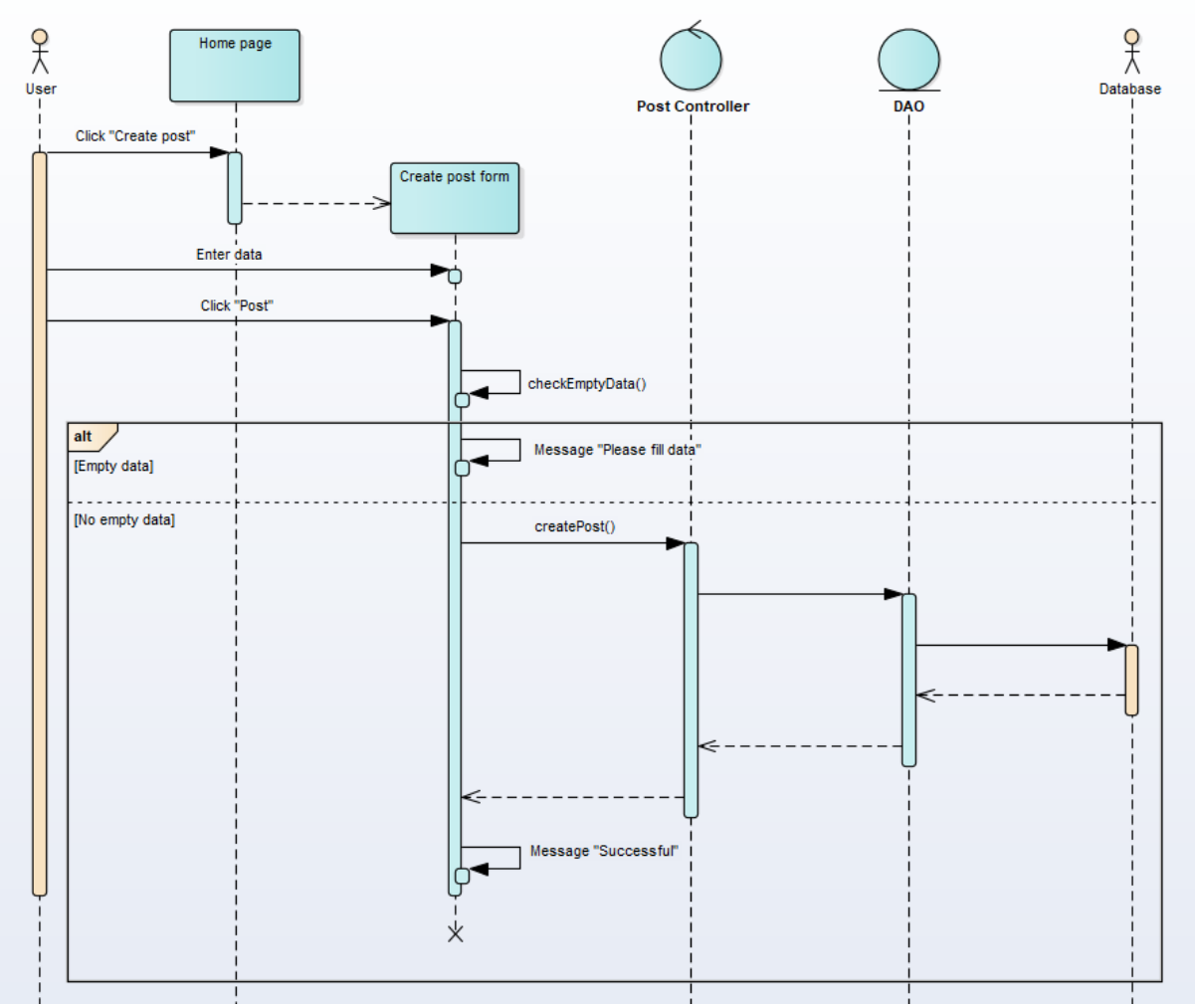


Figure 29: Sequence diagram Reset password


User
Home page
Post Controller
DAO
Database
Click "Create post"
Create post form
Enter data
Click "Post"
checkEmptyData()
alt
[Empty data]
Message "Please fill data"
[No empty data]
createPost()
Message "Successful"

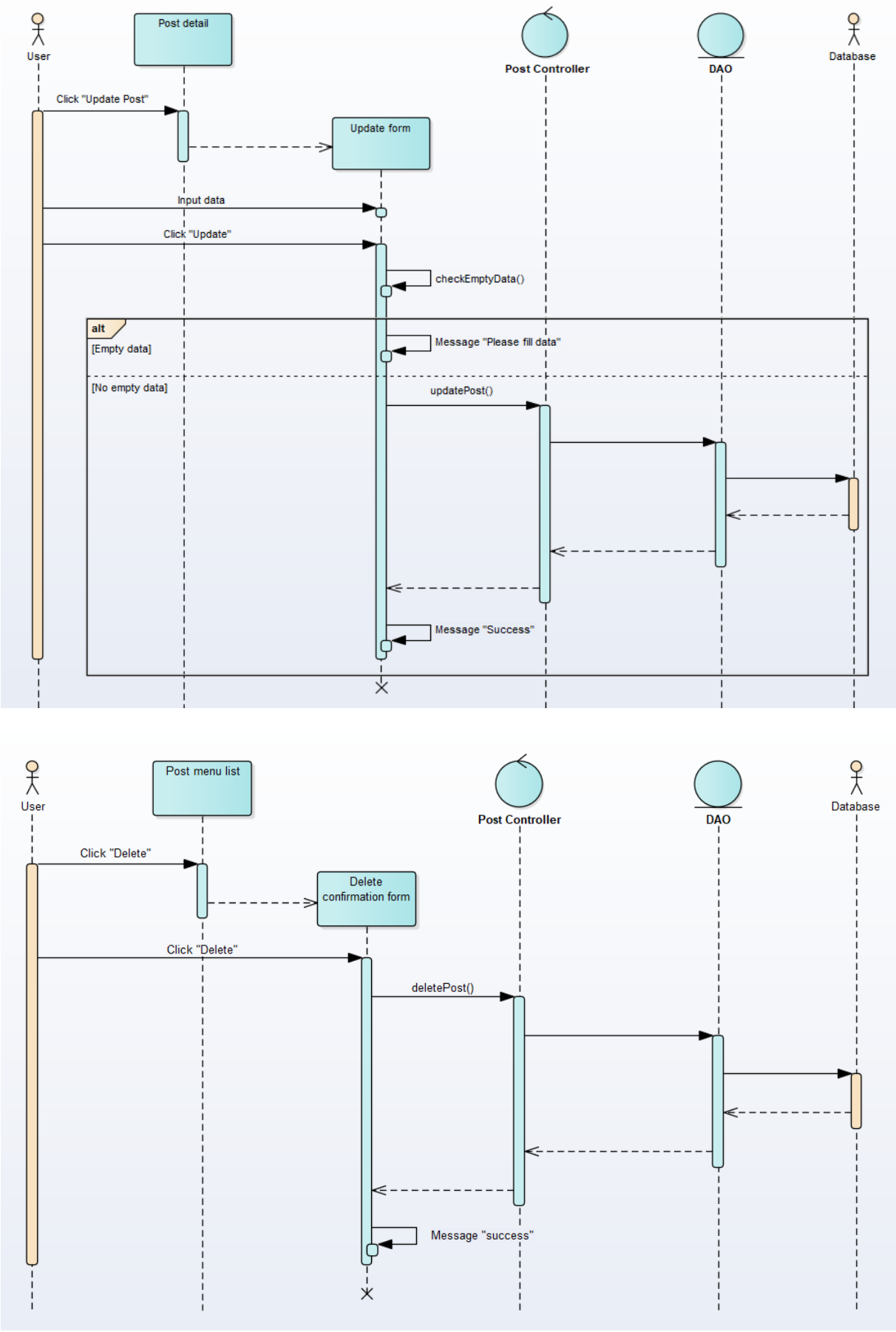


Figure 30: Sequence diagram Manage post

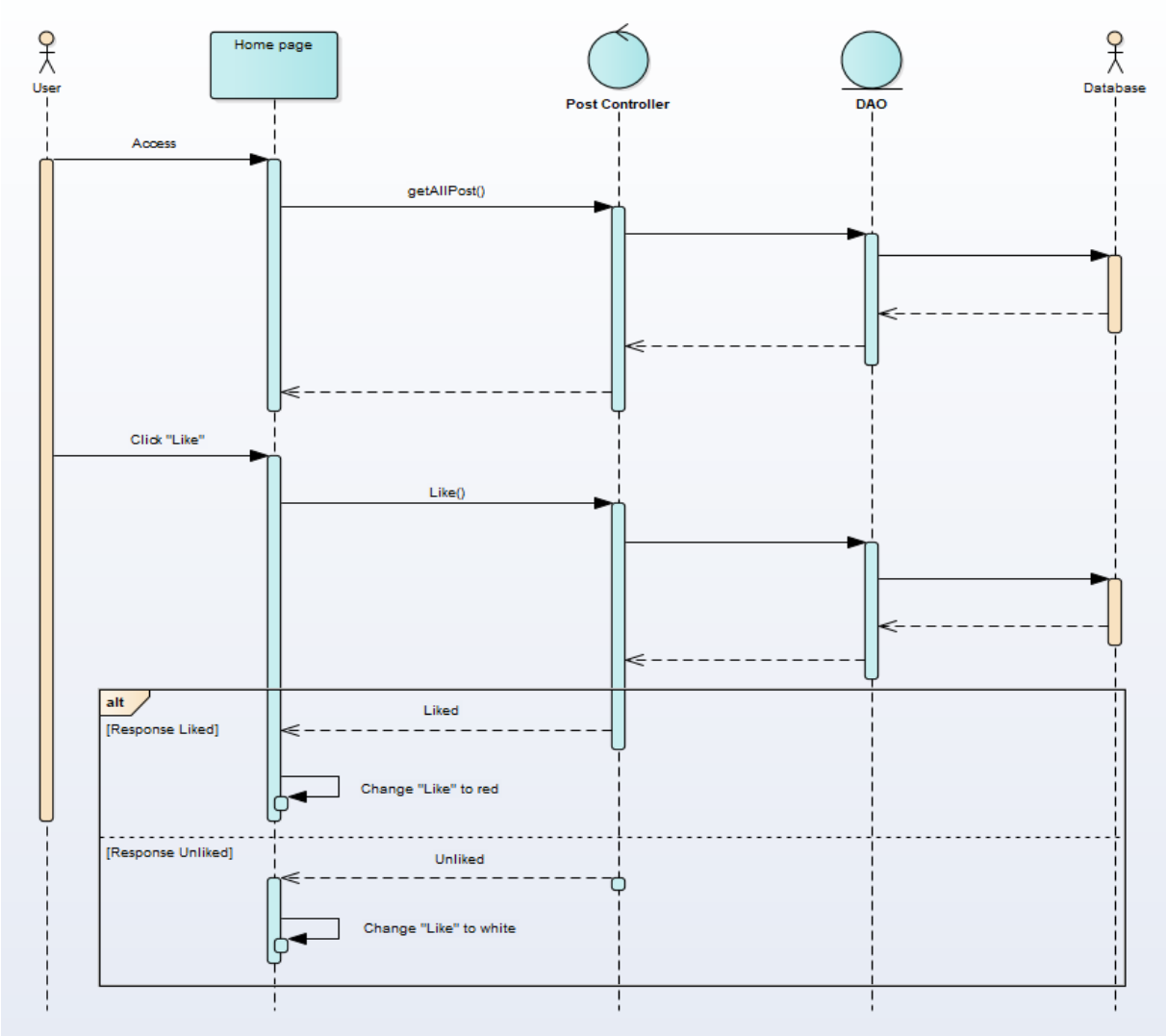


Figure 31: Sequence diagram Like post

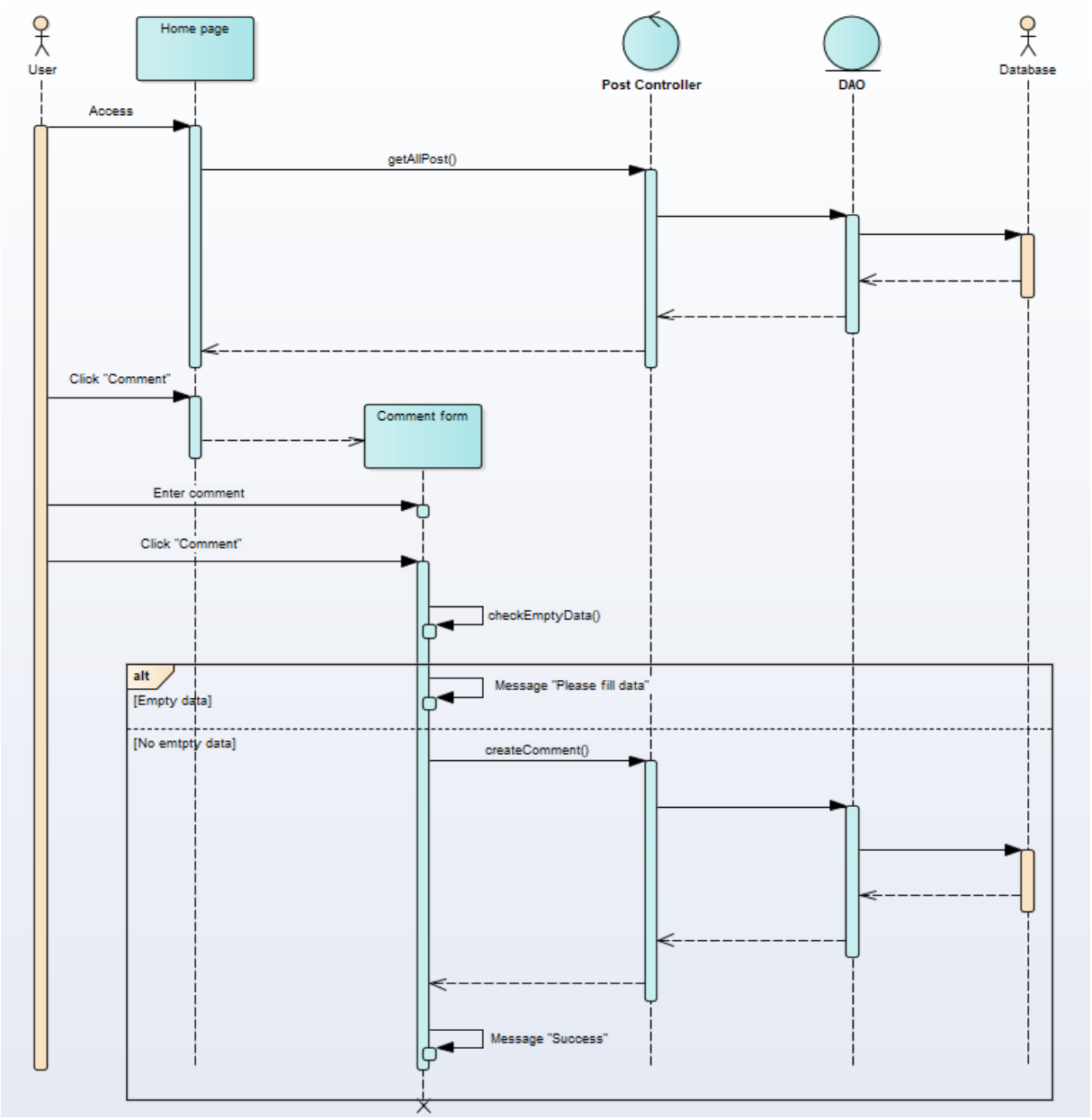


Figure 32: Sequence diagram Comment post

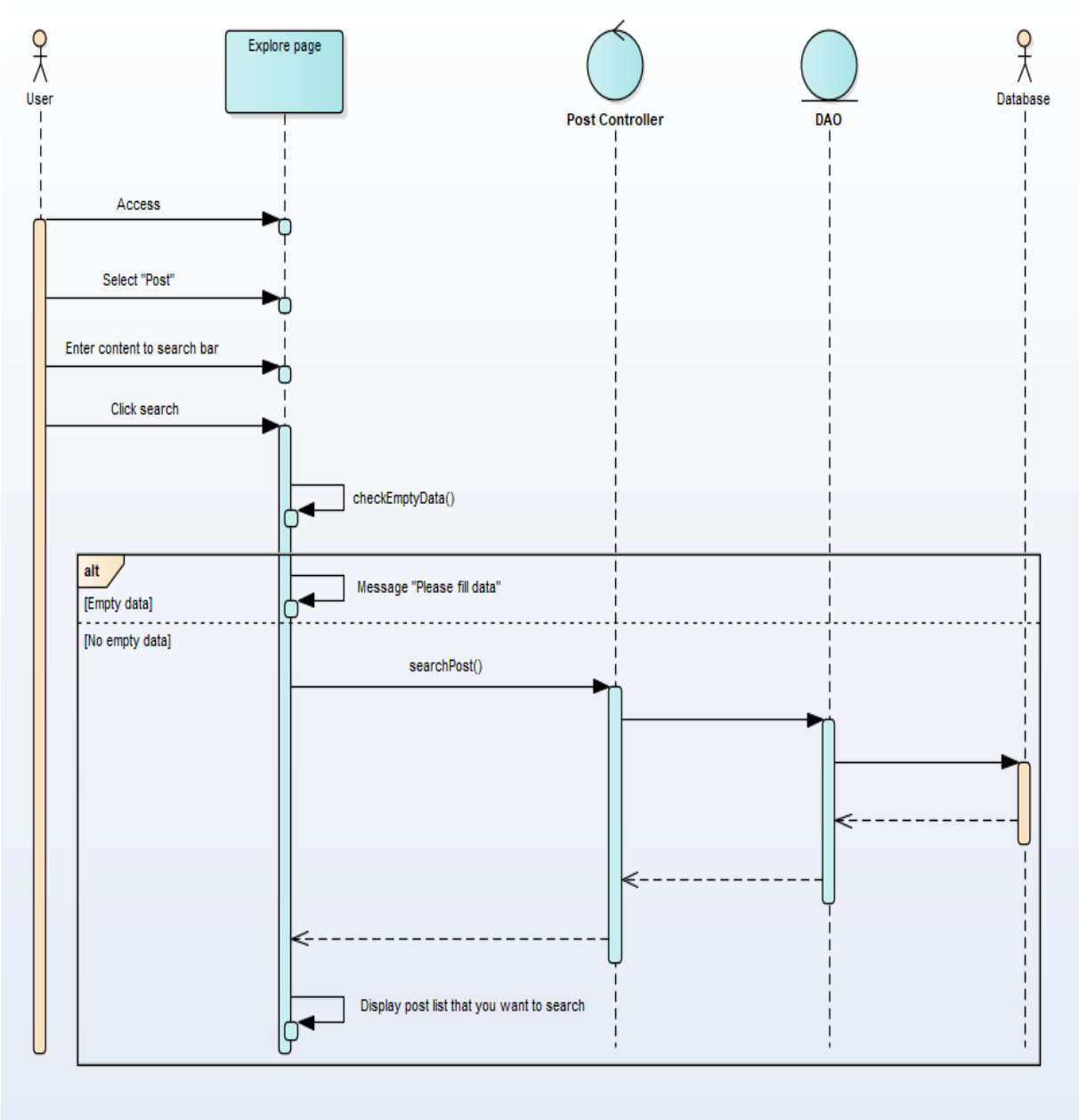


Figure 33: Sequence diagram Search post

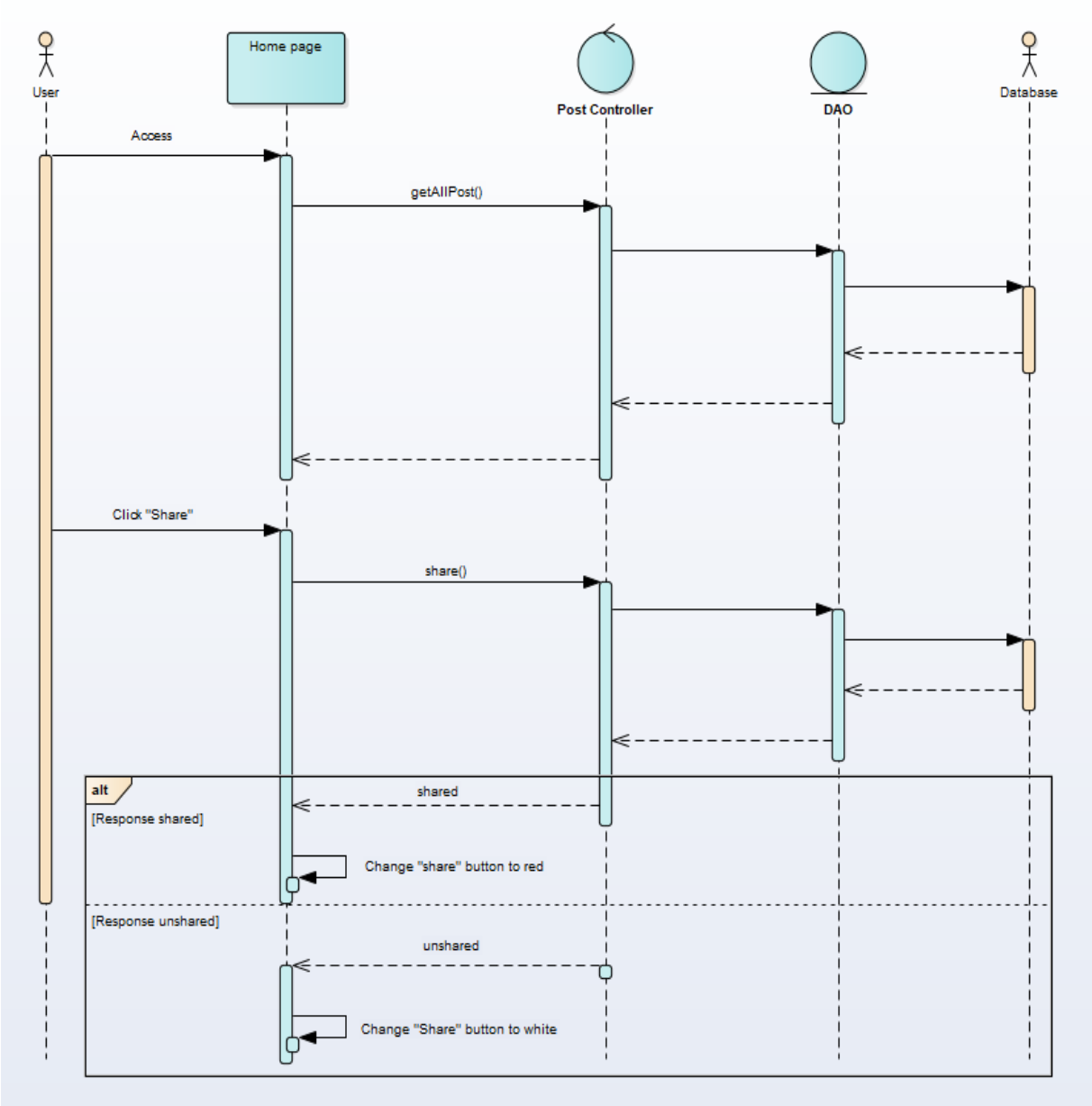


Figure 34: Sequence diagram Share post

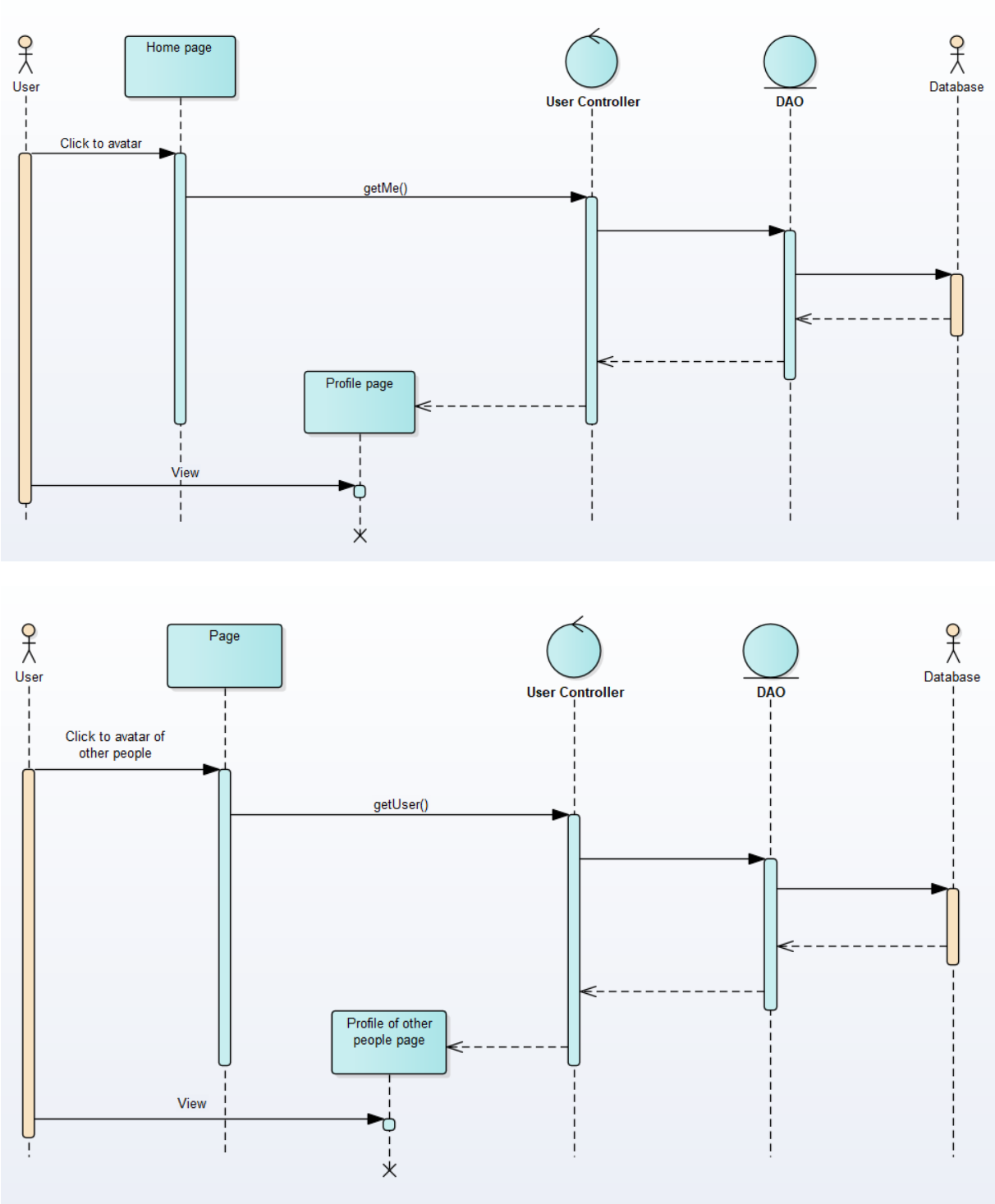


Figure 35: Sequence diagram View information user

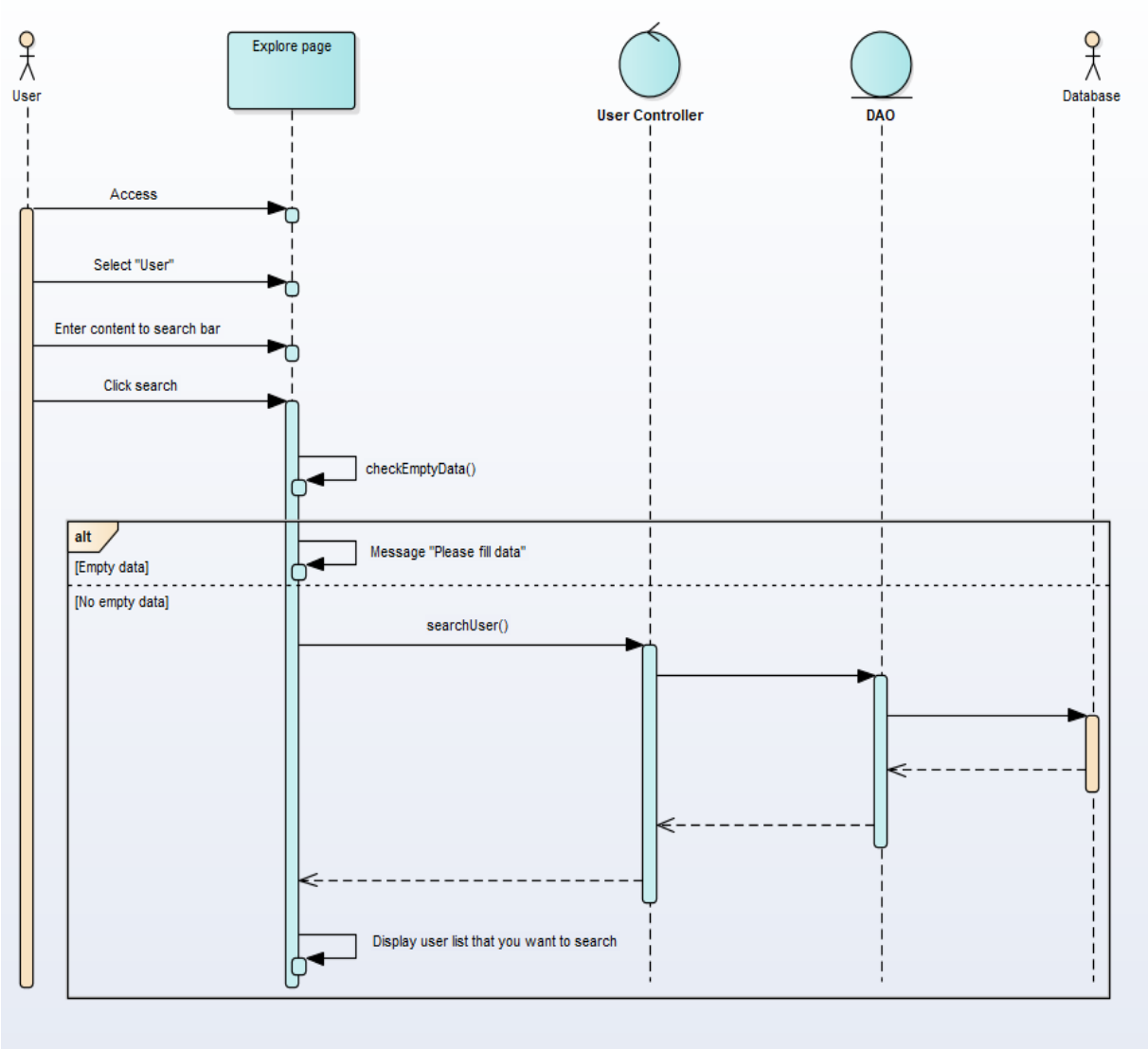
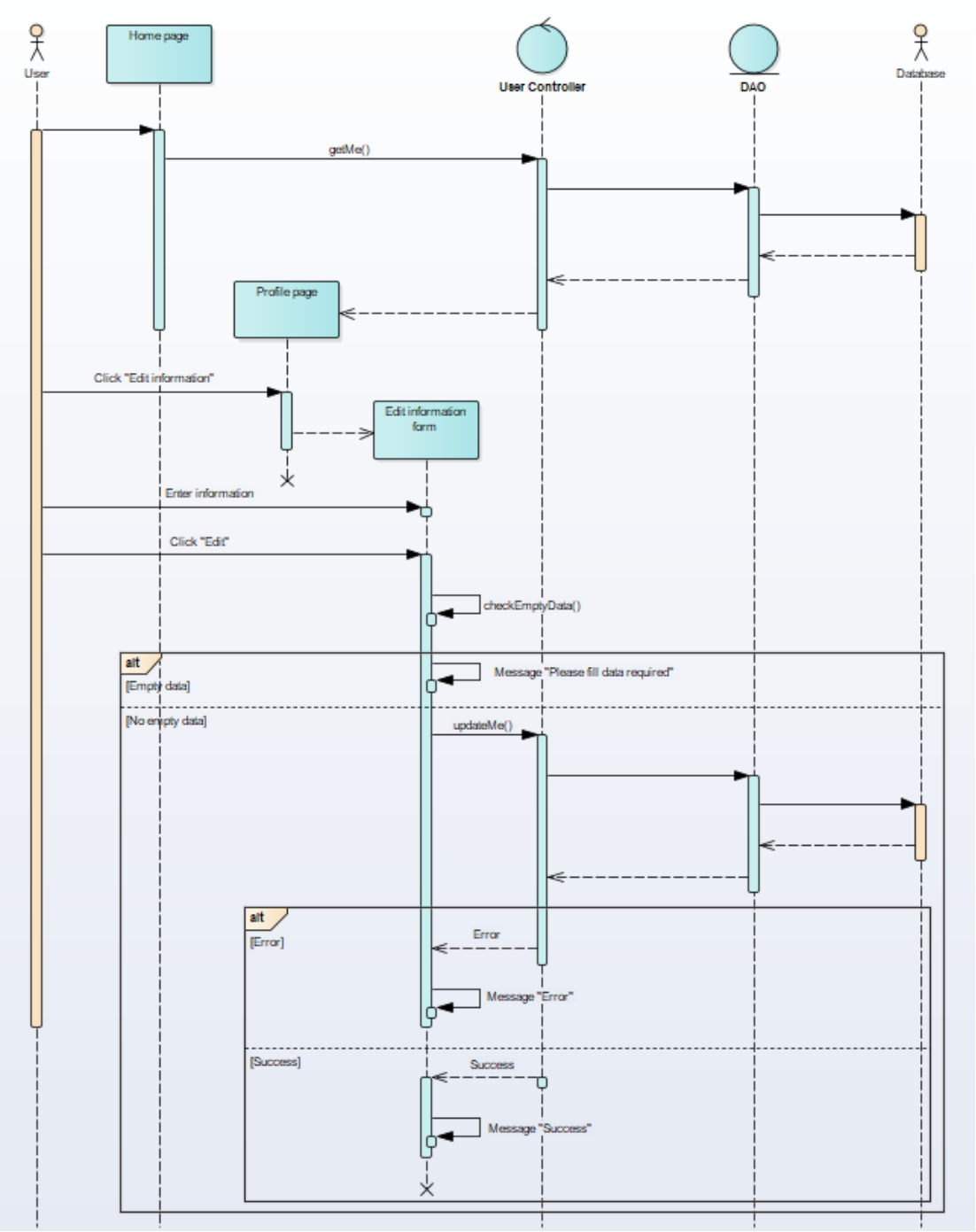


Figure 36: Sequence diagram Search user


User
Home page
User Controller
DAO
Database
getMe()
Profile page
Click "Edit information"
Edit information form
Enter information
Click "Edit"
checkEmptyData()
alt
[Empty data]
Message "Please fill data required"
[No empty data]
updateMe()
alt
[Error]
Error
Message "Error"
[Success]
Success
Message "Success"

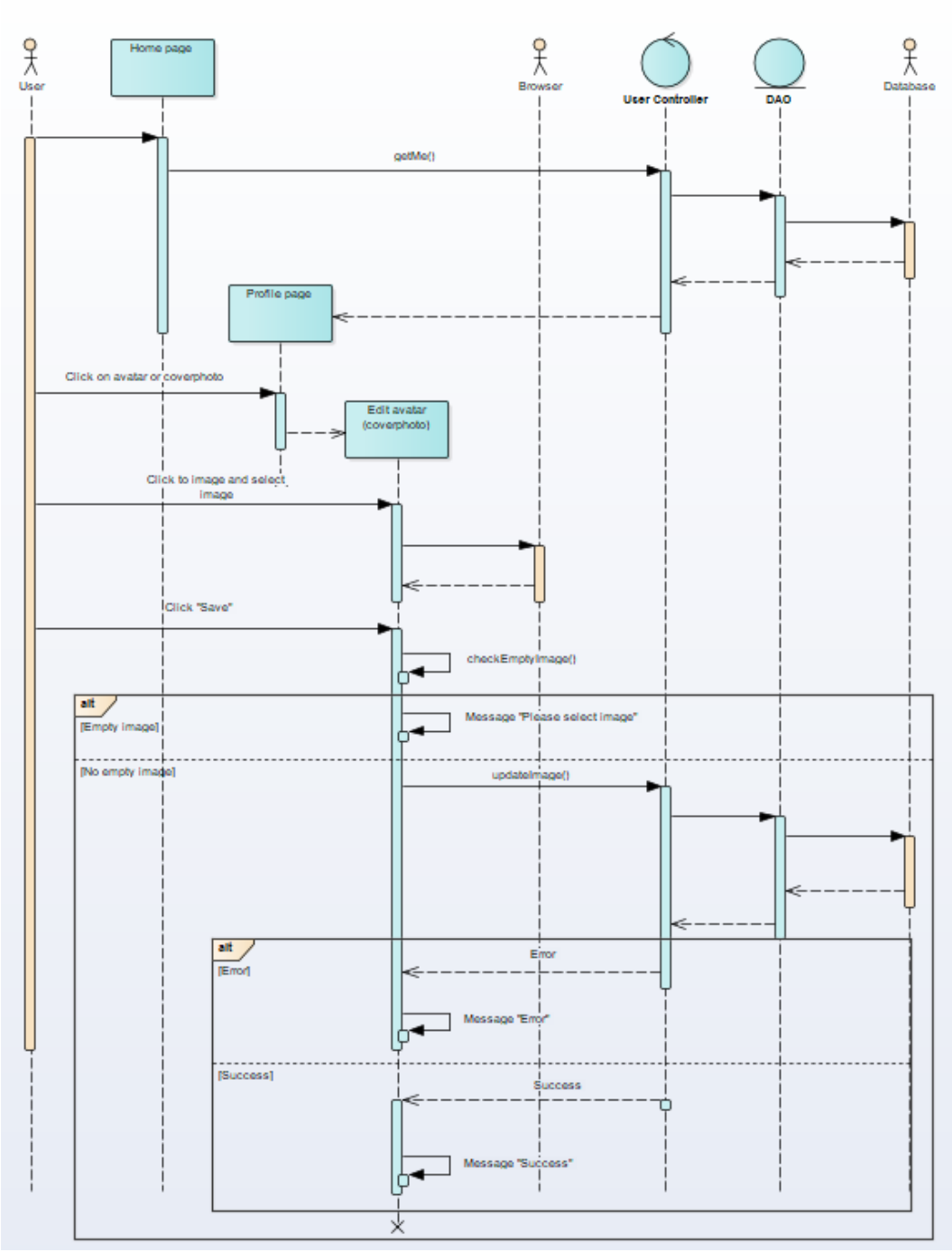


Figure 37: Sequence diagram Edit personal information

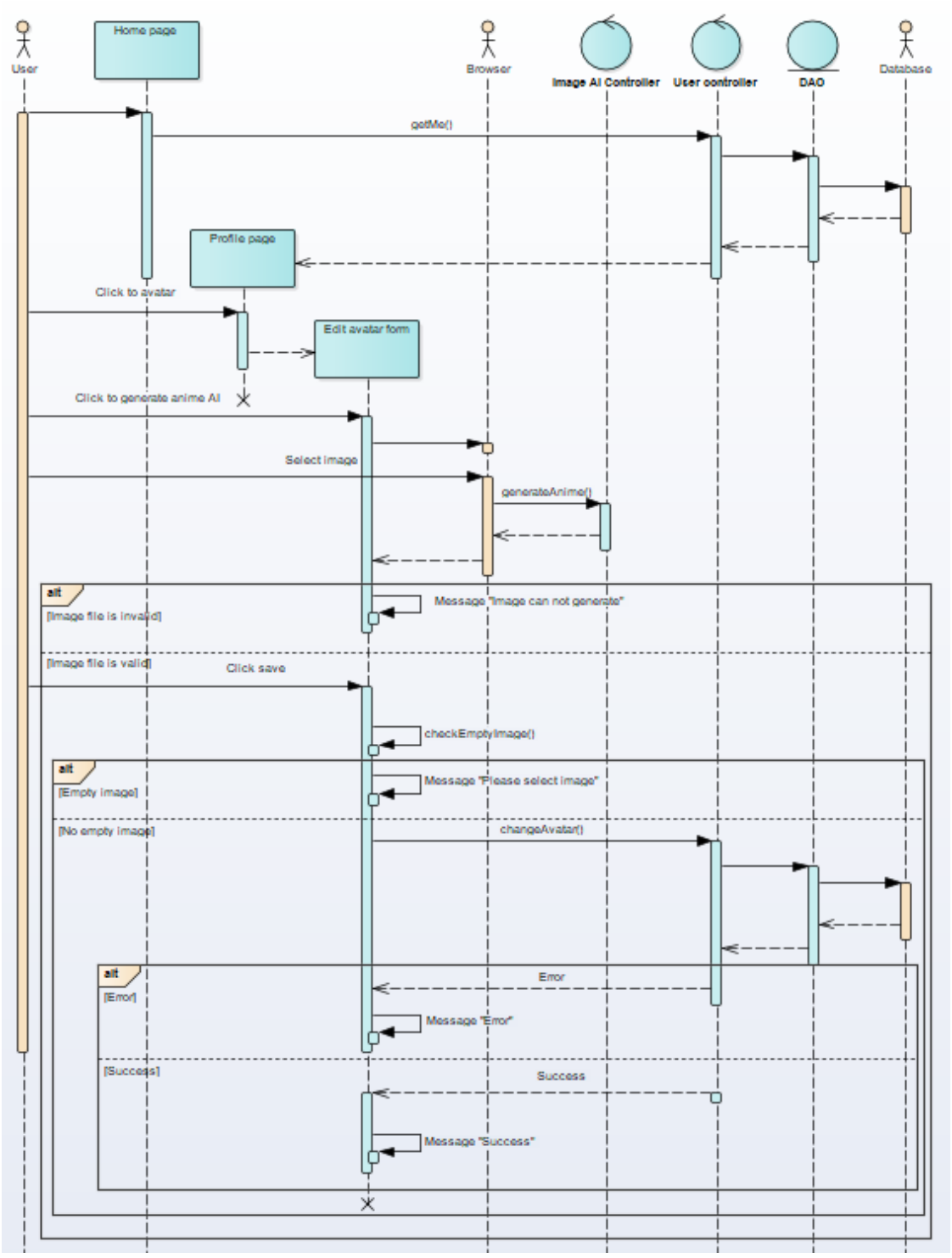


Figure 38: Sequence diagram Create avatar anime

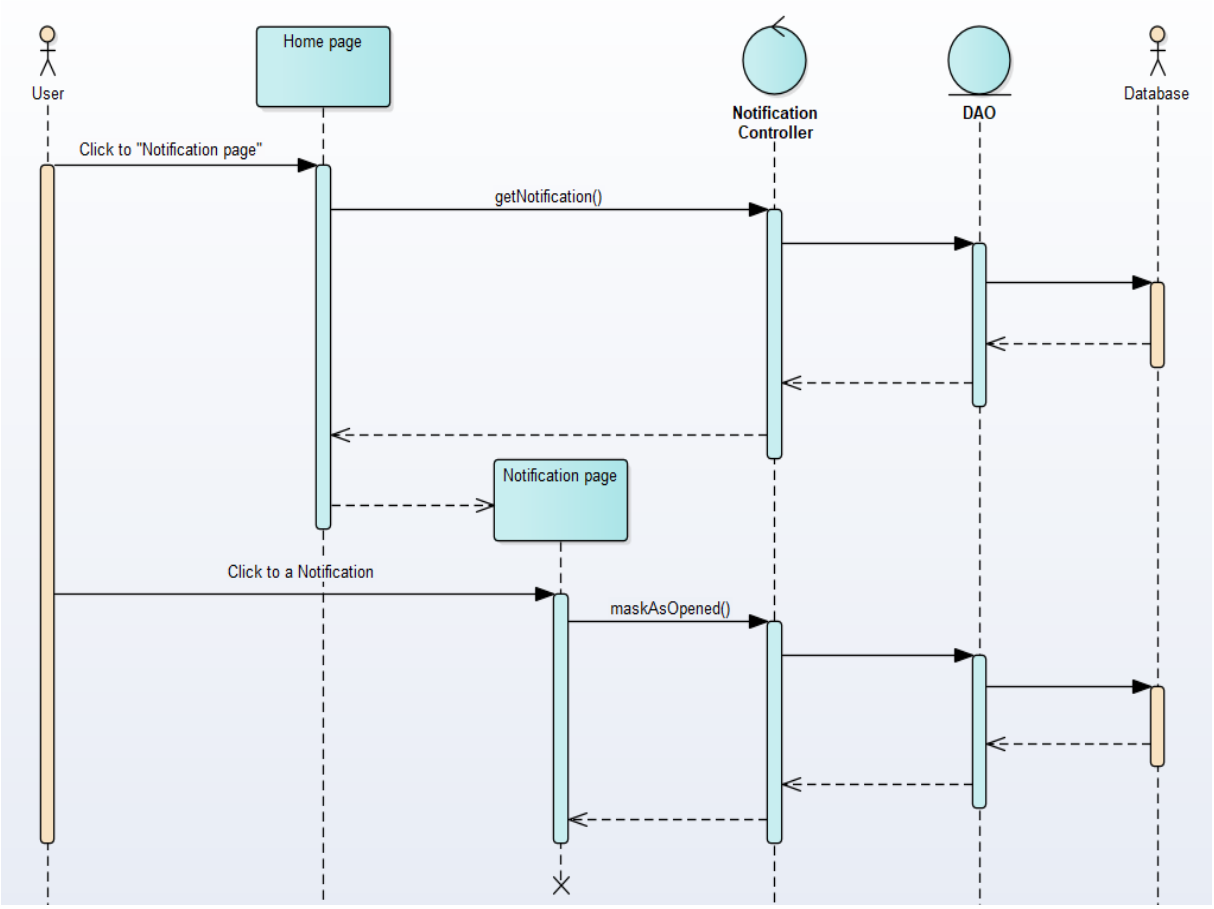


Figure 39: Sequence diagram Receive and view Notification

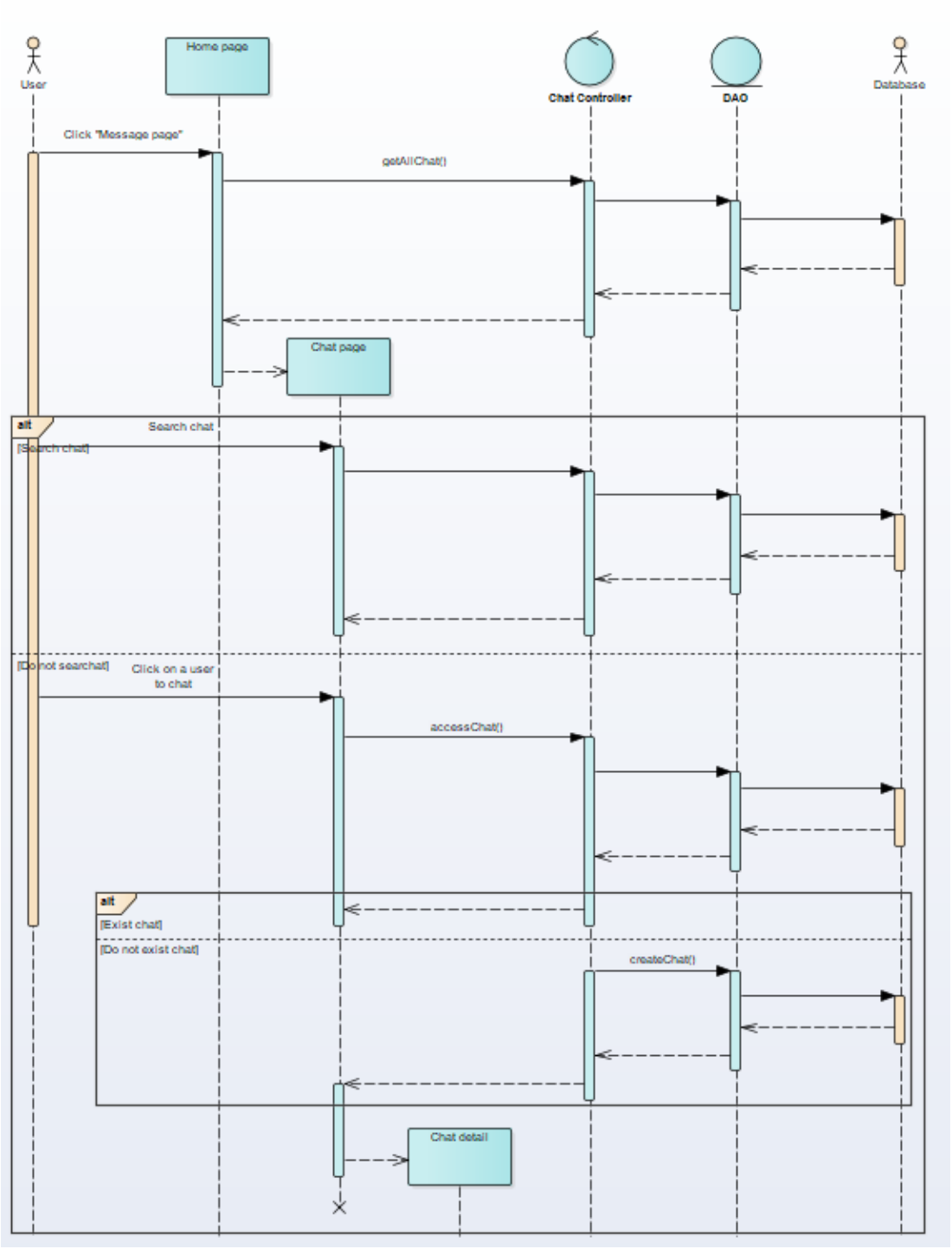
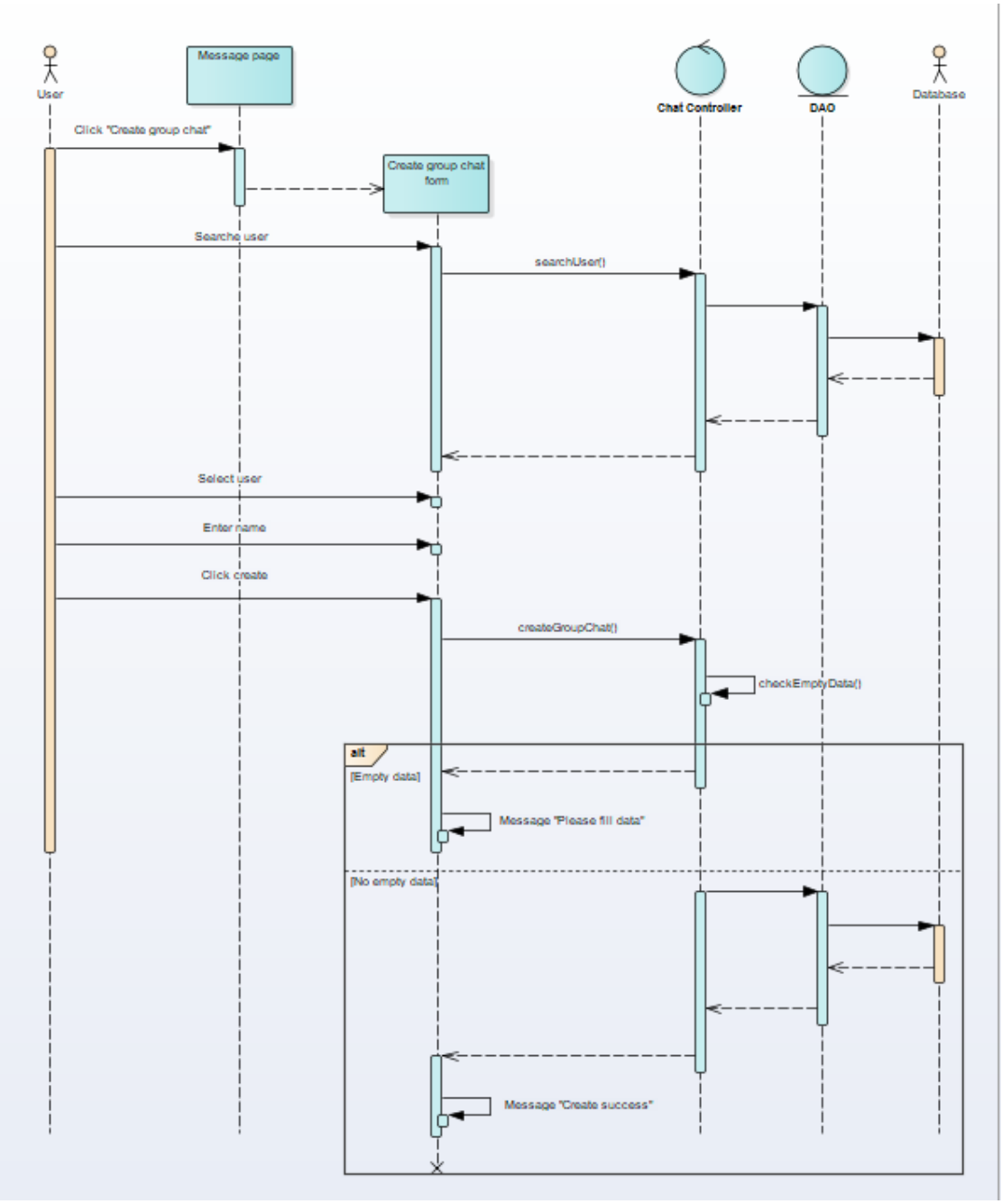
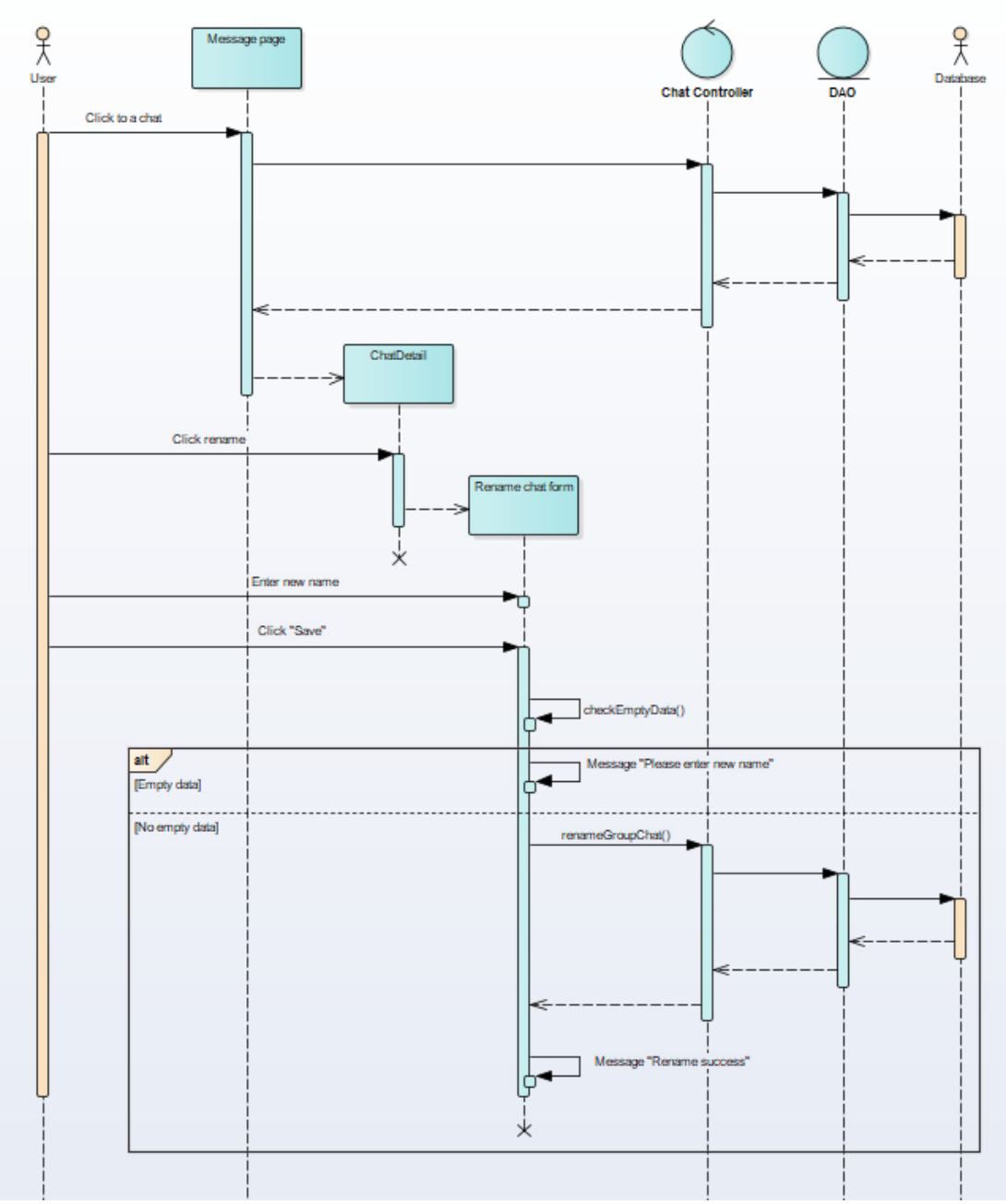
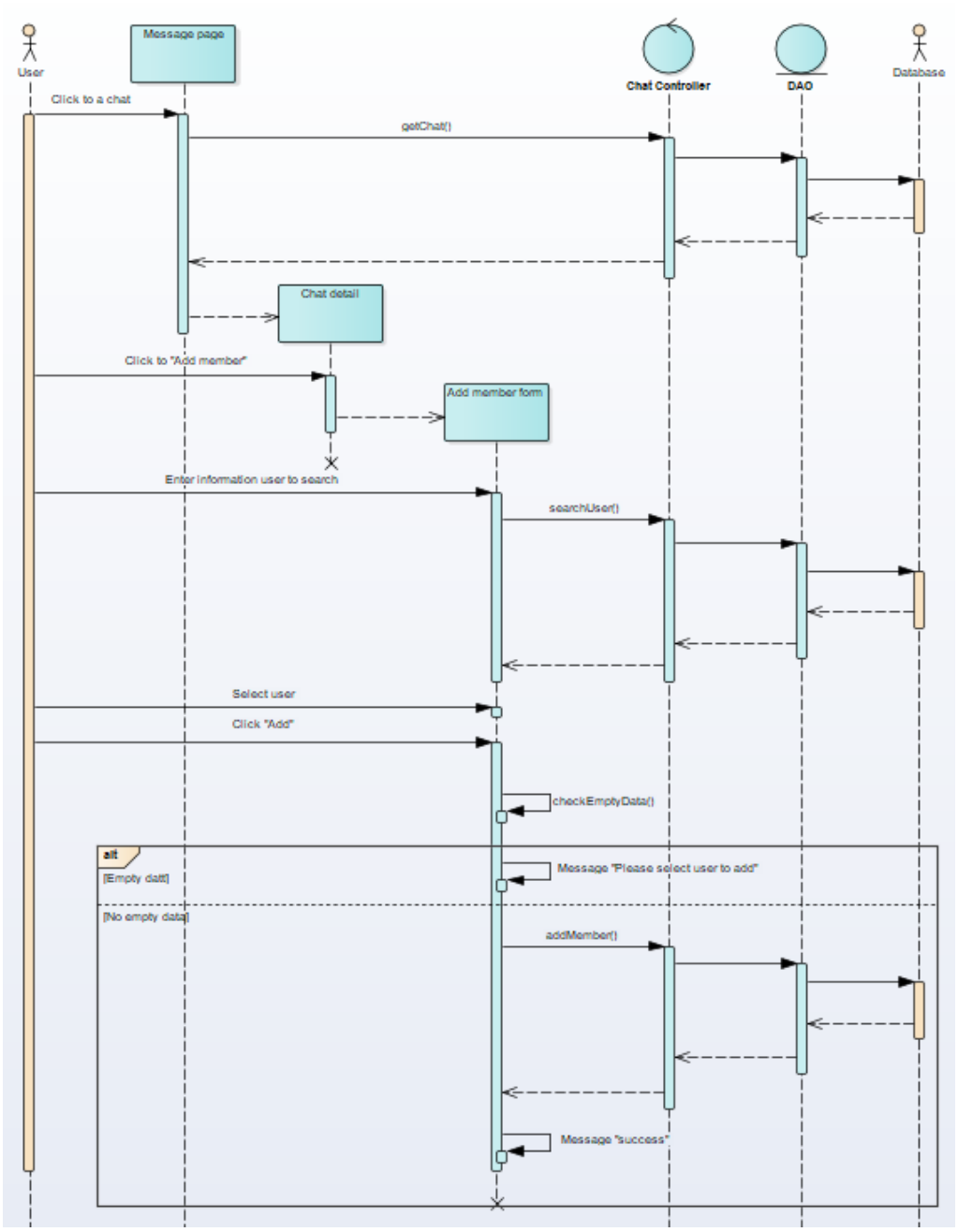


Figure 40: Sequence diagram Manage chat


User
Message page
Chat Controller
DAO
Database
Click "Create group chat"
Create group chat form
Searche user
searchUser()
Select user
Enter name
Click create
createGroupChat()
checkEmptyData()
alt
[Empty data]
Message "Please fill data"
[No empty data]
Message "Create success"


User
Message page
Chat Controller
DAO
Database
Click to a chat
ChatDetail
Click rename
Rename chat form
Enter new name
Click "Save"
checkEmptyData()
alt
[Empty data]
Message "Please enter new name"
[No empty data]
renameGroupChat()
Message "Rename success"


User
Message page
Chat Controller
DAO
Database
Click to a chat
getChat()
Chat detail
Click to "Add member"
Add member form
Enter information user to search
searchUser()
Select user
Click "Add"
checkEmptyData()
alt
[Empty data]
Message "Please select user to add"
[No empty data]
addMember()
Message "success"

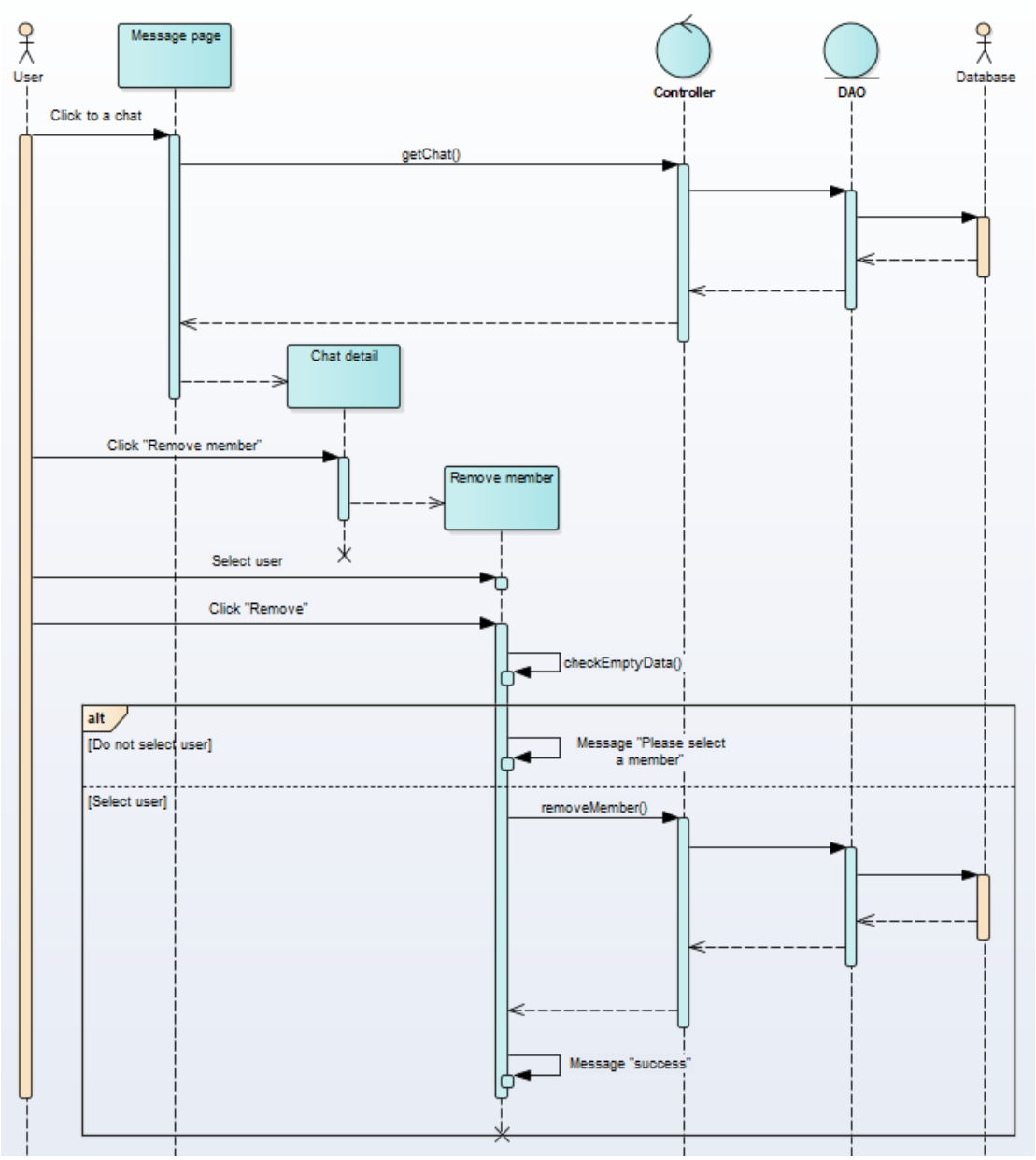


Figure 41: Sequence diagram Manage group chat

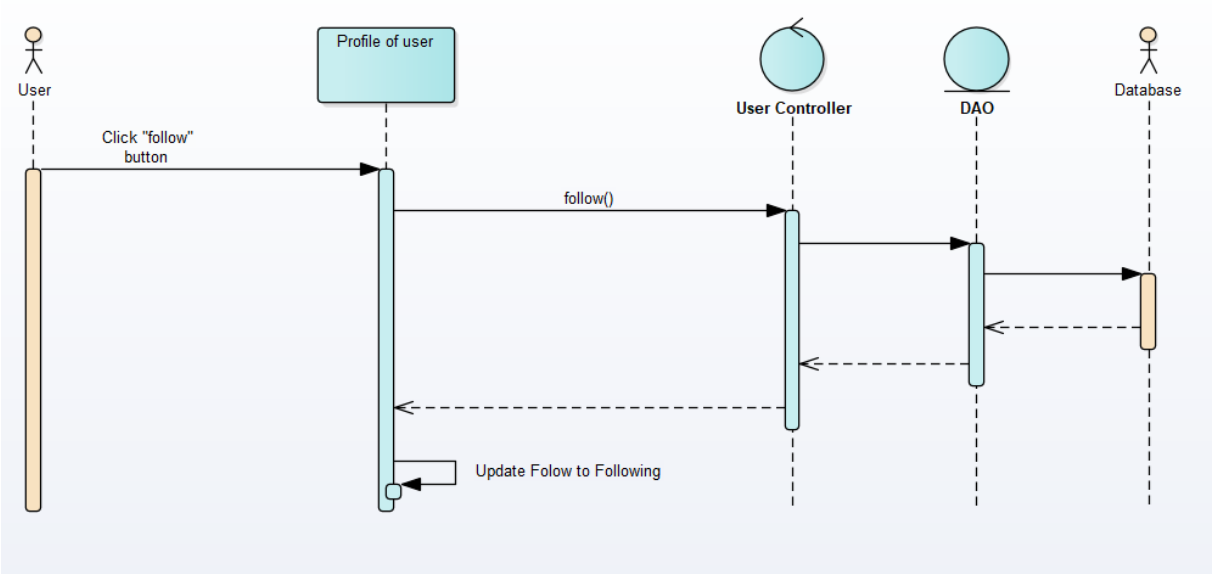


Figure 42: Sequence diagram Follow user

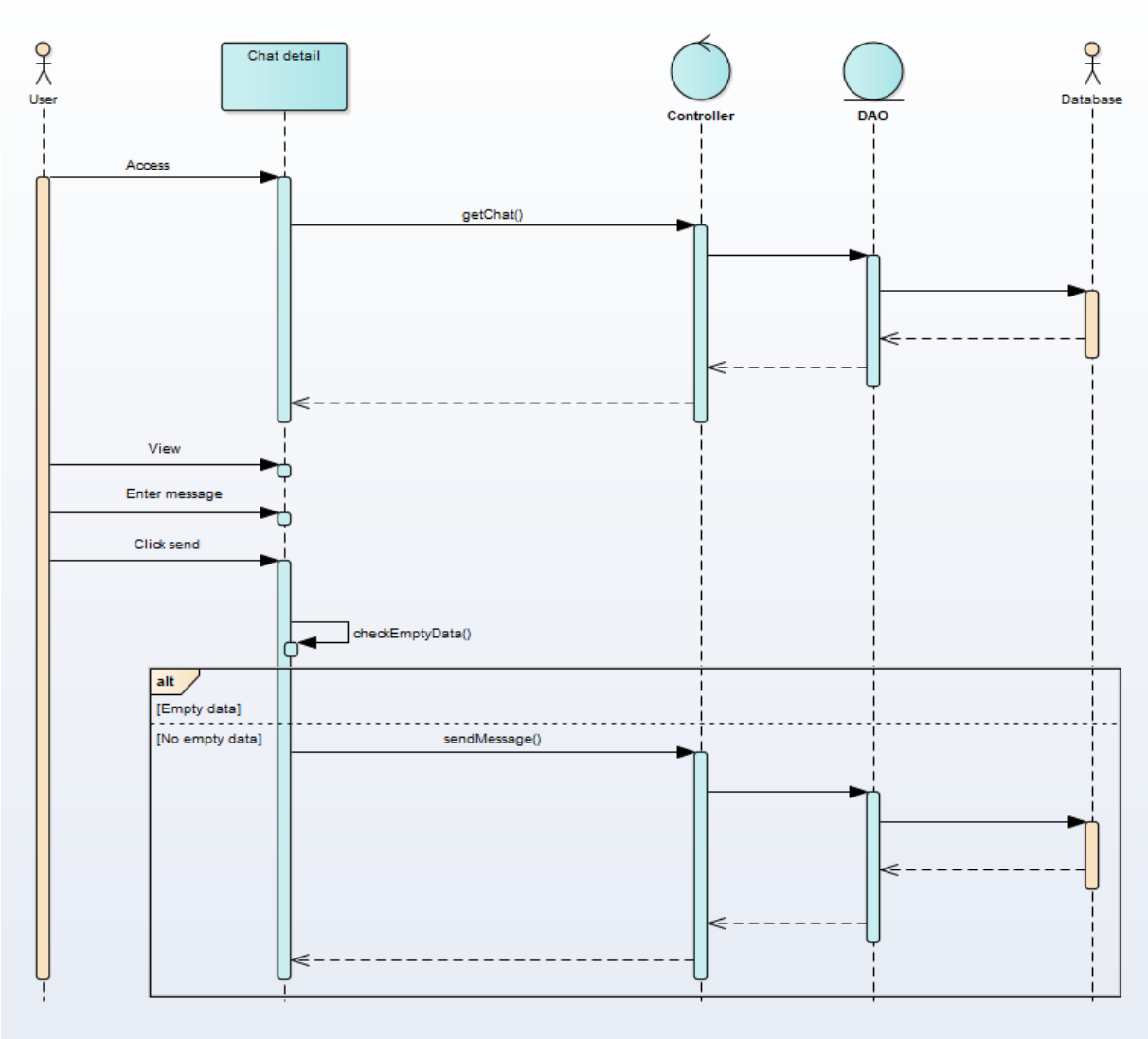


Figure 43: Sequence diagram Receive and send message

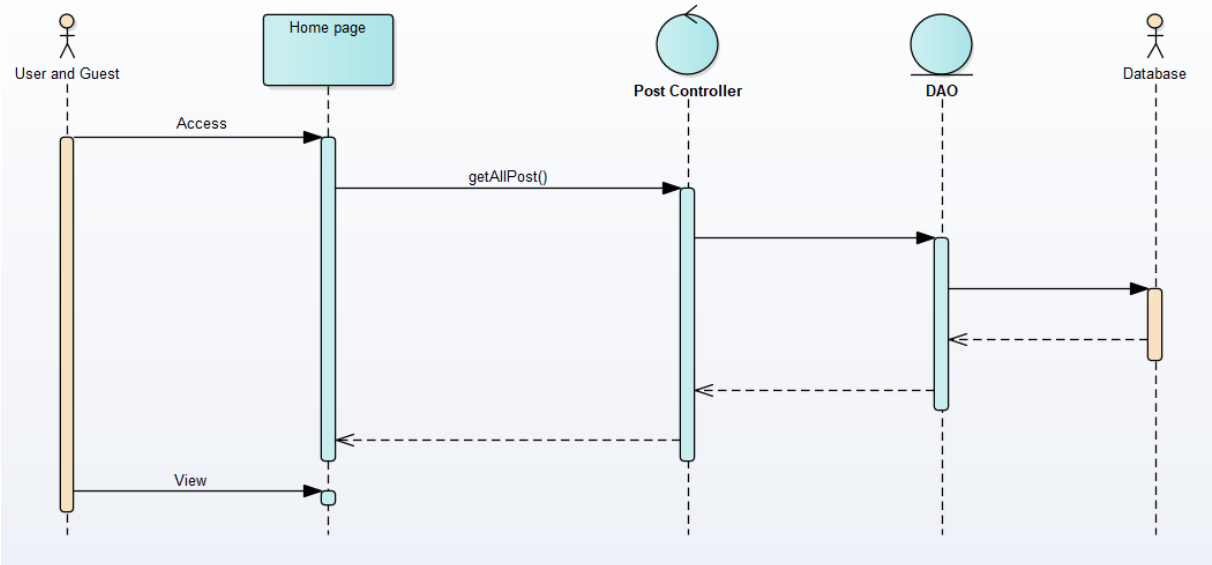


Figure 44: Sequence diagram View post

CHAPTER IV. APLLYING MACHINE LEARNING TO BLOCK NUDE IMAGES

4.1. Introduction

Social media platforms have implemented important nudity detection and blocking functionality to ensure user safety and maintain community guidelines. By using image processing technologies and algorithms, our website is able to identify objectionable content and quickly remove it, creating a safer online environment.

We use Object detection to detect:

[

"FEMALE_GENITALIA_COVERED",

"FACE_FEMALE",

"BUTTOCKS_EXPOSED",

"FEMALE_BREAST_EXPOSED",

"FEMALE_GENITALIA_EXPOSED",

"MALE_BREAST_EXPOSED",

"ANUS_EXPOSED",

"FEET_EXPOSED",

"BELLY_COVERED",

"FEET_COVERED",

"ARMPITS_COVERED",

"ARMPITS_EXPOSED",

"FACE_MALE",

"BELLY_EXPOSED",

"MALE_GENITALIA_EXPOSED",

"ANUS_COVERED",

"FEMALE_BREAST_COVERED",

"BUTTOCKS_COVERED",

]

4.2. Inference flow

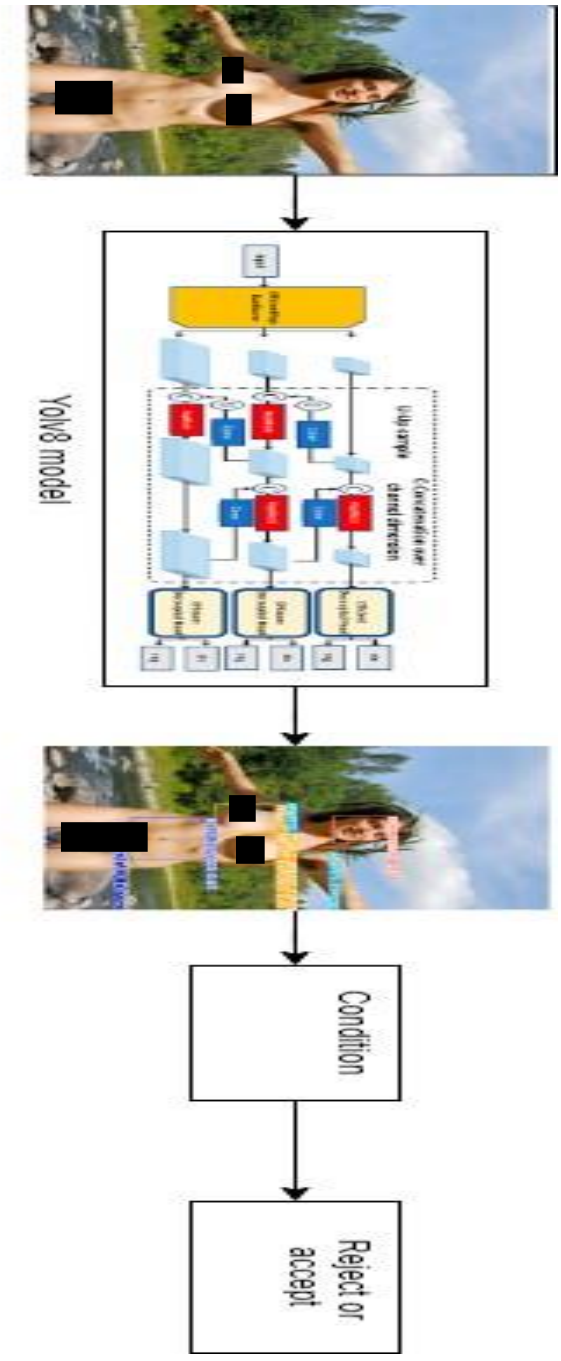


Figure 45: Inference flow of Validate Nude image

4.3. Implementation

Table 22: Description nudenet.py

	Description
__labels	It is a list that contains labels corresponding to different types of sensitive content that can be detected in an image.

_read_image function	This function takes an image in the form of a NumPy array (img) and a target size (target_size). It performs the reading and preprocessing of the image as a preprocessing step before feeding it into the nude detection model. It converts the image to RGB format, resizes it while maintaining the aspect ratio, and pads it to make it a square image. Finally, the function returns the preprocessed image data and the original size of the image.
_postprocess function	This function is used to post-process the output from the nude detection model. It takes the output from the model (output), the original size of the image (img_width and img_height), and the input size of the model (input_width and input_height) as inputs. This function extracts information about the position, score, and label of the detected objects from the output of the model. It then applies Non-Maximum Suppression (NMS) to reduce overlapping detections and returns a list of the final detections.
NudeDetector class	This is the main class for detecting sensitive content in images. In the constructor (__init__ method), the class creates a session of onnxruntime.InferenceSession to load the nude detection model from an ONNX file. It also retrieves the input size of the model. The detect method takes an image path (image_path) as input and performs the preprocessing and detection steps using the loaded model. Finally, it returns a list of processed detections.

```python
import os
import cv2
import numpy as np
import onnxruntime

__labels = [
    "FEMALE_GENITALIA_COVERED",
    "FACE_FEMALE",
    "BUTTOCKS_EXPOSED",
    "FEMALE_BREAST_EXPOSED",
    "FEMALE_GENITALIA_EXPOSED",
    "MALE_BREAST_EXPOSED",
    "ANUS_EXPOSED",
    "FEET_EXPOSED",
    "BELLY_COVERED",
    "FEET_COVERED",
    "ARMPITS_COVERED",
    "ARMPITS_EXPOSED",
    "FACE_MALE",
    "BELLY_EXPOSED",
    "MALE_GENITALIA_EXPOSED",
    "ANUS_COVERED",
    "FEMALE_BREAST_COVERED",
    "BUTTOCKS_COVERED",
]


def _read_image(img :np.array, target_size=320):
    # From ultralytics

    img_height, img_width = img.shape[:2]
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

```

```
    # Calculate the aspect ratio
    aspect = img_width / img_height

    if img_height > img_width:
        new_height = target_size
        new_width = int(target_size * aspect)
    else:
        new_width = target_size
        new_height = int(target_size / aspect)

    # Resize the image preserving aspect ratio
    img = cv2.resize(img, (new_width, new_height))

    # Pad the shorter side to make the image square
    pad_x = target_size - new_width  # Width padding
    pad_y = target_size - new_height  # height padding

    img = np.pad(
        img,
        ((pad_y // 2, pad_y - pad_y // 2), (pad_x // 2, pad_x - pad_x // 2), (0, 0)),
        mode="edge",
    )

    image_data = np.array(img) / 255.0
    image_data = np.transpose(image_data, (2, 0, 1))
    image_data = np.expand_dims(image_data, axis=0).astype(np.float32)

    return image_data, img_width, img_height


def _postprocess(output, img_width, img_height, input_width, input_height):
    outputs = np.transpose(np.squeeze(output[0]))
    rows = outputs.shape[0]
    boxes = []
    scores = []
    class_ids = []
    x_factor = img_width / input_width
    y_factor = img_height / input_height

    for i in range(rows):
        classes_scores = outputs[i][4:]
        max_score = np.amax(classes_scores)

        if max_score >= 0.2:
            class_id = np.argmax(classes_scores)
            x, y, w, h = outputs[i][0], outputs[i][1], outputs[i][2], outputs[i][3]
            left = int((x - w / 2) * x_factor)
            top = int((y - h / 2) * y_factor)
            width = int(w * x_factor)
            height = int(h * y_factor)
            class_ids.append(class_id)
            scores.append(max_score)
            boxes.append([left, top, width, height])
```

```
    indices = cv2.dnn.NMSBoxes(boxes, scores, 0.50, 0.45)

    detections = []
    for i in indices:
        box = boxes[i]
        score = scores[i]
        class_id = class_ids[i]
        detections.append(
            {"class": __labels[class_id], "score": float(score), "box": box}
        )

    return detections

```

```
class NudeDetector:
    def __init__(self):
        self.onnx_session = onnxruntime.InferenceSession(
                                r"./weights/best.onnx",
            providers=["CUDAExecutionProvider", "CPUExecutionProvider"],
        )
        model_inputs = self.onnx_session.get_inputs()
        input_shape = model_inputs[0].shape
        self.input_width = input_shape[2]  # 320
        self.input_height = input_shape[3]  # 320
        self.input_name = model_inputs[0].name

    def detect(self, image_path):
        preprocessed_image, image_width, image_height = _read_image(
            image_path, self.input_width
        )
        outputs = self.onnx_session.run(None, {self.input_name: preprocessed_image})
        detections = _postprocess(
            outputs, image_width, image_height, self.input_width, self.input_height
        )

        return detections
```

Figure 46: Nudenet.py

```python
@app.post("/api/v1/validateImage")
async def upload_image(file: UploadFile):
    try:
        # Check if the uploaded file is an image
        if not file.content_type.startswith("image/"):
            return JSONResponse(content={"message": "Uploaded file is not an image"}, status_code=400)

        # Read the image content
        image_bytes = file.file.read()

        # Process the image (e.g., resize or apply some operations)
        image = Image.open(io.BytesIO(image_bytes))
        predict= model_nude(image)
        print(predict)
        for i in predict:
            if 'EXPOSED' in i["class"]   and i["score"]>=0.6 and i["class"]!='BELLY_EXPOSED':

                return JSONResponse(content={"message":0})

        return JSONResponse(content={"message":1}, status_code=200)

        # Return the generated image as a response

    except Exception as e:
        return JSONResponse(content={"message": f"Error: {str(e)}"}, status_code=500)
```

Figure 47: API validate image

4.4. Experimental results

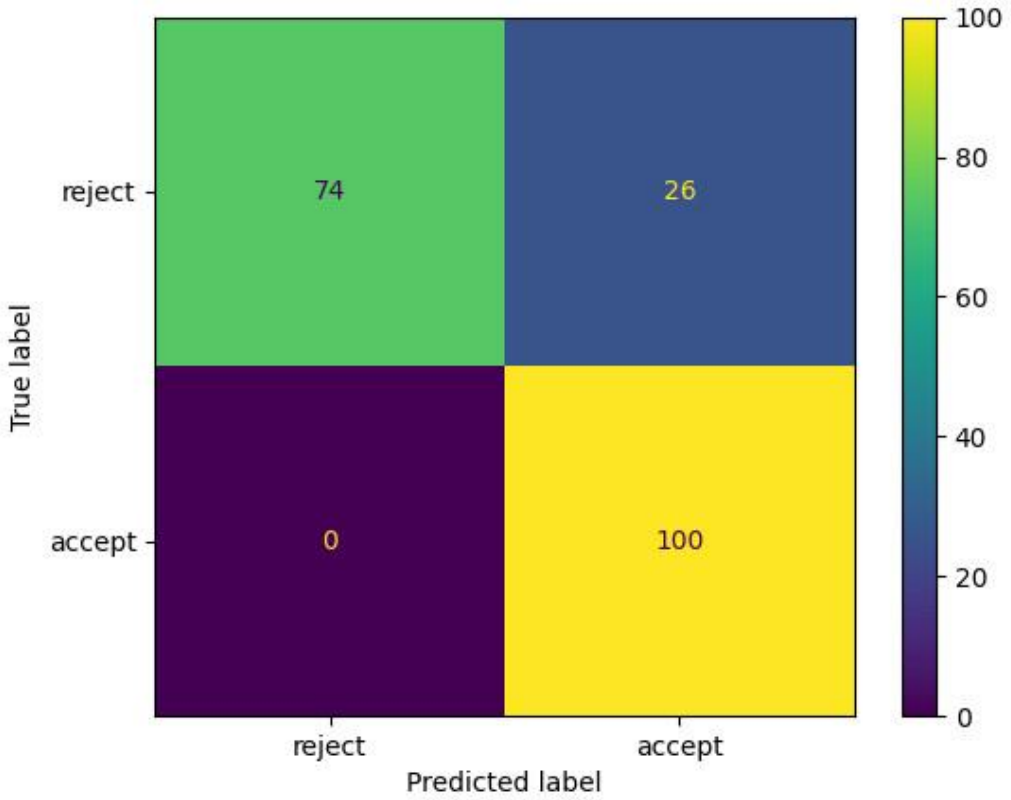


Figure 48: Confusion matrix

	precision	recall	f1-score	support
0.0	1.00	0.74	0.85	100
1.0	0.79	1.00	0.88	100
accuracy			0.87	200
macro avg	0.90	0.87	0.87	200
weighted avg	0.90	0.87	0.87	200

Figure 49: Classification report

Our Test set includes 100 nude samples and 100 non-nude samples corresponding to 100 reject cases and 100 accept cases.

Based on the confusion matrix we can calculate,

- The model correctly identified almost all non-nude photos
- The model only correctly identified 74% of nude photos and missed 26% of nude photos

Based on classification report

- Precision: 0.79 score is okay but still not high, proving that the model also predicts Nude images with not very high accuracy, with a high error of 21%, however the accuracy of non-nude images is good at 1.0.
- Recall: scorte is 0.74, a temporary score, proving that the model did not miss too many nude photos

➢ Through experimental results, we see that the model has the ability to effectively filter Nude images but needs to be improved in the future

CHAPTER V. SYSTEM INTERFACE

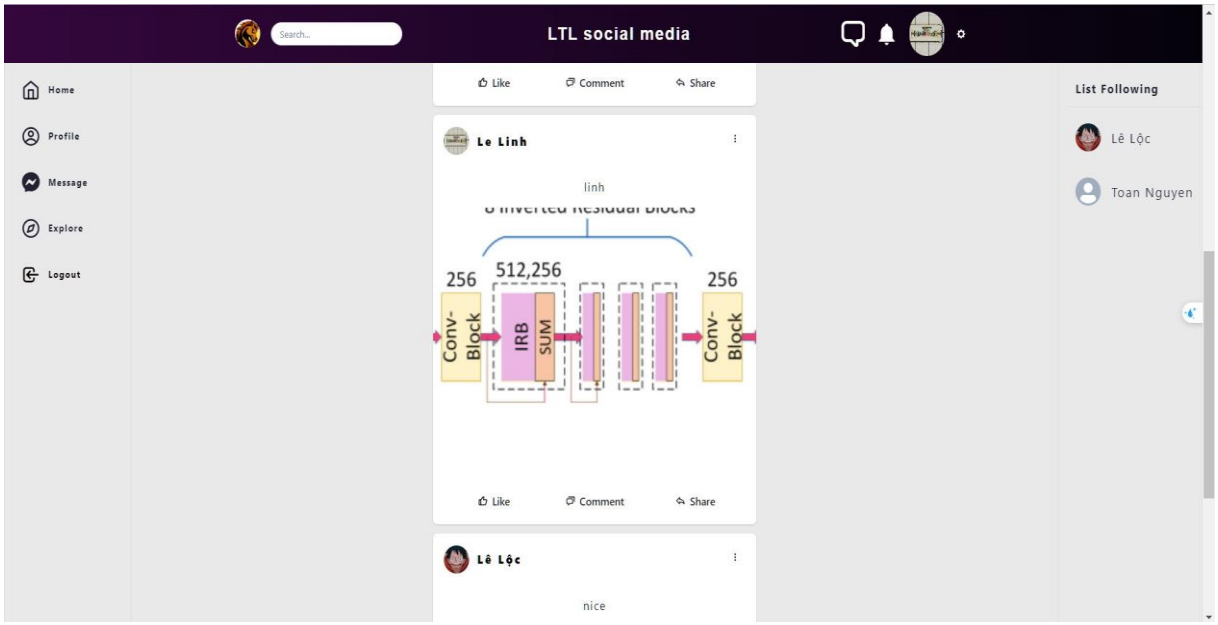


Figure 50: Home page

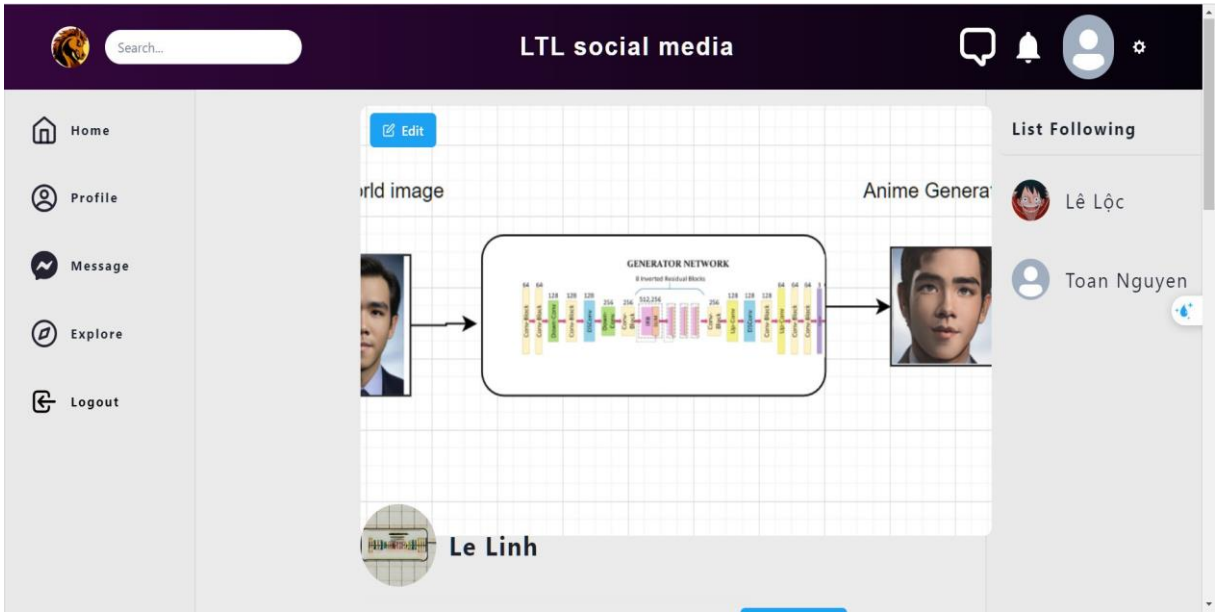


Figure 51: Profile page

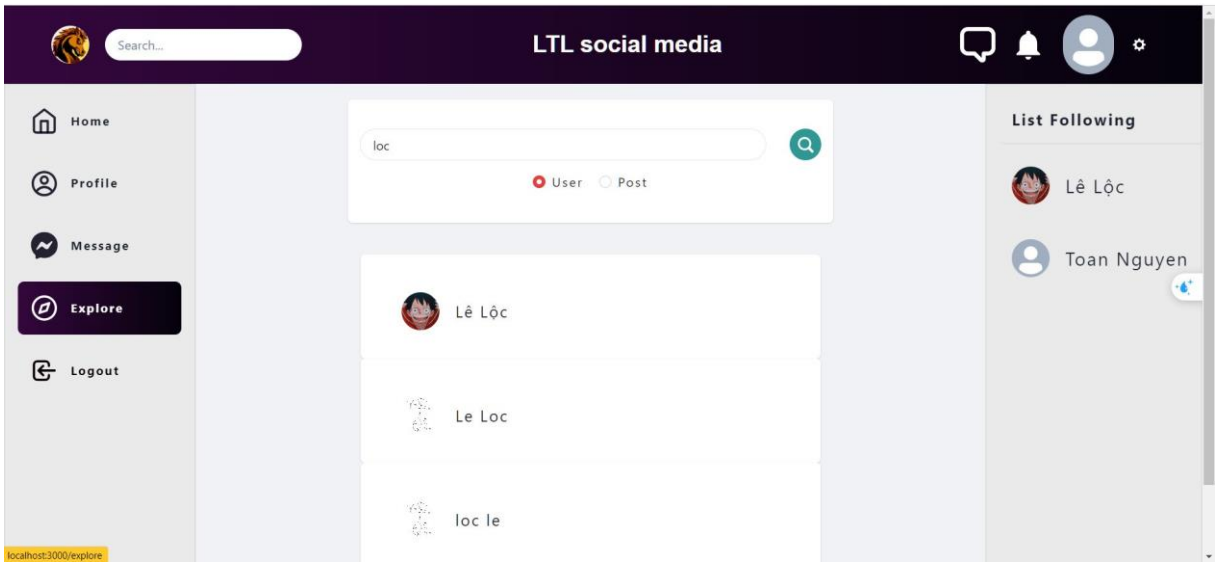


Figure 52: Explore page (search user)

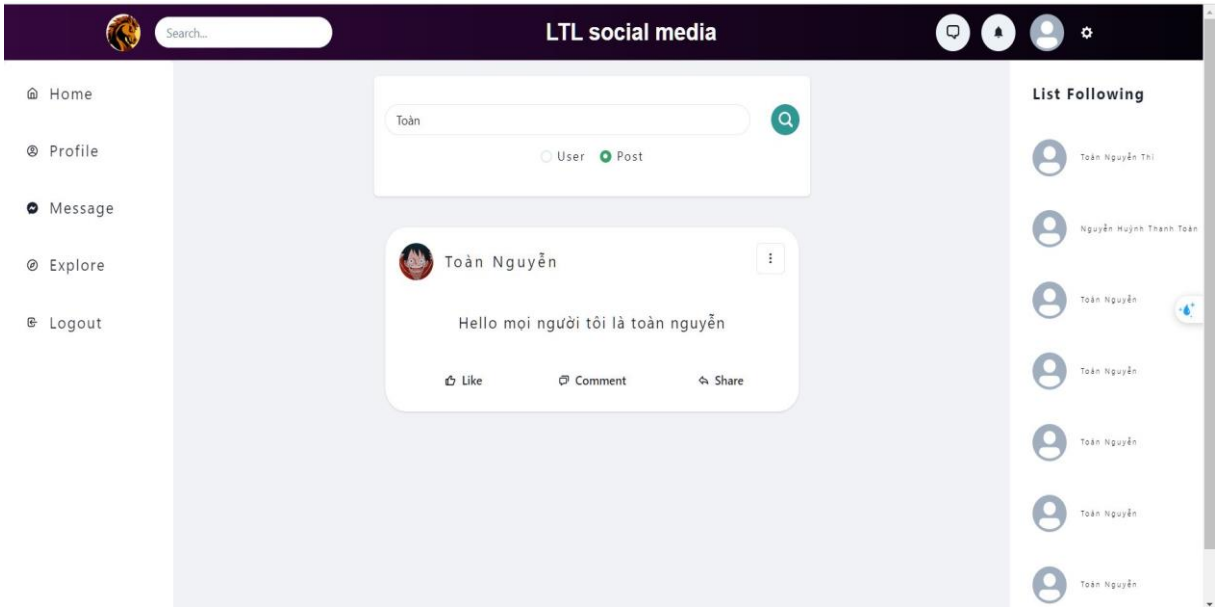


Figure 53: Explore page (search post)

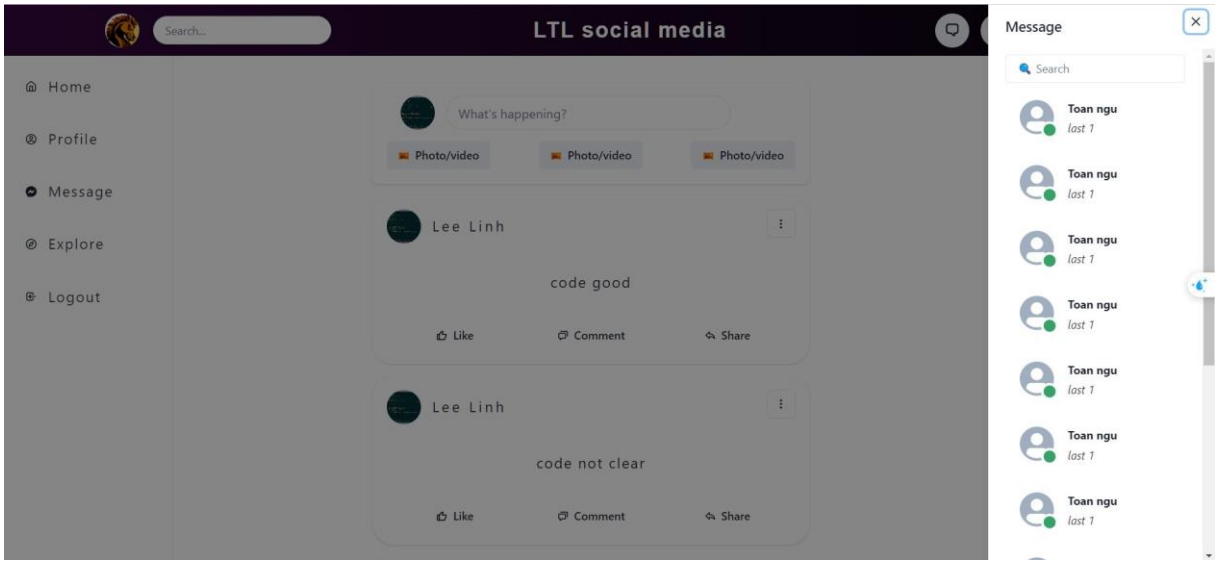


Figure 54: List chat

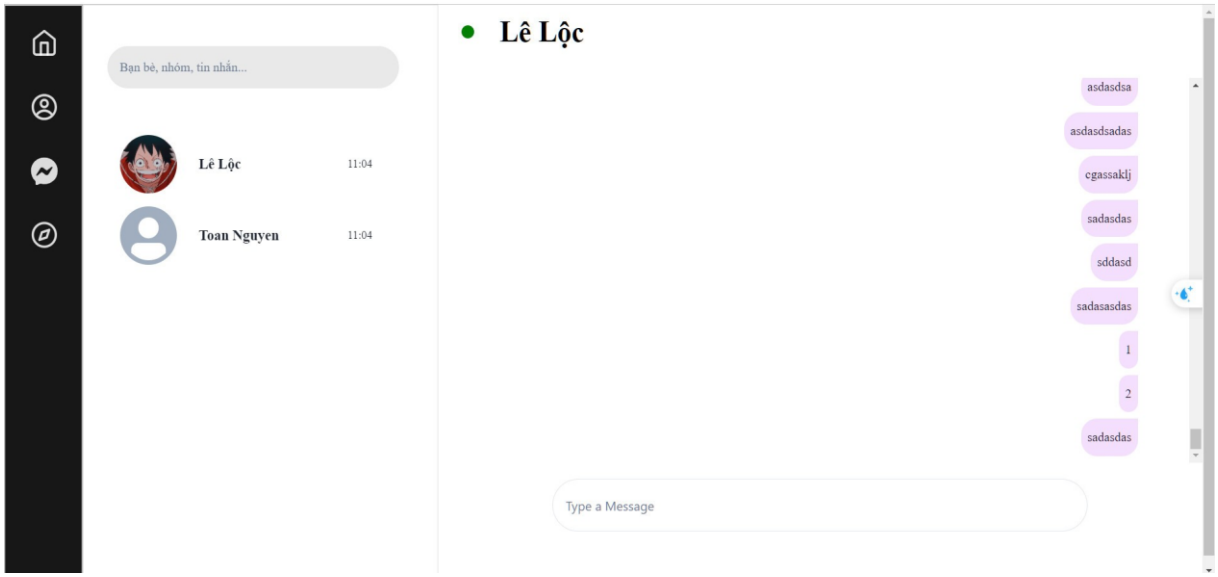


Figure 55: Message page

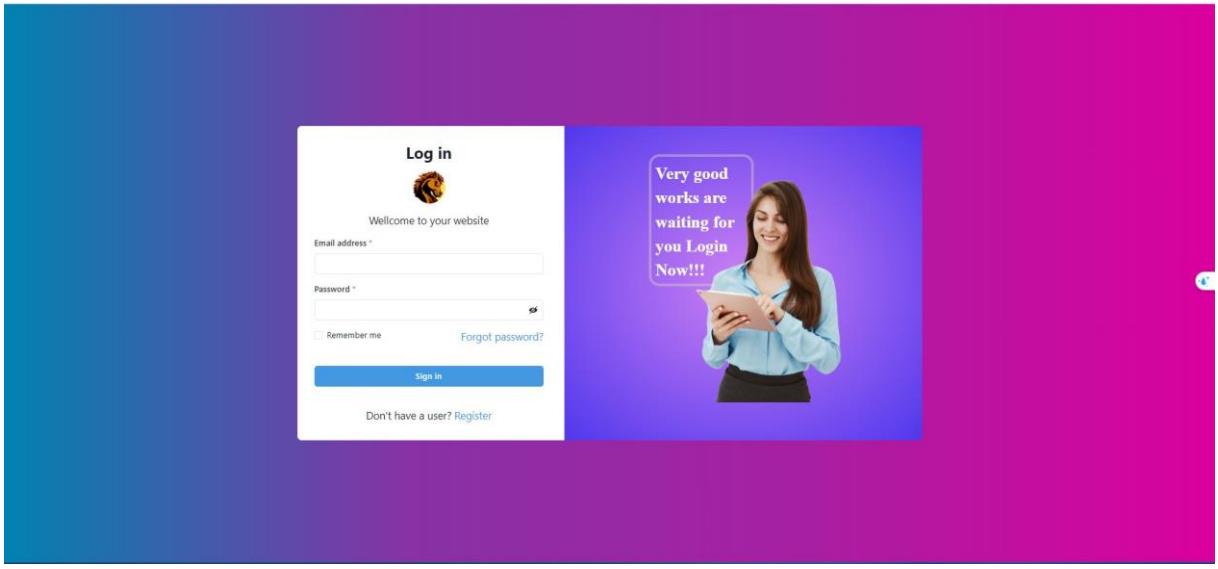


Figure 56: Log in form

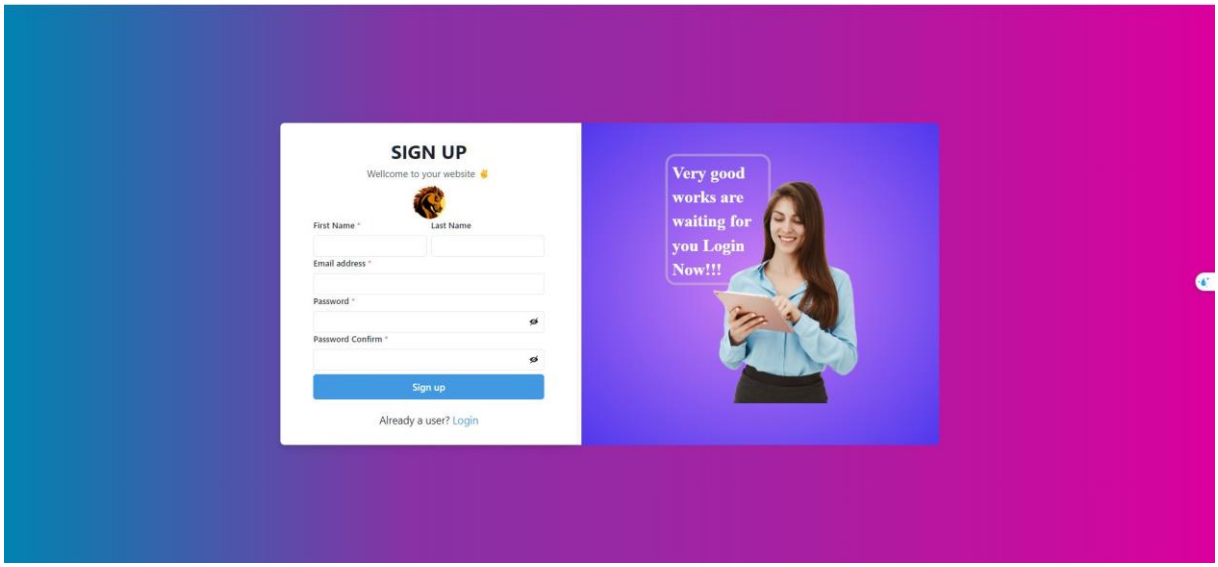


Figure 57: Sign up form

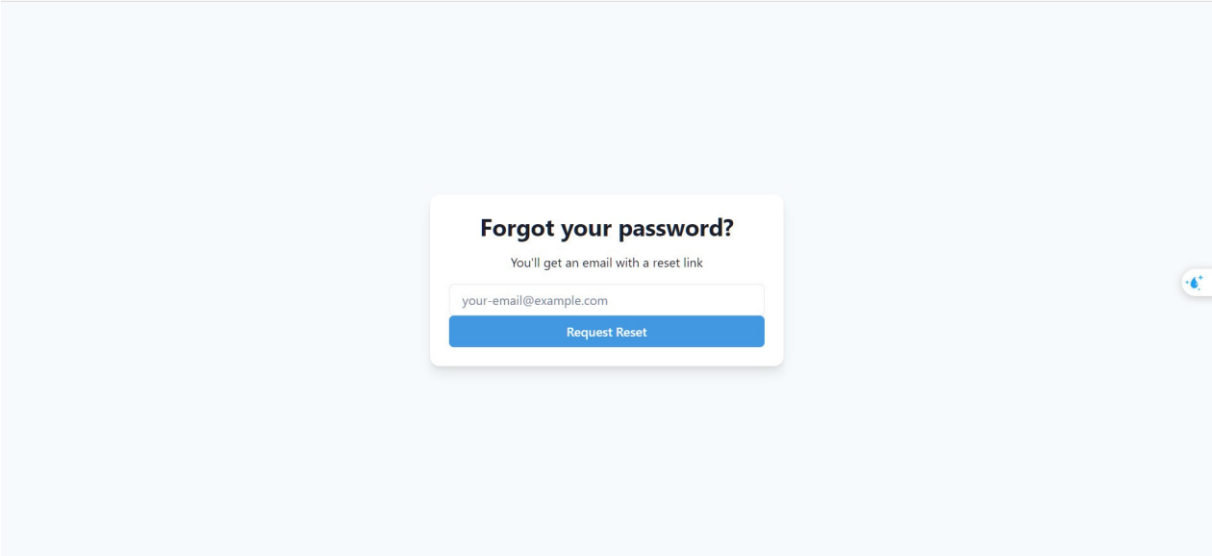


Figure 58: Forgot password form

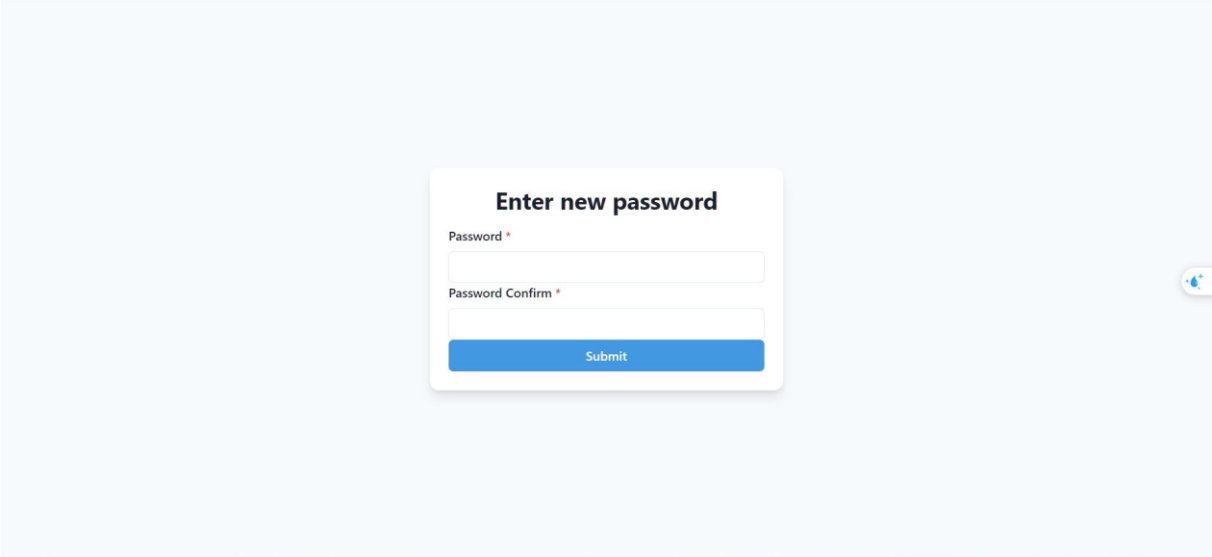


Figure 59: Reset password form

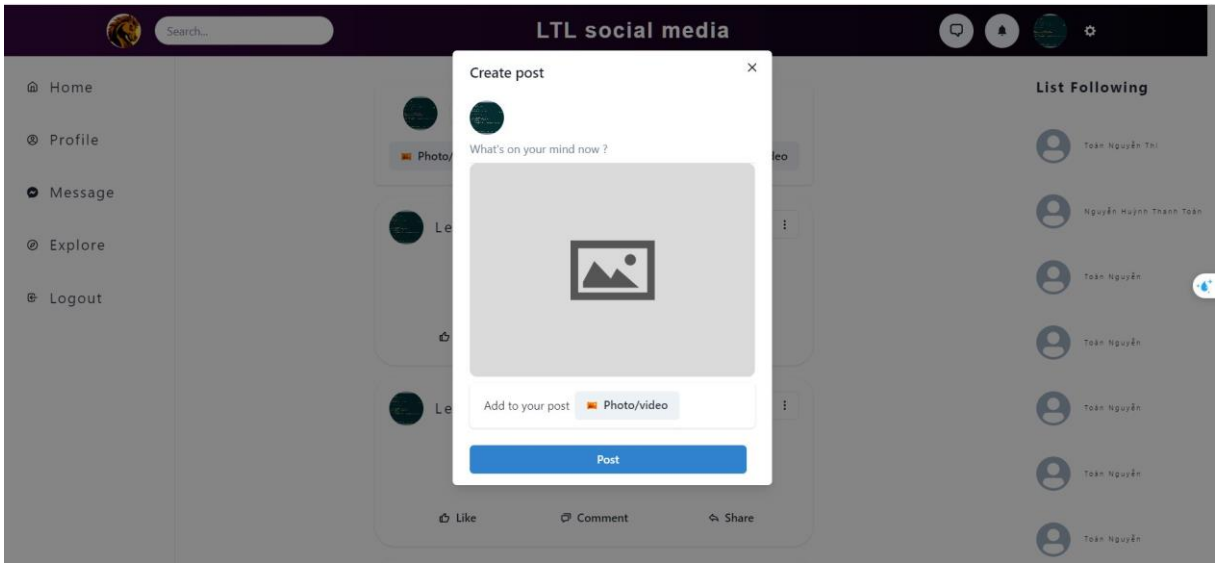


Figure 60: Create post form

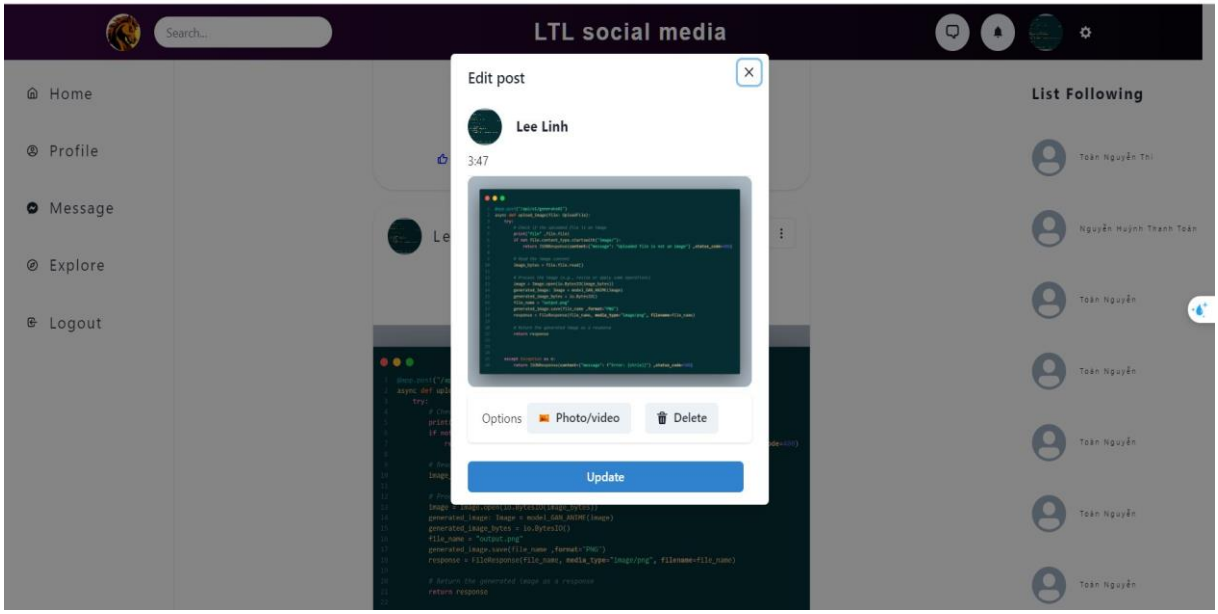


Figure 61: Edit post form

Figure 62: Post detail page

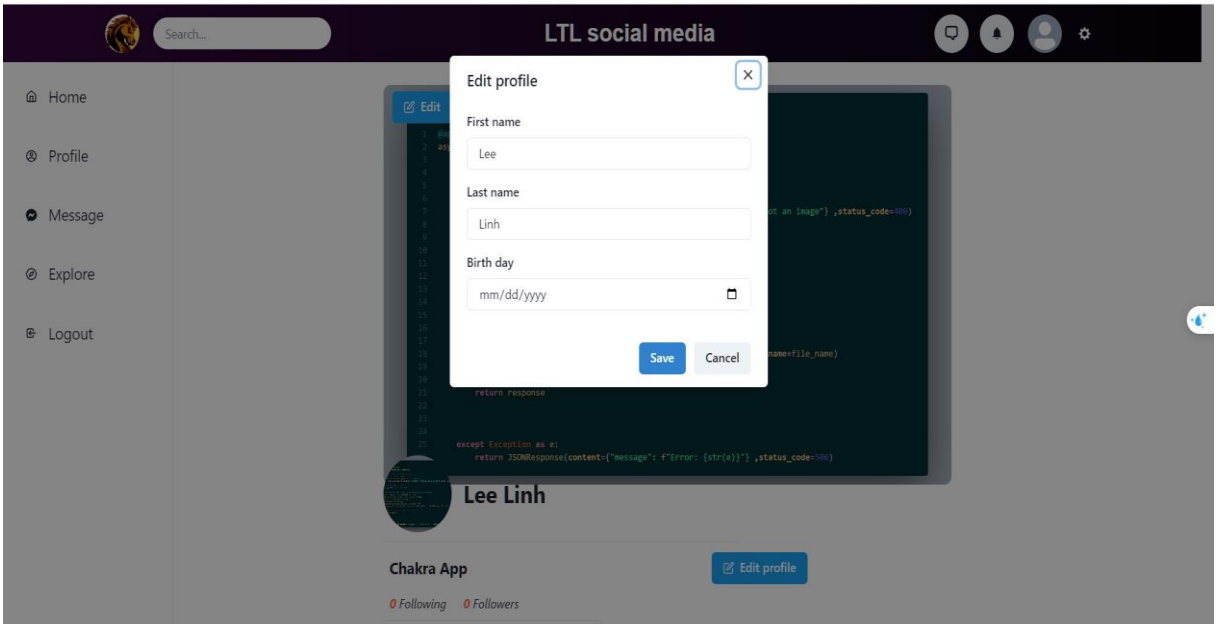


Figure 63: Edit information form

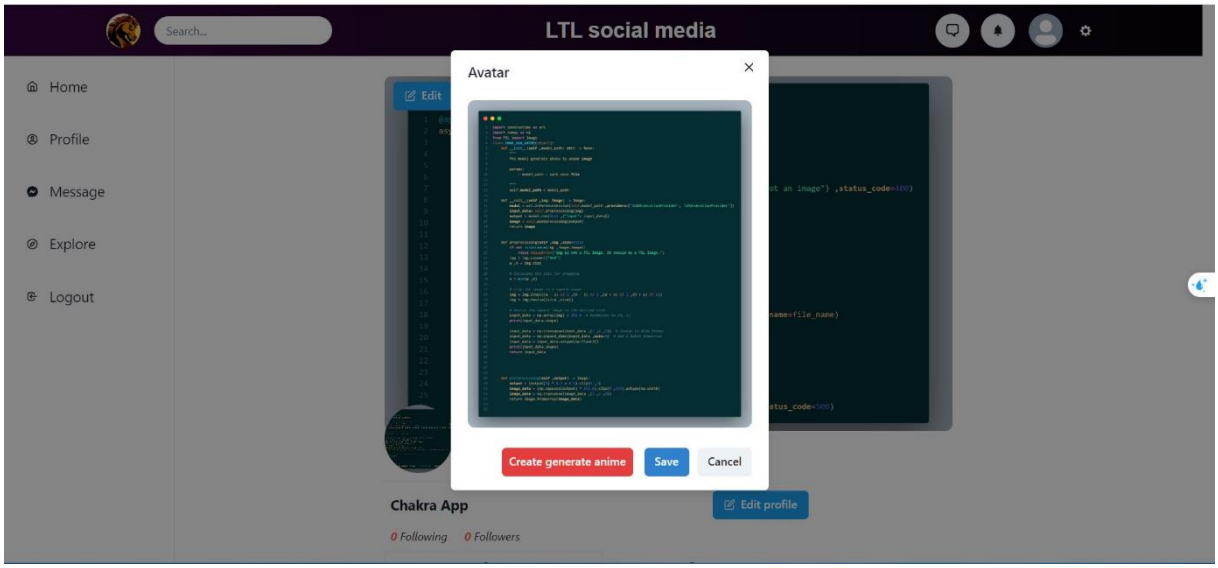


Figure 64: Edit avatar form

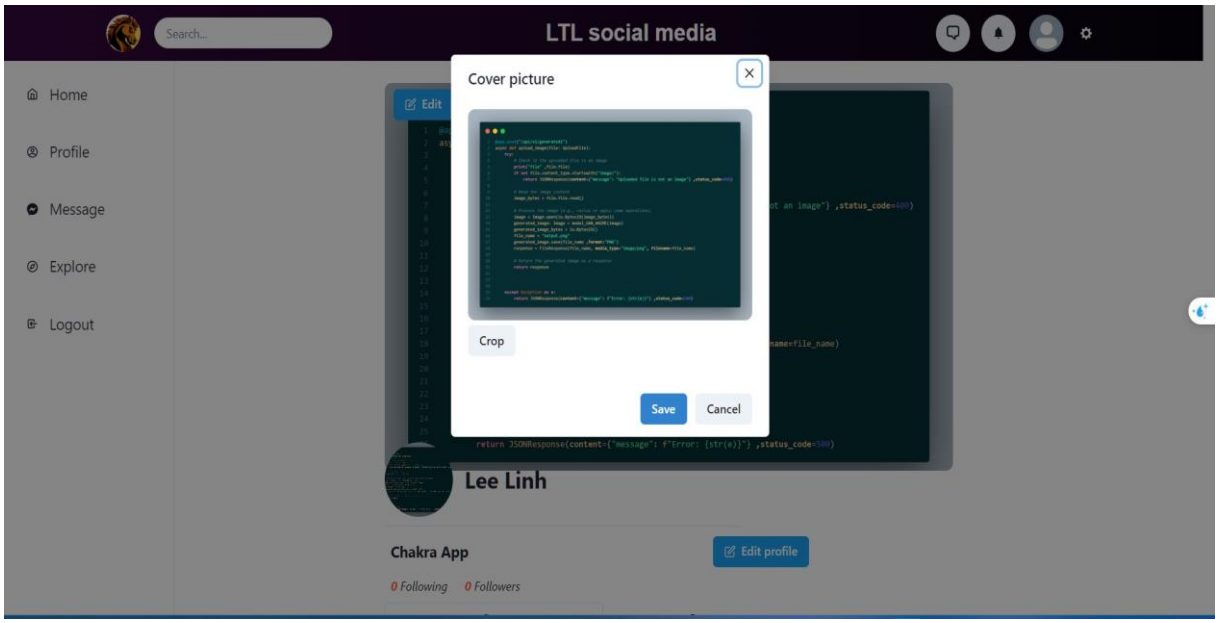


Figure 65: Edit cover photo form

CHAPTER VII. CONCLUSION

6.1. Learning from project

Building a social media website that applies machine learning (ML) to process input images using the MERN stack (MongoDB, Express.js, React.js, Node.js) provided several valuable learning experiences, including:

Full-stack web development: The project offered an opportunity to gain proficiency in the MERN stack, which encompasses both front-end and back-end development. Learning to work with MongoDB for data storage, Express.js for server-side development, React.js for building the user interface, and Node.js for server-side runtime environment provided a comprehensive understanding of full-stack web development.

Integration of ML algorithms: Incorporating ML algorithms into the website required understanding how to integrate them within the MERN stack. This involved setting up endpoints in the Node.js server to handle ML requests, passing data to ML models, and integrating the ML results into the React.js front-end for display.

Data management and querying: Working with MongoDB involved learning how to manage and query data efficiently. Understanding MongoDB's document-based structure, creating schemas for image data, and implementing queries to retrieve and manipulate data were important aspects of the project.

User authentication and security: Implementing user authentication and ensuring secure user interactions were crucial for the social media website. Learning how to use authentication libraries, such as bcryptjs jwtwebtoken, ... and implementing secure API routes to protect user data were essential components of the project.

6.2. Difficulties

ML model integration: Integrating ML models into the website's architecture can be complex. Ensuring smooth communication between the ML models and the rest of the stack, handling data preprocessing, and managing ML model updates or retraining can pose difficulties.

Scalability and performance: As the website grows and handles a larger user base, ensuring scalability and performance becomes crucial. Optimizing server-side code, implementing caching mechanisms, and scaling the web application to handle increased traffic are challenging tasks.

Complex UI/UX requirements: Designing an intuitive and visually appealing user interface (UI) that allows users to interact with image processing features can be challenging. Balancing functionality, responsiveness, and aesthetic appeal while providing a seamless user experience requires careful attention to detail.

Data privacy and security: Handling user data and ensuring its privacy and security are paramount. Implementing secure authentication, encrypting sensitive data, and following best practices for data storage and access control can be demanding.

6.3. Future development

Improved ML algorithms: Continuously enhancing ML algorithms to improve image processing accuracy, explore additional features like object recognition or sentiment analysis, and leverage advanced ML techniques such as deep learning or reinforcement learning.

Enhanced user experience: Further refining the UI/UX design to provide a seamless and engaging experience for users. This could involve adding interactive features, optimizing page load times, and incorporating user feedback for iterative improvements.

Mobile application development: Expanding the project to include mobile app development using frameworks like React Native or building native apps. This would allow users to access the social media website and its ML-based image processing features through mobile devices.

Community building and social features: Implementing features that foster community interaction, such as user profiles, following/follower systems, likes/comments, and sharing capabilities. These enhancements can enhance user engagement and create a more dynamic social media experience.

Integration with cloud services: Exploring integration with cloud services like AWS or Google Cloud to leverage their ML services, storage options, and scalability features. This could enhance the website's capabilities, improve performance, and streamline ML model deployment and management.

TASK ASSIGNMENT

Table 23: Task assignment

ID	Name	Task
20110012	Nguyễn Thanh Toàn	Back-end (90%) Write report (50%)
20110377	Lê Khánh Linh	AI (100%) Back-end (10%) Front-end (40%) Write report (20%)
20110379	Lê Tấn Lộc	Front-end (60%) Write report (30%)

REFERENCE

[1]. Node.js – Node.js guides - https://nodejs.org/en/guides

[2]. Mongoose – Mongoose guides - https://mongoosejs.com/docs/index.html

[3]. Socket.IO – Socket.IO guides - Client API | Socket.IO

[4]. Home - Ultralytics YOLOv8 Docs

[5]. https://chakra-ui.com/getting-started

[6]. AnimeGANv2 – Xin Chen – August, 2020 – AnimeGANv2 (tachibanayoshino.github.io)