DevOverFlow - An Advanced Q&A Platform with AI Integration

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

in

Information Technology

by

ESHAN GUPTA

20BIT0262

Under the guidance of

Dr. Suganya P.

School of Computer Science Engineering and Information Systems (SCORE)

VIT, Vellore

May,2024

DECLARATION

I hereby declare that the thesis entitled "**DevOverFlow - An Advanced Q&A Platform with AI Integration**" submitted by me, for the award of the degree of Bachelor of Technology in Information Technology to VIT is a record of bonafide work carried out by me under the supervision of Dr. Suganya P.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Date: 08.05.2024

Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled "**DevOverFlow - An Advanced Q&A Platform with AI Integration**" submitted by Eshan Gupta & 20BIT0262, School of Computer Science Engineering and Information Systems (SCORE), VIT, for the award of the degree of Bachelor of Technology in Information Technology, is a record of bonafide work carried out by him / her under my supervision during the period, 02. 01. 2024 to 20.04.2024, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfils the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place: Vellore

Date: 08.05.2024

Signature of the Guide

Internal Examiner **External Examiner**

Head of the Department

School of Computer Science Engineering and Information Systems (SCORE)

ACKNOWLEDGEMENTS

The opportunity VIT gave me to carry out my Capstone Project was an excellent learning and professional development opportunity. Thus, I am incredibly fortunate to have the chance to learn and grow and to utilize the skills I learnt during my degree to create this project.

I want to thank my professors, my guide Dr Suganya P. and my panel members Dr. Balaganesh N and Prof. Brindha K for providing excellent advice and assistance. I owe them a debt of gratitude and would like to express my appreciation for their efforts.

This is a significant step forward in my professional development for me. In order to achieve my professional goals, I will work hard to put my newly acquired skills and knowledge to the greatest possible use, and I will continue to enhance them.

Eshan Gupta

Executive Summary

DevOverFlow is an ambitious capstone project that aims to create a sophisticated Question and Answer (Q&A) platform, integrating the functionalities of StackOverflow with innovative features. The key distinguishing factor is the implementation of AI-powered text generation, facilitated by OpenAI technology, to provide users with instant and accurate answers.

The project leverages cutting-edge technologies, including NextJS version 13.5 as the core framework, MongoDB as the database, and Tailwind CSS for the presentation layer. TypeScript is the language of choice, ensuring robust and maintainable code, with strategic use of React. The architecture is grounded in NextJS fundamentals, encompassing client-server dynamics, runtime versus build time considerations, and versatile rendering strategies like SSR, ISR, SSG, and CSR.

The platform promotes user-friendly interactions by allowing Markdown input for questions and ensuring full responsiveness across diverse devices. Users can pose questions, receiving AI-generated answers alongside community responses. Active participation earns users' badges and reputation points, contributing to an incentivized and dynamic knowledge-sharing ecosystem. DevOverFlow aims to redefine Q&A platforms by combining AI innovation with a rich feature set for an unparalleled user experience.

The project's key objectives include implementing AI-powered text generation, creating a user-friendly interface, developing a comprehensive authentication system, designing a scalable backend architecture, and enhancing user experience through responsive design and interactive features. The proposed architecture consists of a frontend layer, backend infrastructure, AI-powered text generation, a recommendation system, authentication and access control, and responsive design with performance optimization.

Contents

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

1. Q&A - Question and Answer
2. NLP - Natural Language Processing
3. SSR - Server-Side Rendering
4. ISR - Incremental Static Regeneration
5. SSG - Static Site Generation
6. CSR - Client-Side Rendering
7. API - Application Programming Interface
8. JSON - JavaScript Object Notation
9. BSON - Binary JSON
10. RDBMS - Relational Database Management System
11. HTML - Hypertext Markup Language
12. SIASN - State Civil Apparatus Information System
13. FCP - First Contentful Paint
14. LCP - Largest Contentful Paint
15. TTI - Time to Interactive
16. TBT - Total Blocking Time
17. SEO - Search Engine Optimization

1. Introduction

DevOverFlow is a novel Question and Answer (Q&A) platform designed to rival established platforms like StackOverflow. It leverages cutting-edge technologies including NextJS 13.5, MongoDB, Tailwind CSS, and TypeScript to deliver an exceptional user experience. A key differentiator is the integration of OpenAI's AI-powered text generation, providing instant and accurate answers alongside community-driven responses. The platform boasts a comprehensive feature set including user authentication, robust search, a reputation system, and Markdown support for questions. DevOverFlow aspires to redefine Q&A platforms by fostering a dynamic knowledge-sharing environment through the power of AI and a rich feature set.

i. Objective

The primary objective of the DevOverFlow project is to address the shortcomings of existing Q&A platforms by leveraging advanced technologies to deliver instant and accurate answers to user queries. The project specifically focuses on integrating AI-powered text generation capabilities, provided by OpenAI technology, into a Q&A platform inspired by the functionalities of StackOverflow. This approach seeks to enhance the responsiveness and effectiveness of the platform, providing users with a more engaging and efficient knowledge-sharing experience.

1.2 Motivation

The document delves into the rich landscape of research and development efforts surrounding Q&A platforms, highlighting the drive to improve user engagement, content relevance, and system efficiency. The motivation behind DevOverFlow is rooted in these evolving needs, with a particular emphasis on exploring the integration of artificial intelligence (AI) and natural language processing (NLP) techniques to facilitate automatic response generation. Additionally, the project is inspired by the growing interest in AI-powered recommendation systems that suggest relevant questions, answers, and resources based on user preferences and browsing history.

1.3 Background

The background context for the DevOverFlow project underscores the significance of robust authentication systems, scalable architectures, and responsive designs to ensure seamless user interactions across diverse devices. The document also acknowledges the advancements in backend technologies, such as server-side rendering (SSR), incremental static regeneration (ISR), and client-side rendering (CSR), which have contributed to improved performance, scalability, and user experience. Furthermore, the project's development is influenced by the growing popularity of TypeScript and React in web development, as these technologies enable code maintainability, type safety, and component reusability.

Also, we can discuss about the papers that were used for getting an idea about the background of the different technologies used in the project-:

1.3.1 Literature Survey

"NextJS File-Based Routing - A Review by Krutika Patil, Published in International Journal of Trend in Scientific Research and Development" [1]

In this paper the author discusses that Next Js is quickly gaining popularity as a full-stack React Js framework and a React framework for Production. Next Js helps us build efficient websites with solid Search Engine Optimization. Next.js, developed by Vercel, has become a popular framework for developing React applications. It provides useful features such as server-side rendering, static site generation, and an intuitive file based routing system, revolutionizing how developers construct their applications. This paper delves into the intricacies of the file-based routing system in Next.js, discussing its principles, benefits, potential issues, and use cases bolstered by tangible coding examples.

Keywords: NextJs, file-based routing, full-stack, web development, react-router

“Improving Performance Of NextJS App And Testing It While Building A Badminton Based Web App by S. Sasikumar, S. Prabha and Chandra Mohan in The International Conference on Innovative Computing & Communication (ICICC) 2022” [2]

In this paper, the author delves into the intricate process of building complete web applications, highlighting the significant time and effort developers invest in amalgamating various technologies. The discussion centers on the pivotal role of frameworks like Next.js in simplifying this endeavor. Next.js adeptly consolidates packages and configuration files, offering developers a streamlined and organized workflow. Notably, it stands out as a full-stack web application framework, enabling developers to seamlessly integrate frontend and backend code within a singular environment. This characteristic not only facilitates ease for developers but also expedites product delivery. However, challenges arise with full-stack frameworks like Next.js, particularly in the compilation process during production builds. Identifying opportunities for enhancement, the author explores techniques and coding patterns gleaned from the development of a badminton data analytics-based web application using Next.js, aiming to elucidate methods for enhancing the performance of production builds.

Keywords: next.js, client-side rendering, server-side rendering, data analytics, routing, dynamic routing, Image Optimization, page pre-rendering, lazy loading

“Modern Front End Web Architectures with React. Js and Next. Js by Mohammad Fariz Syah Lazuardy and Dyah Anggraini in International Research Journal of Advanced Engineering and Science” [3]

In this study, the author delves into the evolution of web development technologies, particularly focusing on the distinction between front-end and back-end development. Traditionally, full-stack developers juggled responsibilities on both ends of web applications. However, contemporary practices have separated front-end and back-end development, allowing front-end developers to concentrate solely on user interface design and interaction.The advent of libraries like React.js has significantly

impacted front-end development, offering robust capabilities for client-side rendering. Nonetheless, integrating React concepts with server-side rendering necessitates the adoption of frameworks like Next.js. Next.js extends React's functionality, making it a preferred choice for building front-end applications, such as the State Civil Apparatus Information System (SIASN).

This research endeavors to dissect the merits and demerits of both React.js and Next.js within the context of SIASN web application development. By scrutinizing the technologies' strengths and weaknesses, the study sheds light on the procedural aspects of building SIASN applications from the front-end perspective. It offers insights into the advantages and disadvantages of employing React.js and Next.js as the primary technologies for crafting the user interface of SIASN applications.

Keywords: SSR, CSR, Data Fetching, React.js, Next.js.

“Application of TypeScript Language: A Brief Overview by Saptarshi Bhattacharyya and Asoke Nath in International Journal of Innovative Research in Computer and Communication Engineering 2007” [4]

In this paper, the authors delve into the functionality of the TypeScript Compiler, which interprets TypeScript programs and translates them into JavaScript. TypeScript, with its wide-ranging applications in Internet webpage design, is the focal point of this study. The paper offers a comprehensive examination of the scope, applications, and challenges associated with the TypeScript language. Its primary objective is to articulate the essence of TypeScript by providing a succinct definition of its type system across key language constructs.

JavaScript, while ubiquitous, is often regarded as inadequate for the development and maintenance of large-scale applications. TypeScript emerges as an extension of JavaScript, aiming to mitigate this limitation. The resulting JavaScript code from TypeScript is compatible with a diverse array of execution environments, allowing for immediate deployment. Notably, the TypeScript compiler is extensively utilized within Microsoft for crafting substantial JavaScript applications.

Keywords: TypeScript, JavaScript, Transpilation, Transpiler, CoffeeScript.

“Understanding TypeScript by Gavin Bierman, Martin Abadi and Mads Torgersen in European Conference on Object-Oriented Programming ECOOP 2014” [5]

In this paper, the author delves into the realm of TypeScript, an extension of JavaScript designed to streamline the development of large-scale JavaScript applications. While inherently every JavaScript program can be considered a TypeScript program, TypeScript offers a specialized module system, classes, interfaces, and a comprehensive gradual type system. The aim of TypeScript is to facilitate a seamless transition for JavaScript programmers, supporting well-established programming idioms without necessitating major rewrites or annotations. Notably, TypeScript's type system is intentionally not statically sound. The primary objective of this paper is to encapsulate the essence of TypeScript by providing a precise definition of its type system concerning a core set of language constructs. The main contribution lies in presenting a robust, mathematical formalization, and introducing a refactoring approach into a safe inner fragment along with an additional layer of unsafe rules.

Keywords: Type System, Operational Semantic, Object Type, Call Signature, Return Type

“MongoDB – a comparison with NoSQL databases by Hema Krishnan, M. Sudheep Elayidom and T. Santhanakrishnan in International Journal of Scientific and Engineering Research 2016” [6]

In this paper, the author explores the evolving landscape of web-based applications and their evolving data management needs. Traditional relational databases have long provided a robust set of features and ensured strict data consistency. However, the considerable costs associated with storing and manipulating data in relational database systems have prompted the development of NoSQL databases. These databases offer enhanced scalability and support for diverse data types.

Among the various NoSQL databases, MongoDB stands out for its exceptional scalability, performance, and availability. Specifically designed for internet and web-based applications, MongoDB operates on a document-based model that effortlessly accommodates unstructured data. Unlike traditional relational databases, MongoDB eliminates the need for costly and time-consuming migrations when application requirements evolve.A key feature of MongoDB lies in its use of BSON, a JSON-like format, to encode documents. This paper delineates the advantages of MongoDB over other NoSQL databases and explores its application in sentiment analysis. Through this discussion, the paper underscores the significance of MongoDB in meeting the evolving demands of modern web-based applications.

Keywords: MongoDB, NoSQL, Document, Sharding, Replication, Indexing

“A Study on Mongodb Database by Kavya S. in International Journal for Scientific Research & Development| Vol. 3, Issue 10, 2015” [7]

In this paper, the author delves into the core features, advantages, and applications of non-relational databases, particularly MongoDB, highlighting its superiority over relational databases in big data contexts. MongoDB is contrasted with MySQL for comparison purposes. The discussion emphasizes MongoDB's key attributes including flexibility, scalability, auto-sharding, and replication, underscoring its pivotal role in big data and real-time web applications. By exploring MongoDB's versatility and robustness, the paper argues for its preeminence in handling the demands of contemporary data-intensive environments. Throughout, the author establishes MongoDB as a leading technology in the realm of database management, shedding light on its pivotal role in facilitating the storage and retrieval of vast amounts of data efficiently and effectively.

Keywords: NoSQL, MongoDB, auto sharding, aggregation

“A Review on Various Aspects of MongoDb Databases by Anjali Chauhan in International Journal of Engineering Research & Technology (IJERT), Vol. 8 Issue 05, May-2019” [8]

In this paper, the author explores the prominence of MongoDB among NoSQL databases, particularly highlighting its efficacy in constructing data warehouses, chiefly owing to its adeptness in leveraging the "sharding-nothing cluster architecture." As an open-source database, MongoDB presents an optimal solution for crafting high-performance data warehouses. The paper conducts a comprehensive review of MongoDB, delving into its multifaceted aspects and outlining several key issues for consideration. These issues serve as potential avenues for future research endeavors within MongoDB databases. By delineating these areas, the paper lays the groundwork for further exploration and scholarly inquiry, thereby contributing to the ongoing discourse surrounding MongoDB's utilization and enhancement in the realm of data management and analysis.

Keywords: No-SQL, MongoDB, Database, RDBMS, Nonrelational databases

“Website Development Technologies: A Review by Pratiksha D Dutonde, Shivani S Mamidwar , Monali Sunil Korvate , Sumangala Bafna, Prof. Dhiraj D Shirbhate in International Journal for Research in Applied Science & Engineering Technology (IJRASET) Volume 10 Issue I Jan 2022” [9]

In this paper, the author delves into the essence of Service Science, which forms the bedrock of the knowledge system and internet services, revolving around the provider/client paradigm. The central focus lies in devising a methodology applicable to the augmentation of internet services, such as websites, web applications, and eCommerce platforms. The primary objective is to engineer a methodology capable of instilling structure into highly unstructured problem domains, thereby aiding in the advancement and prosperity of internet services. The proposed innovation, termed the Web Development Life Cycle (WDLC), is a novel approach derived from pre-existing methodologies and customized to suit the unique requirements of web development.

A pivotal aspect of this paper is the elucidation of the distinct phases constituting the WDLC, offering a comprehensive understanding of its implementation and potential impact on the domain of internet services. Through meticulous delineation, this paper intends to illuminate the path towards a systematic and effective approach to web development, underscoring the significance of structured methodologies in navigating the complexities inherent in the digital landscape.

Keywords: Web Development, Application Development, Technologies, eCommerce

"A Survey on Current Technologies for Web Development by Akhil Krishna and Dr. Padmashree T. in International Journal of Engineering Research & Technology (IJERT) Vol. 9 Issue 06, June-2020" [10]

In this paper, the author delves into the myriad options, formats, languages, frameworks, and technological tools available to web-based application developers. Through a comprehensive survey, the paper identifies and compares various technologies crucial for crafting web applications. It is observed that despite substantial progress in addressing the connectivity issues of the web, the landscape of web-based application technologies remains fragmented and diverse. The study concludes that while connectivity challenges have largely been mitigated, the absence of a cohesive model specifically tailored to the domain of web-based applications persists. The research underscores the need for a more unified and structured approach to address the complexities inherent in web application development, signaling opportunities for streamlining processes and enhancing the overall efficacy of web-based solutions.

Keywords: Technology, NoSQL, MySQL, Angular, NodeJS

“Research and Analysis of the Front-end Frameworks and Libraries in E-Business Development by YongKang Xing, JiaPeng Huang in 2019 11th International Conference ICCAE” [11]

This paper delves into the evolving landscape of web technology, particularly the transformation of Hypertext Markup Language (HTML)5 under the aegis of the worldwide web consortium, catapulting front-end development into the forefront of internet history. Amidst this evolution, a plethora of front-end development frameworks and libraries such as React, Angular, and Vue have emerged. Consequently, the paramount task of selecting an appropriate framework or library to fortify e-Business initiatives and enhance user experience has become imperative in web development discourse.

Commencing with an overarching examination of the leading frameworks and libraries in front-end development, this paper scrutinizes their individual performances within web services. Through a comprehensive analysis of research data across various dimensions, it delineates the merits and demerits of each framework and library based on distinct commercial criteria. In conclusion, this paper underscores the significance of these contributions and forecasts potential future trajectories for front-end development in the realm of e-Business.

Keywords: Css, Front-end, Back-end, Business

“Research and Analysis of the Front-end Frameworks and Libraries in Web Development by Arnav Awasthi, Shubham More, Warren Viegas in International Journal for Research in Applied Science & Engineering Technology (IJRASET) Volume 10 Issue IV Apr 2022” [12]

In this paper, the author explores the burgeoning landscape of online technology, projecting Hypertext Markup Language (HTML)5 as a pivotal global web consortium driving the forefront of internet history. Amidst this evolution, numerous front-end development frameworks emerge, with Angular and React exemplifying popular choices. The selection of an appropriate framework or library for launching and

expanding e-businesses significantly impacts user experience in web development. The paper initiates with an introduction to prevalent frameworks and libraries, coupled with insights into web performance analysis services. Through comprehensive research analysis, the study delineates the merits and demerits of each framework. Culminating in a summary of findings and contributions, the paper speculates on the future trajectory of front-end development. As the digital landscape continues to evolve, understanding the nuances of front-end technologies becomes paramount for navigating the ever-changing terrain of online business and user engagement.

Keywords: Front-end; JavaScript; Web Development; e-Business; HTML5.

These were the papers that were referred for getting an idea about the background of the technologies used in the project.

2. Project Description and Goals

The DevOverFlow project is an ambitious undertaking that aims to create a sophisticated Question and Answer (Q&A) platform, integrating the functionalities of StackOverflow with innovative features. The primary distinguishing factor lies in the implementation of AI-powered text generation, facilitated by OpenAI technology, to provide users with instant and accurate answers.

The key objectives of the DevOverFlow project include:

1. Creating a user-friendly interface: The project places a strong emphasis on developing a user-friendly interface for posing questions and accessing responses. This focus on user experience is intended to foster community engagement and facilitate seamless knowledge sharing among users.

2. Developing a comprehensive authentication system: The project aims to implement a robust authentication system to manage user accounts, permissions, and access control effectively. This will ensure secure interactions and protect user data from unauthorized access.

3. Designing a scalable backend architecture: The project leverages the capabilities of NextJS and MongoDB to create a scalable backend infrastructure that can handle the demands of a thriving Q&A platform. This approach is designed to ensure optimal performance, reliability, and data integrity.

4. Implementing an AI-powered text generation system: The project seeks to integrate advanced AI and natural language processing (NLP) capabilities to generate instant and accurate responses to user queries using Open-AI API. This feature is designed to enhance the responsiveness and effectiveness of the Q&A platform, setting it apart from traditional Q&A platforms.

5. Enhancing user experience through responsive design and interactive features: The project prioritizes responsive design to ensure seamless access across diverse devices. Additionally, it incorporates interactive features, such as upvoting and downvoting questions and answers, saving questions for future reference, following tags, badges, and reputation points, to create an engaging and dynamic knowledge-sharing ecosystem.

By addressing the shortcomings of existing Q&A platforms and integrating cutting-edge technologies, the DevOverFlow project aspires to redefine the Q&A landscape, providing users with an unparalleled experience that combines AI innovation with a rich feature set.

3. Technical Specification

The DevOverFlow project adopts a robust technical architecture built on NextJS version 13.5, MongoDB, and other cutting-edge technologies to ensure scalability, performance, and maintainability. The technical specification of the project can be broken down into the following key components:

1. Frontend Layer:

The presentation layer of the DevOverFlow platform is developed using NextJS, incorporating React components to create dynamic user interfaces. Tailwind CSS is utilized for responsive styling, ensuring an optimal user experience across diverse devices. This frontend layer is designed to provide a seamless and visually appealing interface for users to interact with the Q&A platform.

2. Backend Infrastructure:

The backend of the DevOverFlow project consists of API routes, server-side form validation, and authentication mechanisms, all implemented using the NextJS framework. MongoDB serves as the database management system, facilitating efficient data storage, retrieval, and manipulation. This backend infrastructure is responsible for handling the core functionalities and data management of the Q&A platform.

3. AI-Powered Text Generation:

A key aspect of the DevOverFlow project is the integration of OpenAI API technology to generate instant and accurate answers to user queries. The platform employs natural language processing (NLP) techniques to analyse and interpret user input, generating relevant responses in real-time.

4. Recommendation System:

The DevOverFlow project incorporates a robust recommendation system that utilizes classic algorithms to suggest relevant questions, answers, and resources based on user preferences and browsing history. This feature is designed to enhance the user experience by surfacing content that is tailored to the individual's interests and needs.

5. Authentication and Access Control:

The DevOverFlow project features a comprehensive authentication system that manages user accounts, permissions, and access control. This system ensures secure interactions and protects user data from unauthorized access. The project utilizes the Clerk authentication service to establish the authentication system, and user data is stored in the MongoDB database to maintain data privacy.

6. Responsive Design and Performance Optimization:

The DevOverFlow platform is designed to be fully responsive, adapting seamlessly to various screen sizes and resolutions. To optimize performance and minimize latency, the project employs strategies such as server-side rendering (SSR), incremental static regeneration (ISR), and client-side rendering (CSR).

4. Design Approach and Details

Figma Link-: Link **Prototype Link**-: Link

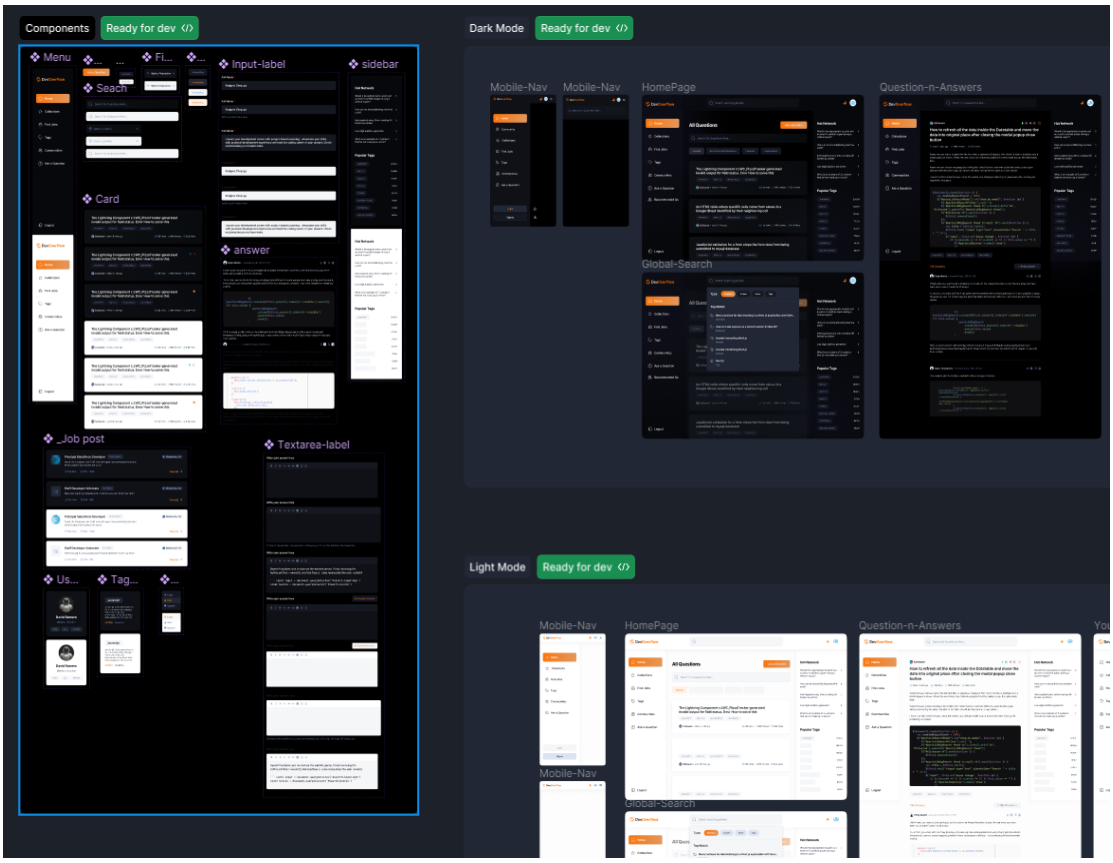

Fig 4.1 Design at Figma Link

The DevOverFlow project has been designed with a strong focus on creating an intuitive and visually appealing user interface. The design of the platform has been entirely conceived in Figma, a leading UI/UX design tool, ensuring a cohesive and polished aesthetic across all the key modules and functionalities with utmost responsivity across all the available screen sizes.

4.1 Design Approach/Materials and Methods

The design approach for DevOverFlow has been centred around delivering a user experience that is both visually engaging and highly responsive. The platform is available in both light and dark themes, catering to the preferences of a diverse user base. I have meticulously crafted each element, to create a visually harmonious and accessible interface. The use of Figma as the primary design tool has enabled me to iterate quickly, and ensure pixel-perfect implementation.

4.2 Codes and Standards

The DevOverFlow project adheres to industry-standard coding practices and design guidelines to ensure the highest levels of quality and maintainability. The frontend codebase follows the best practices of the React, NextJS ecosystem, leveraging the robust features of the NextJS framework. The backend development adheres to established API design principles and embraces modular, scalable architecture. Throughout the development process, I have been vigilant in upholding accessibility standards and web content guidelines to make the platform inclusive and user-friendly.

In addition, the project embraces the principles of clean code, emphasizing readability, maintainability, and extensibility. Consistent naming conventions, proper code formatting, and comprehensive documentation have been prioritized to facilitate collaboration and future enhancements. Furthermore, the project incorporates rigorous testing practices, including unit tests, integration tests, and end-to-end tests, to ensure the robustness and reliability of the codebase. Continuous integration and deployment processes have been implemented to streamline the development workflow and enable rapid iterations while maintaining code quality. By adhering to these industry-standard codes and best practices, the DevOverFlow project lays a solid foundation for long-term sustainability, scalability, and future growth.

4.3 Constraints, Alternatives and Trade-offs

In the course of designing and developing DevOverFlow, I have navigated various constraints, considered alternative approaches, and carefully evaluated trade-offs. The need to balance cutting-edge features with a streamlined user experience has been a constant challenge. Similarly, I had to make decisions around the optimal balance between server-side and client-side rendering to maximize performance across diverse device capabilities and network conditions. The integration of AI-powered text generation has also introduced unique design considerations. The alternatives for this particular website currently in the market is StackOverflow.

Resource constraints, such as limited computational power and API usage quotas, necessitated careful optimization to prevent performance bottlenecks and ensure scalability. Additionally, privacy and security considerations have played a crucial role in shaping the authentication and data management strategies employed in the project.

5. Schedule, Tasks and Milestones

Review Number	Date and Time	Work Presented on the review	Targets till next review
Review 0	11.01.2024, 2:00 P.M.	Overview and objective of the project and Idea.	Completing the Figma Design and initial documentation like problem definition and literature review.
Review 1	15.02.2024, 1:00 P.M.	Presented the Figma design of the website with problem definition, literature review, proposed architecture and objectives.	Start the implementation of the project and complete 70% of the project by next review. Suggestions regarding Generate AI Answer feature and recommendation algo.
Review 2	05.04.2024, 7:00 P.M.	70% implementation including major features like local search, auth, posting a question, posting an answer, community, collections, profile and tags page was ready.	Features to be completed by the final review-: • Pagination • Generate AI Answer • Global Search • Jobs Page • Recommendation algo
Review 3	22.04.2024	100% project with complete documentation is ready.	Tips for presentation during the final review and some minor changes.
Final Review	08.05.2024	Final Presentation of the project.	NA

- **Project Diary**

S.No.	Date	Key Points	Remark by Guide	Guide Name
1	06-01-2024	About Project Topic	Title discussion	Dr. Suganya P
2	14-01-2024	Abstract and Objective		Dr. Suganya P
3	18-01-2024	Literature Survey	Suggested for 10 Papers	Dr. Suganya P
4	20-01-2024	Doubts		Dr. Suganya P
5	30-01-2024	Architecture		Dr. Suganya P
6	08-02-2024	Discuss about review 1 and PPT		Dr. Suganya P
7	20-02-2024	Discuss about Auth / Home-Page Features	Suggested some features like views	Dr. Suganya P
8	28-02-2024	Discuss about backend implementations	Suggested some secure methods for backend	Dr. Suganya P
9	07-03-2024	Discuss for implement functionalities on ask a question page		Dr. Suganya P
10	14-03-2024	Trends for the website		Dr. Suganya P
11	21-03-2024	Community Page features	Suggested features for filters	Dr. Suganya P
12	28-03-2024	Discussion on other pages		Dr. Suganya P
13	02-04-2024	Discuss about review 2 and PPT		Dr. Suganya P
14	08-04-2024	Pagination Feature implemented	Suggested recommendation algo idea	Dr. Suganya P
15	12-04-2024	Global Seach and Recommendation Algo implemented		Dr. Suganya P
16	16-04-2024	Jobs Page and Generate AI Answer feature Implemented	Documentation doubts for final review were cleared	Dr. Suganya P
17	20-04-2024	100% implementation of the project completed	Documentation finalization.	Dr. Suganya P

- **Milestones**

i. Completed Project Ideation and proposal finalization – Review 0
ii. Completed abstract and problem definition – Review 0
iii. Completed Literature Survey and defined project objectives- Review 0
iv. Finalized project architecture, system design and UI design – Review 1
v. Development environment setup and initial coding – After Review 1
vi. Implemented Auth using Clerk and theme switching – Review 2
vii. Implemented 70% of the website, features such as Home Page, Community, Collections, Tags and Ask a question Page – Review 2
viii. Functionalities like upvoting, downvoting, answering a question, Filtering and local search were implemented – Review 2
ix. Website Responsivity was also implemented – Review 2
x. Performed Testing on lighthouse and found fantastic scores (Performance – 86, Accessibility – 94, Best Practices – 100, SEO – 100) – Review 2
xi. Implemented features such as Generate AI Answer, Recommendation Algo, Pagination (Remaining 30%) – Review 3
xii. Jobs Page was also implemented as everything was on time using Rapid API – Review 3
xiii. Conducted thorough testing and achieved excellent Lighthouse scores – Review 3
xiv. Deployed the final web application – Review 3
xv. Completed the final documentation – Review 3
xvi. Project presentation and final demonstration – Final Review

6. Project and Code Demonstration

GitHub Link-: https://github.com/eshan1925/stackoverflow_nextjs13
Deployed Site Link-: https://devoverflow-eshan.vercel.app/

This project is built with a modern tech stack, utilizing the following technologies:

- **Next.js:** A React framework for server-rendered and statically generated web applications, offering optimal performance and SEO benefits.
- **Tailwind CSS:** A utility-first CSS framework that provides a rapid and responsive approach to styling, allowing for quick customization and design iteration.
- **MongoDB:** A flexible NoSQL database well-suited for storing and managing various data structures, perfect for this project's needs.
- **Clerk:** A user authentication service that simplifies user management and secure login functionalities, ensuring a smooth user experience.
- **OpenAI Api:** Publicly available code interfaces that allow applications to interact and share data seamlessly. Used for generate AI Answer feature in the website.
- **Rapid API:** Rapid API is a large online marketplace where developers can discover and connect to a vast collection of APIs (Application Programming Interfaces) to enhance their projects. I have used this for building up the jobs page.

Now coming to the implementation of the website so here is the entire project structure of the website-:

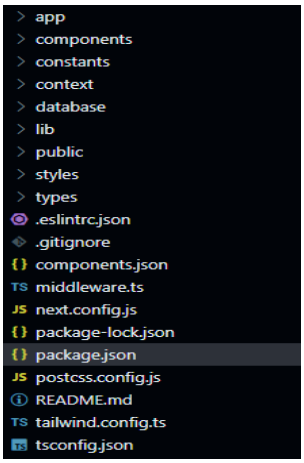

Fig 6.1 – Project Directory

Here's an explanation of each folder and file in the Next.js project directory provided, based on the image:

Folders

- **app:** Contains the core application components and logic and based on the folder name the Next-Js routing takes place.

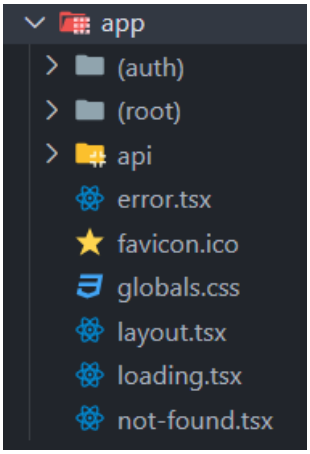

Fig 6.2 – App Folder Structure

Now here under auth folder we have code for the auth components of the website which we have coded using the Clerk library. In the root folder we have all the sub folder which will basically define the rooting of the project. The circular brackets here signify the ignorance of the name of these folders while Next JS compiles the route. Api consists of all the code that is used by external APIs.

- Now talking about the root folder's structure-:

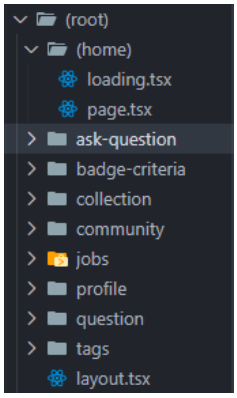

Fig 6.3 – Root Folder Structure

So, this folder consists of all the available routes by their folder names as you can see, each page consists of a file called page.tsx which has all the code of that particular page.

Now lets only discuss about the code for the homepage of the website which is under (root) > home > page.tsx

```
import HomeFilters from "@/components/home/HomeFilters";
import Filter from "@/components/shared/Filter";
import NoResult from "@/components/shared/NoResult";
import QuestionCard from "@/components/card/QuestionCard";
import LocalSearch from "@/components/shared/search/LocalSearch";
import { Button } from "@/components/ui/button";
import { HomePageFilters } from "@/constants/filters";
import Link from "next/link";
import { getQuestions, getRecommendedQuestions } from
"@/lib/actions/question.action";
import { SearchParamsProps } from "@/types";
import Pagination from "@/components/shared/Pagination";
import { auth } from "@clerk/nextjs";

// All the components that are required are imported above

// Now the main async function Home starts from here-:
export default async function Home({ searchParams }:
SearchParamsProps) {

// Calling auth

  const {userId} = auth();

// Fetching all the questions according to query of the user
  let result;
  if(searchParams?.filter==='recommended'){
    if(userId){
      result=await getRecommendedQuestions({
        userId,
        searchQuery: searchParams.q,
        page:searchParams.page? +searchParams.page:1,
      });
    }else{
      result={
        questions:[],
        isNext:false,
      }
    }
  }else{
    result= await getQuestions({
      searchQuery: searchParams.q,
      filter: searchParams.filter,
      page:searchParams.page? +searchParams.page:1,
    });

  }
```

```
// displaying all the data in the component using tailwind css and
regular html
  return (
    <>
      <div className="flex w-full flex-col-reverse justify-between
gap-4 sm:flex-row sm:items-center">
        <h1 className="h1-bold text-dark100_light900">All
Questions</h1>
        <Link href="/ask-question" className="flex justify-end max-
sm:w-full">
          <Button className="primary-gradient min-h-[46px] px-4 py-
3 !text-light-900">
            Ask a Question
          </Button>
        </Link>
      </div>

      <div className="mt-11 flex justify-between gap-5 max-sm:flex-
col sm:items-center">
        <LocalSearch
          route="/"
          iconPosition="left"
          imgSrc="/assets/icons/search.svg"
          placeHolder="Search for Questions..."
          otherClasses="flex-1"
        />
        <Filter
          filters={HomePageFilters}
          otherClasses="min-h-[56px] sm:min-w-[170px]"
          containerClasses="hidden max-md:flex"
        />
      </div>
      <HomeFilters />
      <div className="mt-10 flex w-full flex-col gap-6">
        {result.questions.length > 0 ? (
          result.questions.map((question) => (
            <QuestionCard
              key={question._id}
              _id={question._id}
              title={question.title}
              tags={question.tags}
              author={question.author}
              upvotes={question.upvotes}
              views={question.views}
              answers={question.answers}
              createdAt={question.createdAt}
            />
          ))
        ) : (
          <NoResult
```

```
            title="There's no question to show"
            desc="Be the first to break the silence! 🚀 Ask a
Question and kickstart the
          discussion. our query could be the next big thing others
learn from. Get
          involved!"
            link="/ask-question"
            linkTitle="Ask a Questions"
          />
        )}
      </div>
      <div className="mt-10">
        {/* Here this + is used for conversion of string to integer
*/}
        <Pagination
        pageNumber={searchParams?.page ? +searchParams.page : 1}
        isNext={result.isNext}
        />
      </div>
    </>
  );
}
```

- **components:** This houses all the reusable React components for building the UI.

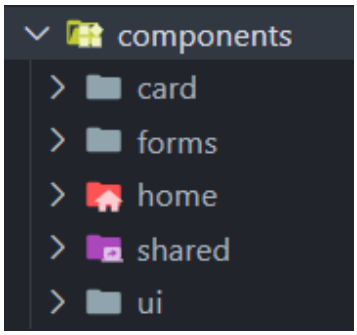

Fig 6.4 – Components Folder Structure

All the reusable components of the website are kept inside these folders, Cards folder contains various cards like Answer Card, Question Card, Job Card and User Card. Similarly other folders contain several other components.

- **constants:** Likely stores fixed values or configurations used throughout the application.
- **context:** It contain Theme providers for managing application state.
- **database:** It contains code for interacting with the MongoDB database.

- **lib:** It contains helper functions or utility code and general action functions used across the project.

  First let us see the general structure of the lib folder-:

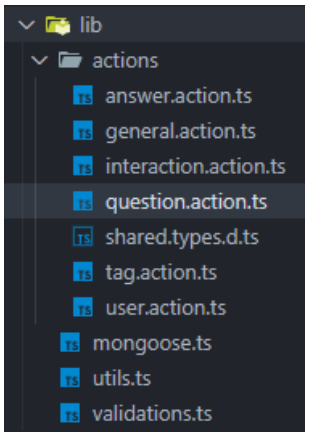

Fig 6.5 – Lib Folder Structure

Now lets us see how do we code an interaction, for displaying here I have used question.action.ts, this file consists of all the backend code for question component, ranging from fetching the question to creating a question.

```
/* eslint-disable no-unused-vars */
"use server";

import Question from "@/database/question.model";
import Tag from "@/database/tag.model";
import { connectToDatabase } from "../mongoose";
import {
  CreateQuestionParams,
  DeleteQuestionParams,
  EditQuestionParams,
  GetQuestionByIdParams,
  GetQuestionsParams,
  QuestionVoteParams,
  RecommendedParams,
} from "./shared.types";
import User from "@/database/user.model";
import { revalidatePath } from "next/cache";
import Answer from "@/database/answer.model";
import Interaction from "@/database/interaction.model";
import { FilterQuery } from "mongoose";

// Imported all the required Parameters, Database Models, mongoose
functions above…
```

```
// Starting with the functions-:

// Function to get Questions
export async function getQuestions(params: GetQuestionsParams) {
  try {
    connectToDatabase();
    const { searchQuery, filter, page = 1, pageSize = 5 } = params;

    const skipAmount = (page - 1) * pageSize;
    const query: FilterQuery<typeof Question> = {};

    if (searchQuery) {
      query.$or = [
        { title: { $regex: new RegExp(searchQuery, "i") } },
        { content: { $regex: new RegExp(searchQuery, "i") } },
      ];
    }

    let sortOptions = {};

    switch (filter) {
      case "newest":
        sortOptions = { createdAt: -1 };
        break;
      case "frequent":
        sortOptions = { views: -1 };
        break;
      case "unanswered":
        query.answers = { $size: 0 };
        break;

      default:
        break;
    }
    const questions = await Question.find(query)
      .populate({ path: "tags", model: Tag })
      .populate({ path: "author", model: User })
      .skip(skipAmount)
      .limit(pageSize)
      .sort(sortOptions);

    const totalQuestions = await Question.countDocuments(query);
    const isNext = totalQuestions > skipAmount + questions.length;

    return { questions, isNext };
  } catch (error) {
    console.log(error);
    throw error;
  }
}
```

```
// Function to create a question

export async function createQuestion(params: CreateQuestionParams) {
  "use server";
  try {
    connectToDatabase();
    const { title, content, tags, author, path } = params;
    const question = await Question.create({
      title,
      content,
      author,
    });

    const tagDocuments = [];

    // Create the tags or get them if they are already updated
    for (const tag of tags) {
      const existingTag = await Tag.findOneAndUpdate(
        { name: { $regex: new RegExp(`^${tag}`, "i") } },
        { $setOnInsert: { name: tag }, $push: { questions: question._id } },
        { upsert: true, new: true }
      );

      tagDocuments.push(existingTag._id);
    }

    await Question.findByIdAndUpdate(question._id, {
      $push: { tags: { $each: tagDocuments } },
    });

    // Create an interaction record for the user's ask_question action
    await Interaction.create({
      user: author,
      action: "ask_question",
      question: question._id,
      tags: tagDocuments,
    });
    // Increment author's reputation by +5 points for creating a question

    await User.findByIdAndUpdate(author, { $inc: { reputation: 5 } });
    console.log(path);
    revalidatePath(path);
  } catch (error) {
    console.log(error);
  }
}
```

```
// Function to getQuestionByID
export async function getQuestionById(params: GetQuestionByIdParams) {
  try {
    connectToDatabase();
    const { questionId } = params;
    const question = await Question.findById(questionId)
      .populate({
        path: "tags",
        model: Tag,
        select: "_id name",
      })
      .populate({
        path: "author",
        model: User,
        select: "_id clerkId name picture",
      });

    return question;
  } catch (error) {
    console.log(error);
    throw error;
  }
}

// function to upvote a function
export async function upvoteQuestion(params: QuestionVoteParams) {
  try {
    connectToDatabase();
    const { questionId, userId, hasupVoted, hasdownVoted, path } = params;
    let updateQuery = {};
    if (hasupVoted) {
      updateQuery = { $pull: { upvotes: userId } };
    } else if (hasdownVoted) {
      updateQuery = {
        $pull: { downvotes: userId },
        $push: { upvotes: userId },
      };
    } else {
      updateQuery = { $addToSet: { upvotes: userId } };
    }

    const question = await Question.findByIdAndUpdate(questionId,
updateQuery, {
      new: true,
    });

    if (!question) {
      throw new Error("Question not found");
    }

    // Increment the user's reputation by +1/-1 for upvoting/downvoting
```

```
    await User.findByIdAndUpdate(userId, {
      $inc: { reputation: hasupVoted ? -1 : 1 },
    });

    // Increment the author's reputation by +10/-10 for receving an
upvote/downvote to the question
    await User.findByIdAndUpdate(question.author, {
      $inc: { reputation: hasupVoted ? -10 : 10 },
    });
    revalidatePath(path);
  } catch (error) {
    console.log(error);
    throw error;
  }
}

// function to downvote a question
export async function downvoteQuestion(params: QuestionVoteParams) {
  try {
    connectToDatabase();
    const { questionId, userId, hasupVoted, hasdownVoted, path } = params;
    let updateQuery = {};
    if (hasdownVoted) {
      updateQuery = { $pull: { downvotes: userId } };
    } else if (hasupVoted) {
      updateQuery = {
        $pull: { upvotes: userId },
        $push: { downvotes: userId },
      };
    } else {
      updateQuery = { $addToSet: { downvotes: userId } };
    }

    const question = await Question.findByIdAndUpdate(questionId,
updateQuery, {
      new: true,
    });

    if (!question) {
      throw new Error("Question not found");
    }

    // Increment the users reputation

    // Increment author's reputation
    await User.findByIdAndUpdate(userId, {
      $inc: { reputation: hasdownVoted ? -2 : 2 },
    });

    await User.findByIdAndUpdate(question.author, {
      $inc: { reputation: hasdownVoted ? -10 : 10 },
```

```
    });
    revalidatePath(path);
  } catch (error) {
    console.log(error);
    throw error;
  }
}

// function to delete a question
export async function deleteQuestion(params: DeleteQuestionParams) {
  try {
    connectToDatabase();

    const { questionId, path } = params;

    await Question.deleteOne({ _id: questionId });
    await Answer.deleteMany({ question: questionId });
    await Interaction.deleteMany({ question: questionId });
    await Tag.updateMany(
      { questions: questionId },
      { $pull: { questions: questionId } }
    );

    revalidatePath(path);
  } catch (error) {
    console.log(error);
  }
}

// function to edit a question
// eslint-disable-next-line no-undef
export async function editQuestion(params: EditQuestionParams) {
  try {
    connectToDatabase();

    const { questionId, title, content, path } = params;

    const question = await Question.findById(questionId).populate("tags");

    if (!question) {
      throw new Error("Question not found");
    }

    question.title = title;
    question.content = content;

    await question.save();

    revalidatePath(path);
  } catch (error) {
    console.log(error);
```

```
  }
}

export async function getHotQuestions() {
  try {
    connectToDatabase();
    const hotQuestions = await Question.find({})
      .sort({ views: -1, upvotes: -1 })
      .limit(5);

    return hotQuestions;
  } catch (error) {
    console.log(error);
    throw error;
  }
}
```

- **public:** It stores static assets that can be directly accessed in the browser, such as images, fonts, or stylesheets.
- **styles:** It contains global or project-wide CSS styles.
- **types:** It contain TypeScript type definitions for improved code structure and type safety.

Files

- **.eslintrc.json:** It holds configurations for ESLint, a static code analysis tool.
- **.gitignore:** It specifies files or patterns to be ignored by Git version control.
- **components.json:** It consists of the folder routes and details for shadcn library.
- **middleware.ts:** It contains middleware functions for specifying the public and private routes for Clerk Auth.
- **next.config.js:** Configures the Next.js application's behavior.
- **package-lock.json:** It is generated by npm and lists the exact package versions required by your project.
- **package.json:** It specifies the project's dependencies and configurations.

```
  {
"name": "nextjs13",
  "version": "0.1.0",
  "private": true,
  "scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start",
```

```
    "lint": "next lint"
  },
  "dependencies": {
    "@clerk/nextjs": "^4.29.12",
    "@hookform/resolvers": "^3.3.4",
    "@radix-ui/react-dialog": "^1.0.5",
    "@tailwindcss/typography": "^0.5.12",
    "@tinymce/tinymce-react": "^5.0.1",
    "class-variance-authority": "^0.7.0",
    "clsx": "^2.1.0",
    "eslint-config-prettier": "^9.1.0",
    "eslint-config-standard": "^17.1.0",
    "eslint-plugin-tailwindcss": "^3.15.1",
    "html-react-parser": "^5.1.10",
    "lucide-react": "^0.368.0",
    "mongodb": "^6.5.0",
    "mongoose": "^8.3.1",
    "next": "latest",
    "prettier": "^3.2.5",
    "prismjs": "^1.29.0",
    "query-string": "^9.0.0",
    "react": "latest",
    "react-dom": "latest",
    "react-hook-form": "^7.51.3",
    "svix": "^1.21.0",
    "tailwind-merge": "^2.2.2",
    "tailwindcss-animate": "^1.0.7",
    "zod": "^3.22.4"
  },
  "devDependencies": {
    "@types/node": "latest",
    "@types/prismjs": "^1.26.3",
    "@types/react": "latest",
    "@types/react-dom": "latest",
    "autoprefixer": "latest",
    "encoding": "^0.1.13",
    "eslint": "latest",
    "eslint-config-next": "latest",
    "postcss": "latest",
    "tailwindcss": "latest",
    "typescript": "latest"
  }
}
```

- **postcss.config.js:** It has configurations for PostCSS, a tool for processing CSS.
- **README.md:** It contains project documentation or instructions for setup.
- **tailwind.config.ts:** Configuration file for Tailwind CSS.
- **tsconfig.json:** Configuration file for TypeScript, a superset of JavaScript that adds optional static typing.

Now let’s talk about the project architecture-:

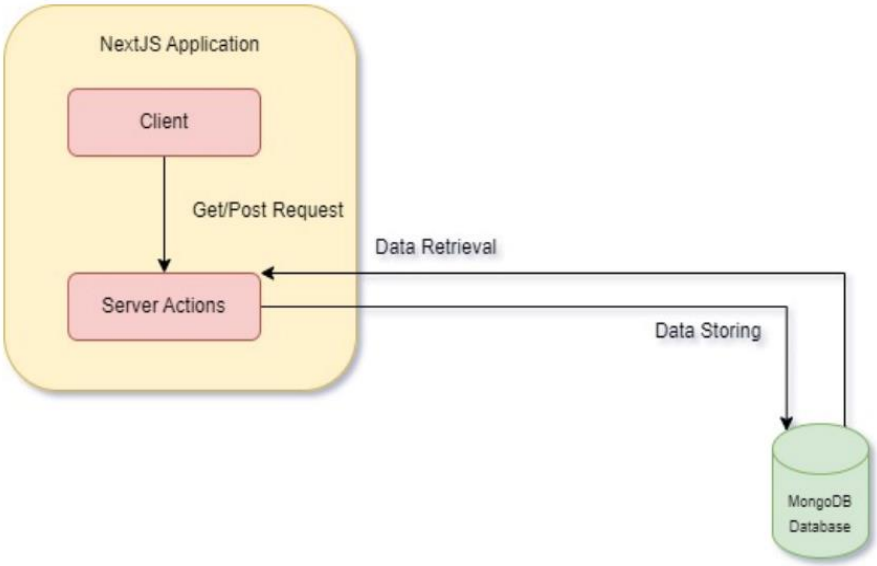

Fig 6.6 Architecture Diagram

The figure depicts a client-server system with Next.js serving as the client-side framework. Here’s a breakdown:

- **Client-side (Next.js application):** This initiates data requests and displays retrieved information. It utilizes Get/Post requests to interact with the server.
- **Server-side:** Handles data retrieval and storage. It fulfills client requests and communicates with the MongoDB database.
- **Data Storage (MongoDB database):** Stores the application’s data. The server retrieves and stores data from this database.

In essence, this architecture separates the user interface (client-side) from the data logic and storage (server-side). This promotes better organization and maintainability.

Now let’s talk about each and every page of the project and how it looks on the website-:

[1] Home Page

This page is the landing page of the website and consists of 3 main sections-:

i. Left Side Bar-: This is from where the user can navigate to different sections of the website.

ii. All Questions Section-: All the questions are displayed here and a user can search for the questions, apply filter, ask a new question and view and answer the questions.

iii. Right Side Bar-: This page consists of Top Questions and popular tags on the website.

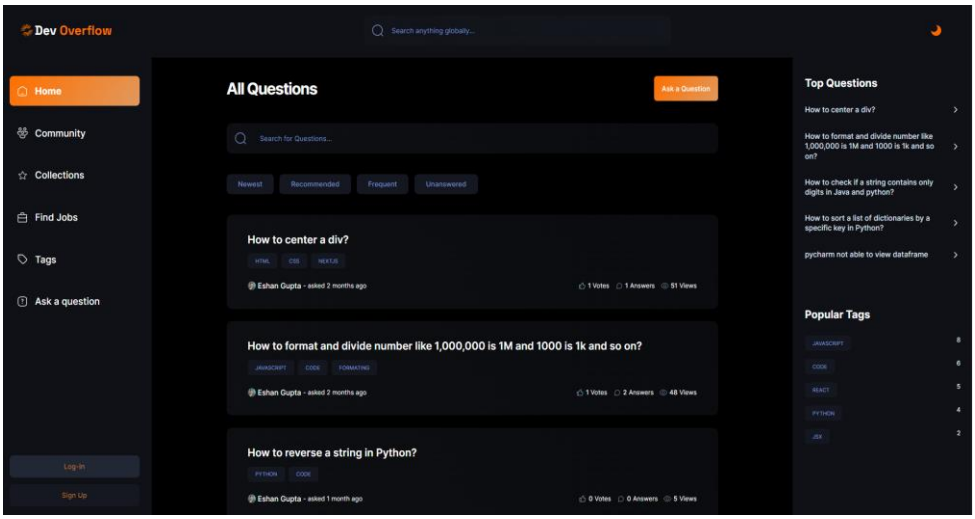

Fig 6.7 Home Page

[2] Auth Page

This page is used for authentication i.e., login and signup page. It is powered by clerk. User can create an account and can login using GitHub and Google.

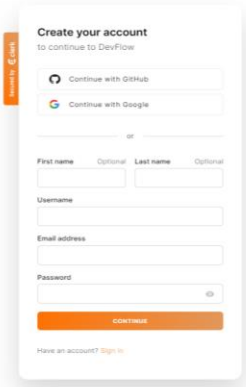

Fig 6.8 Auth Page

[3] View Question Page and Answer the question Page

This page is the view a question page, under this page a user can view answers and question details, can give new answers using the Generate AI Answer button. User can also upvote or downvote a question and an answer. With this user is also given an option to save a question for future use.

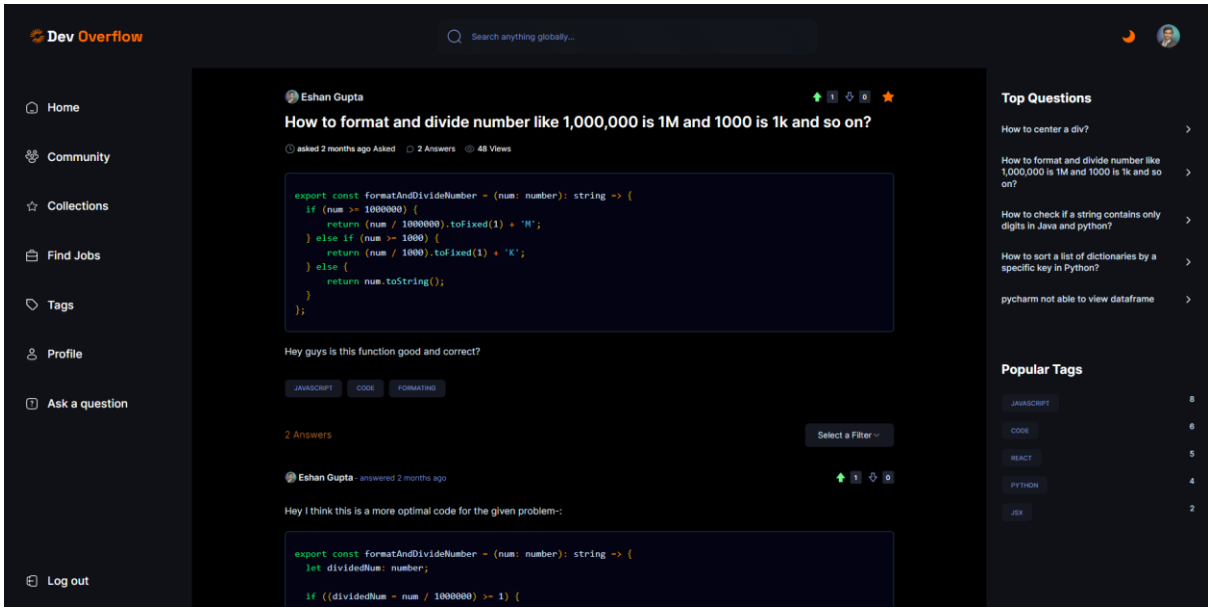

Fig 6.9 View Question and Answer the question Page

[4] Community Page

This page is used for viewing all the current users of the website, different filters like most active user, oldest user and new user can be used. Local search functionality can also be used to search user by his/her name.

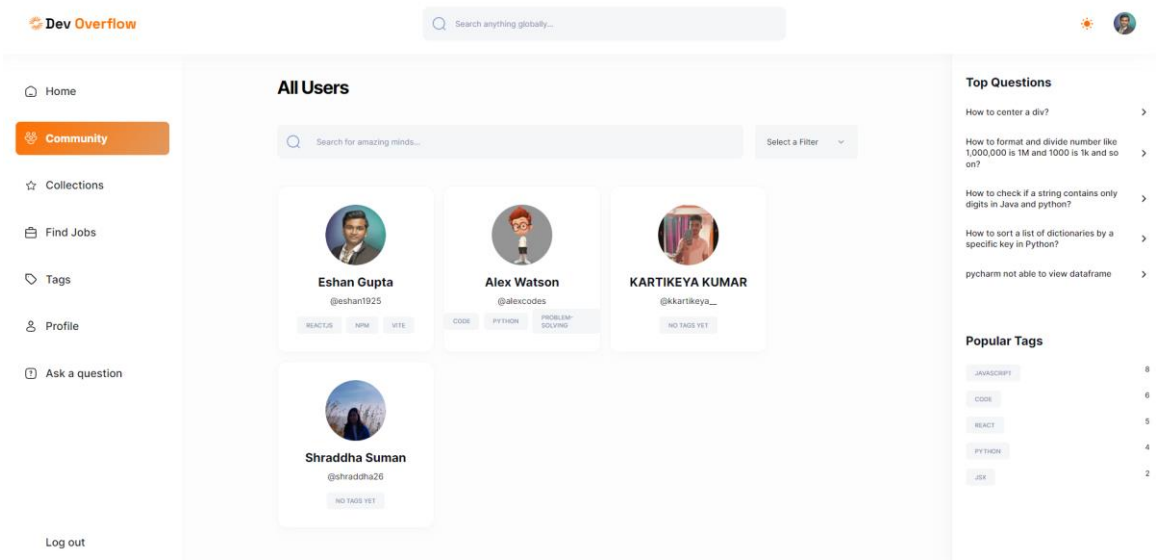

Fig 6.10 Community Page

[5] Collections Page

This page can be used for viewing the saved questions by a user. Filters and local search works here as well.

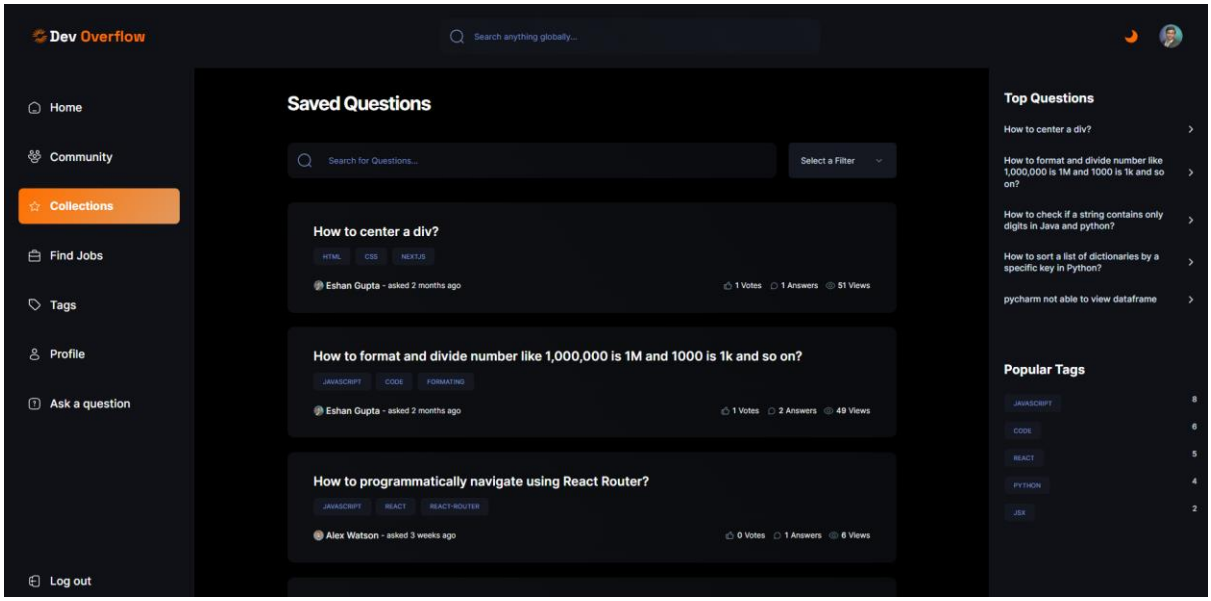

Fig 6.11 Collections Page

[6] Jobs Page

This is an additional feature of the website where a user can come and look for new job openings, these jobs are fetched from RapidAPI.

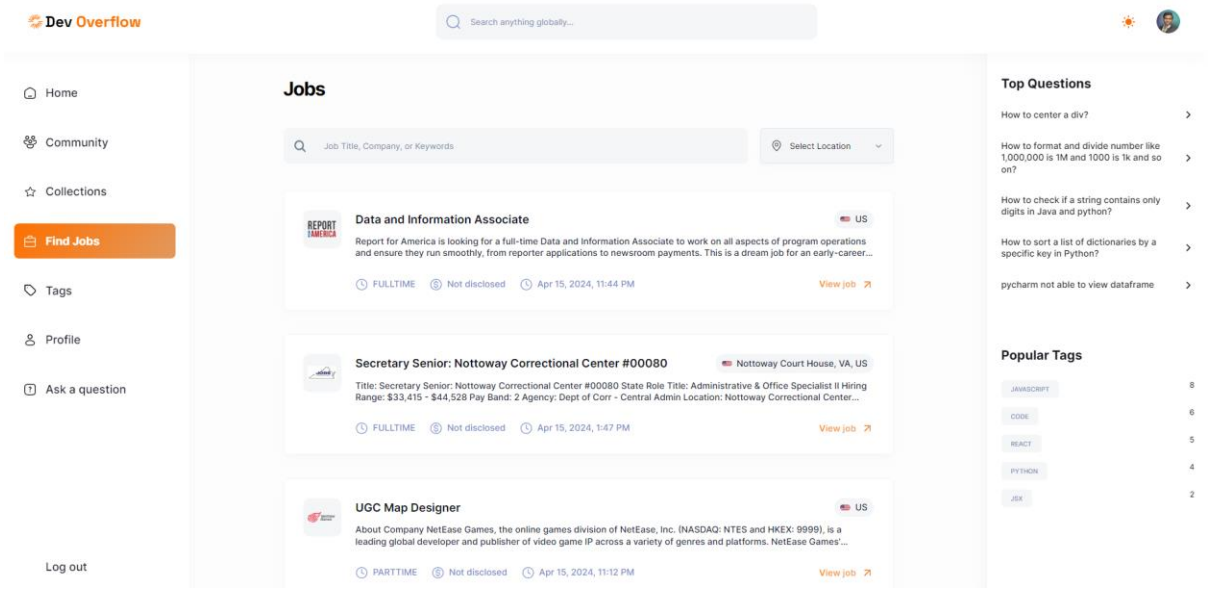

Fig 6.12 Find Jobs Page

[7] Tags Page

This page can be used for viewing all of the tags that are used in the website while creating questions. The filters and local search work here as well.

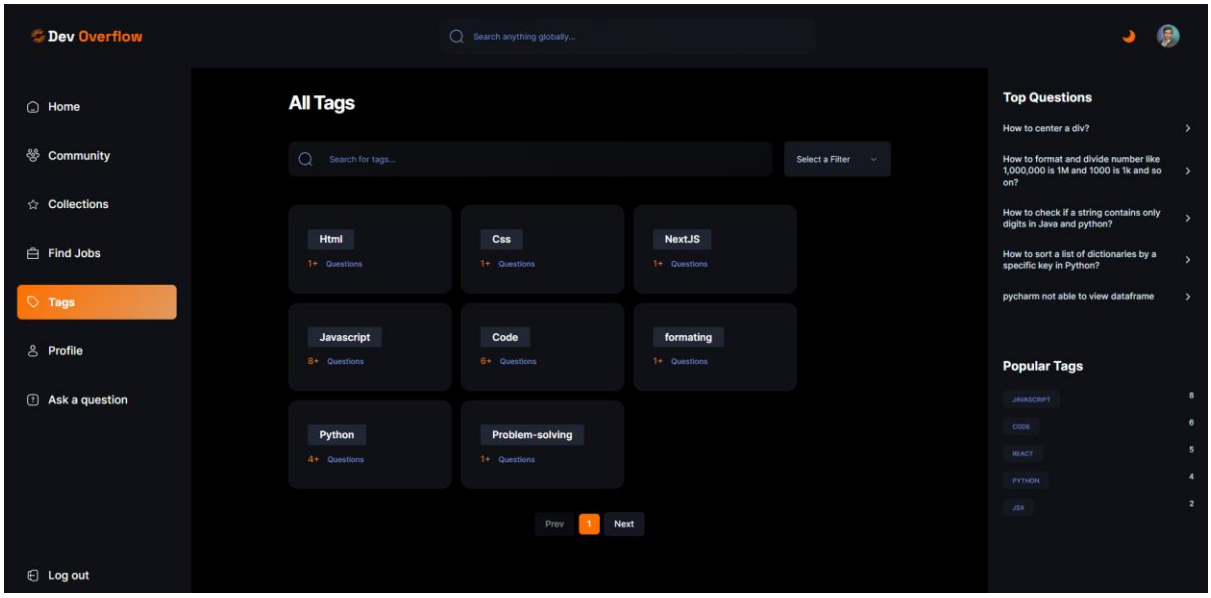

Fig 6.13 Tags Page

[8] Profile Page

This page is used for viewing a user profile, it shows all the details of the user such as how many badges a user has, what is his/her reputation, number of questions and answers a user has given.

Reputation and badges are calculated using number of views and interactions from a user.

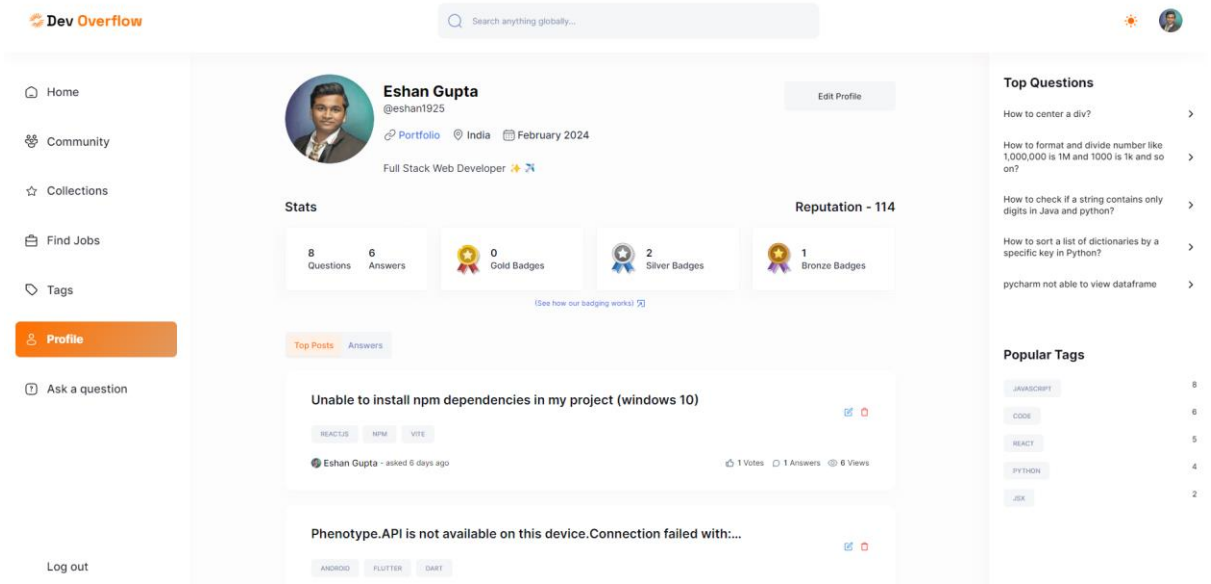

Fig 6.14 Profile Page

[9] Ask a Question Page

This page is used for creating a question, the user has to give a title to the question and a description, with this he also has to add 3 tags related to the question and then he/she can post the question to be visible at all the questions section.

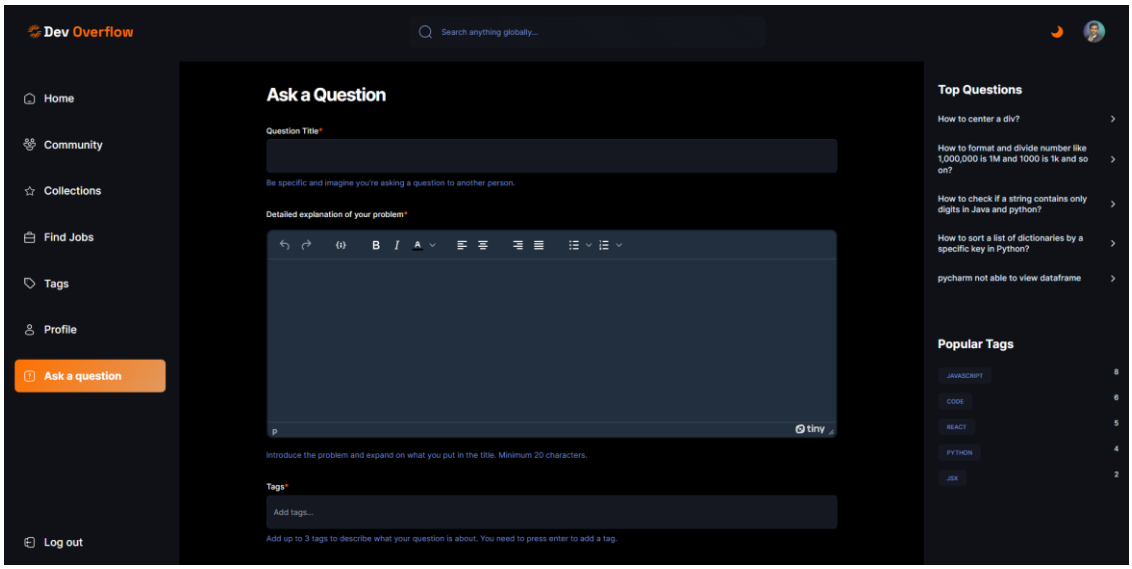

Fig 6.15 Ask a Question Page

[10] Edit Profile Page

This is the edit profile page the user can edit his/her profile here, like name, location, portfolio etc.

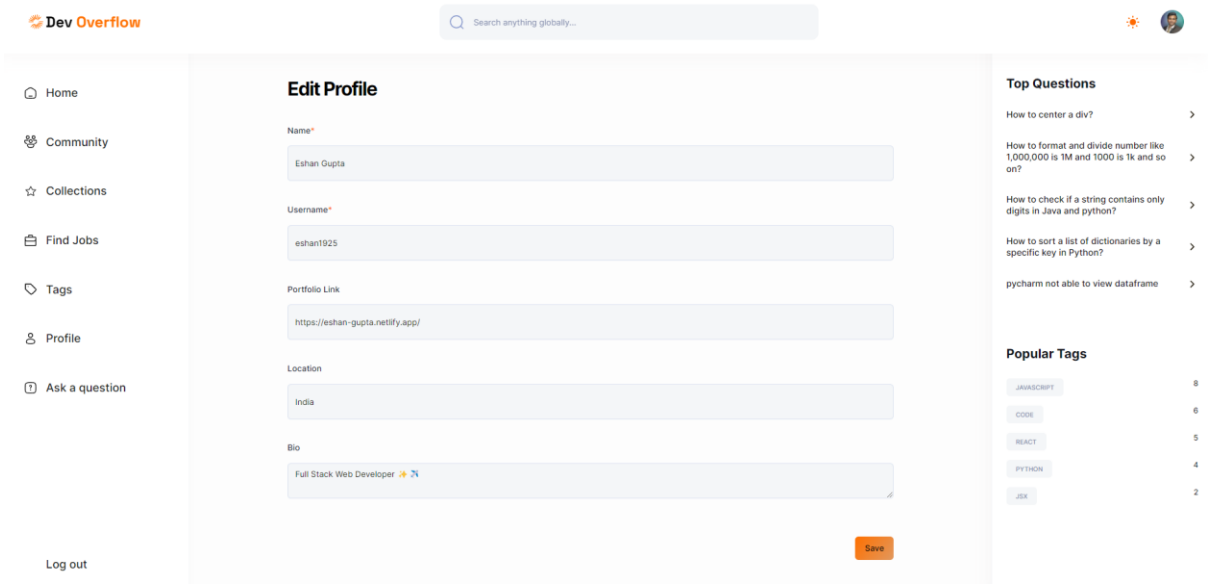

Figure 6.16 Edit Profile Page

Now let's discuss about the MongoDB Schemas or tables that we have used in our project-:

[1]. **User Schema-:**

Tab 6.1 User Schema

Field	**Field Type**	**Required**
clerkId	String	Yes
Name	String	Yes
Username	String (Unique)	Yes
Email	String (Unique)	Yes
Password	String	No
Bio	String	No
Picture	String	Yes
Location	String	No
portfolioWebsite	String	No
Reputation	Number (Default:0)	No
Saved	Array of Schema.Types.ObjectId (ref: 'Question')	No

This was the schema used for creating a user and storing his information.

[2]. **Question Schema-:**

Tab 6.2 Question Schema

Field	Field Type	Required
Title	String	Yes
Content	String	Yes
Tags	Array of Schema.Types.ObjectId (ref: "Tag")	No
Views	Number (Default: 0)	No
Upvotes	Array of Schema.Types.ObjectId (ref: "User")	No
Downvotes	Array of Schema.Types.ObjectId (ref: "User")	No
Author	Schema.Types.ObjectId (ref: "User")	No
Answers	Array of Schema.Types.ObjectId (ref: "User")	No
createdAt	Date (Default: Date.now)	No

This was the schema used for creating a question and storing all the information related to it.

[3]. **Tag Schema-:**

Tab 6.3 Tag Schema

Field	Field Type	Required
Name	String (unique)	Yes
Description	String	Yes
Questions	Array of Schema.Types.ObjectId (ref: 'Question')	No
followers	Array of Schema.Types.ObjectId (ref: 'User')	No
createdAt	Date (Default: Date.now)	No

This was the schema used for creating a tag and storing all the information related to it.

[4]. **Answer Schema-:**

Tab 6.4 Answer Schema

Field	Field Type	Required
Author	Schema.Types.ObjectId (ref: "User")	Yes
Question	Schema.Types.ObjectId (ref: "Question")	Yes
Content	String	Yes
Upvotes	Array of Schema.Types.ObjectId (ref: "User")	No
Downvotes	Array of Schema.Types.ObjectId (ref: "User")	No
CreatedAt	Date (default: Date.now)	Np

This was the schema used for creating an answer and storing it.

[5]. **Interaction Schema-:**

Tab 6.5 Interaction Schema

Field	Field Type	Required
User	Schema.Types.ObjectId (ref: "User")	Yes
Action	String	Yes
question	Schema.Types.ObjectId (ref: "Question")	No
Answer	Schema.Types.ObjectId (ref: "Answer")	No
Tags	Array of Schema.Types.ObjectId (ref: "Tag")	No
CreatedAt	Date (default: Date.now)	No

This was the schema used for recording an interaction and storing it in the database. This is also one of the important schemas as it used majorly in writing the recommendation algorithm and also for calculating the reputation.

The entire project code and the commit history for this project can be found at the GitHub link provided.

Link -: https://github.com/eshan1925/stackoverflow_nextjs13

The live and hosted version of this website can be visited using the below provided link.

Link-: https://devoverflow-eshan.vercel.app/

7. Cost Analysis/ Result and Discussion

The development of DevOverFlow, an AI-powered Q&A platform, has yielded promising results while maintaining a cost-effective approach. By leveraging open-source technologies such as NextJS, React, and MongoDB, the project has significantly reduced the overhead costs associated with proprietary software licenses.

One of the primary expenses incurred was the integration of OpenAI's text generation API, which provided the AI capabilities central to the platform's functionality. The expense can be recorded as a n expense of 5.90$. However, the long-term benefits of delivering accurate and instantaneous answers to users outweigh the initial investment in this cutting-edge technology.

The use of Tailwind CSS and responsive design principles has ensured that DevOverFlow is accessible across a wide range of devices, maximizing its potential user base without incurring additional costs for developing separate mobile applications.

The project's focus on performance optimization through techniques like server-side rendering (SSR) and incremental static regeneration (ISR) has resulted in a seamless user experience while minimizing infrastructure costs associated with hosting and maintaining the application.

Overall, a period of 4 months or 1 semester was utilised for building this project, and I had a chance to learn about different technologies associated with the project like NextJS, MongoDB, Tailwind-CSS and using different external libraries in the project.

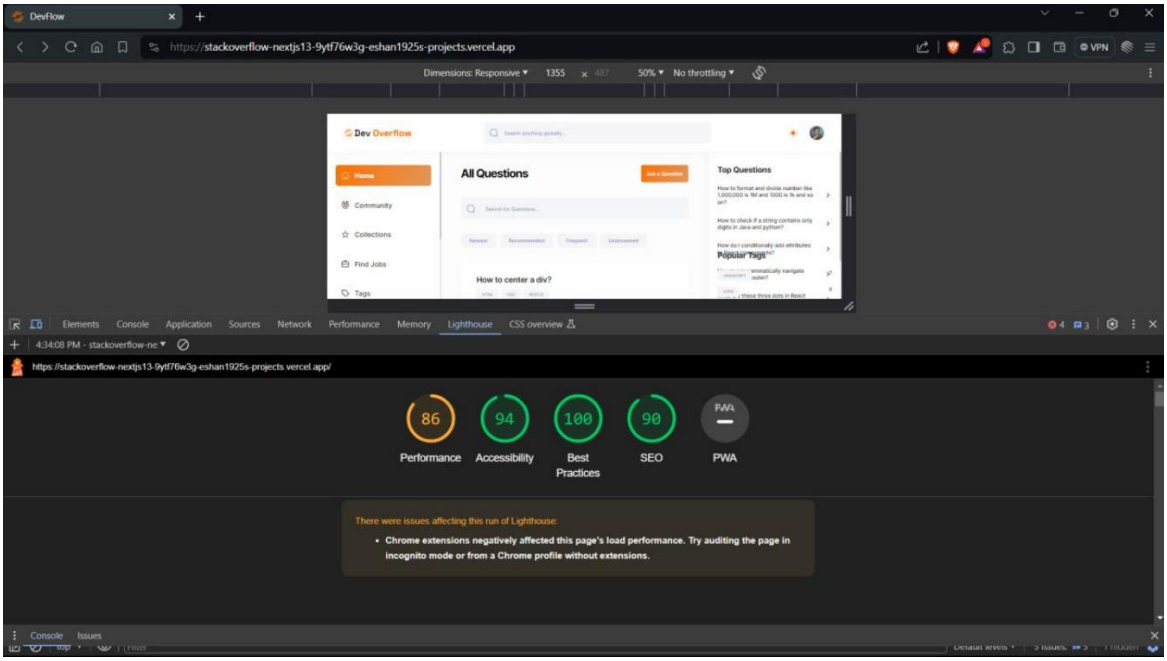

Fig 7.1 Performance Metrics

Now talking about the results obtained on performing the most used test, Lighthouse, DevOverFlow's performance is exceptional. The website scores an impressive 86 in performance, ensuring fast load times and a smooth user experience. Its accessibility score of 94 demonstrates an inclusive design catering to users with disabilities. Adherence to best practices earns a perfect 100, reflecting clean, efficient code. Furthermore, a strong SEO score of 90 indicates excellent search engine optimization, improving visibility and organic traffic. These outstanding Lighthouse results solidify DevOverFlow as a high-performing, accessible, and well-optimized web application, providing a seamless experience for all users.

Overall, the development of DevOverFlow has demonstrated the feasibility of creating a feature-rich, AI-driven web application within reasonable budgetary constraints. The strategic selection of cost-effective technologies, combined with a focus on performance and scalability, has laid the foundation for a sustainable and financially viable platform that can continuously evolve to meet the ever-changing needs of its users.

8. Summary

DevOverFlow is an ambitious capstone project that aims to create a sophisticated Question and Answer (Q&A) platform, integrating the functionalities of StackOverflow with innovative features. The primary distinguishing factor is the implementation of AI-powered text generation, facilitated by OpenAI technology, to provide users with instant and accurate answers.

The project leverages cutting-edge technologies, including NextJS version 13.5 as the core framework, MongoDB as the database, and Tailwind CSS for the presentation layer. TypeScript is the language of choice, ensuring robust and maintainable code, with strategic use of React. The architecture is grounded in NextJS fundamentals, encompassing client-server dynamics, runtime versus build time considerations, and versatile rendering strategies like SSR, ISR, SSG, and CSR.

The platform promotes user-friendly interactions by allowing Markdown input for questions and ensuring full responsiveness across diverse devices. Users can pose questions, receiving AI-generated answers alongside community responses. Active participation earns users' badges and reputation points, contributing to an incentivized and dynamic knowledge-sharing ecosystem. In summary, DevOverFlow aspires to redefine Q&A platforms by combining AI innovation with a rich feature set for an unparalleled user experience.

The design of DevOverFlow has been entirely conceived in Figma, offering a cohesive and visually appealing user interface available in both light and dark themes. The project adheres to industry-standard coding practices, design guidelines, and accessibility standards to deliver a high-quality and inclusive platform.

In summary, DevOverFlow is an ambitious capstone project that aims to redefine the Q&A landscape by integrating cutting-edge AI technology, a rich feature set, and a user-centric design approach. By addressing the shortcomings of existing platforms and providing an unparalleled user experience, DevOverFlow aspires to become a leading Q&A platform that fosters dynamic knowledge sharing and collaboration.

9. References

[1] Krutika Patil, "NextJs File-Based Routing - A Review," Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-7 | Issue-4, August 2023.

[2] Sasikumar, S., Prabha, S., and Mohan, Chandra, "Improving Performance Of Next.Js App And Testing It While Building A Badminton Based Web App," Proceedings of the International Conference on Innovative Computing & Communication (ICICC) 2022, May 27, 2022.

[3] Mohammad Fariz Syah Lazuardy and Dyah Anggraini, "Modern Front End Web Architectures with React.Js and Next.Js," International Research Journal of Advanced Engineering and Science, Volume 7, Issue 1, pp. 132-141, 2022.

[4] Saptarshi Bhattacharyya and Asoke Nath, "Application of TypeScript Language: A Brief Overview," International Journal of Innovative Research in Computer and Communication Engineering, 2007.

[5] Bierman, G., Abadi, M., Torgersen, M., "Understanding TypeScript," In: Jones, R. (eds) ECOOP 2014 – Object-Oriented Programming. ECOOP 2014. Lecture Notes in Computer Science, vol 8586. Springer, Berlin, Heidelberg, 2014.

[6] Krishnan, Hema, Elayidom, M.Sudheep, and Santhanakrishnan, T., "MongoDB – a comparison with NoSQL databases," International Journal of Scientific and Engineering Research, Vol. 7, pp. 1035-1037, 2016.

[7] Kavya S., "A study on MongoDB Database," Published in IJSRD - International Journal for Scientific Research & Development, Vol. 3, Issue 10, 2015.

[8] Anjali Chauhan, "A Review on Various Aspects of MongoDB Databases," Published in International Journal of Engineering Research & Technology (IJERT), Vol. 8 Issue 05, May-2019.

[9] Pratiksha D Dutonde, Shivani S Mamidwar, Monali Sunil Korvate, Sumangala Bafna, and Prof. Dhiraj D Shirbhate, "Website Development Technologies: A Review," Published in International Journal for Research in Applied Science & Engineering Technology (IJRASET), Volume 10 Issue I, January 2022.

[10] Akhil Krishna and Dr. Padmashree T., "A Survey on Current Technologies for Web Development," Published in International Journal of Engineering Research & Technology (IJERT), Vol. 9 Issue 06, June-2020.

[11] Xing, Yongkang, Huang, JiaPeng, and Lai, YongYao, "Research and Analysis of the Front-end Frameworks and Libraries in E-Business Development," Published in 2019 11th International Conference ICCAE.

[12] Arnav Awasthi, Shubham More, and Warren Viegas, "Research and Analysis of the Front-end Frameworks and Libraries in Web Development," Published in International Journal for Research in Applied Science & Engineering Technology (IJRASET), Volume 10 Issue IV, April 2022.

APPENDIX A

- **Setting up the project-:**

For setting up this project locally follow the steps given below-:

Step1 – Open this link (https://github.com/eshan1925/stackoverflow_nextjs13) in your browser.

Step2 – Now copy this link to your console.

Step3 - Open a terminal or command prompt on your local machine.

Step4 - Navigate to the directory where you want to clone the repository using the "cd" command.

Step5 - Clone the repository using the following command:

"git clone https://github.com/eshan1925/stackoverflow_nextjs13.git"

Step6 - Change the current working directory to the cloned repository folder:

"cd stackoverflow_nextjs13"

Step7 – Type the following command to open VS Code:

"code ."

Step8 – Type the following command to install all the node modules:

"npm i"

Step9 – Setup the .env.local using the .env.global template given in the repository.

Step10 – Type the following command in the terminal:

"npm run dev"

You have successfully cloned the repo and now your website should be visible on the address-: http://localhost:3000

- **Deployment Instructions-:**

For deploying this project, you need to raise a pull request to the main branch of the repository and get you code merged to it, once that happens you will be able to see the changes on the live website.

- **Future Scopes-:**

  • **Blog Section:** Implementing a dedicated blog section within the platform, allowing users to write, publish, and share their thoughts, insights, and experiences related to various topics. This feature would foster a vibrant community of knowledge-sharing and encourage user-generated content.

  • **Open to All Users:** After a successful beta phase, opening the website to a wider audience, transitioning from a closed or invite-only system to a publicly accessible platform. This move would enable a larger user base to benefit from the advanced Q&A functionalities, AI-powered text generation, and the wealth of knowledge available on the platform.

  • **Enhanced User Profiles:** Expanding user profiles to include additional customization options, such as personalized cover photos, bios, and the ability to showcase their expertise through badges, certifications, or portfolio sections.

  • **Gamification:** Introducing gamification elements to incentivize user engagement and participation. This could include leaderboards, achievement systems, and rewards for consistent contributions, fostering a competitive and engaging environment.

  • **Mobile Applications:** Developing native mobile applications for iOS and Android platforms, ensuring a seamless experience for users on-the-go and enabling push notifications, offline access, and other mobile-specific features.

  By implementing these future scopes, the project aims to continuously evolve, offering a more comprehensive and engaging experience for its users, while solidifying its position as a leading Q&A platform in the industry.

- **3rd Party Libraries-:**

  Various 3rd party libraries were used for implementation of the project and some of them are as follows-:

  - Clerk – This was used for the purpose of Authentication of the users.
  - TinyMCE – This was used for incorporating the feature of the editor (ask a question)
  - ShadCN UI – This was used for using a lot of predefined UI Components.
  - OpenAI API – This library was used for enabling the users with Generate AI Answer feature.