



DevOps Project





INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)
Dundigal, Hyderabad – 500 043

Certificate

This is to certify that it is a bonafied record of practical work done by Mr. / Ms.

_____ bearing the roll no. _____

of _____ class _____ branch in

the _____ laboratory during the academic

year _____ under our supervision.

|

Head of the department

Lecturer – in charge

Signature of External Examiner

Signature of Internal Examiner

Snapgram: Building a Scalable Social Media App with React and Appwrite

INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

Dundigal, Hyderabad – 500 043, Telangana



**Department of
CSE(AI&ML)**

**Bachelor of Technology
in
CSE(AI&ML)**

-BY

D.Sonic_paul

(23951A66M4)

DECLARATION

I hereby declare that this report on the Snapgram project represents my original work, based on the development of a modern social media application. The project was designed and implemented by me, leveraging various technologies and tools as described. All information provided is accurate to the best of my knowledge, and any external resources or inspirations (e.g., React Query documentation for debouncing) have been appropriately utilized. This report is submitted to fulfill the requirement for documenting the project.

Place: Hyderabad

Date:03-07-25

Signature of the Student

Roll No: 23951A66M4

ABSTRACT

The Snapgram project is a modern social media application designed to deliver a stunning user interface, native mobile-like experience, and high performance. Unlike typical Instagram clones, Snapgram incorporates advanced features such as infinite scroll, robust authentication, and seamless file management, addressing challenges like complex server setup, security, and API optimization. Built using React, TypeScript, Tailwind CSS, Shadcn UI, React Query (TanStack Query), and Appwrite Cloud, the application offers a scalable backend and responsive frontend. Key features include user authentication, post creation/editing, liking/saving posts, profile management, and an Explore page with debounced search and infinite scroll. The project was successfully deployed to Vercel, overcoming CORS issues with Appwrite integration. This report details the project's introduction, methodology, results, and conclusions, highlighting its technical achievements and user-focused design.

CONTENTS

Chapter 1	Introduction	1
	1.1 About Snapgram	3
	1.2 Requirements	4
	1.3 Prerequisites	5
Chapter 2	Methodology	7
Chapter 3	Results and Discussions	10
Chapter 4	Conclusion	16

CHAPTER 1

INTRODUCTION

1.1 About Snapgram

Snapgram is a modern social media platform designed to provide a seamless, engaging, and high-performance user experience, distinguishing itself from typical Instagram clones. The application features a stunning UI with a native mobile feel, achieved through a carefully selected tech stack and innovative features like infinite scroll. Snapgram addresses common development challenges, including complex server setups, security concerns, user authentication, file storage, and API performance optimization. By leveraging Appwrite Cloud, the project delivers a hassle-free, cost-effective backend compared to alternatives like Firebase, enabling rapid development and scalability. Key features include:

- **Robust Authentication:** Supports email/password, magic URLs, and over 30 social sign-in options.
- **Home Page:** Displays posts with a “Top Creators” section for discoverability.
- **Explore Page:** Features infinite scroll and debounced search for efficient content discovery.
- **Post Interactions:** Allows liking, saving, creating, editing, and deleting posts.
- **Profile Management:** Showcases user posts, liked posts, and editable profiles.
- **Responsive UI:** Includes a mobile-friendly bottom bar for a native app experience.

The project emphasizes performance, scalability, and user engagement, making it a standout social media application.

1.2 Requirements

The Snapgram project was developed to meet the following requirements:

- **User Experience:** Deliver a visually appealing, responsive UI with a native mobile feel across web and mobile devices.
- **Performance:** Optimize API calls and data fetching for seamless interactions, including infinite scroll and real-time search.
- **Security:** Implement robust authentication and permission-based access to ensure data privacy and security.
- **Scalability:** Use a serverless backend to handle growing user bases and content without complex server management.
- **File Management:** Provide secure, effortless file storage and retrieval for post images.
- **Cost-Effectiveness:** Utilize a backend solution (Appwrite Cloud) that offers advanced features at a lower cost than competitors like Firebase.
- **Feature Set:** Include core social media functionalities (post creation/editing, liking/saving, profile management) and advanced features (infinite scroll, debounced search).
- **Deployment:** Ensure the application is production-ready with a streamlined deployment process.

1.3 Prerequisites

To develop and run Snapgram, the following prerequisites were necessary:

Development Environment:

- Visual Studio Code (VS Code) for coding and terminal access.
- Node.js and npm for managing dependencies and running the Vite development server.

Frontend Technologies:

- React for building the UI.
- TypeScript for type safety.
- Vite for fast project setup and development.
- Tailwind CSS and Tailwind Animate for styling and animations.
- Shadcn UI for customizable UI components.
- React Router DOM for navigation.
- React Query (TanStack Query v5.0) for data fetching, caching, and mutations.
- React Dropzone for drag-and-drop file uploads.

Backend Technologies:

- Appwrite Cloud account for backend services (authentication, databases, storage).
- Appwrite SDK for integration with the frontend.

Deployment:

- GitHub account for version control and repository hosting.
- Vercel account for deploying the frontend application.

Knowledge:

- Familiarity with React, TypeScript, and modern JavaScript.
- Understanding of RESTful APIs, database relationships, and CORS policies.
- Experience with state management (React Context) and data fetching (React Query).

CHAPTER 2

METHODOLOGY

The development of Snapgram followed a structured approach, broken down into key phases, as described below in the order of implementation:

1. Appwrite Cloud Setup

- Created an account on Appwrite Cloud and initialized a project.
- Configured authentication (email/password, social sign-ins), databases (with collections for posts, users, saves), and storage (media bucket for images).
- Set up environment variables in .env for project, database, and collection IDs.

2. Project Initialization

- Created a new project folder and opened it in VS Code.
- Used Vite to set up a React project with TypeScript (npm create vite).
- Ran npm run dev to start the development server on localhost:5173.
- Set up main.tsx and App.tsx as the root files for rendering and routing.

3. Frontend Setup

- Installed and configured Tailwind CSS and Tailwind Animate for styling.
- Set up React Router DOM for public (Sign In, Sign Up) and private (Home, Explore, etc.) routes.
- Created folder structure: _auth for authentication pages and _root for protected pages.
- Developed AuthLayout.tsx, SignInForm.tsx, SignUpForm.tsx, RootLayout.tsx, and HomePage.tsx.

4. Authentication Implementation

- Built Sign-Up and Sign-In forms using Shadcn UI, React Hook Form, and Zod for validation.
- Integrated Appwrite authentication in src/lib/appwrite/config.ts and api.ts.
- Created createUserAccount and saveUserToDB functions to register users and store their data.
- Used React Context (AuthContext.tsx) to manage authentication state with useState and useEffect.

- Implemented sign-out functionality with navigation to Sign-In/Sign-Up pages.

5. Database and Storage Configuration

- Created collections (Posts, Users, Saves) with relationships:
- Posts: Many-to-one with Users (creator), many-to-many (likes).
- Saves: Many-to-one with Users and Posts.
- Defined attributes (e.g., caption, imageUrl, tags) and indexes for efficient searching.
- Configured storage for image uploads in a media bucket.

6. Navigation Components

- Developed TopBar, LeftSidebar, and BottomBar for navigation:
- TopBar: Included logo, user info, and sign-out button.
- LeftSidebar: Featured profile link and navigation links with hover effects.
- BottomBar: Mobile-only navigation mimicking native apps.

7. Post Creation and Editing

- Built CreatePost.tsx and EditPost.tsx with a reusable PostForm.tsx component.
- Implemented FileUploader.tsx with react-dropzone for drag-and-drop image uploads.
- Created createPost, updatePost, and deletePost functions in api.ts for CRUD operations.
- Used React Query mutations (useCreatePost, useUpdatePost, useDeletePost) for seamless data updates.

8. Home and PostDetails Pages

- Developed HomePage.tsx to display recent posts using useGetRecentPosts.
- Created PostCard.tsx and PostStats.tsx for post rendering and interactions (likes, saves).
- Built PostDetails.tsx to show post details, edit/delete buttons (creator-only), and PostStats.

9. Explore Page with Infinite Scroll and Search

- Developed Explore.tsx with a search bar (useState for searchValue) and infinite scroll.
- Implemented getInfinitePosts and searchPosts in api.ts for fetching posts.
- Used useInfiniteQuery (useGetPosts) for infinite scroll and useQuery (useSearchPosts) for search.

- Created useDebounce hook to optimize search by delaying API calls, reducing server load.
- Rendered SearchResults.tsx or GridPostList.tsx based on search input.

10. Deployment

- Deployed to Vercel by importing the GitHub repository and adding environment variables.
- Resolved CORS issues by adding *.vercel.app as a trusted host in Appwrite's platform settings.

CHAPTER 3

RESULTS AND DISCUSSION

The Snapgram project achieved its objectives, delivering a fully functional social media application with the following results:

User Experience:

- Achieved a stunning, responsive UI with a native mobile feel using Tailwind CSS, Shadcn UI, and a mobile-friendly BottomBar.
- Implemented intuitive navigation with TopBar, LeftSidebar, and BottomBar, ensuring seamless access across devices.
- Enhanced content discovery with the Explore page's infinite scroll and debounced search, improving user engagement.

Performance:

- Optimized data fetching with React Query's auto-caching, refetching, and mutation handling, reducing API latency.
- Used debouncing to limit search API calls, minimizing server strain (e.g., delaying calls by 500ms).
- Achieved smooth infinite scroll using useInfiniteQuery, fetching posts in chunks with cursorAfter.

Security and Scalability:

- Leveraged Appwrite's permission-based authentication and collection-level access controls to ensure secure user data.
- Used Appwrite's serverless backend for scalability, handling growing user bases without manual server management.
- Validated form inputs with Zod and handled edge cases (e.g., deleting corrupted files) to enhance reliability.

Feature Implementation:

- Successfully implemented core features: authentication, post creation/editing/deletion, liking/saving, and profile management.

- Added advanced features like infinite scroll, drag-and-drop file uploads, and real-time search, setting Snapgram apart from typical social media apps.
- Ensured creator-only access for edit/delete actions using `useUserContext`, enhancing security.

Deployment:

- Deployed the application to Vercel, achieving a production-ready state.
- Resolved CORS issues by configuring Appwrite, demonstrating problem-solving skills.

Discussions:

Challenges Overcome:

- Simplified complex server setup and security concerns using Appwrite Cloud, saving development time compared to alternatives like Firebase.
- Optimized API performance with React Query and debouncing, addressing potential server overload from frequent search queries.
- Handled CORS issues during deployment, ensuring seamless integration between Vercel and Appwrite.

Lessons Learned:

- Mastered advanced React Query features (`useInfiniteQuery`, mutations) for efficient data management.
- Gained expertise in modular component design by reusing `PostForm` for both creation and editing.
- Learned the importance of debouncing for optimizing real-time features like search.

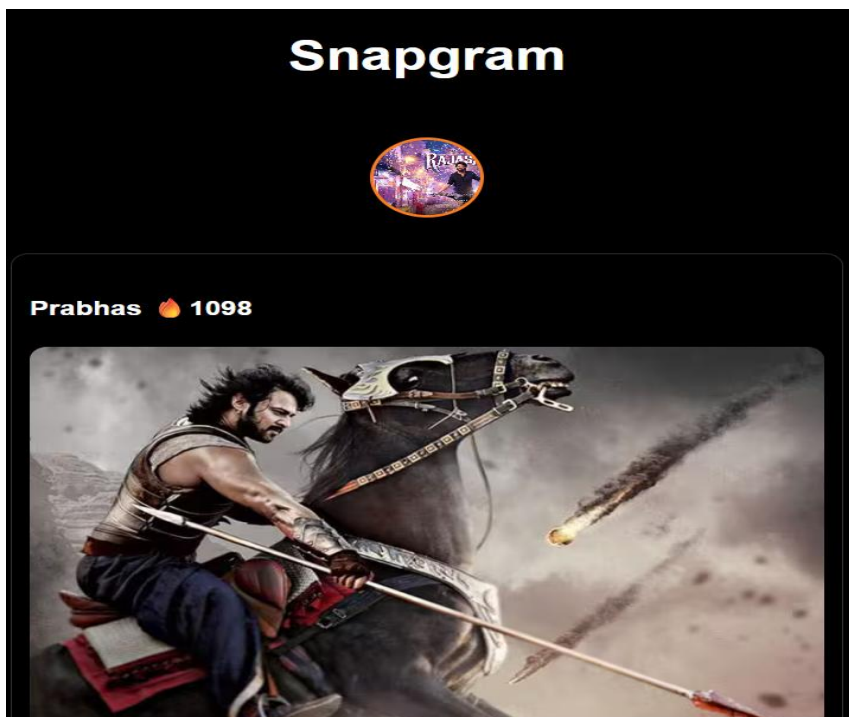
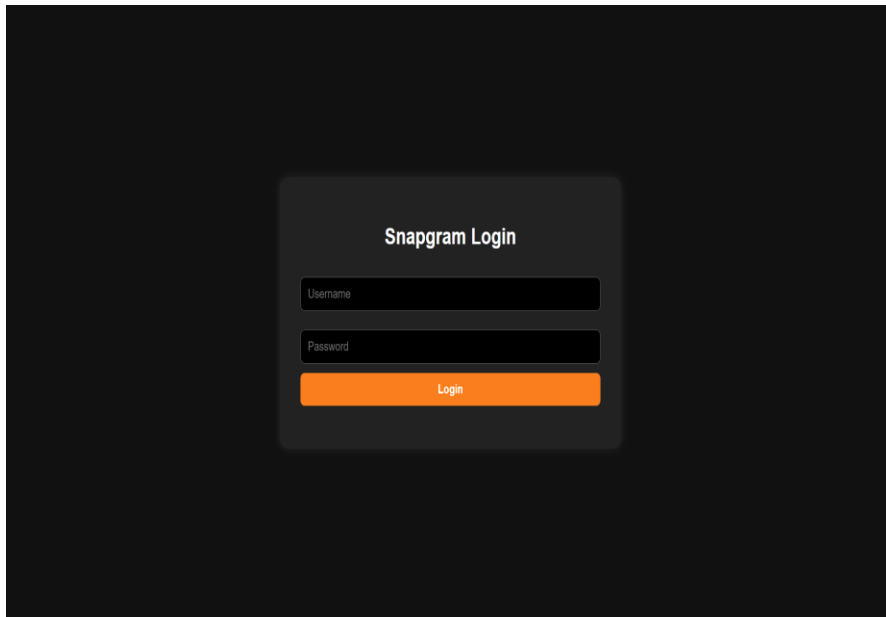
Limitations:

- The Top Creators feature (mentioned in Part 1) was not implemented in the provided scope, potentially limiting discoverability of influential users.
- Advanced filtering for the Explore page (e.g., beyond “All”) was planned but not detailed.

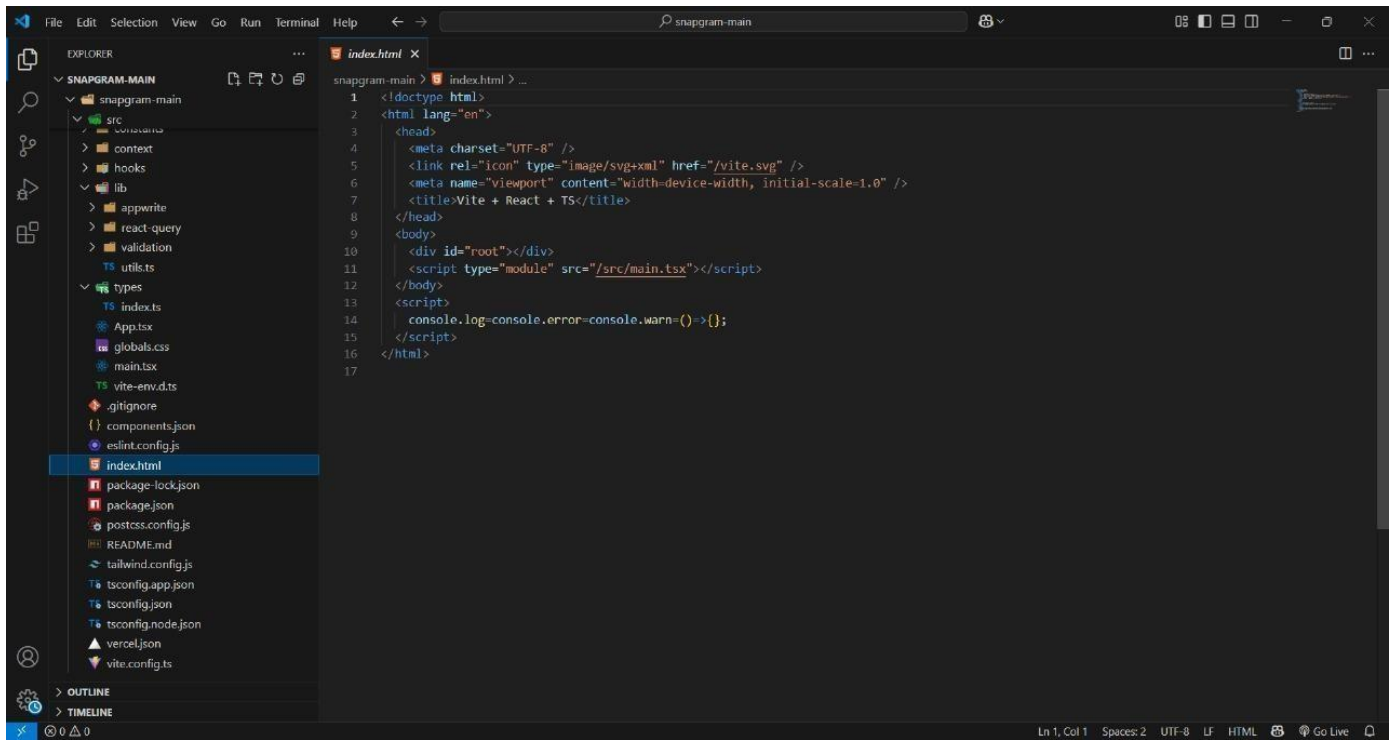
Future Improvements:

- Implement the Top Creators page to highlight influential users.
- Add advanced filters (e.g., by tags or location) to the Explore page.

Execution:



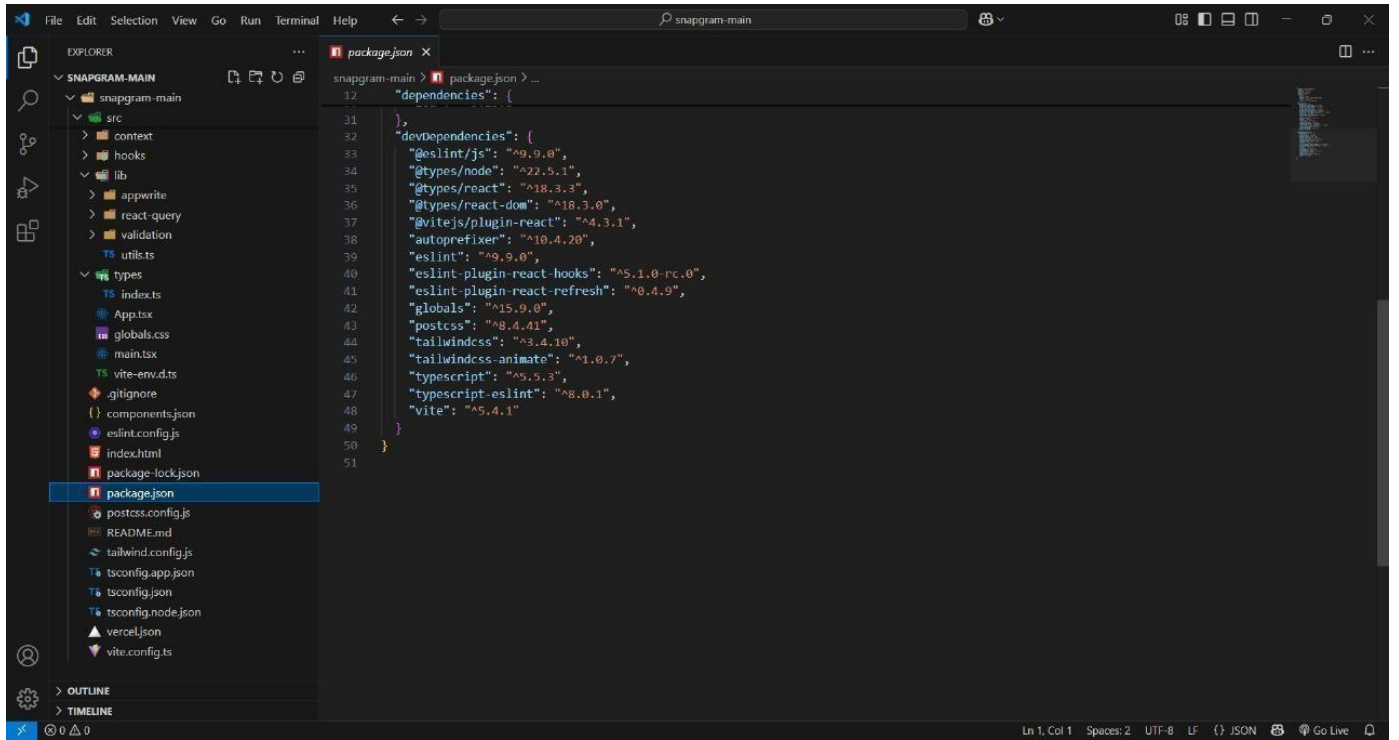
Source Code:

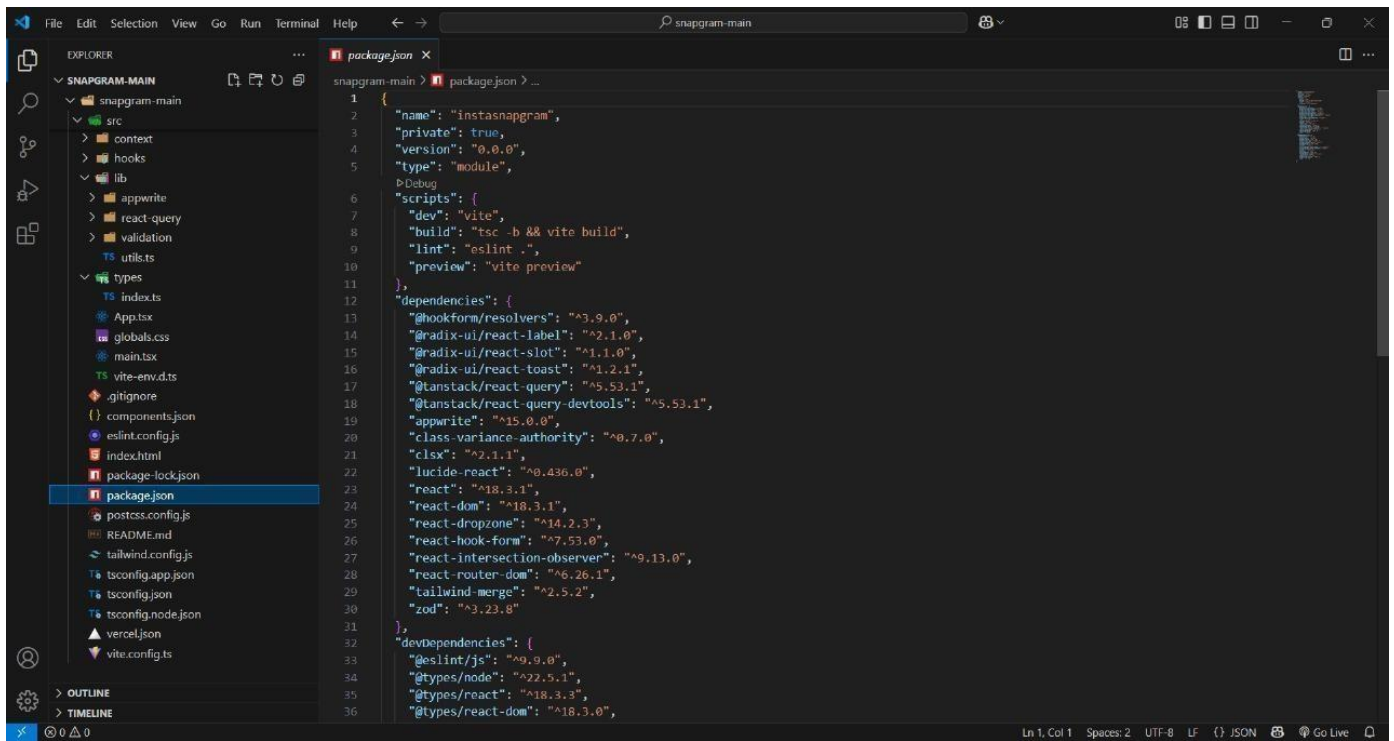


The screenshot shows a VS Code editor window with the file explorer on the left and the editor on the right. The file explorer shows the project structure for 'snapgram-main', with 'index.html' selected. The editor displays the following HTML code:

```
1 <!doctype html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <link rel="icon" type="image/svg+xml" href="/vite.svg" />
6     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7     <title>Vite + React + TS</title>
8   </head>
9   <body>
10    <div id="root"></div>
11    <script type="module" src="/src/main.tsx"></script>
12  </body>
13  <script>
14    console.log=console.error=console.warn={()=>{}};
15  </script>
16 </html>
17
```

The status bar at the bottom indicates the current position is Line 1, Column 1, with 2 spaces, in UTF-8 encoding, using the HTML language.





CHAPTER 4

CONCLUSION

The Snapgram project successfully delivered a modern, high-performance social media application that meets the requirements of a responsive UI, robust functionality, and scalable backend. By leveraging a tech stack of React, TypeScript, Tailwind CSS, Shaden UI, React Query, and Appwrite Cloud, the project addressed key challenges like server complexity, security, and API optimization

Standout features include infinite scroll, debounced search, and drag-and-drop file uploads, which enhance user engagement and distinguish Snapgram from typical social media clones. The application was deployed to Vercel, with CORS issues resolved to ensure production readiness. The project demonstrates proficiency in frontend and backend development, state management, and performance optimization, while offering opportunities for future enhancements like the Top Creators page and advanced filtering. Snapgram stands as a testament to innovative design and efficient development practices, ready to scale with user growth.