INTERNATIONAL UNIVERSITY

VIETNAM NATIONAL UNIVERSITY, HCM CITY

School of Computer Science & Engineering

PROJECT

Lecturer: Assoc.Prof. Nguyen Van Sinh

M.Sc. Nguyen Trung Nghia

Course: Web Application Development

Mini Instagram

Group 3

Nguyễn Hùng Quốc	ITCSIU21101
Văn Bảo Khánh	ITITWE19026

Ho Chi Minh, June 2024

CONTENTS

List of figures

I. Introduction

1. Project

Mini Instagram is a clone application of Instagram with other UI. We first aimed to use Microservices Architecture to develop this project. But, since we do not have enough experience, we cannot connect FE(ReactJS) and BE(Spring Boot) together and we developed a Firebase Server (Backend as a Services) instead of Spring Boot to demo.

Deployment: https://minigram-web.vercel.app/

Front End: https://github.com/AesirHres/minigram_web.git

Back End Firebase

Back End Spring Boot: https://github.com/quocnguyenhung/Mini-Instagram (cannot connect)

2. Team

Name	Student ID	Contribute
Van Bao Khanh	ITITWE19026	50%
Nguyen Hung Quoc	ITCSIU21101	50%

3. Gantt Chart

Figure I.3. Gantt Chart

When we began this project, we planned to develop this project following this Gantt Chart. We used Agile since the technology and functionalities of our application are suitable with this method. Although we must complete this project earlier for an internship application, this plan is our backbone.

II. Requirements Analysis

1. **Use Case Diagram**

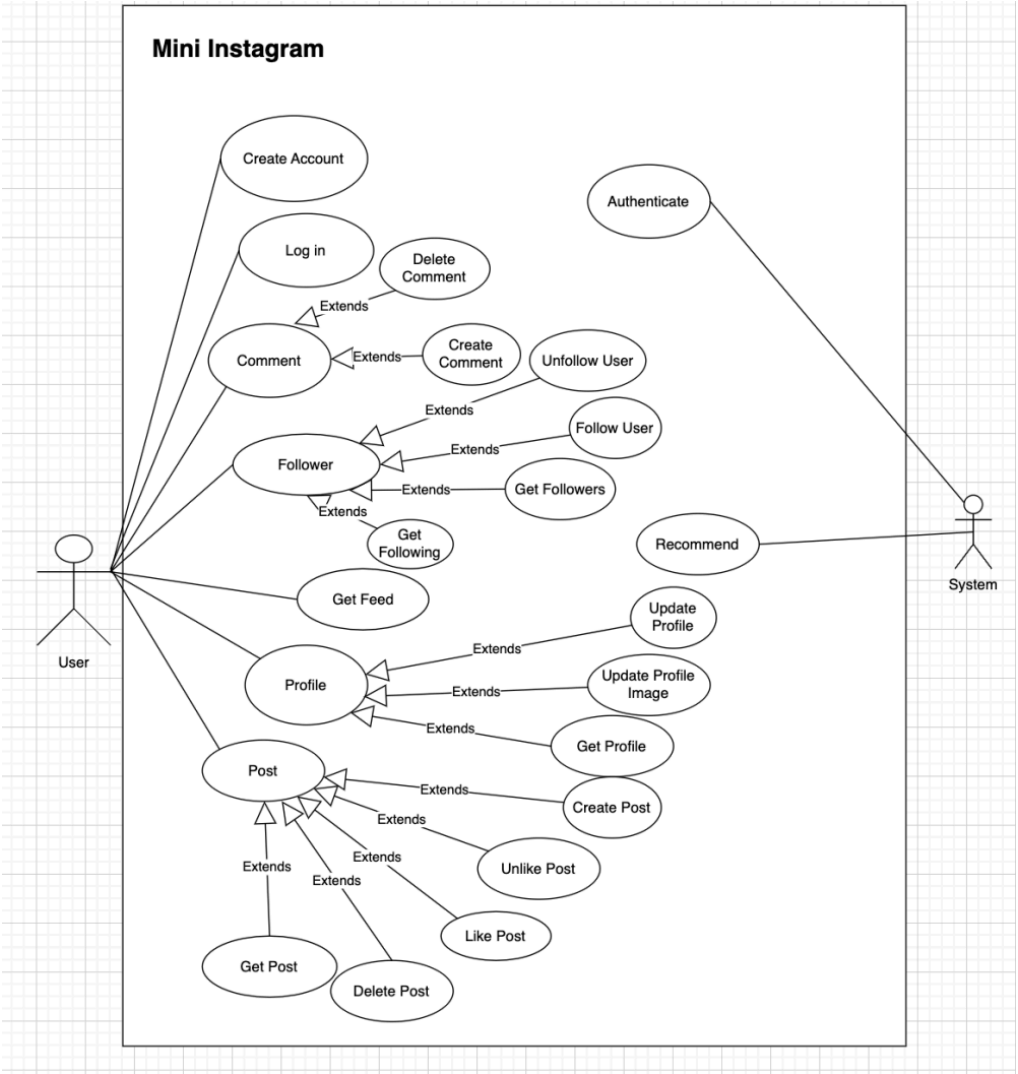


Figure II.1. Use case diagram

Actors:

- User
- System

Use Cases:

- Create account
- Log in
- View Profile
- Edit Profile
- Create Post

- Like Post
- Comment on Post
- Follow User
- Unfollow User
- View Feed
- View Followers
- View Following

Detailed Explanations for Each Use Case

USE CASE #1:

Name: Create account
Inputs: User details (e.g., name, email, password)
Outputs: User account creation confirmation
Basic Course:

1. User navigates to the registration page.
2. User enters their details.
3. System validates the information.
4. System creates a new user account.
5. User receives confirmation of account creation.

   Precondition: User is not logged in and does not have an existing account.
   Postcondition: User account is created and user can log in.
   User Story: As a new user, I want to register for an account so that I can access the platform's features.

USE CASE #2:

Name: Log in
Inputs: User credentials (username/email and password)
Outputs: User session
Basic Course:

1. User enters login credentials.

2. System validates the credentials.
3. System creates a session for the user.
4. User is redirected to the homepage.

   Precondition: User has a registered account.
   Postcondition: User is logged into their account.
   User Story: As a user, I want to log in to my account so that I can access personalized features.

USE CASE #3:

Name: View Profile
Inputs: User ID
Outputs: User profile information
Basic Course:

1. User navigates to a profile page.
2. System retrieves and displays the profile information.

   Precondition: Profile exists in the system.
   Postcondition: Profile information is displayed.
   User Story: As a user, I want to view profiles so that I can see information about myself and other users.

USE CASE #4:

Name: Edit Profile
Inputs: Updated user information
Outputs: Confirmation of profile update
Basic Course:

1. User navigates to the edit profile page.
2. User updates their profile information.
3. System validates and saves the updated information.
4. User receives confirmation of the profile update.

Precondition: User is logged in.
Postcondition: Profile information is updated and saved in the system.
User Story: As a user, I want to edit my profile so that I can keep my information up to date.

USE CASE #5:

Name: Create Post
Inputs: Content (image, caption)
Outputs: New post
Basic Course:

1. User navigates to the create post page.
2. User uploads an image and writes a caption.
3. System saves the post and displays it in the feed.

Precondition: User is logged in.
Postcondition: New post is created and visible in the user's feed.
User Story: As a user, I want to create posts so that I can share my photos with others.

USE CASE #6:

Name: Like Post
Inputs: Post ID
Outputs: Updated like count
Basic Course:

1. User views a post.
2. User clicks the like button.
3. System updates the like count for the post.

Precondition: User is logged in and viewing a post.
Postcondition: The post's like count is increased by one.
User Story: As a user, I want to like posts so that I can show appreciation for content I enjoy.

USE CASE #7:

Name: Comment on Post
Inputs: Comment text
Outputs: New comment
Basic Course:

1. User views a post.
2. User writes and submits a comment.
3. System saves and displays the comment on the post.

   Precondition: User is logged in and viewing a post.
   Postcondition: The comment is added to the post and visible to others.
   User Story: As a user, I want to comment on posts so that I can engage in discussions about the content.

USE CASE #8:

Name: Follow User
Inputs: User ID to follow
Outputs: Updated follow status
Basic Course:

1. User views another user's profile.
2. User clicks the follow button.
3. System updates the follow status and notifies the followed user.

   Precondition: User is logged in and viewing another user's profile.
   Postcondition: The user is now following the selected user.
   User Story: As a user, I want to follow other users so that I can see their posts in my feed.

USE CASE #9:

Name: Unfollow User
Inputs: User ID to unfollow
Outputs: Updated follow status
Basic Course:

1. User views a profile they are following.
2. User clicks the unfollow button.
3. System updates the follow status.

   Precondition: User is logged in and following the selected user.
   Postcondition: The user is no longer following the selected user.
   User Story: As a user, I want to unfollow users when I no longer wish to see their posts in my feed.

USE CASE #10:

Name: View Feed
Inputs: None
Outputs: List of posts from followed users
Basic Course:

1. User navigates to the feed page.
2. System retrieves and displays posts from users the current user is following.
   Precondition: User is logged in.
   Postcondition: User sees a feed of posts from followed users.
   User Story: As a user, I want to view my feed so that I can see updates from users I follow.

USE CASE #11:

Name: View Followers
Inputs: User ID
Outputs: List of followers
Basic Course:

1. User navigates to the followers list page.

2. System retrieves and displays the list of followers.

   Precondition: User profile exists.
   Postcondition: User sees a list of their followers.
   User Story: As a user, I want to see who follows me so that I know who is interested in my posts.

USE CASE #12:

Name: View Following
Inputs: User ID
Outputs: List of followed users
Basic Course:

1. User navigates to the following list page.
2. System retrieves and displays the list of users the current user is following.
   Precondition: User profile exists.
   Postcondition: User sees a list of users they are following.
   User Story: As a user, I want to see who I am following so that I can manage my connections.
2. **Sequence Diagram**
   All the sequence diagrams of this app are simple because Firebase internally does everything. But the specialty is authentication sequence diagrams.

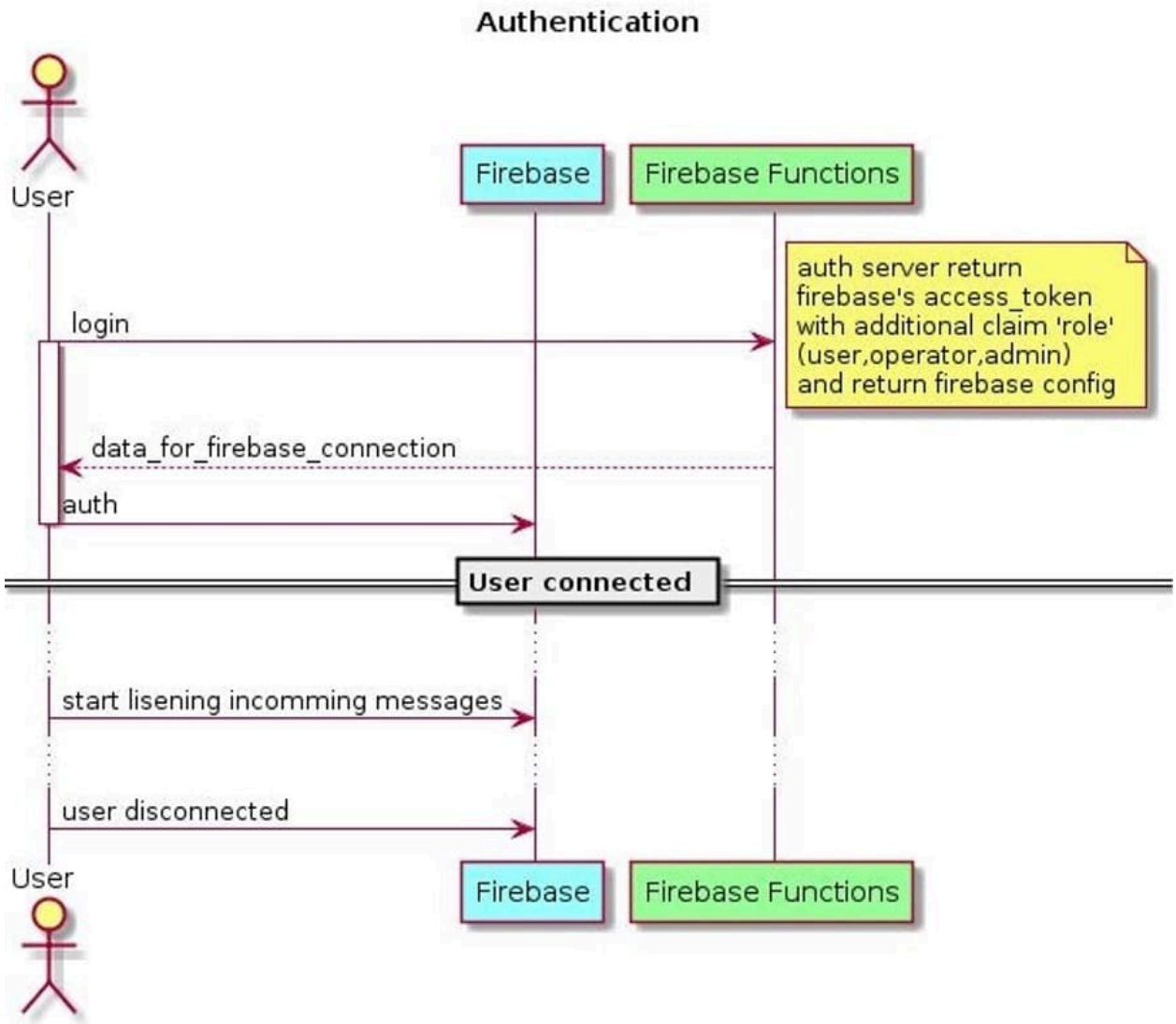


Figure II.2. Authentication Sequence Diagram

3. Architecture

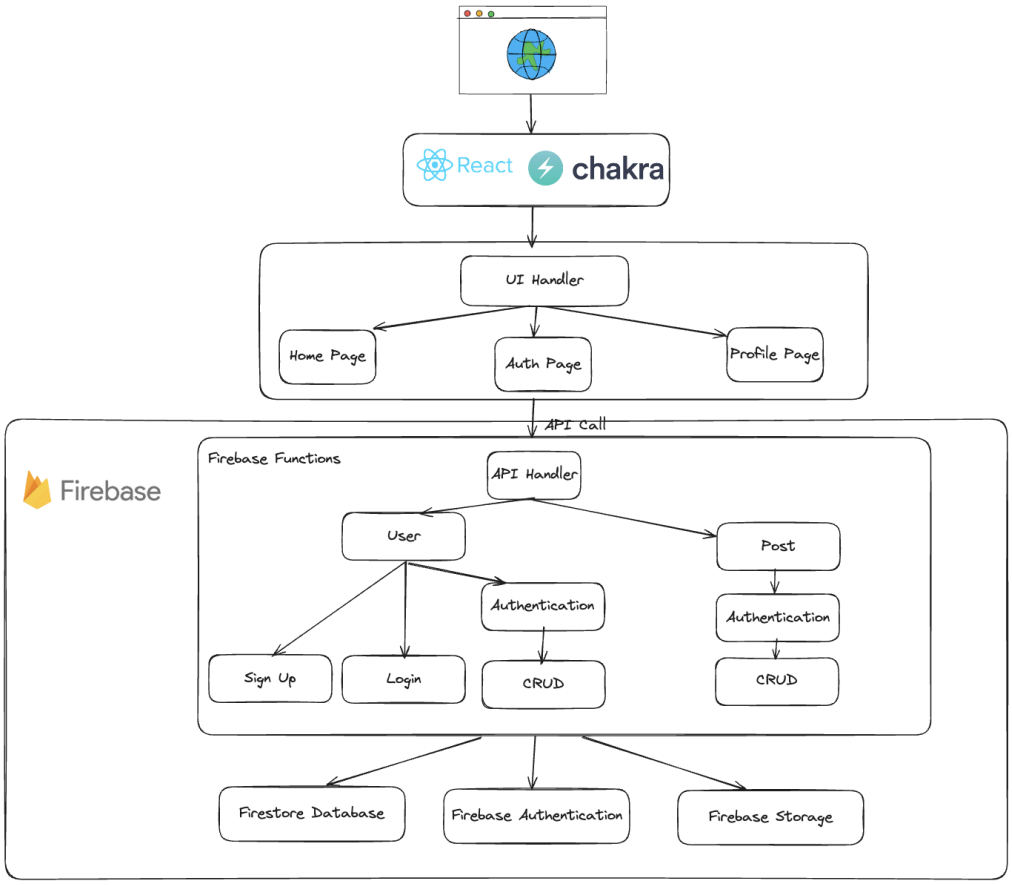


Figure II.3. Architecture of Mini Instagram

High-Level Architecture

The high-level architecture of the Instagram clone is divided into two main parts: the front-end (React) and the back-end (Firebase). The interactions between these parts are facilitated through API calls, enabling seamless data flow and real-time updates.

User Interface

- Users interact with the application through a web browser.
- React components, styled with Chakra UI, handle the presentation and user interactions.

UI Handler

- Manages the routing and rendering of different pages:
  - **Home Page:** Displays user posts and allows interactions such as liking and commenting.
  - **Auth Page:** Manages user login and registration.
  - **Profile Page:** Displays and allows editing of user profile information.

API Calls

- The front-end makes API calls to Firebase for authentication, data retrieval, and storage.

Firebase Functions

- **API Handler:** Routes API requests to the appropriate Firebase services.
- **User Module:** Handles user-related operations such as sign-up, login, and profile management.
- **Post Module:** Manages operations related to posts, including creation, retrieval, updating, and deletion.

Firebase Services

- **Firestore Database:** Stores user data, posts, and comments, supporting real-time data synchronization.
- **Firebase Authentication:** Manages user authentication and secure access to the app.
- **Firebase Storage:** Stores user-uploaded images and links them to posts in the Firestore Database.

4. Entity Relationship Diagram (ERD)

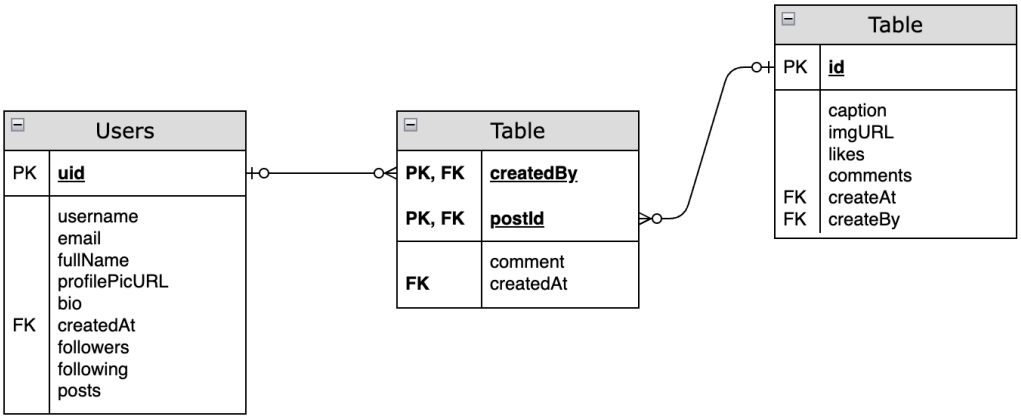


Figure III.4. ERD of Mini Instagram

5. Class Diagram

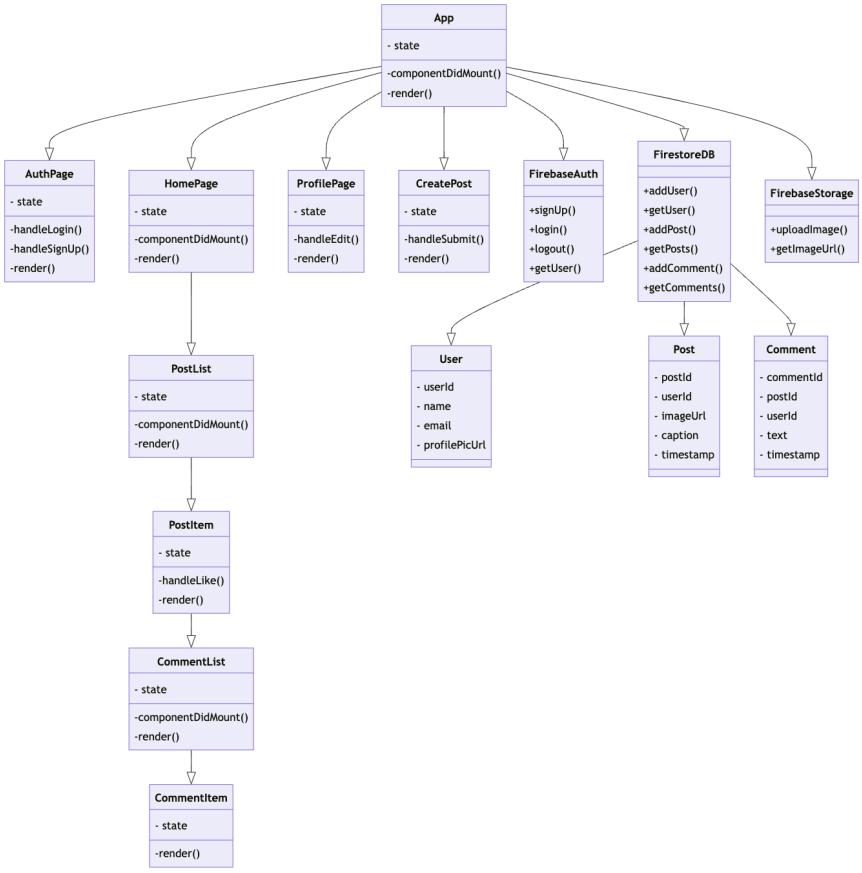


Figure III.5. Class diagram of Mini Instagram

React Components:

- **App:** The root component that manages global state and routing.
- **AuthPage:** Manages user authentication (login, registration).
- **HomePage:** Displays the main feed with posts.
- **ProfilePage:** Allows users to view and edit their profile.
- **CreatePost:** Manages the creation of new posts.
- **PostList:** Displays a list of posts.
- **PostItem:** Represents a single post, including likes and comments.
- **CommentList:** Displays a list of comments for a post.
- **CommentItem:** Represents a single comment.

Firebase Services:

- **FirebaseAuth:** Manages user authentication processes.
- **FirestoreDB:** Handles CRUD operations for users, posts, and comments in the Firestore database.
- **FirebaseStorage:** Manages the upload and retrieval of images.

Data Models:

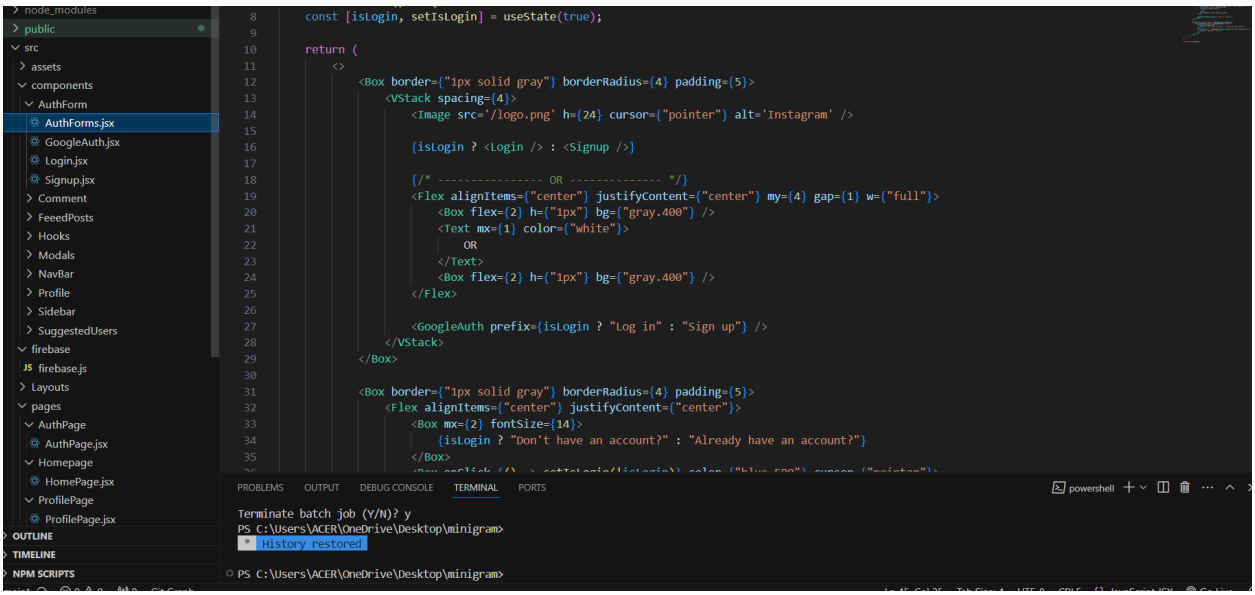
- **User:** Represents user information, including userId, name, email, and profilePicUrl.
- **Post:** Represents a post, including postId, userId, imageUrl, caption, and timestamp.
- **Comment:** Represents a comment, including commentId, postId, userId, text, and timestamp.

IV. Development

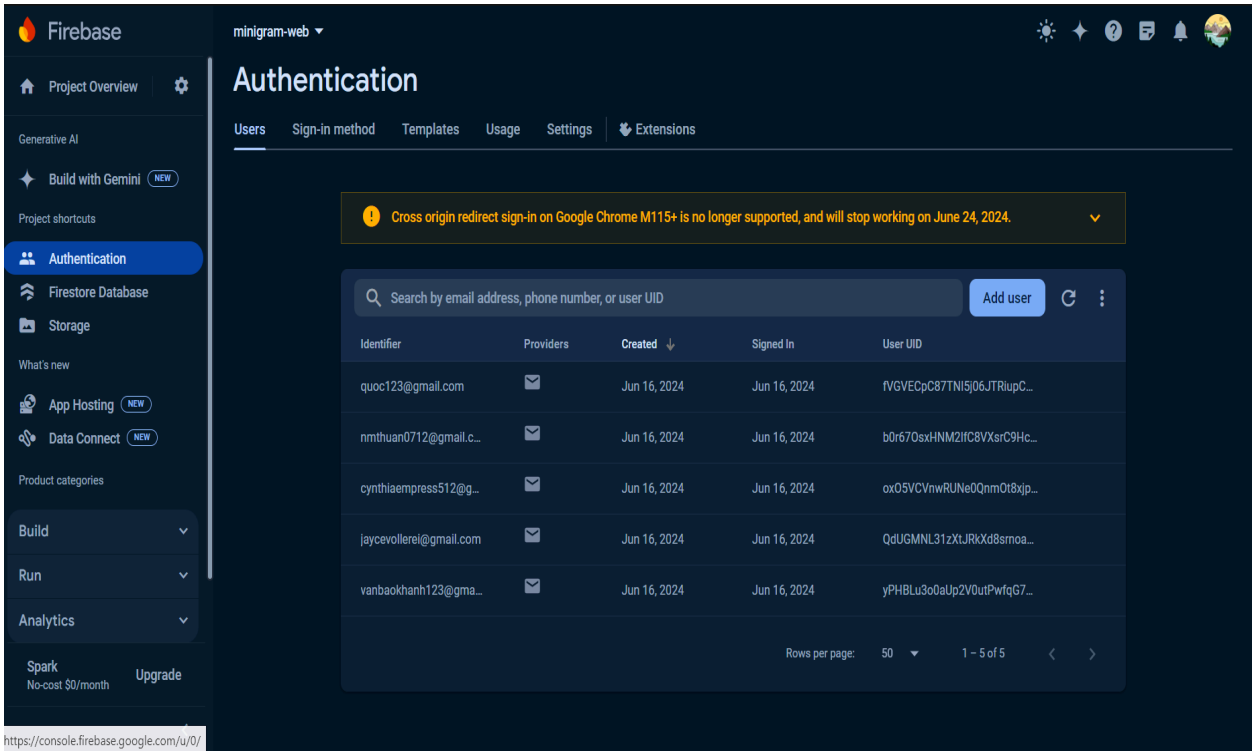
Key Features and Implementation

1. **User Authentication**
    - Implemented using Firebase Authentication, supporting email/password and social logins.

- React components handle the UI for login and registration forms, while Firebase handles the backend authentication logic.

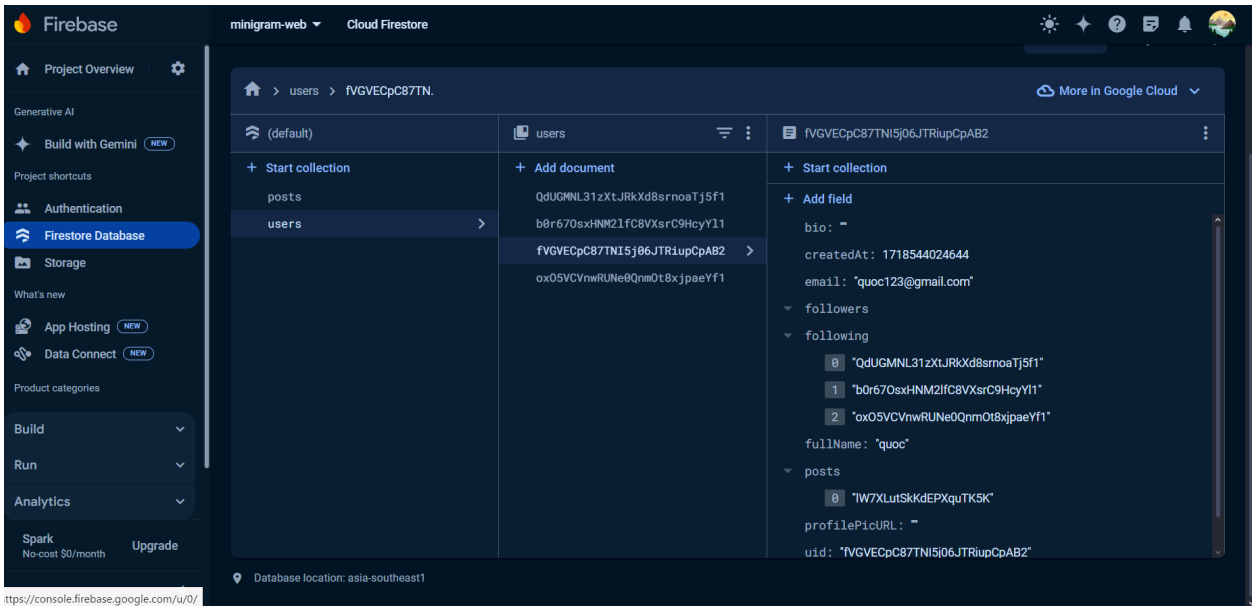


2. **User Profiles**
   - Users can create and edit their profiles, including profile pictures and bio information.
   - Profile data is stored in Firestore and retrieved as needed.



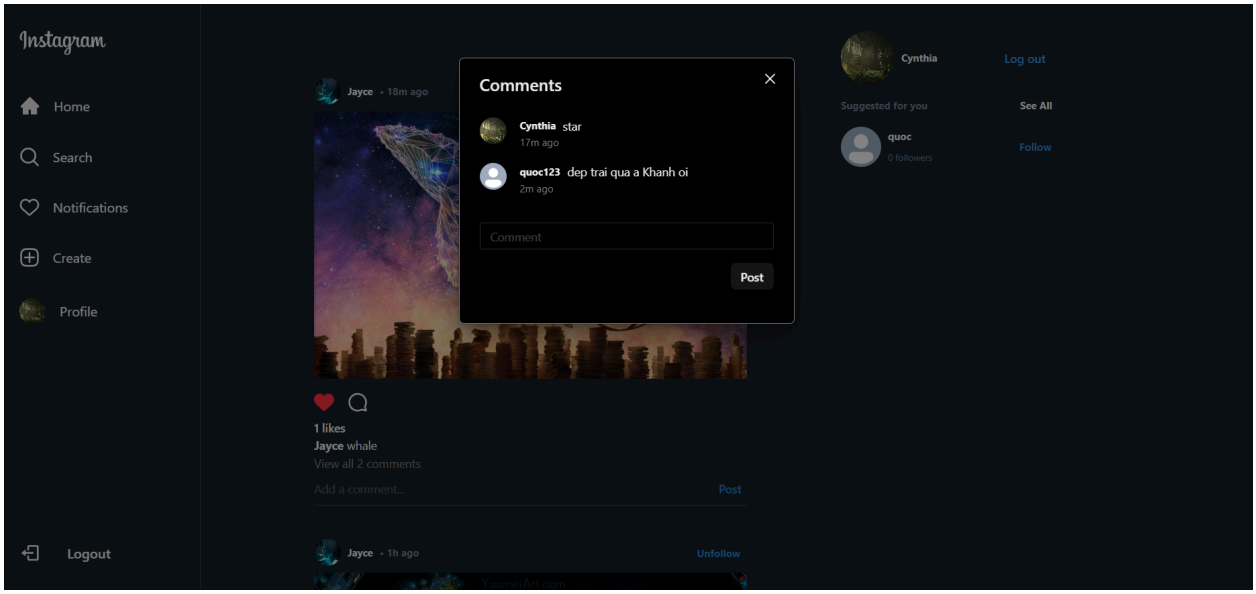
User with database

3. **Post Creation and Display**
    - Users can create posts with text and images.
    - Images are uploaded to Firebase Storage, and post metadata (e.g., image URL, caption, timestamp) is stored in Firestore.
    - Posts are displayed on the Home Page, with real-time updates reflecting new posts and interactions.

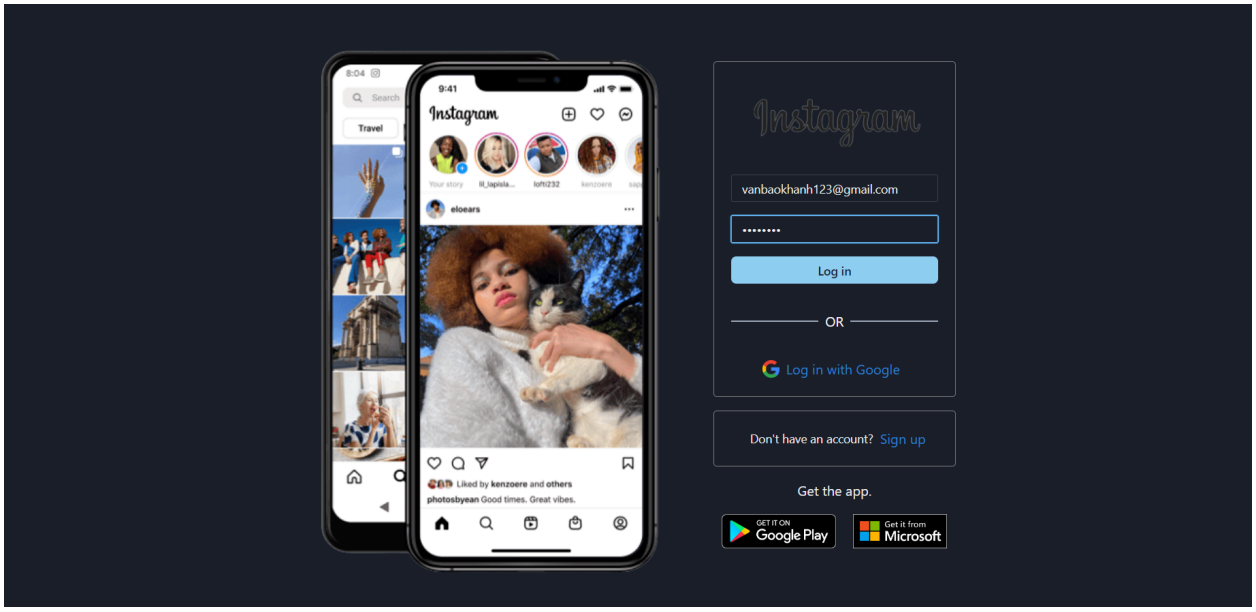


Database with firebase

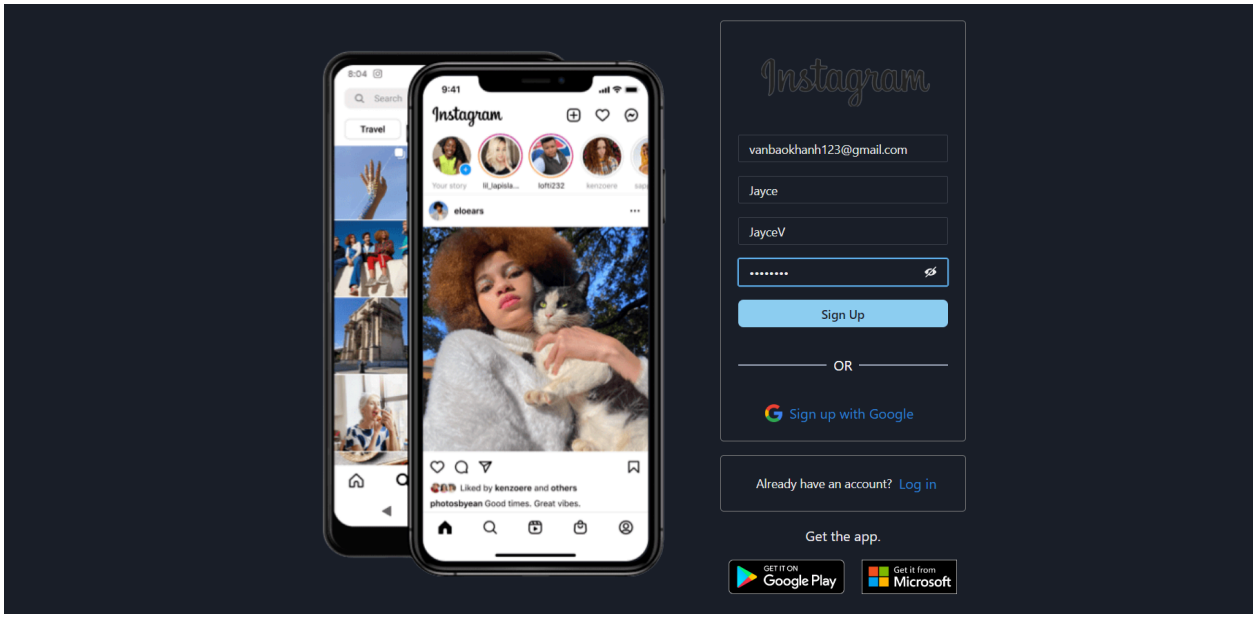
4. **Real-Time Interactions**
    - Users can like and comment on posts.
    - Firestore listeners provide real-time updates, ensuring that the UI reflects the latest data without requiring manual refreshes.



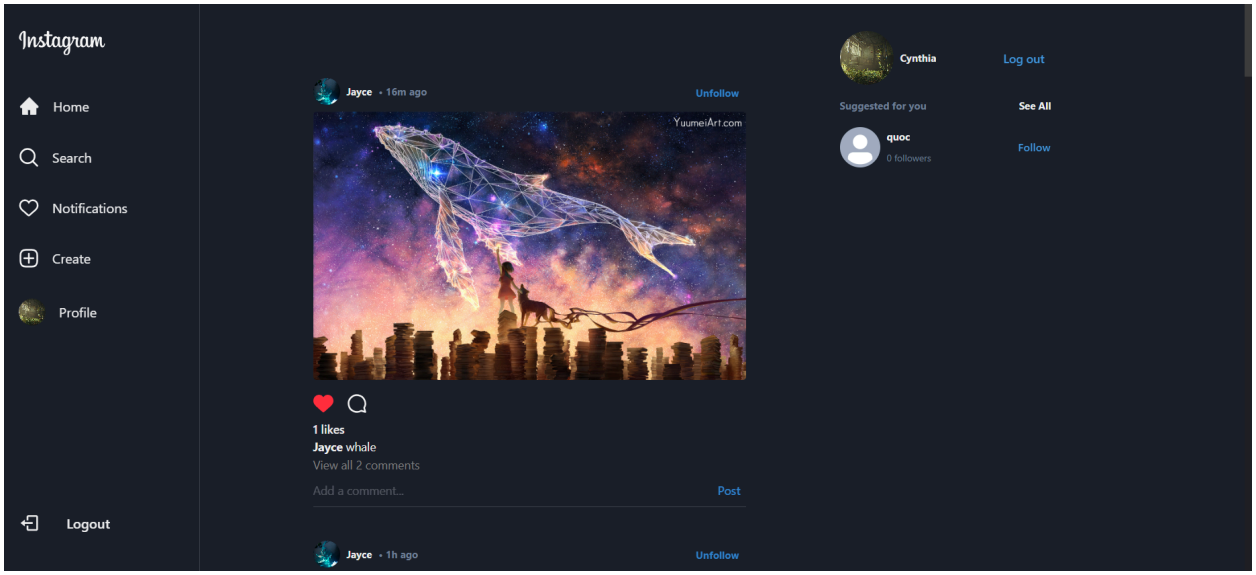
V. Testing



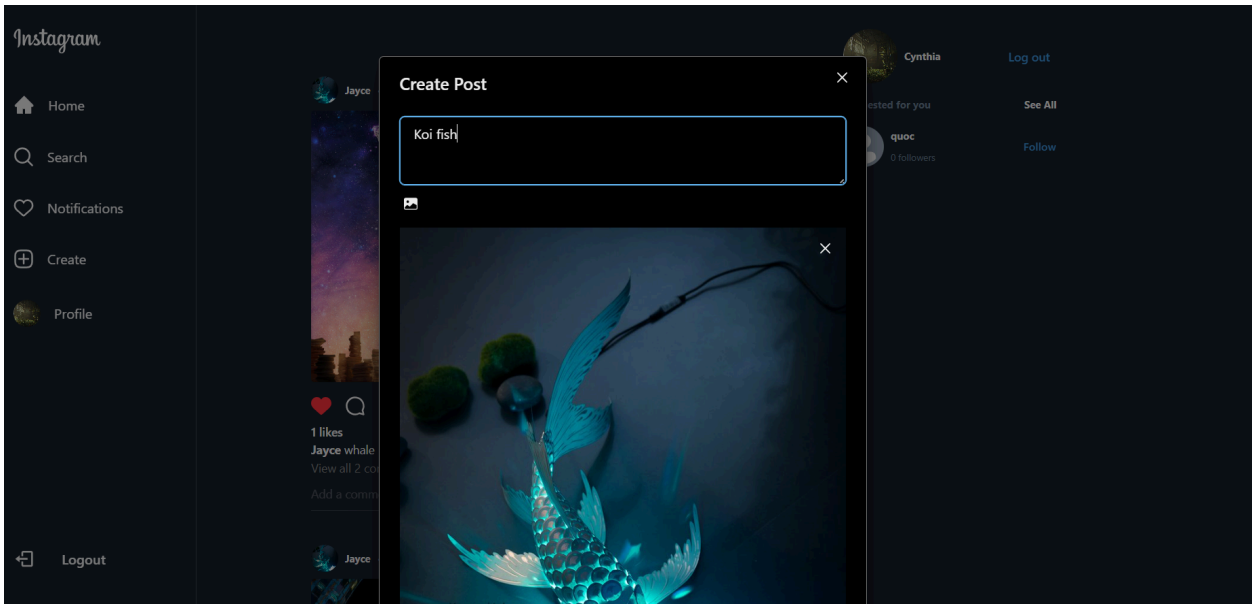
Login Page



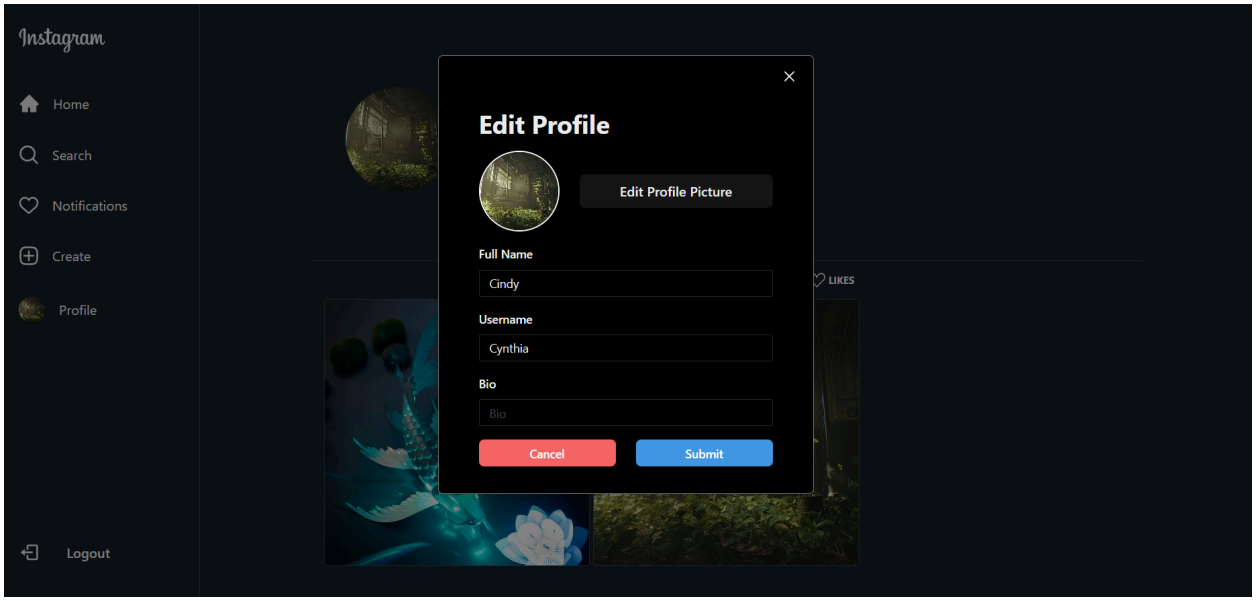
Sign Up Form



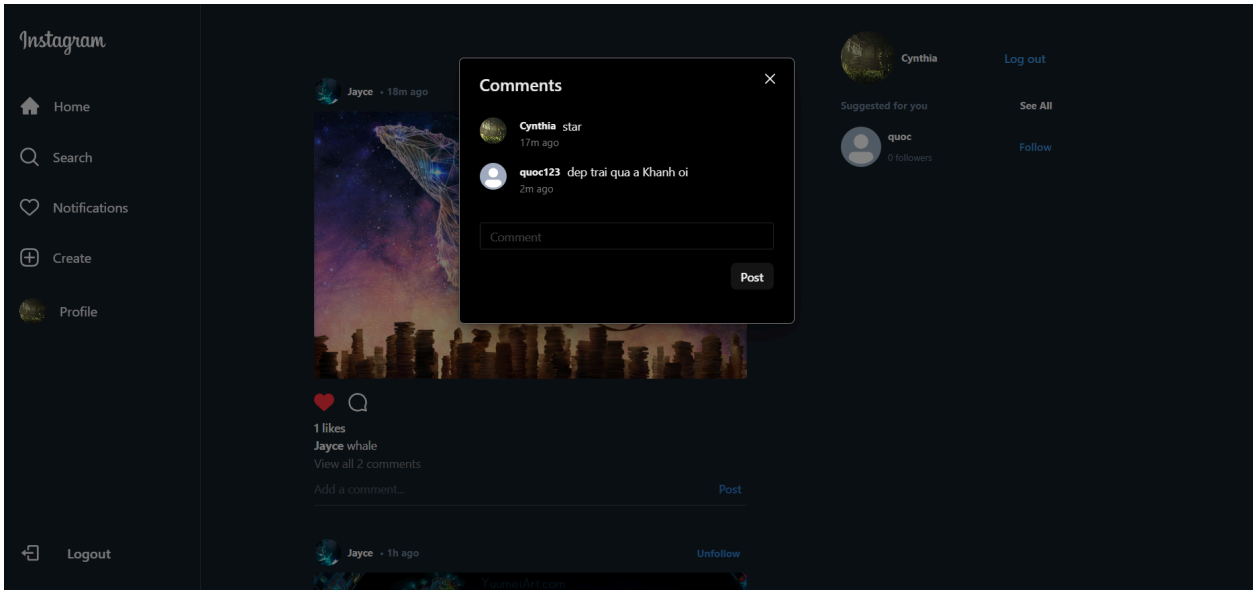
Homepage



Create Post

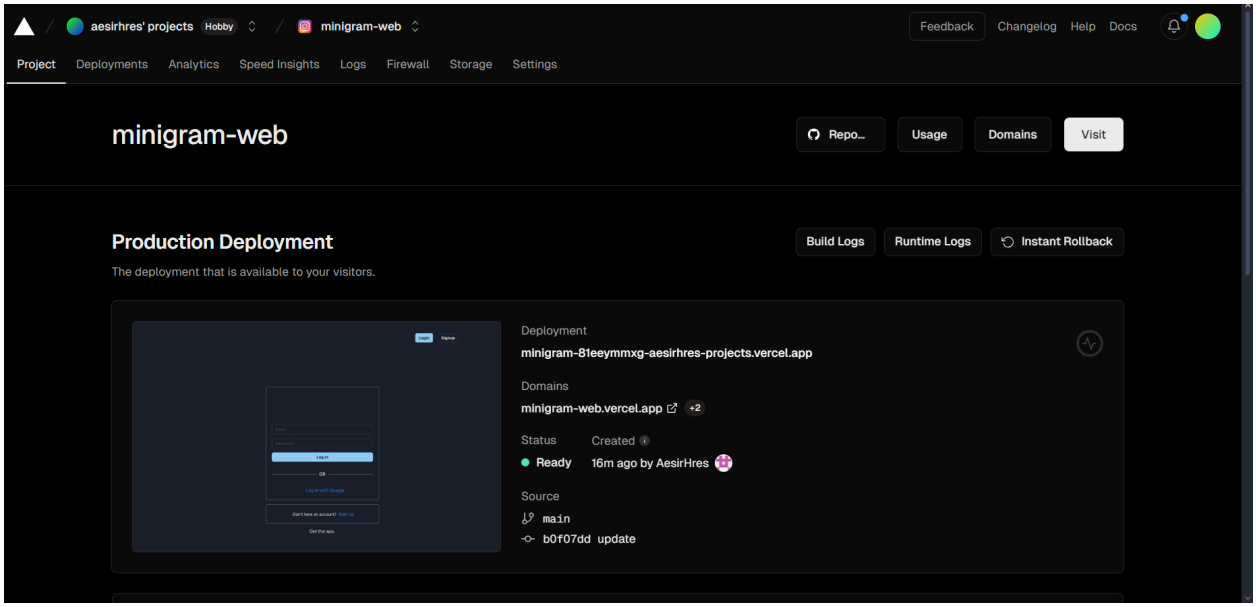


Edit Profile



Likes & Comment

VI. Deployment



Deployment with Vercel

VI. Future Work

Direct Messaging

Implement a direct messaging feature to allow users to communicate privately.

Notifications

Add real-time notifications for likes, comments, and new followers.

Stories Feature

Implement a feature similar to Instagram Stories, allowing users to post temporary content.

VII. Conclusion

The Instagram clone project successfully demonstrates the integration of React and Firebase to build a modern, scalable, and real-time web application. The architecture effectively separates concerns, ensuring a clean and maintainable codebase while leveraging the strengths of both technologies to deliver a seamless user experience. The project lays a solid foundation for further enhancements and can serve as a reference for similar full-stack development projects.

References

- https://forum.freecodecamp.org
- https://vercel.com/docs
- https://v2.chakra-ui.com/getting-started/vite-guide
- https://firebase.google.com/docs?hl=en&authuser=0&_gl=1*1skubjd*_ga*MTk1Mjc5NzE1MC4xNzE3MjA1MzEw*_ga_CW55HF8NVT*MTcxODU0MjgxOS4xMi4xLjE3MTg1NDU2MjEuNTkuMC4w
- https://legacy.reactjs.org/docs/getting-started.html