Community cloud for knowledge/skill sharing

Nikolajs Lozko

A dissertation presented for the degree of
Bachelor of Computer Science (Honours)

School of Computer Science
Department of Computer Science
University College Cork
Ireland
April 24, 2023

Supervisor: Prof. Dan Grigoras

Declaration of Originality

In signing this declaration, you are conforming, in writing, that the submitted work is entirely your own original work, except where clearly attributed otherwise, and that it has not been submitted partly or wholly for any other educational award. I hereby declare that:

- This is all my own work, unless clearly indicated otherwise, with full and proper accreditation;
- With respect to my own work: none of it has been submitted at any educational institution contributing in any way to an educational award;
- With respect to another's work: all text, diagrams, code, or ideas, whether verbatim, paraphrased or otherwise modified or adapted, have been duly attributed to the source in a scholarly manner, whether from books, papers, lecture notes or any other student

Date Created: April 24, 2023

April 24, 2023

Signed by:

_____

Nikolajs Lozko

Acknowledgements

I would like to express my gratitude to the following people who have contributed to the completion of this project:

My supervisor Dan Grigoras, for providing guidance, support, and valuable feedback throughout the project. Your insights, expertise, and guidance were essential in shaping this project.

My friends and family, for their support and encouragement, especially during the challenging times of this project. Your love and belief in me have kept me motivated and inspired.

The participants who took part in the user interview and quality testing. The feedback and suggestions have been crucial in improving the project's design and architecture.

Thank you all for your contributions and support, much appreciated.

Abstract

YoungDeveloperHub is a website application made to assist students in learning and enhancing their abilities like technical and non-technical skills such as programming or teamwork skills. The platform makes use of cloud computing resources to deliver an extensive and engaging learning environment. With YoungDeveloperHub, students have access to a wide range of learning materials and tools. This is made possible through posts, These posts were made by other students in their relevant fields. Thus, creating opportunities to learn from each other. They can find interactive exercises, lessons, and jobs or internships that are available. Along with a personalized dashboard that tracks students' posts made. The program also features a social community where students can interact with the admin to request relevant topics. YoungDeveloperHub offers the tools and assistance students need to improve as learners and developers, regardless of their level of programming experience. The issue of traditional learning environments' lack of personalization and restricted access to educational resources and too much irrelevant information posted by anyone online. Goal to give students all across the world a scalable and accessible platform by employing cloud services. Also provide a variety of instructional tools and content because the cloud permits enormous storage and processing capacities. Furthermore, real-time data processing and analysis are possible because of Hub's cloud-based architecture. The utilization of cloud services offers students a safe and dependable platform for collaboration and connection. Within a safe and secure environment, they can collaborate on group projects by leaving comments and feedback, sharing their work, and exchanging feedback. In general, YoungDeveloperHub's usage of cloud services allows it to offer students a thorough and individualized learning experience, assisting them in becoming better learners and programmers.

Contents

Chapter 1

Introduction

This chapter we provided an overview of the platform and represents the context of the project, including the problem it seeks to solve and the solution it offers. This also explains the approach taken to develop the platform and the significance of cloud computing technology in delivering an accessible and scalable learning environment.

The goal of this project was to develop a program that would be useful to a group of young students or developers who are in pursuit of their early career or degree in Computer Science. A place where they can exchange knowledge, familiarity, and skill in order to support one another while they pursue their goals as coders. The web app was created with the intention of making it simple for the group to share information while collaborating toward the common goal of enhancing members' skills. This project involved choosing a certain population, specifying the requirements for the app, and then carrying it out through deployment using cloud services. To initiate the application's creation, an analysis was made to examine what necessities and specifications were required for the intended audience. This examination involved thorough research as well as interacting with community members in order to identify which features would be most advantageous for them. This was achieved by interviewing [D] people who are enrolled in the Universities or talking to graduate developers. By using this information, an outline of necessary app requirements was created.

Modern software practices such as agile methodology alongside consistent integration and deployment were utilized during construction. Programming languages including React.js, Node.js, and Swift along with Cloud Technologies such as MongoDB Atlas and AWS played fundamental roles in building this new technology. The resulting output is a program that offers various services and the ability to help the targeted audience. These comprise forums for conversation, exchange of information, and easy access to the most relevant educational materials. The web app is positioned on a cloud base that can be effortlessly approached by users from Computer Science. Evaluation of the final product required trial runs conducted by Computer Science students who provided comments towards discovering areas needing improvement as well as future optional futures that this app can have. All in all, this project manifests technology's potential in enabling communication between students and young developers to create that bridge while still being user-focused to create that advancement in better and more effective learning.

1.1 Background and context of the project

Computer Science students deal with a lot of challenges in pursuing their degree. There is limited access to educational resources, especially during a pandemic, a lack of personalization, and sometimes students are restricted in communication and collaboration with each other. In today's age, there is a growing need for a personalized and accessible learning platform that can help students overcome these challenges and enhance their abilities not only grow as developers but evolve as the best students who help each other in their pursuits of education, experiences and learning.

The YoungDeveloperHub platform is designed to address these challenges and provide students and young developers with an engaging learning platform just like Canvas but more concentrated on team collaboration. The platform offers a wide range of learning materials such as students sharing their lecture notes, exam strategies, graduate roles, and their experiences. The coding tool was implemented to give some interactive exercises as well as a personalized dashboard that tracks students' and developers' posts made by them. Additionally, the integration of popular posts to show and give recognition to those users for their effort in helping other students and developers.

The need for a personalized learning platform is not new. There are numerous systems and applications in the market that are available with similar features and benefits but however, but many of these systems have limitations, such as high costs, limited scalability, and convoluted content that is most relevant to students and young developers. Once again YoungDeveloperHub platform goal is to address these limitations by utilizing cloud computing technology to deliver a scalable, cost-effective, and accessible learning environment that can be updated and expanded easily.

Cloud computing technology has been rapidly advancing in past recent years. The use of cloud services enables platforms to provide storage and processing capacities. This offers a variety of features and content to the users. The use of cloud services makes development faster, cost-effective, and scalable. Nevertheless, the cloud-based architecture allows for real-time data processing and analysis, which can be used to personalize the learning experience for each student and young developer.

The start of the development of this platform was motivated by the need to provide a better learning experience for students and young graduates in pursuit of their degree or their dream future job. As a student, having experienced the challenges and limitations of traditional learning environments firsthand, especially during the pandemic, it showed its flaws such as that there was a need for that kind of platform to be needed to improve learning for students and young developers. Having spent time interviewing [D] current students, graduates, and graduate developers with their expressed criticism that a platform like that, if it existed could have improved their interaction with their University colleagues during their year's studies, such as working on a team-based project, and their personal skills such as teamwork and communication. This showed the need for a personalized and accessible learning platform that could enhance students' skills and provide a supportive community between them. The YoungDeveloperHub platform is the result of this.

In conclusion, the YoungDeveloperHub platform is a cloud-based learning platform designed to address the challenges and limitations of traditional learning environments that are either too convoluted or irrelevant information that makes life much more complicated than it has to be. This platform provides an engaging learning environment, with tools and a supportive community that create this safe space for students to get motivated and stay on track with their pursuits and their goals. The use of cloud computing technology enables the platform to deliver a scalable, cost-effective, and accessible learning experience that can be updated and expanded easily in the future.

1.2 Objectives and goals of the project

This platform was designed with specific objectives and goals in mind, which are developer-based and research-based objectives. These objectives are to provide students and young developers with a comprehensive and engaging learning environment that would enhance their skills. Which also is to increase collaboration and communication among each other. This platform will solve the issue of not being limited only to laboratory hours spent working together or just a few small team-based projects that happen rarely during the four years of their degree. The following specific goals were identified at the start:

1.2.1 List of Developer-Based Objectives

1. Full Stack Application with modern UI features and backend
2. AWS cloud service (IaaS) hosted and uses range of cloud services
3. Dashboard-style application that let students from all over the world discuss internships, courses, notes, CV advice, etc.
4. Personalized sharing and learning experience for each student and developer, based on their learning styles, knowledge, and skills.
5. A wide range of learning materials posted by users and tools, including online coding tools.
6. Create a social community circle where students and young developers can interact with each other and share their work together on projects, jobs, etc.
7. Implementation of cloud computing services to deliver a scalable, cost-effective, and sharing learning environment.
8. Admin feature that keeps track of learning materials and tools provided by users that are up-to-date, relevant, and informative.
9. Feedback system from students to improve the platform in long term and ensure that it meets their specific needs.
10. Successful deployment of the application on cloud service (EC2)

1.2.2 List of Research-Based Objectives

- To assess the effectiveness of the platform in promoting knowledge sharing and collaboration among the target audience of young students and developers pursuing a career or degree in Computer Science.
- To identify the key features and functionalities that are most valued by the platform's users, as well as areas for improvement and further development.
- To explore the potential impact of cloud computing technology on the accessibility and scalability of the learning environment, and its implications for the future of education.
- To compare the platform to other similar tools and resources available to the target audience, and to evaluate its strengths and weaknesses in relation to those alternatives.

1.3 Importance of the Platform: Design and Goals

These objectives and goals were identified through a combination of research, analysis, and interviews [D] with students and young developers. The goal of providing a knowledge and skill-sharing platform between those targeted users. Having students and young developers share their learning materials, tools (e.g IDE like vscode for programming), skills, and experiences (during internships) was made clear at the start as necessary to ensure that the resources posted by these enhance their learning together as a team. This social community circle made by this platform was seen as a way to promote collaboration and communication for these users to grow as better students and work as real software engineers in the future.

Utilizing cloud computing technology was seen as the best way to provide a scalable and accessible learning environment. The platform could be accessed from anywhere in the world, making it ideal for studying remotely. Referring back to the pandemic, once it started laboratory hours and working together were limited to online calls and there was a lack of collaboration between students. Providing this platform to these users not only opens a new space for opportunities to work remotely but also gives a chance to spread awareness that they are not alone in this pursuit of this degree or the future dream job of becoming a developer. Additionally, the use of cloud services enabled the platform to provide enormous storage and processing capacities, making it possible to offer a variety of content that can be shared between users.

The need to establish that the learning materials, knowledge, skills, and tools that were posted were up-to-date, relevant, and informative was seen as essential to keep this in order. A feature of introducing an admin that can keep control of what is posted by the users, similar like a management tool. The feedback from early users that created accounts in the platform was used in such a way to continually improve the platform and make sure that it remained true to its goals and objectives that were outlined initially.

In conclusion, the platform was designed with specific objectives and goals in mind,

including providing a personalized learning experience and offering a wide range of learning materials, knowledge, skills, etc. Hence creating a social community circle platform, that utilizes cloud computing services, that provides up-to-date and relevant content, that incorporates feedback from students and young developers. These objectives and goals were identified through a combination of research, analysis, and interviews with programming students and were recognized as essential to enhancing to create of a community cloud for knowledge/skill sharing.

Chapter 2

Literature Review

2.1 Overview of existing systems and applications in the field

This part of the project focused on identifying existing research related to the use of technology to support collaborative learning and knowledge sharing among students and young developers, these include platforms such as StackOverflow, e-learning sites such as LinkedIn Learning, Skillshare, and many more. Several studies have demonstrated the benefits of collaborative learning in higher education, including improved critical thinking, problem-solving, and communication skills. These online discussion forums, social media platforms, and project management tools have a lot of useful features but they lack to have a specific target audience just specifically computer science students and young developers. Anyone can create an account and post anything. Too much irrelevant information can make it jarring for anyone who is new to the field and be overwhelmed. This project aims to address by developing a custom application tailored to the needs of this particular community. The literature review played a critical role in informing the design and development of the application, by identifying best practices and trends in the field, as well as potential challenges and limitations.

2.2 Review of relevant literature and research studies

Collaborative learning and knowledge sharing are key components of many higher education programs, particularly in fields like computer science and software engineering. In these fields, students are often required to work together on group projects and assignments, and to share their knowledge and skills to achieve common goals. Collaborative learning can help to improve critical thinking, problem-solving and teamwork skills, as well as create a sense of community that support students that can be available outside laboratory hours. A sense of safe space and feeling that you are not alone in this pursuit of the degree, interships or that dream graduate role. Similarly, this interaction between Computer Science students and developers helping each other will enhance learning outcomes and their skills at the same time

by providing students with access to a wider range of resources, experience, and expertise from each other. This should eliminate the factor of students looking elsewhere on Google search which again provides way too much information, especially the chances it's irrelevant to the student or developer.

Research has shown that students who engage in a collaborative environment are more likely to succeed academically and develop the skills needed for success in their future careers. From experience and working in a software company collaboration, communication and teamwork play a vital part of completing a task and this should give the members of this app an early start to be working like not a student but more like a software engineer.

2.3 Identification of gaps, limitations and opportunities for the project

Despite the growing popularity of collaborative learning and knowledge sharing in higher education, there is a lack of specifically focused on collaboration and knowledge sharing among students and developers. There is a need for a platform that explores the potential of technology to enhance collaboration and knowledge sharing only among students and developers. While there are a variety of digital tools and platforms available that can support collaborative learning and knowledge sharing, there is limited places where student and developers have a place to work together to the common goal of sharing and helping each other. Some web apps may be too complex or too technical, while others may not provide the necessary features and functionalities to support collaboration and knowledge sharing which can be relevant to the users. As such, there is a need for research that evaluates existing apps and identifies gaps and limitations in their functionality, usability, and suitability for student and developers. Another issue here may be differences in learning styles and preferences that need to be taken into account when designing and implementing collaborative learning and knowledge-sharing tools. This is the main reason to have a specific group instead anyone is just able to create an account and post anything they want. Targeting only a specific group can change this dynamic. This supports students' approach to learning and tries to create an emphasis on teamwork. Addressing these challenges and limitations is important in order to ensure that this platform is effective and accessible, to solve issues of people being lost in millions of information and making life difficult for them. As mentioned Computer Science area is evolving constantly and technologies changes. Other platforms are focused on general populations of students rather than specific communities or subgroups. By addressing these issues above in the project this should create the project that will create that environment we are after for. Aim is to contribute to a better understanding of technology that can be used to support collaborative learning and knowledge sharing among student developers.

2.4 Opportunities for innovation and improvement

There are several limitations and areas for future improvement that should be noted. One limitation of the current application is it is limited to one community. As it is

designed specifically for the community of young student and developers who share their experience, knowledge, and skills. Future iterations of the application could be expanded to include other communities or to support collaboration and knowledge sharing. Additionally, while the current application includes several key features and functionalities, there may be additional features or third-party integrations that need to be explored such as assistance from AI or a feature where students can nominate students for their contributions to being active students that contribute in helping other people in the platform. Overall, while the current application is effective in supporting the needs of the target community.

Based on the findings and limitations of this project, several tweaks can be made for future research and development in the field of collaborative learning and knowledge sharing. First feature, explore the potential of the application to support other communities beyond just young students and developers. For example, have the ability to host different communities that has a slightly different version of the app, which does the same things as it did for the original target community. Maybe even going further having each specific University course have their sub community and people interact and learn from each. Keeping the integrity of having specific audience but giving an opportunity not only Comnputer Science student to have this platfrom only to them. Second feature, the application could be further developed to include additional features or third-party integrations, that could enhance its usefulness and effectiveness such integrating set of management tools or API's. Big example is integrating Canvas API, so that people can have ability share lecture notes on specific course material, making life easier for people to learn from each other. For example how to write notes better in class. Finally the third and final, research could be expanded into exploring the potential of using Machine Learning for the application to able predict what student is looking for. Overtime the app may get convoluted with information and instead implemntation search bar, an AI could make it more effective for a user to find material faster. Overall, these recommendations highlight the potential for continued research and development in the field of collaborative learning and knowledge sharing, and suggest several directions for future work in this area.

Chapter 3

Project Requirements

In this chapter, the goals and requirements of the project are identified and discussed. The chapter outlines the technologies, tools and cloud services utilized in the development of the platform, including a comparison of the popular cloud computing services, AWS and Azure. The chapter also discusses the criteria used to assess the success of the project and identifies potential risks and challenges.

3.1 Project Goals and Scope

Diving into more detail on a more technical level of what the application needs to achieve using the appropriate technology and tools. The scope of work includes the front-end, back-end, and cloud service integration with the following tools and languages. This includes the appropriate language for the client and server side. Also having the best cloud service provider that will help to achieve the goals outlined previously. The main goals were to choose the best languages, and the most cost-effective and well-optimized cloud services.

To develop a full-stack application Vscode IDE was chosen due to its capabilities in terms of available libraries, quick testing features and ability quickly host the web app on localhost in web preview feature. For the front end, React.js was chosen, as React makes it easier to create interactive user interfaces. The requirement for the simple views for each state, which is easier to create in the application, and also React efficiently updates and renders components when data changes. Those views make code more predictable and easier to debug.

For the back-end that will be hosted on the AWS cloud services, Node.js was chosen as it supports many frameworks and offers high performance for real-time applications which we are developing to achieve. The application will be a dashboard-style platform that allows students from all over the world to interact with each other and share information on internships, courses, notes, CV advice, and other related topics which are important because the best and most optimized back-end languages are used as a lot of data and background required.

The front-end of the application will be developed using a JavaScript framework such as React.js. The user interface will be requiremented to provide a seamless and intuitive user experience. The front-end will be responsible for displaying the

content, handling user interactions, and providing real-time updates. The Material-UI was chosen due to its features that suit very well the application, it offers a comprehensive suite of UI tools to aid in new features faster. It comes with pre-made UI, a fully-loaded component library, and easy-to-convert already own requirement systems to production-ready components.

The back end of the application will be developed using MongoDB and Node.js. The back-end will provide a dedicated system for saving users' credentials and posting content. It will also manage user authentication and authorization, data storage and retrieval, and API integration. API will be used from the Rapid API host site that offers a wide range of different APIs. The one for developing the online coding compiler is Judge0 Extra CE which is the most advanced open-source online code execution system in the world.

3.2 AWS vs Azure

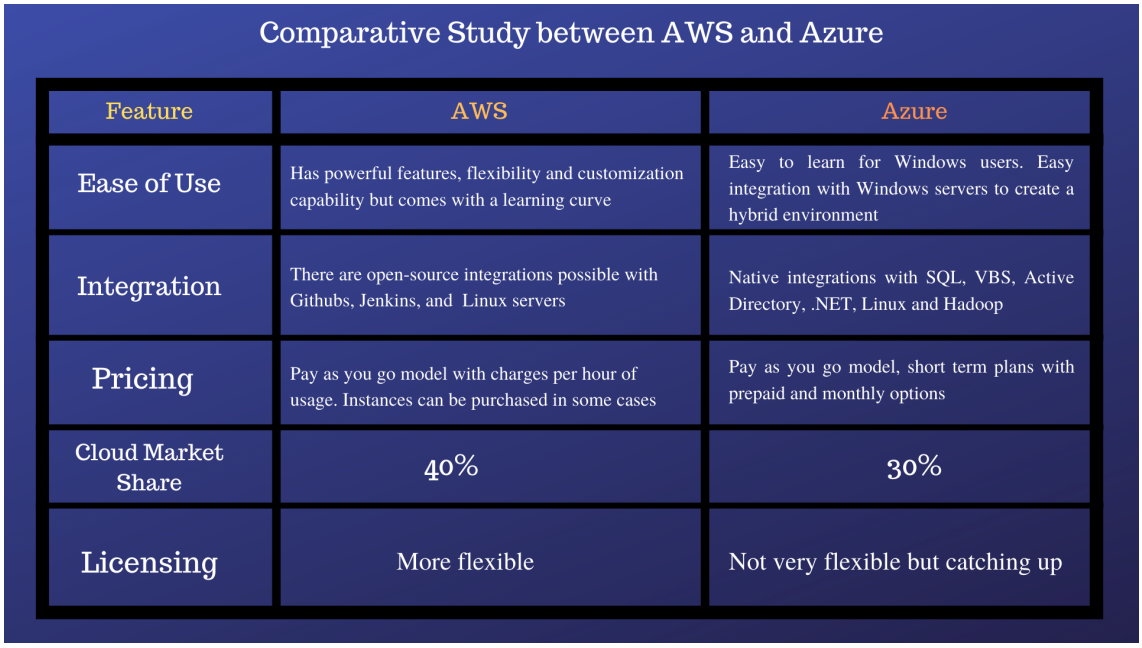


Figure 3.1: The figure above shows a comparison between AWS and Azure, two of the most popular cloud computing platforms[1]

Figure 3.1 shows a comparison of features of AWS and Azure services. This information was from extensive reasearch on these articles to learn about computing technolgies of AWS and Azure Services [2], [3], [4], [5].

3.2.1 Pricing

Pricing is a major factor to consider when choosing between AWS and Azure. Both platforms offer a pay-as-you-go model, where you only pay for what you use. However, AWS tends to be more cost-effective for smaller workloads, while Azure may be more cost-effective for larger workloads as our application is not being developed for a huge workload [6].

3.2.2 Services

Both AWS and Azure offer a similar range of cloud services, including computing, storage, database, and networking services. However, AWS has a larger and more mature service offering, with more than a hundred of services available, compared to Azure's around a hundred of services. More services provided means better support, which makes it easier to implement the client and server-side easier as more options to pick from [6].

3.2.3 Integration

Both AWS and Azure offer strong integration capabilities, with support for a wide range of programming languages, operating systems, and third-party software. However, AWS has a stronger focus on open-source software and has more community support with loads of documentation, video tutorials, and online courses, which may make it easier to integrate with the application's tools [6].

3.2.4 Scalability

Both AWS and Azure offer scalable cloud solutions that can handle large workloads and provide high availability. However, AWS has a stronger track record when it comes to scalability and has been used to handle some of the world's largest workloads, including big companies such Netflix as one of the examples. Following the track record, it is better to pick more reliable services that prove loads of professional companies use such as the tech giants [6].

3.2.5 Ease of Use

Both AWS and Azure have a user-friendly interface and offer comprehensive documentation and support. However, AWS is often criticized for its complexity and can be more challenging to set up and configure compared to Azure. This is where Azure wins over AWS but this is not much of a disadvantage compared the above challenges that AWS deals better than Azure services [6].

AWS and Azure are both strong cloud platforms that offer similar services and capabilities. The choice between them depends on specific needs, preferences, and the budget for which AWS is chosen. From the comparison, the cloud services for the application will be hosted on AWS, and a range of services will be used to achieve the project goals. For instance, Amazon S3 will be used for data storage and retrieval, while Amazon SES will handle email communications and verification. Amazon EC2 will be used for hosting the application, and other potential services could be used such as Amazon Route 53 for domain name registration and Amazon Cloud Services for content delivery will be integrated[6] .

Overall, the project objectives are to develop a reliable, scalable, and secure full-stack application that meets the goals of the project and the community of young students and developers. The application is built using the appropriate languages for front-end, and back-end and with the appropriate IDE (VsCode) [7]. The Cloud provider AWS is the most cost-effective and well-optimized in offering these cloud services. The end result will be a platform that achieves knowledge-sharing and

skills among students and developers, who are pursuing their degrees and roles as developers [6].

3.3 Technology Stack

The requirement of the application will be centered around providing a user-friendly experience that incentives collaboration, knowledge-sharing, and a community between young students and developers. The summary of the application stack is that it is made of three parts. One is the front-end, the second is the back-end and the third and final is the cloud services. The application is developed using modern technologies, libraries, and frameworks that enable the integration of various features and functionalities with cloud services.

- Front-end requirement
- Back-end requirement
- Cloud Service requirement

3.3.1 Front-end Requirement

The front-end of the web app is developed using a JavaScript framework **React.js**. The front-end focuses on a modern, user-friendly user interface, that allows for users easy navigation and also is intuitive to the user, which is easy to use. The UI also is optimized using Swift language if possible, if not then use standard React.js library or normal CSS (depending on time), for mobile devices such as iOS and desktop devices such as MacOS. Through the use of WKWebView, making sure that users can access the platform from multiple devices. Using a platform-native view called a WKWebView object, it is easy to integrate the web app into the user's mobile devices. Together with the native views of the app, a WKWebView displays HTML, CSS, and JavaScript content to enable a complete experience on mobile devices.

The interface will feature a navigation menu, profile page, admin page, topics page, and colorful interface that allow users to have a good experience when using and traversing the app, either through desktop or mobile devices. Also, users will also be able to customize their profiles such as their username and email.

The front-end requirement will also include features such as real-time updates which are the search functionality in different categories of topics and notifications to the emails when a new post is made in their favorite topics when they subscribe to them. Real-time updates will allow users receive the latest information and updates as they happen through the posts. Search functionality will allow users to easily find relevant content within a category or the most popular posts. Notifications keep users up to date about new posts or other important updates such as changes in the topics. Another notification feature is the request topics to the administrator of the web app platform. Users can notify the admin through a page called ”leave feedback” on important changes that are needed to improve the app in long-term.

3.3.2 Back-end Requirement

The back-end of the application is developed using **Node.js, MongoDB Atlas** for the database, and **AWS** for cloud services. Node.js is a single-threaded, open-source, cross-platform runtime environment for building fast and scalable server-side and network applications which is ideal for the web app building. It runs on the JavaScript runtime engine, and it uses event-driven, non-blocking I/O architecture (enables a single process to handle several requests at once. The I/O operations are delegated to the system rather than blocking the process as it waits for them to finish, allowing it to move on to the next line of code.), which again makes it efficient and applicable for real-time applications. The requirement will focus on creating a cost-effective, robust, and scalable system that can handle large amounts of data and traffic from the user's interaction with the web app. The back-end will be broken down into controllers, models, routes and validators. These are responsible for managing user authentication, authorization, data storage and API integration.

In Node.js, controllers imports required dependencies such the user and models, AWS SDK, and various helper functions that exports multiple functions for example to handle user registration. The models define data structure that contains schema that includes properties such as name of the user, routes define endpoints and handle HTTP responses, and validators ensure user input meets criteria for creating a password and preventing common attacks like SQL injection attacks. Building this requirement is crucial for making a scalable and maintainable web application for this project.

Breakdown of the structure of the back-end

The breakdown of the back-end structure is shown in the figure below (Figure 3.2). It includes several key components, including controllers, models, routes, and validators.

To conclude the backend requirement that includes features such as routing which refers, to how an application responds to a client request to endpoints, through the HTTP request method GET, POST, etc. In addition, it deals with the user profile authentications, a connection with the cloud database, connection with AWS cloud services such as S3 service in the controller's directory, content management such as posts that include image files such as png, jpeg, etc. or pdf files, and handling data storage. Content management enables users to create, publish, and share their own content. This creates the experience of connection with others that share the same interests and skills

3.3.3 Cloud Service Requirement

There is a range of different cloud services integrated to achieve the features of this web app. The cloud services that are used in this app are AWS EC2 for deployment. Amazon S3 will be used for data storage and retrieval, while Amazon SES will handle email communications such as verification and email notifications. As mentioned before Amazon EC2 will be used for the hosting of the application, and other services such as Amazon Route 53 for domain name registration on the world wide web.

- Controllers
- Models
- Routes
- Validators

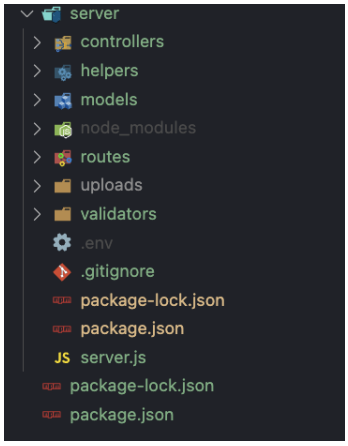


Figure 3.2: Breakdown of the structure of the back-end

The cloud service requirement will focus on creating a cost-effective, scalable, and well-optimized system that can handle large amounts of traffic and data posted by users. The requirement also includes features such as scaling which is the ability to increase or decrease the resources to meet changing demand from users on the platform that is being built. Load balancing distributes traffic and workloads to ensure that no single server or machine is under-loaded, overloaded, or idle. In general, having these cloud services integrated, there is a huge benefit of the web app being optimized to various constrained parameters such as execution time, response time, and system stability which overall improves the performance. Cloud computing offers high availability which means, that infrastructure allows a system to continue functioning as per normal, even when certain components fail, ensuring that the platform remains available and responsive even during periods of high traffic or demand to large numbers of users and data. This is a critical requirement to the project as the web app as it should able to be robust in handling all the oncoming traffic.

To end these sections, the requirements of the application are optimized for the best performance, scalability, and user engagement. The front-end, back-end, and cloud service requirement will work together simultaneously to create a seamless and intuitive user experience that encourages knowledge-sharing and skills platform, and also creates that community amongst young students and developers that are pursuing their degrees and or their dream roles as developers.

3.4 Acceptance Criteria and Metrics

Acceptance criteria (AC) are the prerequisites that a piece of software or an application must fulfill in order to be accepted by a user, a client, or other systems.

They specify the feature behavior from the perspective of users and each user interaction. The parameters of user stories are set by Acceptance Criteria. They give developers a piece of accurate information on functionality that helps in determining whether the feature is finished and functions as intended for the user [8].

The methodology used is BDD (behavior-driven development) which was learned during work experience, which gives developers and testers a consistent framework to use when deciding when to start and stop testing a certain feature. Due to the upfront description of the system's behavior, writing test cases takes less effort [9].

There is also the use of test-driven development (TDD), in which tests are written ahead of any code. TDD makes ensuring that the code is written to match the requirements outlined in the user stories and that it is extensively tested before being incorporated into the system. This approach helps in detecting any issues or mistakes with the code at the early stages of the development, consequently allowing for simpler and less costly resolution. The team was able to use TDD to make sure the code was of good quality and followed the user stories' defined acceptance criteria [10].

Figure 3.3 shows an example of an acceptance criteria template that includes these statements. This template can be used by the development team to define and test features, ensuring that they meet the requirements outlined in the user storyboards.

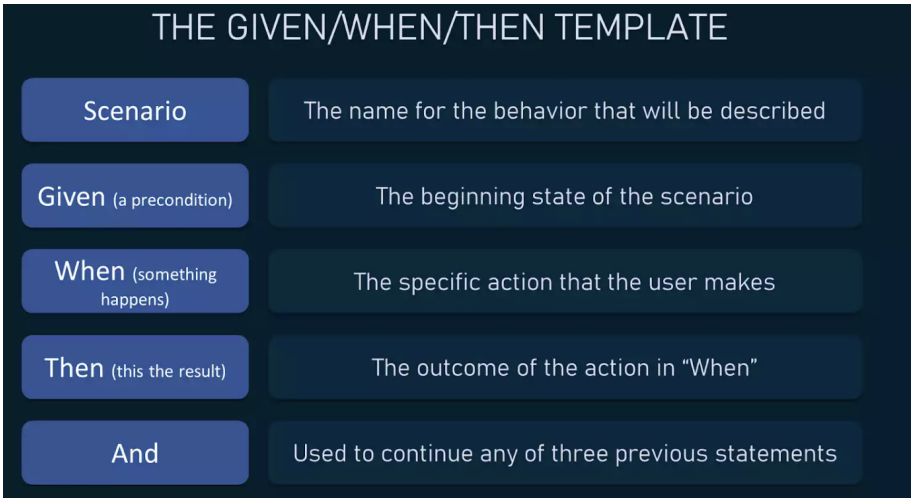


Figure 3.3: Example of Acceptance Criteria Template

3.4.1 Acceptance Criteria

- Contains a modern up-to-date and friendly user interface that allows for easy navigation and interaction with the content provided by the web app.
- Allow users to create an account and share with others their knowledge or interests and or skills.
- Allow users to create, publish, and share their own content (posts), such as internships, courses, notes, and CV advice through images, pdf files, or descriptions in posts.

- Search feature that allows users to easily find relevant content in the topics category.
- The app must provide real-time updates and notifications to keep users informed about new topics or posts being submitted or created.
- A well-optimized UI for mobile devices and desktops, ensuring that users can access the platform from any device.
- The app must be secure and protect user data, with appropriate security protocols and backups in case of attacks or downtime of the system.
- The whole app must be scalable and able to handle large amounts of data with traffic, use of automatic scaling (a feature that allows to autonomously scale cloud services like server capacity or virtual machines up or down depending on circumstances like traffic) and load-balancing (that no single server or system is overloaded, underloaded, or idle by distributing traffic and workloads) features.
- The application must be cost-effective to maintain, with the optimized cloud provider and development tools.

In this figure 3.4, a more in-depth look and research done during the process of picking the best acceptance criteria using the TEAMWORK management tool

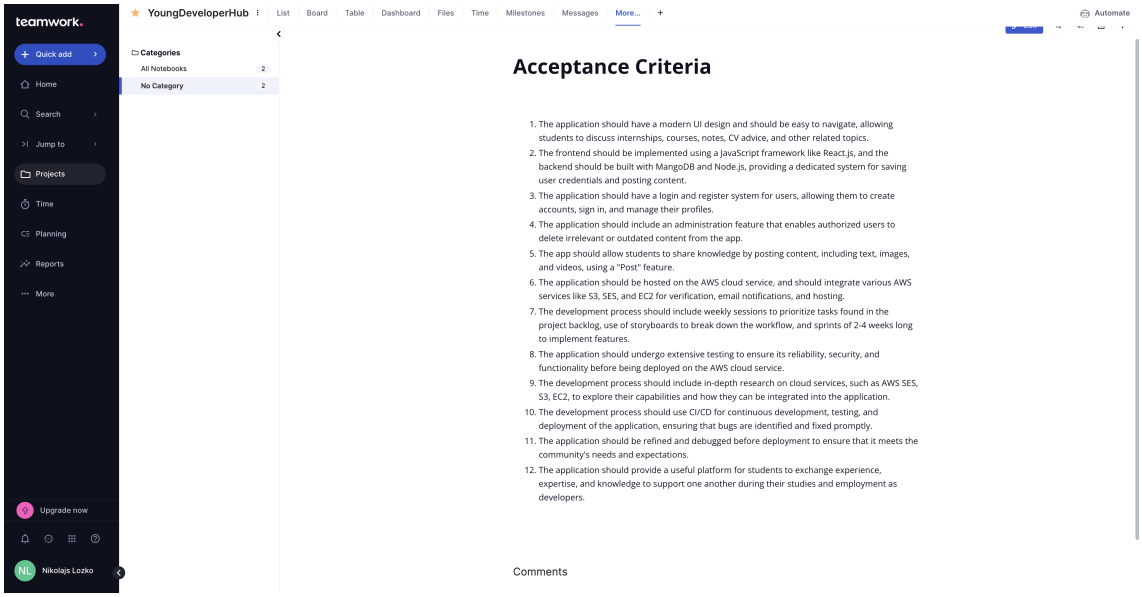


Figure 3.4: Acceptance Criteria for a feature in a software development project. The criteria include scenarios, acceptance tests, and the expected outcomes for each scenario.

3.4.2 Metrics

- User engagement metrics such as the number of verified users, frequency of visits, and interactions with certain posts or topics in different categories in the platform.

- Content metrics such as the number of posts, interactions, and popularity with posts. Also comments, as well as the quality and relevance of the content to the young students and developers.
- Performance metrics such as page load time, server response time, and error issues.
- Security metrics such as vulnerabilities, penetration tests (SQL injection attacks), and compliance with industry standards as the minimum requirement.
- Cost metrics such as cloud service usage and optimization, development and deployment usage and optimization, and cost-effectiveness.

Overall, the acceptance criteria and metrics for the application will meet the needs of our platform that is being built, while also providing a reliable, efficient, and cost-effective web app that utilizes knowledge, skills sharing, and community place for the targeted audience.

Figure 3.5 More in-depth look and research done during the process of picking the best Metrics using the TEAMWORK management tool

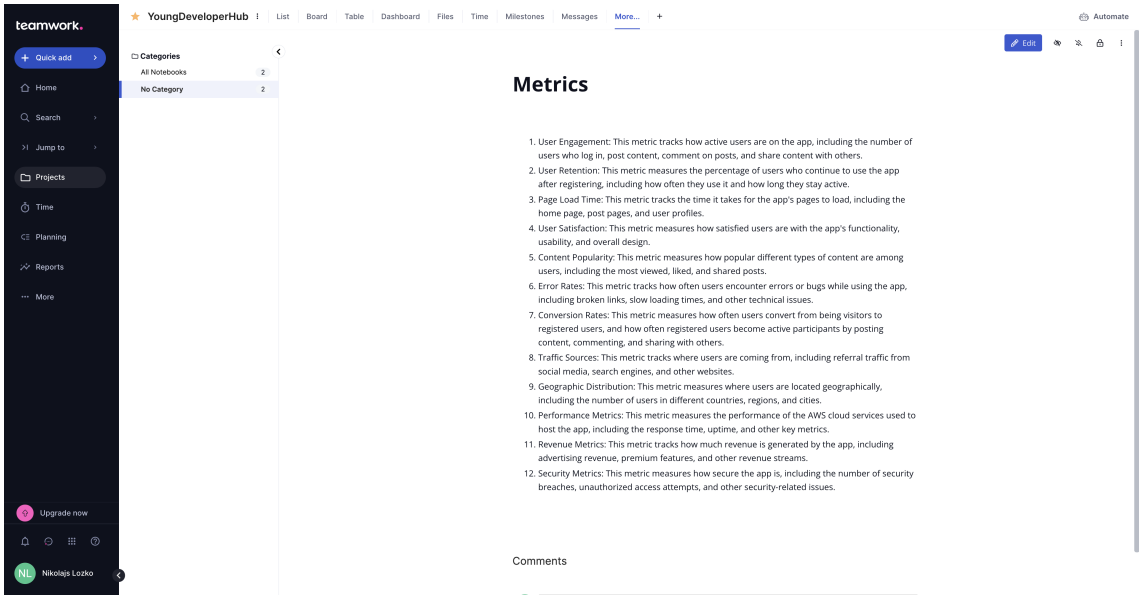


Figure 3.5: Metrics for evaluating the performance of a software development project.

This figure is described as an example of Metrics for evaluating the performance of a software development project.

3.5 Risks and Mitigation Strategies

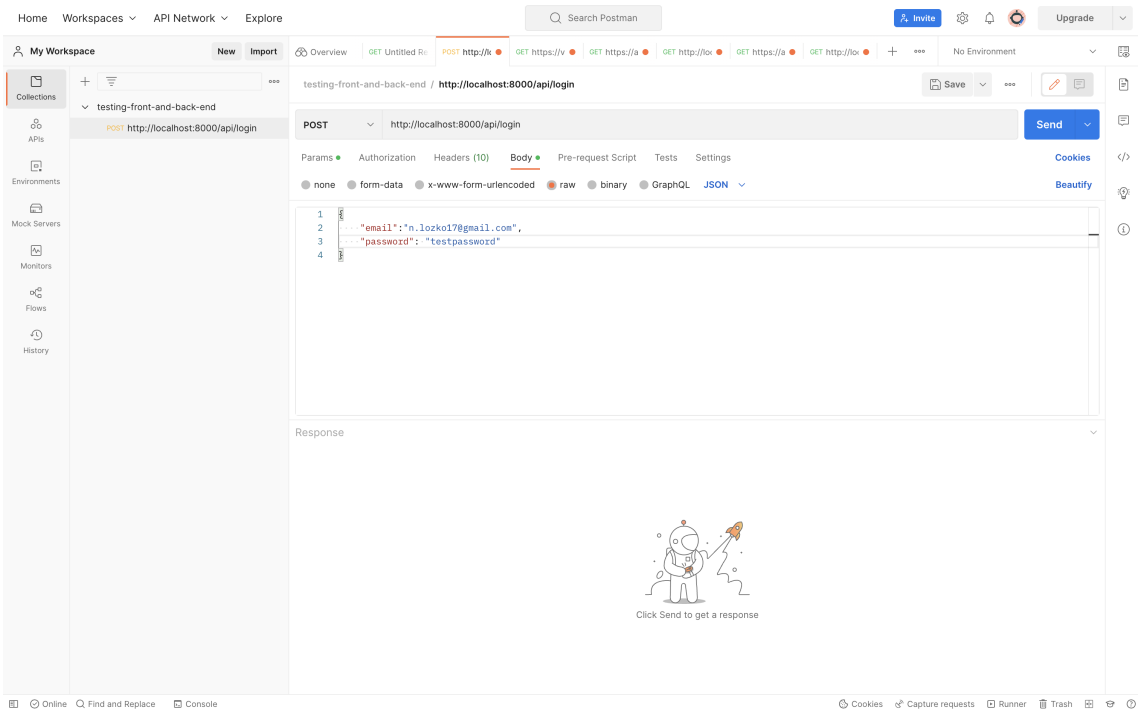
3.5.1 Technical Risks:

The risks were assessed that the development may encounter technical issues during the implementation of the project, such as compatibility issues with cloud services or problems with the application's performance when trying to use cloud services such as connection to S3 service, SES sending emails to spam folder to user's inbox

or MongoDB Atlas not being connected in the back-end. These were the first issues that were in assessment and found in development.

Mitigation Strategy: To mitigate this risk, in the development is to conduct regular testing and debugging to identify the cause of technical issues as soon as possible. One way was to use Postman (Postman is a cross-platform application that includes tools to use for both test API calls and confirm their responses. It is an HTTP client that tests HTTP requests, using a graphical user interface, through different types of responses that need to be validated and verified that are working as intended) to test HHTP requests and other services such as cloud services before making full implementation with the front-end, so is compatibility and optimal performance.

Figure 3.5.1 shows an example of mitigating the risk of the project.

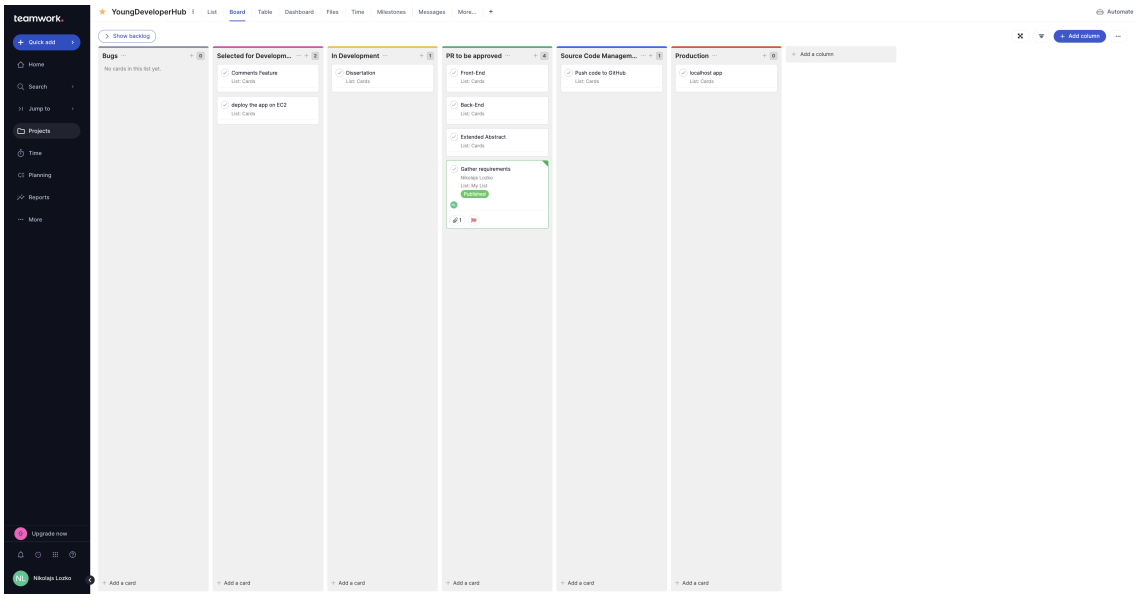


3.5.2 Time Management Risks:

The development may encounter delays in completing certain sprints, causing the project to fall behind schedule.

Mitigation Strategy: To mitigate this risk, the development is to prioritize tasks and set realistic timelines for completion that were outlined in the Project Outline report with the key dates. Furthermore conduct weekly sessions with the supervisor or storyboards review that the most essential tasks in the two or four-week-long sprints are completed accordingly and in the proper order. Additionally, storyboards and sprints are broke down the workflow and manage time efficiently through management software from a company "TEAMWORK" where the experience was gained.

3.5.2 Breaking down the development sprints and having specific feature done in the certain timeframe.



3.5.3 Security Risks:

There is a risk that the application may be vulnerable to cyber attacks, which may compromise user data and other information that may be confidential e.g user credentials.

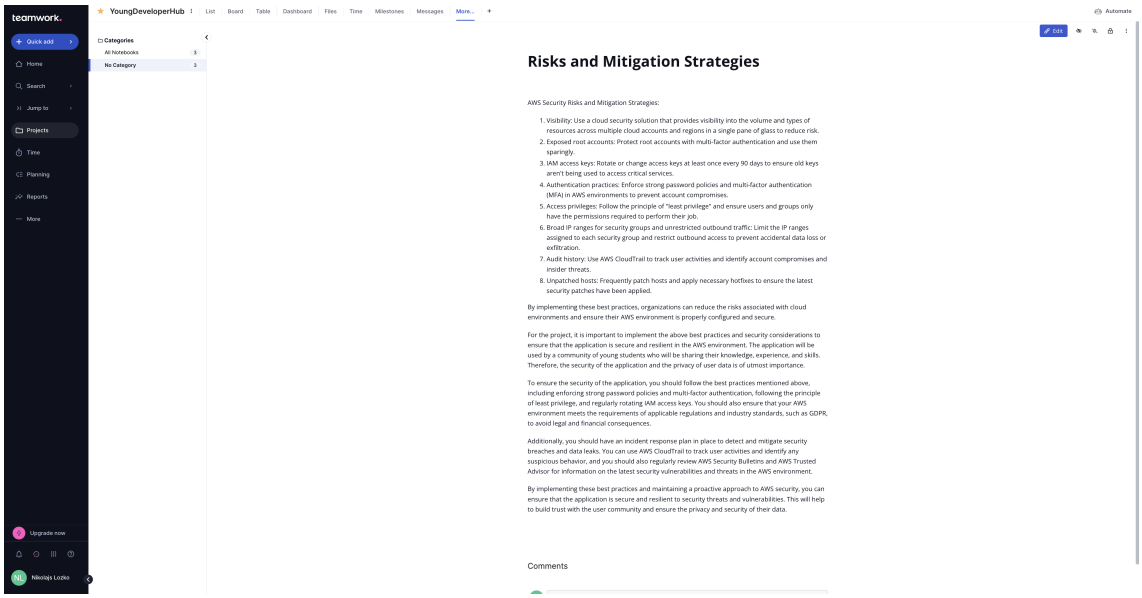
Mitigation Strategy: To mitigate this risk, implement robust security measures that comply with industry standards, such as encrypting sensitive data like passwords with salt, using multi-factor authentication for activating user accounts or sign-in, and regularly looking at security protocols that are available.

3.5.4 User Adoption Risks:

There is a risk that the target audience may not adopt the application as expected, resulting in low user engagement and adoption.

Mitigation Strategy: To mitigate this risk, conduct extensive user research and testing is to ensure that the application meets the acceptance criteria. It is vital also that the web app holds up to the needs and expectations of its target audience. The platform has to prioritize user experience and features that are user-friendly and easy to use.

Figure 3.5.4 shows an example of research done in the development of potential risks.



Chapter 3 has provided an overview of the project requirements such as the front, back, and cloud requirements. Having established acceptance criteria with risks and mitigation. The most important also having the goals and scope of work that is needed in the development of the YoungDeveloperHub platform. The chapter also covered the assessment of the project's success and identified potential risks and challenges. In Chapter 4, will shift more focus to the requirement decisions made for the platform. This chapter will explore the user interface, modern requirement principles, and the role of requirement and management tools in the web app. By the end of this chapter, readers will gain an understanding of the requirement choices that make the YoungDeveloperHub platform a user-friendly and visually appealing application.

Chapter 4

Design and Architecture

This Chapter focuses on the design and architecture decisions made for the Full Stack Application with modern UI front-end and back-end, which is then hosted on the AWS cloud service, so the platform can assist young students and developers in sharing their experience, expertise, skills, and knowledge in order to support one another while pursuing their degrees and in their employment as developers. The design and architecture of the application is based on a dashboard-style platform that allows young students and developers to discuss internships, courses, notes, CV advice, and other related topics in their degrees or roles. The chapter dives into user interface design and architectures and modern trends in the industry that are made in front-end, back-end, and cloud. Some examples that are discussed such as accessibility, consistency, and intuitive and purposeful design and architectures, which are used to create a visually appealing and user-friendly application. The role of design and architecture tools in the design and architecture is vital to create initials look of the platform, such as Figma is used to deliver these design and architectures. It is also important in highlighting their importance in improving progress in building the platform as it ensures that design and architecture goals were met and set a clear path of what needs to be achieved, polished, or modified. By the end of this chapter, readers is to have gained insight into the design and architecture choices made for this app.

4.1 Overview of the system design and architecture

The system design and architecture for the application is to consist mainly of a modern user interface that contains a dashboard that allows users to discuss internships, courses, notes, CV advice, and other related topics. That front-end interacts with back-end that has an established a connection database and to cloud services.

From the front-end side of the application, the decisions is made to built the interface using a JavaScript framework such as React.js. Reason for this choice is that React.js provides multiples features, frameworks and libraries that can make the web app high quality. It allows for the creation of highly responsive user interfaces that can handle complex interactions between components and rendering with each pages that are

loaded. For the developers, React.js, can build high quality, feature-rich, intuitive and engaging application for the targeted users. It also uses a component-based architecture that makes it easy to create reusable UI elements that can be utilised to build bigger and more complex interfaces that are needed for YoungDeveloperHub platform. This create the opportunity for the app to be flexible and more modular. In addition, React.js has a large and active community of developers, which means that there are loads of resources available for learning, debugging, tutorials, examples and libraries such as Material-UI. Overall, this makes ideal choice for building highly responsive and feature rich user web app that can meets minimum standard in the industry for modern design and architecture of the web application.

From the back-end side of things, Node.js is a powerful and efficient choice for building back-end architecture due to its capabilities in handling user data storage, retrieval and how it integrates very well with front-end and cloud services which is vital to the project. Server-side JavaScript run-time enables to build with ease, fast and scalable web application, making it an ideal design and architecture decision for building modern web app that require high performance and also being responsive. By leveraging Node.js, the architecture of the back-end is to robust that can handle large amounts of data with ease, requests, connections and create seamless integration with client-side and cloud services. Nevertheless Node.js is great part of the technology stack for the app that can result in a high-quality architecture that meets acceptance criteria of the application.

Design and architecture decision for selecting AWS as the cloud provider and services is that it can lead to a highly scalable, secure, and reliable host environment. AWS offers a broad range of cloud services that meets the acceptance criteria and goals established in earlier chapters for the project. From the storage and hosting it is ideal cloud provider for the project. By utilizing these services that AWS offers such as S3, SES, and EC2, the web app can have a scalable infrastructure that can easily be modified or adapted to the needs. As the application is in development stages these things then can be applied. AWS also offers powerful tools for scaling resources which allows the application to handle spikes in traffic when more users join the platfrom, this is essential to the overall app as it achieves being robust and reliable. To add more on the reason for this design and architecture decision, it also offers a highly secure hosting environment that can help to protect confidential user data and prevent unauthorized access. This range of security features and tools, such as firewalls, encryption, and access controls, that can ensure that the website application is highly secure. The other services also offer a range of other functionalities, such as email verification and management, that can enhance the overall functionality and reliability of the application.

To summarise this, the system design and architecture decisions from all aspects of the application is vital as it provides a solid foundation in development process and a reliable, secure, and scalable platform for young students and developers to share their knowledge, experience, and skills with each other.

4.2 Tools and Technologies Used

- **Designs**

  Figma is used for designing, architecting, and prototyping the user interface of the application. It enables a preview of the app and gains some crucial feedback on the app's design and architecture before it's built. Documentation for Figma can be found at [11]. All the original designs can be found here as shown in Appendix A using Figama B.1

- **Frontend**

  The frontend of the application is built using React.js and Material-UI. Documentation for React.js [12] and for Material-UI [13].

- **Backend**

  The backend of the application is built using Node.js. Documentation for Node.js can be found at [14].

- **Cloud Database (MongoDB Atlas)**

  MongoDB Atlas is used as the cloud database for the application. Documentation for MongoDB Atlas can be found at [15].

- **Cloud Services**

  AWS cloud services such as S3, SES, and EC2 are used for the application. Documentation for AWS cloud services can be found at [16].

- **CI/CD (LaunchDarkly)**

  Continuous Integration and Continuous Deployment (CI/CD) is implemented using LaunchDarkly. Documentation for LaunchDarkly can be found at [17].

- **Management**

  TEAMWORK Project and Management Software is used to manage the project. Storyboards are used to break down the workflow. Documentation can be found at [18].

4.3 System Components and Interactions

The application design and architecture is to consist of the following components [19]:

4.3.1 User Interface and Experience

The UI/UX (user interface or user experience) is designed and architected using Figma and implemented using React.js with Material-UI. The UI/UX allows users to navigate the app with ease, create and manage their own profiles, post, and view topics, and interact with the admin.

4.3.2 Authentication and Authorization

These components are implemented using Node.js and MongoDB Atlas. Which allows users to create accounts, log in, reset passwords and access the app's features based on their roles and permissions.

4.3.3 Data Storage

The data storage component is to use MongoDB Atlas to store user data, posts, and other content. AWS S3 is used to store media files such as images, docs, links, pdf, and videos.

4.3.4 API and Back-end Services

The API and backend services are implemented using Node.js and is to interact with the data storage component to retrieve and update data. AWS SES is to be used to send email notifications to users as one of the examples from the many.

4.3.5 Cloud Infrastructure

The app is hosted on AWS EC2 instances for scalability and reliability. The cloud infrastructure use of AWS Route 53 for DNS management, and AWS CloudWatch for monitoring and logging.

4.3.6 External Services

The app is to integrate with external services such as Rapid API [20] for the coding tool integration.

To offer a seamless and safe user experience, these components interact with one another. The user interface communicates with the API and back-end services to gather and display data such as compiling the code. Then sends the output to be verified as a request and then receives the output which is then sent as a response and printed on the user's screen. Meanwhile, the authentication and authorization component limit access to the app's features based on user roles and permissions. The cloud infrastructure makes sure the app is accessible to users, that the app is responsive, and that the data storage component provides persistent storage for user data and content. The integration of external services is the last step in giving users more functionality and value. This again brings this opportunity to code outside laboratory hours and share it with people in the app as a screenshot or a file in the post. [20]

4.4 Design and Architecture decisions and Trade-offs

The following design and architecture decisions and trade-offs have been made during the development of the application during the cloud services integration from the back-end and front-end of this application with provided information [21]:

4.4.1 Scalability vs Complexity

One of the main design and architecture decisions is to prioritize scalability over complexity because it is important to have the best possible app performance. The application's code is to be hosted on AWS EC2 instances to accomplish this, enabling it, to scale up or down in response to demand as how many requests made by users is one of the examples. However, this choice comes with a complexity trade-off, because the application has to operate numerous instances and handle the data that is being provided or fed.

4.4.2 Security vs Usability

Another key design and architecture decision is to prioritize security over usability. To achieve this in this app, the app implements strong password authentications, and multi-factor authentication (MFA) such as an email verification process through SES. However, this decision comes with a trade-off in terms of usability, as many users may find the additional security steps and measures inconvenient and may require more work when using the app. The challenge is to have balance so not to interfere with user experience but also maintain good security for the user.

4.4.3 Flexibility vs Consistency

Also to add to design and architecture decisions is to prioritize flexibility over consistency in terms of user-generated content. To achieve this, the app allows users to post content in various formats and attachments such as images, files, etc. However, this decision comes with a trade-off in terms of consistency, as the app may have to deal with a wide range of content types and formats such as pdf, videos, and images, which makes it more challenging to store, display and organize content in the correct way and keep the UI uniformed and consistent.

4.4.4 Integration vs Complexity

An important design and architecture choice is made to prioritize integration with external services such as Rapid API as one of the examples to provide additional functionality to users. However, this decision comes with a trade-off in terms of complexity, as the app is to have to interact with multiple APIs and data sources, increasing the complexity of the app and requiring additional testing and maintenance over a period of time and during the deployment and have problems with performance as well which needs to be consistently monitored to make sure these services keep working with the application. Having a backup plan or error page in case these services are down is very important

4.4.5 Performance vs Cost

In order to optimize cloud infrastructure performance, a strategic choice is made which is the importance of efficiency over expenses. To mitigate costs, the application is to be hosted through cost-saving alternatives such as AWS EC2 free tier and MongoDB Atlas as DynamoDB atlas charges for cluster use. This decision still enables the app to achieve high performance and availability while keeping the cost

that is required in cloud infrastructure to the minimum. This proves how it is still possible to have a fully functioning app with no extra costs.

By making these design and architecture decisions and trade-offs, the app is scalable, secure, and flexible which meets the needs of young students and developers pursuing their degrees and employment as developers.

4.5 User interface design and architecture and usability considerations

The user interface (UI) of the application is to be designed and architecture with the following principles in mind with provided research [22]:

4.5.1 Simplicity

The UI is to be designed and the architecture is to be simple and easy to use, with a minimalist design and architecture that reduces clutter and distractions. Users are to be able to navigate the app easily and find the features they need without being overwhelmed by too much information.

4.5.2 Consistency

The UI is to be designed and architected with a consistent style and layout across all pages to make it easier for users to navigate and understand the app. Consistent use of colors, typography, and icons is to help users identify different sections of the app and perform actions without confusion to users of this app.

4.5.3 Accessibility

The UI is to be designed and architected with accessibility in mind, with features such as highly vibrant colors, larger fonts, and smart shorcuts in UI to make it easier for users to navigate quickly around the app and get familiar with it fast.

4.5.4 Mobile Responsiveness

The UI is to be designed and the architecture to be responsive and optimized for mobile devices, allowing users to access the app on the go. The app is to be designed and architected with this mindset to make sure that the most essential features are accessible on any size screen like popular devices such as iPhone and Android phones. C.23

4.5.5 Usability Testing

Usability testing is to be conducted throughout the development process to ensure that the UI is intuitive and easy to use for all users. Feedback from users is to be incorporated into the design and architecture process to make sure that the app meets the needs of young students and developers.

By following these UI design and architecture principles and usability considerations, the app is to be able to create a user-friendly and accessible application that is optimized for both desktop and mobile devices.

4.6 Client-side architecture

4.6.1 Components

React.js uses components as its building blocks, allowing for reusable and modular code. Using a range of components to build the user interface of the application, such as forms for user input, cards for displaying content, and models for displaying pop-up messages for example when user logins or creates a new post.

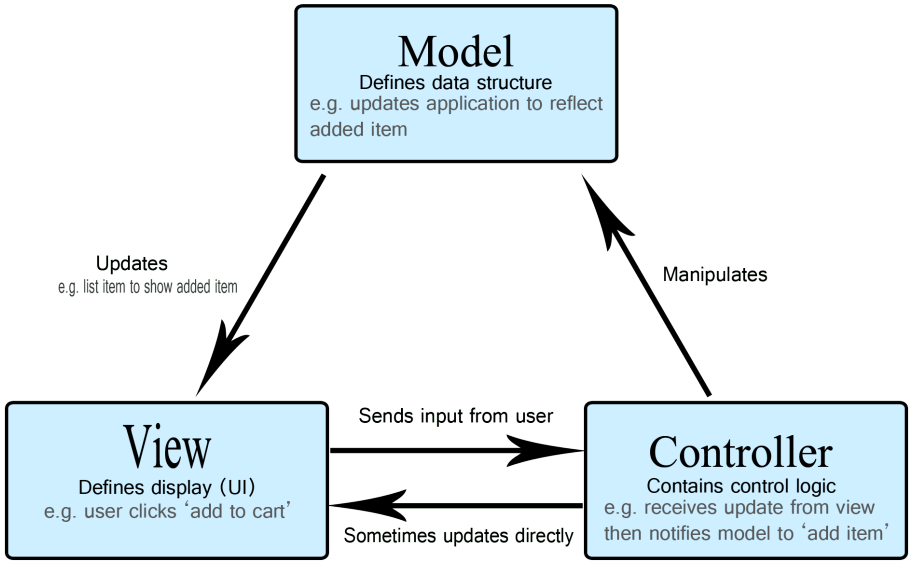
4.6.2 Controllers, Models, Validators, and State Management

Full Stack Application is also utilizes the MVC (Model-View-Controller) architecture pattern, which separates the application into three interconnected parts: models, views, and controllers.

- Models: is to be used to manage the data and the database schema of the application.
- Controllers: is to handle the business logic and user requests, and is to communicate with the models to perform CRUD (Create, Read, Update, Delete) operations on the data.
- Validators: is to be utilized to ensure that the data entered by users is valid and meets the required criteria.

This architecture pattern allows for greater modularity and maintainability of the application's code, as well as providing a clear separation of concerns. By using controllers, models, and validators, we can ensure that the application's logic and data management are handled in a consistent and efficient manner. Information provided by [23]

Figure shows the overview of the architecture designs 4.6.2



4.6.3 UI Framework

To enhance the user interface of the application, the use of a UI framework called Material UI. Material UI provides a range of pre-built components and styles that are designed according to Google's Material Design guidelines [24]. Using Material UI is to help to maintain a consistent and modern design across the application while reducing development time in terms of creating its own components from scratch.

4.6.4 Testing

To ensure the quality and reliability of the client-side code, it is important to thoroughly test it before deployment. Localhost testing can be an effective way to catch issues early and ensure a smoother deployment process. By running tests on the local machine, developers can quickly identify and fix bugs without the need for a dedicated testing environment. This approach can help to streamline the development process and improve the overall quality of the code. Also, use of the postman which is a tool to test API and requests was essential before making changes in the front and back end, this made sure that both sides received and displayed the request and responses.

4.7 Server-side architecture

4.7.1 API

To provide a RESTful API for the client side of the application, is to be using Express.js, a popular and web application framework for Node.js. Express.js provides a range of features for building web APIs, such as routing, middleware, and error handling in the application.

4.7.2 Database

To store and manage data for the application is to be using MongoDB Atlas, a popular and flexible database that is well-suited for data storage. MongoDB Atlas provides a range of features for managing data, such as it cloud-based, having high availability and security, as important as it is used for storing users' confidential information such as passwords, IDs, and roles.

4.7.3 Authentication

To authenticate users in the application, middleware such as Auth and SES are used. These middleware solutions provide a range of authentication strategies, such as local authentication, making it easier to implement user authentication and authorization in the application. By using these middleware solutions, the goal is to ensure the security and privacy of user data and provide a seamless and user-friendly authentication process for our users throughout the whole process.

4.7.4 Testing

To ensure the quality and reliability of the server-side code, is to be using localhost, Postman, launchDarkly, and Node.js. That provides for testing server-side code, such as asynchronous testing.Asynchronous testing helps to identify potential concurrency issues and ensures that the server can handle a high volume of requests without performance issues. Frameworks that test software systems, such as Mocha and Jest, offer pre-installed backing for non-simultaneous testing. This function permits builders to pen evaluation scenarios that process code not happening at the same instant whilst preventing blockage of a server's primary event loop. By using all of these frameworks and tools, such as Node.js, Express.js, MongoDB Atlas, Postman, and LaunchDarkly goal to develop a server-side architecture that is scalable, efficient, and secure. This is to help to ensure that the application can handle a large number of users and requests (POSTMAN), while also maintaining a high level of reliability and security.

4.8 Cloud infrastructure design and architecture and deployment (AWS)

YoungDeveloperHub utilizes cloud computing resources through the Infrastructure as a Service (IaaS) model, specifically with Amazon Web Services (AWS) to host the application. This enables the platform to offer students a wide range of learning materials, tools, etc. AWS's security and compliance features, such as IAM, ensure that the application is secure and compliant with industry standards. With IaaS, the focus can be on developing and deploying the application, without worrying about the underlying infrastructure, providing a reliable and scalable platform for students and developers across the world. The application is to be deployed on the AWS cloud platform, leveraging the following services:

4.8.1 EC2 Instances

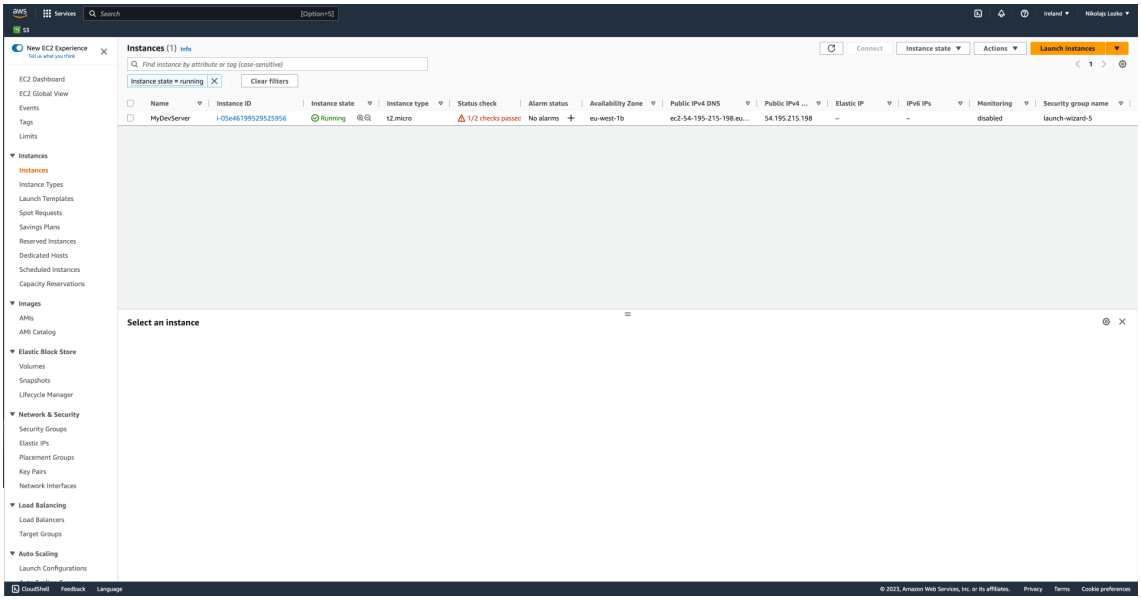
To achieve the goals of the YoungDeveloperHub application, the cloud infrastructure design and architecture, and deployment to make use of Amazon Elastic Compute Cloud (EC2) instance. EC2 provides scalable computing capacity in the Amazon Web Services (AWS) Cloud, allowing the app to launch a virtual server as needed based on demand. EC2 also eliminates the need to invest in hardware upfront which enables the development and deployment of applications much faster.

The app is to be deployed across EC2 instance to ensure high availability, and this instance can be scaled up or down based on demand. This is to enable the app to handle changes in requirements such as growing popularity. Additionally, security and networking can be easily configured through it.

By employing cloud services like EC2, the YoungDeveloperHub platform can provide a variety of instructional tools and content. The enormous storage and processing capacities provided by the cloud permit the platform to offer real-time data processing and analysis. Furthermore, the cloud-based architecture of the Hub allows for a safe and reliable platform for collaboration and connection between users. Within a secure environment, students or young developers can collaborate on posts by leaving comments and feedback, sharing their work, and exchanging feedback.

In summary, the utilization of EC2 instances in the cloud infrastructure design and architecture and deployment of the YoungDeveloperHub application enables the platform to offer students and young developers, a thorough and individualized learning experience, assisting them in becoming better students and or programmers. The scalability and accessibility of the cloud-based platform enable students from all across the world to have access to a safe and dependable learning environment that can help them develop technical and non-technical skills which can be learned outside working or university hours [25].

Example, in this figure 4.8.1 shown the use and practice, of running the AWS EC2 instance.



4.8.2 S3 Bucket

In addition to Amazon EC2 instance, the cloud infrastructure design and architecture and deployment of the YoungDeveloperHub application is to use of Amazon S3 buckets. Amazon S3 (Simple Storage Service) is a highly scalable and durable cloud storage service provided by AWS, which stores data as objects like images or pdf files are the many of examples.

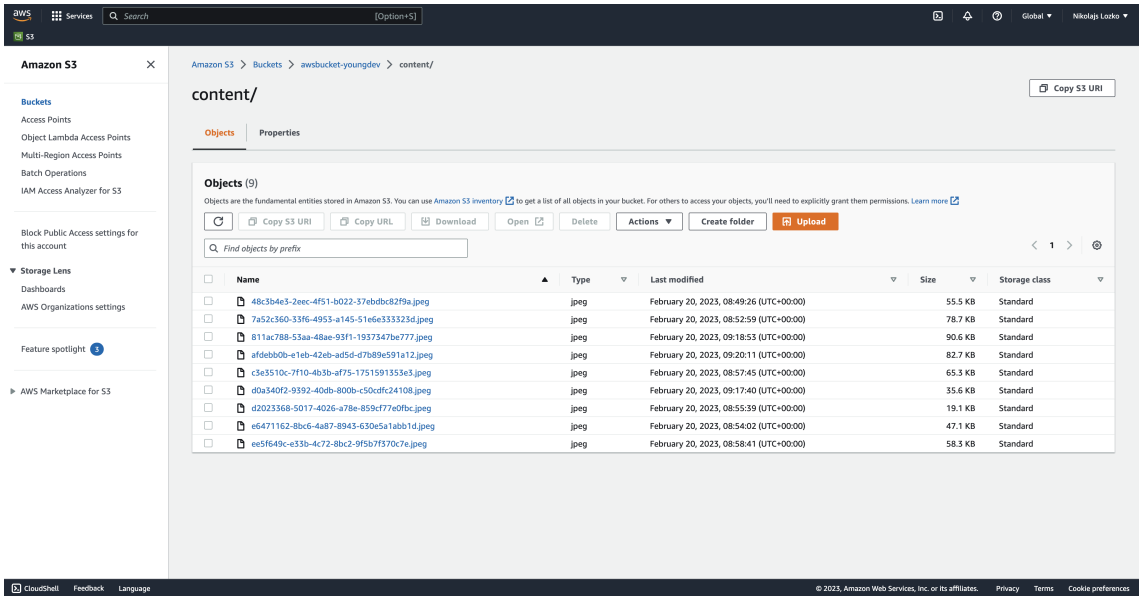
The YoungDeveloperHub application is to use S3 buckets to store and retrieve data related to user profiles, posts, and other learning materials. S3 buckets are designed and architecture for ”99.999999999 percent” durability and can automatically replicate data, providing high availability and durability for the stored data [26] provided by official documentation from Amazon Web Services.

The S3 buckets are to enable the application to store and access a large amount of data at a low cost. The data stored in S3 can be accessed and retrieved from anywhere, using simple web services. S3 buckets also offer built-in security features, such as encryption and access control, ensuring the confidentiality and integrity of the stored data.

The utilization of S3 buckets in the YoungDeveloperHub application is to allow for easy and secure storage and retrieval of user-generated content, lessons, and other learning materials. Additionally, the cloud-based storage service is to provide reliable and secure access to data, ensuring that students have access to the materials they need when they need them.

In summary, the utilization of S3 buckets in the cloud infrastructure design and architecture and deployment of the YoungDeveloperHub application is to provide a scalable and cost-effective solution for storing and accessing loads of different amounts of data securely. All this provides students and young developers with easy and secure access to the materials that they need to enhance their technical and non-technical skills. [26]

In this figure 4.8.2 shown the use and practice, of running the S3 bucket, storing some data as objects.



4.8.3 SES

In addition to Amazon EC2 instances and S3 buckets, the cloud infrastructure design and architecture and deployment of the YoungDeveloperHub application is to also leverage Amazon SES services. Amazon SES (Simple Email Service) is a cloud-based email sending and receiving service provided by AWS.

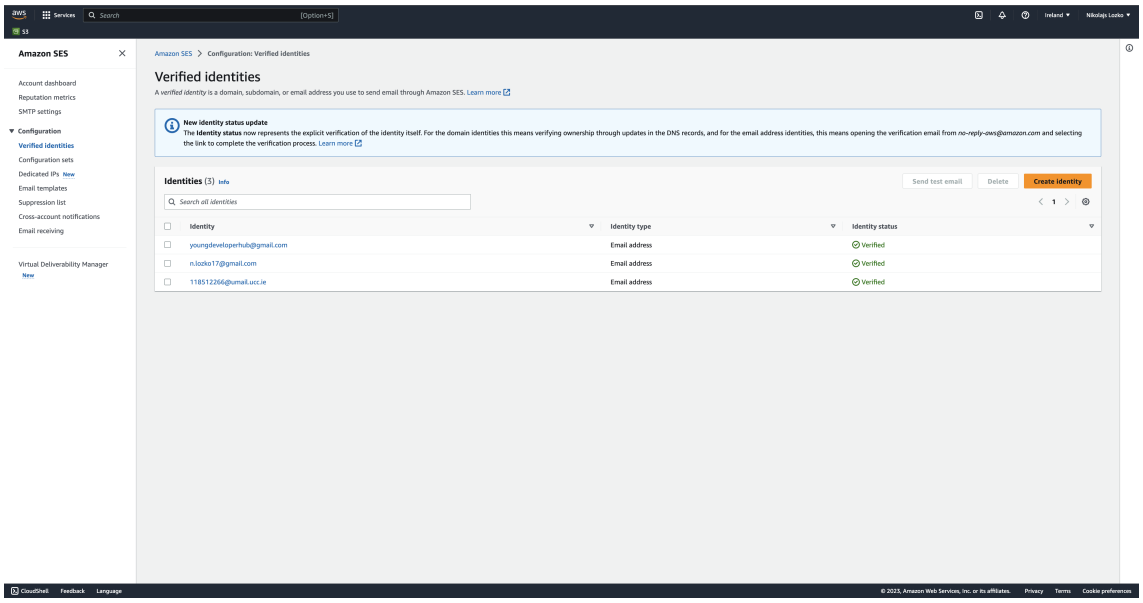
YoungDeveloperHub is to use Amazon SES to send email notifications to users of the YoungDeveloperPlatfrom, informing them about new posts, comments, or replies on their posts, which is reminders about their favorite topics or deadline for job or internship posts made. Amazon SES provides a highly scalable and reliable email service that ensures the delivery of email notifications to users.

The use of Amazon SES in the YoungDeveloperHub application is to provide a cost-effective and scalable solution for sending email notifications to users. Amazon SES can handle large volumes of email messages, providing the ability to send email notifications to thousands of users at once. The service also includes features such as email tracking, verification, and email reputation management, ensuring the high deliverability of email messages.

In addition, Amazon SES is highly flexible, providing multiple ways to integrate with the YoungDeveloperHub application, including SMTP (Simple Mail Transfer Protocol), API (Application Programming Interface), and the AWS SDK (Software Development Kit). Also email.js is to be implemented as it also uses integration with AWS SES for quicker development time and produces the same effect in such verifying or sending emails are the examples.

The utilization of Amazon SES in the YoungDeveloperHub application is to enable efficient communication with students, ensuring they receive timely and relevant notifications about their learning progress, new opportunities, and updates. Additionally, the cloud-based email service is to provide reliable and secure email delivery, ensuring that users receive the email messages they need them.

In summary, the utilization of Amazon SES in the cloud infrastructure design and architecture and deployment of the YoungDeveloperHub application is to provide a scalable, reliable, and cost-effective solution for sending email notifications to students. Amazon SES is to allow for efficient and timely communication with users and administrators as well, ensuring that they receive email messages when they want to sign up for updates on their favorite post or topic and or simply contact an administrator for new topics or give feedback. [27]



For this figure 4.8.3 shown the use verifying and sending emails to users.

4.8.4 Deployment Pipeline

A deployment pipeline is to be set up using LaunchDarkly to automate the deployment process and ensure that new changes to the app are deployed seamlessly. LaunchDarkly is a feature flagging and management platform that allows to manage and control the release of new features of the applications. By using feature flags, the app can release new features to a small subset of users and gradually roll out to larger audiences, reducing the risk of bugs or issues affecting all users.

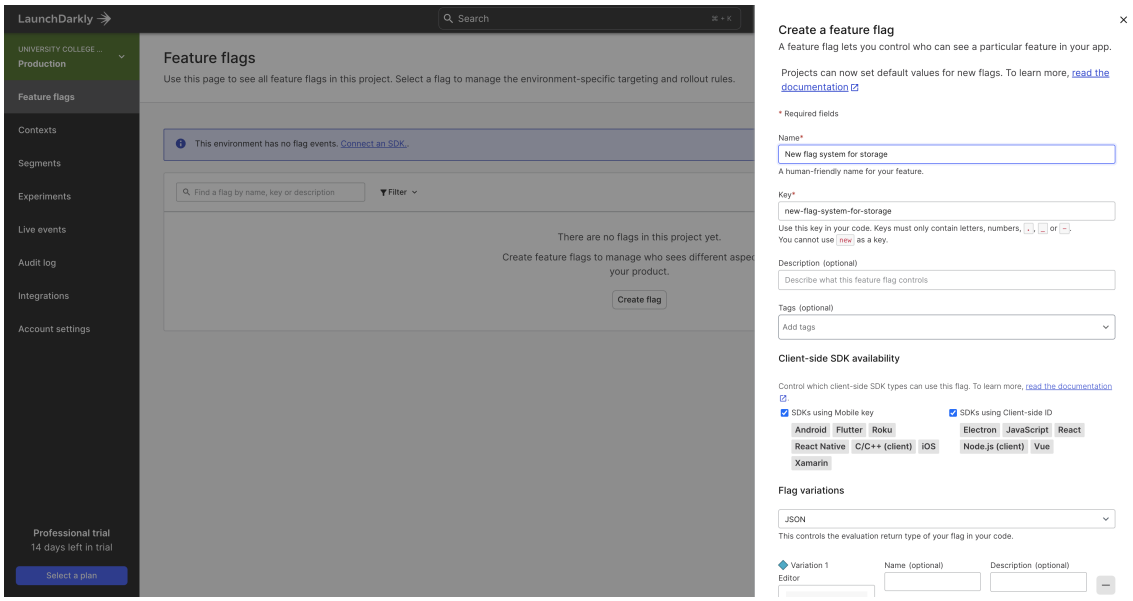
In the deployment pipeline, LaunchDarkly is to be used to manage feature flags and control the release of new features to the YoungDeveloperHub application such improving the handling of unused content that has a low popularity rank. The pipeline is to start with code changes being pushed to the source control repository (GitHub). Once the code has been successfully built and tested, LaunchDarkly is to be used to create a new feature flag to control the release of the new feature or changes made to the existing features.

Once the feature flag is set, the deployment process is to continue using LaunchDarkly to gradually release the new or update the current feature to selected users. This can be done gradually, with a small percentage of users receiving the new feature first, or all at once, depending on the requirements and risks involved in the process of deploying, then see how it will affect the app. Bitrise which an iOS developer tool CI/CD that does pretty much the same thing. That was used for testing for those devices as they have smaller screens [28].

Using LaunchDarkly in the deployment pipeline is to allow for a more controlled and streamlined release process, reducing the risk of issues or bugs affecting all users when they use the platfrom. By using feature flags, the app performance can gather feedback and make improvements to the new feature before releasing it to all users.

In summary, using LaunchDarkly in the deployment pipeline for the YoungDeveloperHub application is to provide a more controlled and streamlined release process. LaunchDarkly's feature flagging and management platform is to enable developers to manage the release of new features and changes, while gradually releasing the new features to selected users at one at the time [29].

4.8.5 LaunchDarkly flagging system



Referring to this figure, use of flag system was tried for CI/CD for the app 4.8.5.

4.9 Conclusion

In conclusion, building a robust and scalable application requires careful consideration of the architecture and technologies used. In this proposal, it is discussed various AWS services such as EC2, S3, and SES, as well as LaunchDarkly, that can be utilized to build and deploy a modern web application such as YoungDeveloperHub. The proposed architecture utilizes EC2 instances and cloud database to provide a scalable and reliable infrastructure for hosting the application. S3 can be used for storing static assets, and SES with email.js can be used for sending emails to users. Additionally, LaunchDarkly can be used to manage feature flags and control the release of new features, which is to help to reduce the risk of bugs or issues affecting all users. Overall, the proposed architecture and the design is to provide a solid foundation for building and deploying a modern web application, while allowing for flexibility and scalability as the application grows. The next chapter is to dig deeper into the timeline and the development, implementation, and testing of the proposed designs and architecture, discussing the various features and components and how they work together to provide a seamless user experience.

Chapter 5

Implementation and Testing

In this chapter, the implementation and testing process for the application is examined, it involves building and deploying the application on the cloud infrastructure, and testing its functionality, security, and performance. To follow an agile development process [30], using sprints, and continuous integration and deployment, to ensure that the application meets the requirements and expectations of its users.

5.1 Implementation process

5.1.1 Sprint planning and backlog

At the start of each sprint, is to hold a sprint planning meeting to prioritize the tasks in the project backlog and ensure that the most crucial tasks are completed punctually and in the correct order. The backlog will be managed using TEAMWORK Project and Management Software and will be regularly revisited and updated throughout the development process.

5.1.2 Development process

Following an iterative development process, using sprints that are typically estimated between 2 to 4 is weeks long, depending on the complexity of the feature. During each sprint, is to will develop and test new features, using modern UI design and backend technologies, such as React.js, MongoDB, and Node.js. Below shown updated dates of the sprints.

- 27th September - Proposal, Designs and research into the technologies for the app
- 18th October - Login and Register System for users
- 10th November - A “Post” feature to share knowledge by students within the application
- 29th November - Administration feature for deleting irrelevant or outdated content

- 8th December - Dashboard for user and admin for view their content and other exclusive features to each roles
- 10th January - Email verifications, Authentications and notifcations
- 29th January - More quality life changes to UI, popularity counter, comments
- 6th February - Coding tool with user requests to Admin
- 20th February -Use AWS cloud services to host the app for use worldwide
- 15th March - More testing before deploying application on AWS cloud service
- 2nd April - Refinements and bug fixing if any found

5.1.3 Continuous integration and deployment

The use of continuous integration and deployment (CI/CD) to automate the development, testing, and deployment of the application. This will involve setting up a pipeline that will automatically build and test the application code, and then deploy it to the cloud infrastructure.

5.1.4 Testing process

Conducting rigorous testing of the application, using a combination of automated testing and manual testing. Automated testing will involve using tools such as LaunchDarkly and Postman to test the functionality of the application, while manual testing which involves testing the user interface and user experience. Through desktop and mobile devices, testing the security and performance of the application on localhost.

5.1.5 Refinements and bug fixing

Upon completion of the testing process, it is vital to perform tests such as debugging the application in order to detect and single out any persisting glitches or bugs. The application's functionality, security, and user experience also is important to be tested, so that security is not compromised by either the bug or tests.

5.2 Code snippets and examples of key features and functionality

In this section, some code snippets and examples of key features and functionality are included in the application. These features and functionality are designed to help young students and developers to share their experience, knowledge, and skills with each other, and include features such as internships, notes, and CV advice.

5.2.1 Login and Register System

18th October - Login and Register System for users

Here's an example of how the login and register system works in the application. Users can create an account by entering their email address and a secure password, which is then encrypted and stored in the database. Once registered, users can log in to their accounts by entering their email addresses and password. As shown in figure C.1

5.2.2 Post Feature

10th November - A "Post" feature to share knowledge by students within the application

The post feature allows users to share their knowledge and expertise with other users by creating a post. Users can include text, images, and links in their posts, and can choose to make their posts public. As shown in figure C.3

5.2.3 Administration Feature

29th November - Administration feature for deleting irrelevant or outdated content

The administration feature allows administrators to delete irrelevant or outdated content from the application. This helps to keep the application up-to-date and relevant to its users. As shown in figure C.5

5.2.4 Dashboard, Profiles and Updates

8th December - Dashboard for user and admin to view their profile and other exclusive features to each roles

The dashboard provides users with an overview of the most recent posts and discussions, alllows access to their own profile and settings. This makes it easy for users to stay up-to-date with the latest information and connect with other users in the community, which creates a sense of social circle for the users.As shown in figure C.8

5.2.5 Email Notifications

10th January - Email verification, authentications and notifications

The application sends email notifications to users when their posts receive comments or likes, or when they receive a message from another user. This helps to keep users engaged with the application and encourages them to stay connected with other users in the community. As shown in figure C.11

5.2.6 UI improvements and additional features

29th January - More quality life changes to UI, popularity counter, comments

It's important to continuously improve the user interface (UI) and add new features to keep the application relevant and engaging for users. This can involve gathering feedback from users and making adjustments to the design to enhance usability and user experience. Additional features can be added based on user needs or emerging trends in the industry, such as integration with new technologies or services. By constantly iterating and improving the application, it can continue to meet the changing needs and expectations of its users. As shown in figure C.16

5.2.7 API integration with more additional features such comments etc.

6th February - Coding tool with user requests to Admin

One potential additional feature for YoungDeveloperHub could be the integration of a commenting system, allowing students to provide feedback and discuss posts with each other. This could be achieved through integration with an existing commenting service, or by developing a custom commenting system within the application. This would enhance the social and collaborative aspects of the platform, encouraging discussion and peer-to-peer learning. As shown in C.21

5.2.8 Cloud services tweaks, tunes, and deployment

20th February to 15th March - Use AWS cloud services to host the app for use worldwide and More testing before deploying the application on AWS cloud service

May need to configure load balancing to distribute traffic evenly across the server, or implement caching to reduce the load on the server and improve response times [31]. Additionally, the integration with other cloud services, such as a database (MangoDB Atlas) or a messaging service (email.js). This can provide additional features like persistent data storage, real-time communication, or background job processing. For example, we could use a cloud-based database to store user information and comments and use a messaging service to notify users of new posts or comments.

Deployment to the cloud also requires careful consideration of security. Need to understand that the application is protected from common threats like SQL injection, cross-site scripting (XSS), and denial-of-service (DoS) attacks [32]. This can be achieved by implementing security best practices like input validation, output encoding, and rate limiting. As shown in C.25

Mobile Responsiveness for iPhone vs Android also is developed in the background

As shown in figure C.23

5.3 Testing methodology and results

In order to ensure the quality and reliability of the application, a thorough testing methodology throughout the development process. The testing methodology consisted of the following steps:

1. **Unit testing:** is to perform unit testing on each component of the application to ensure that it functioned as expected and to catch any bugs or errors early on in the development process. For example, the requests sent by the user are received by the server.

2. **Integration testing:** is to perform integration testing to ensure that all components of the application worked together seamlessly and to identify any potential issues or conflicts. For example, the information from the server-side correctly displays on the client-side, and both communicate without issues.

3. **Goal testing:** is to perform system testing to ensure that the application met all of the requirements and specifications outlined in the project brief.

4. **Acceptance testing:** is to perform testing with quality testers [E] to ensure that the application met the needs and expectations of the target audience.

The acceptance testing phase yielded positive results at first, with quality tester student reporting that they found the application to be user-friendly E, intuitive, and helpful for sharing knowledge and collaborating with other users. The feedback from the testers helped to make additional improvements and refinements to the application before its final release on EC2.

To conclude this part, the testing methodology is essential in ensuring the quality and reliability of the application, and having the acceptance testing phase helped us to ensure that the application met the needs and expectations of the target audience.

5.4 Analysis of tests of load and response

Throughout the testing phase of the project, the use of manual and automated testing methods was to collect some data and analyze the test results. The analysis of test results and performance metrics included the following:

- **Load testing:** The application has to handle a large number of users and posts without experiencing any significant performance issues [33].

- **Response time:** The application needs to be fast and responsive for users. the response time measurements show that the application consistently responded to user requests [**?**].

User that is given early build access, helped to provide feedback on the load testing, response times, and bugs that had been shown in the interview with them. Thus, demonstrated that the application could handle a large number of users and posts without any significant performance issues but had issues on the front-end side, probably from using too many libraries which resulted in some issues in rendering. Also, the response time from their perspective was adequate, as it loaded as any website with modern standards. The application is fast and responsive for users but

indicated some of the interface libraries did not hinder the performance during the tests but had issues with some visual bugs.

The analysis of test results and performance metrics confirmed that the application was able to meet the needs and expectations of the target audience. It provided valuable insights that are to be used to make additional improvements and refinements to the application in the future ahead.

5.5 Optimization and performance tuning

During the development and testing phases of the project, is to identify several areas where it can be optimized and tune the application for better performance and efficiency. The optimization and performance tuning efforts focused on the following areas:

- **Database queries:** is to optimize the database queries to ensure that they are stored fast and efficiently. Use of indexes, caching, and other techniques to reduce the time it took for the application to retrieve and update data from the database.
- **Code optimization:** To optimize the code to ensure that it was clean, maintainable, and scalable. The best practices and design patterns used from [34], reduce the complexity of the code and improve its performance and maintainability.

These small optimization and performance-tuning efforts were small steps to the bigger success of the project. Throughout the development process, making strategic decisions with trade-offs in mind, helped in order to optimize the application's performance and ensure its functionality. Having integrated the application with a variety of other systems and tools in order to provide a more comprehensive and valuable user experience for the community and consistent testing to make sure there are no bigger problems that could arise.

Now that the application has been developed and tested, it is time to evaluate the results and assess the effectiveness of the application in meeting the needs of the users. The next following chapter's discussion is on the evaluation and results which will provide a detailed analysis of the project's success and areas for improvement.

Chapter 6

Evaluation and Results

The final product of this project is collaborative learning and knowledge-sharing the application that is specifically designed for students and developers. The application includes key features such as real-time posting features made by the students or developers through cloud services SES, user profiles, and a popularity counter for the most visited posts. It is also optimized to handle loads of data such as storing images due to the use of cloud service S3 buckets. The application is deployed to a cloud platform using EC2, making it easily scalable and accessible to users around the world when they are not in a laboratory, class, or working.

Focus on design which is learning from each other and having the support of the cloud services. These services made it possible to create the important features listed 3.5.1 for the users. The technical implementation and development process of the project was designed to ensure that the application was able to handle many users and the content they post.

The user feedback and evaluation phase played a vital role in gaining insight into the creation of the project which was questions from graduates [D] and quality tester feedback [E]. This helped to identify areas for future development and improvement. When considering expanding this idea of the app, having multiple communities that log on and view their specific area of degree or expertise is the future goal in expansion to a wider audience.

6.1 The success and effectiveness of the project

The success and effectiveness of this project can be evaluated based on a variety of factors, including the achievement of the developer and research objectives from Chapter 1, and acceptance criteria with metrics from Chapter 3.

Having successfully developed and implemented a full-stack application with a modern user interface with integrated cloud services that are provided by Amazon Web Services. The application meets the requirements outlined by developer objectives, including a dedicated system for saving users' credentials and posting content, as well as a range of cloud services for verification, emails, and hosting the app.

Secondly, user feedback has been overwhelmingly positive but there are a few sug-

gestions such as better search functionality and more collaboration e.g user create their own subgroup with their favorite topics.

Finally, the acceptance criteria and metrics indicate that the application is functioning well and meeting the needs of the users e.g handling many users at each time.

Overall, this project has been in some way successful and effective in achieving its goals basic goals but there a lot of room for exploring new features and improving or expanding the architecture to use more cloud services or use more external services. If the application can continue to grow and develop continuously, it can be a valuable platform for the community.

6.2 Comparison with the initial goals and requirements

Analyzing the developer objectives it has achieved most of the goals that were outlined, but has a few bugs with some minor issues with page refreshes. Nevertheless, in general on the technical or developer level it has been mostly successful. Reflecting on research objectives, there are a lot of things that can be done to make the app better in solving the issues, such how effective this app is in promoting knowledge sharing and collaboration to the targeted audience. In Chapter 1, it was mentioned about other platforms, and how convoluted they are with their audience or content. If YoungDeveloperHub, can keep getting support and more features integrated which will be discussed in the next Chapter 7, it will solve these research objectives. Also exploring more cloud technologies will make this platform not only scalable, and maintainable but also very cost-effective for development which eventually reaches and accomplish these objectives and might go beyond that.

6.3 User feedback and evaluation

User feedback and evaluation are essential components in assessing the effectiveness of the application and its ability to meet the needs of the target audience. In order to gather user feedback, the interview was conducted with a graduate, and the quality tester was to collect information on the user's experiences with the application and give feedback to the developer.

The results of the user feedback and evaluation are positive, with users expressing their satisfaction with the platform it has provided for them to share their experiences and knowledge. Users have reported that the application is easy to use and provides a valuable resource for academic and professional pursuits.

Overall, the user feedback and evaluation have confirmed some of the effectiveness and value of the application to the target audience, and there is an opportunity to have this platform be a resource for young students and developers who pursuing their degrees and roles as developers. The ability to share experiences and knowledge in a semi-controlled environment can bring loads of users and keep the app focused on providing content to the core user base.

6.4 Identification of strengths and weaknesses

Through developing and testing the application, it had several strengths and weaknesses.

Some of the strengths of the application include its modern user interface, efficient cloud infrastructure, and reliable architecture. One of the key weaknesses of the application is the need for more features and bug fixes that were mentioned previously. While it has achieved uptime rates on the cloud and fast page load times, but there is always room for improvement in these and other areas. Additionally, the application is currently limited to a specific community of young students and developers, and there is an opportunity for us to expand the user base and reach a wider audience. Overall, it believes that the strengths of the application outweigh its weaknesses for now but in the long term, there needs to be more support and new features added to keep the user base engaged.

Moving forward, plan to continue developing and refining the application to further enhance its value and functionality for the YoungDeveloper's users. This is explored in the next Chapter 7, the potential new features and integrations.

Chapter 7

Conclusion and Future Work

In this final chapter, is to summarize the key findings of the project and reflect on the successes and challenges that were encountered throughout the development process. Explore potential future directions and areas of further research, and outline the new future goals, to continue developing and refining the YoungDeveloperHub application.

Throughout the project, the use of modern UI features is explored, efficient cloud infrastructure, and a range of functionalities to create this platform.

Moving forward, the project is committed to continuing to develop the application to meet the evolving needs of the users and provide a valuable resource for students and developers who are either pursuing their degrees or careers as developers. The potential new features and integrations can expand the user base to reach a wider audience in multiple communities.

Overall, the project has been a valuable learning experience working as a full-stack developer, and are proud of the work that has been accomplished. Hoping that the application will continue to provide value to its users and serve as a valuable resource for young students and developers.

7.1 Future directions and potential improvements

Moving forward, there are several potential directions and improvements that it can explore to enhance the functionality and value of the application from the feedback of the interview [D] and quality tester [E].

One potential direction is to explore integration with external APIs, such as the Canvas API, to further expand the range of functionalities and resources that the application provides. This integration could enable users to access and interact with their Canvas accounts directly through the application, allowing for seamless integration with their academic studies.

Additionally, it can further optimize and tune the performance of the application to ensure its reliability and scalability. This could involve exploring additional cloud services, implementing caching mechanisms, and optimizing the database queries to improve load times are one of the examples.

This can expand the user base and reach a wider audience. This could involve expanding the marketing efforts, partnering with academic institutions to promote the application, and exploring ways to attract users from outside of the target community.

Overall, there are many potential improvements and directions that it can explore to further enhance the value and functionality of the application, and it is committed to continuing to develop as long it keeps on track with the originally targeted users.

The application has successfully achieved its intended purpose of as a valuable resource for young students and developers to share their experiences and knowledge. Now let's explore things that can come about:

1. **Integration with Canvas API:** To integrate the application with the Canvas API, which will allow Canvas-based online learning management systems, further expanding the reach and functionality of the platform and this can result in better interaction and communication with lecturers.
2. **Advanced search functionality:** Enhance the search functionality of the application, allowing users to more easily find relevant information or resources with filter pills that contain specific keywords, categories, etc.
3. **AI recommended system:** Following trends about artificial intelligence as it has been taken off and most applications use it. Introducing bots or learning models to adapt to user preferences and what their profile is in interest by learning their views or interactions with different posts as one of the examples. This can keep users more engaged and expand the life of the app.
4. **Improved user experience:** Keep continuing to improve the user experience of the application, incorporating user feedback, to create a more intuitive and user-friendly platform that meets the evolving needs of users and modern trends.
5. **Reward system:** For the most dedicated users who spent most of the time either helping or posting to improve the app. A rank or reward should be a feature to be considered to show appreciation for the commitment.

Looking to the future, there are several areas in which the application could be improved or be taken into something bigger. The solid architecture in place, having this new future features such integration with Canvas API, machine learning, enhanced search functionality, and also maybe adding more improved security features can make this platform worth the time and investment.

Overall, we are proud of the success and effectiveness of our project and believe that it represents a valuable resource for young students and developers around the world. We look forward to further developing and optimizing our platform in the years to come, and continuing to provide a valuable service to our users.

Appendix A

A.1 Code Repository

The code repository for the Hub platform is available on GitHub (username: devnikolz project: FYP-Cloud). The repository contains the code used to develop the platform, including the front-end and back-end code, as well as the dependencies.

The front-end code is built using React, a popular JavaScript library for building user interfaces. The back-end code is built using Node.js and Express, for the server-side. The database used for the platform is MongoDB Atlas, a cloud-based database service.

The repository contains separate folders for each component of the platform. Each folder contains the necessary files and code for that component. To access the code repository, simply follow the information provided, to download it. From there, you can clone the repository or download the code directly.

Appendix B

B.1 Original Designs

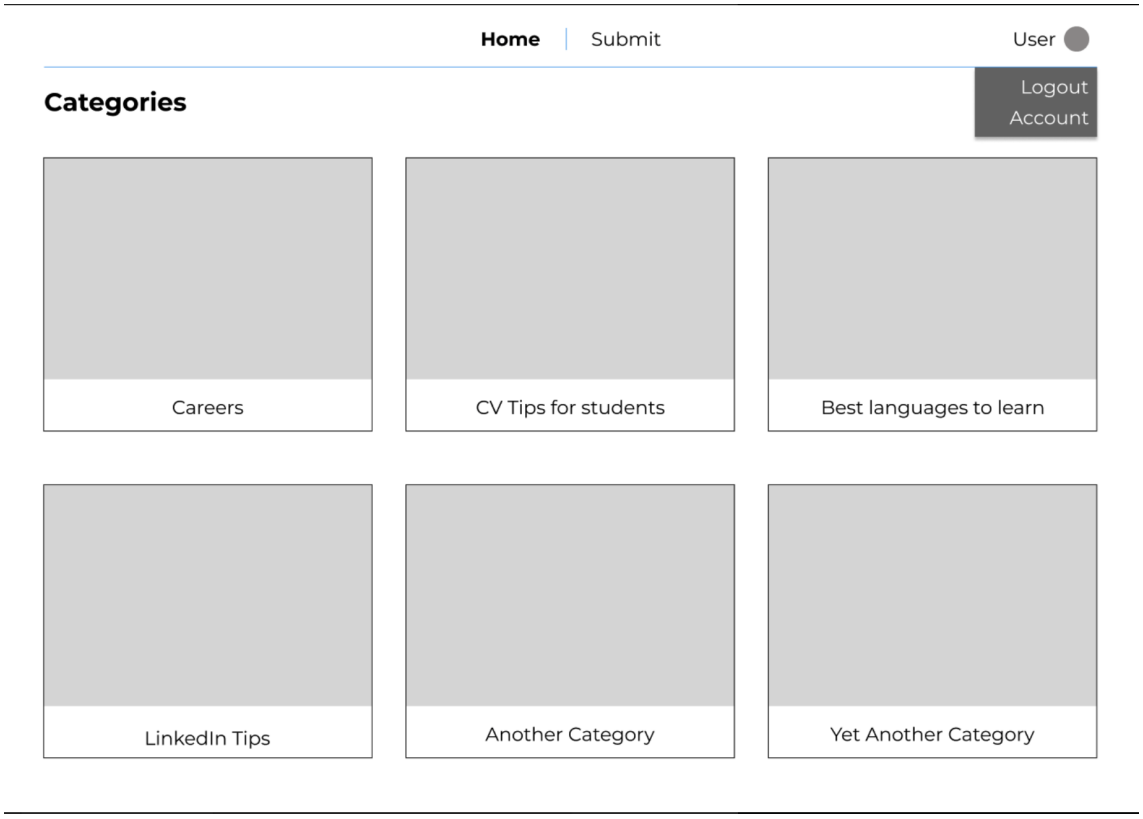


Figure B.1: Original design of homepage

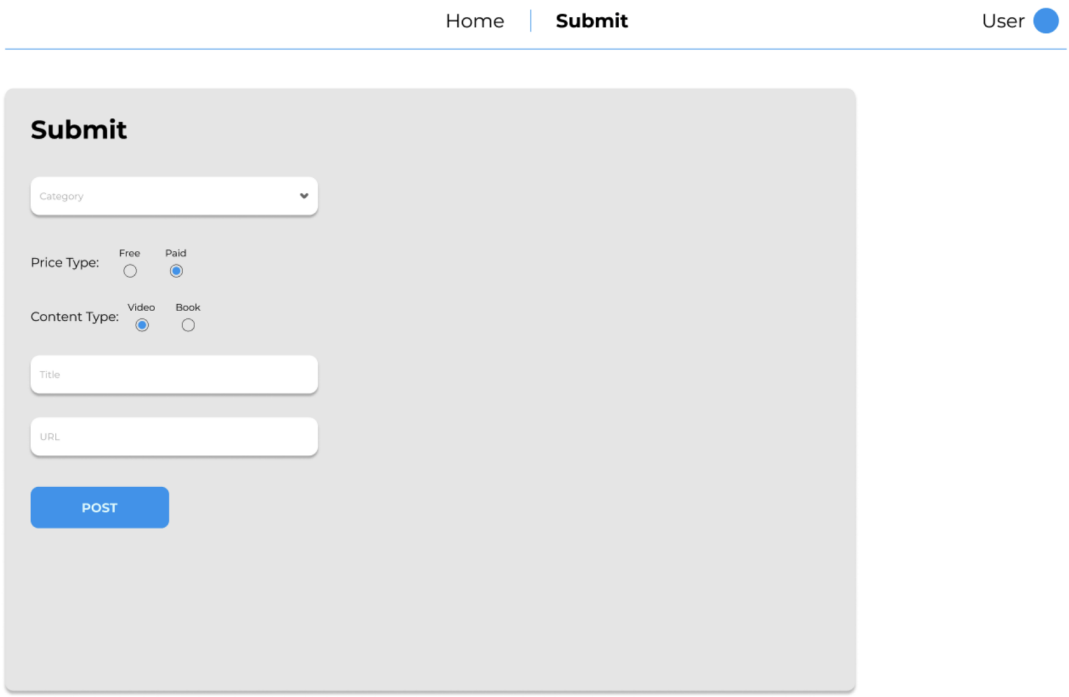


Figure B.2: Original design of submission Page

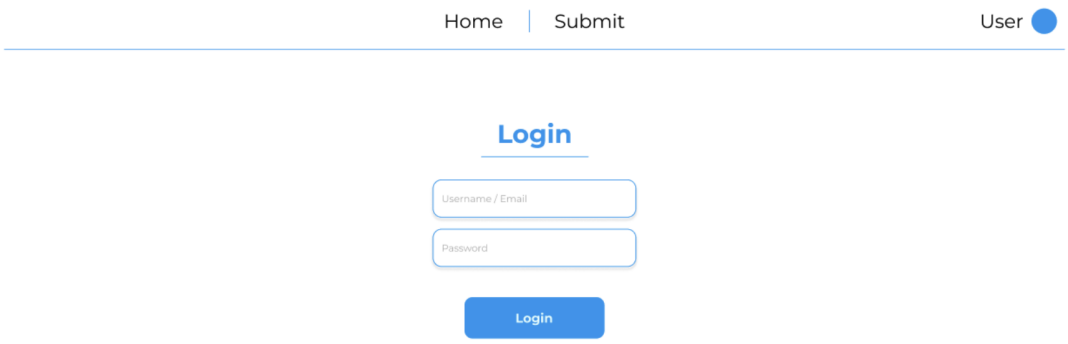


Figure B.3: Original design of login page

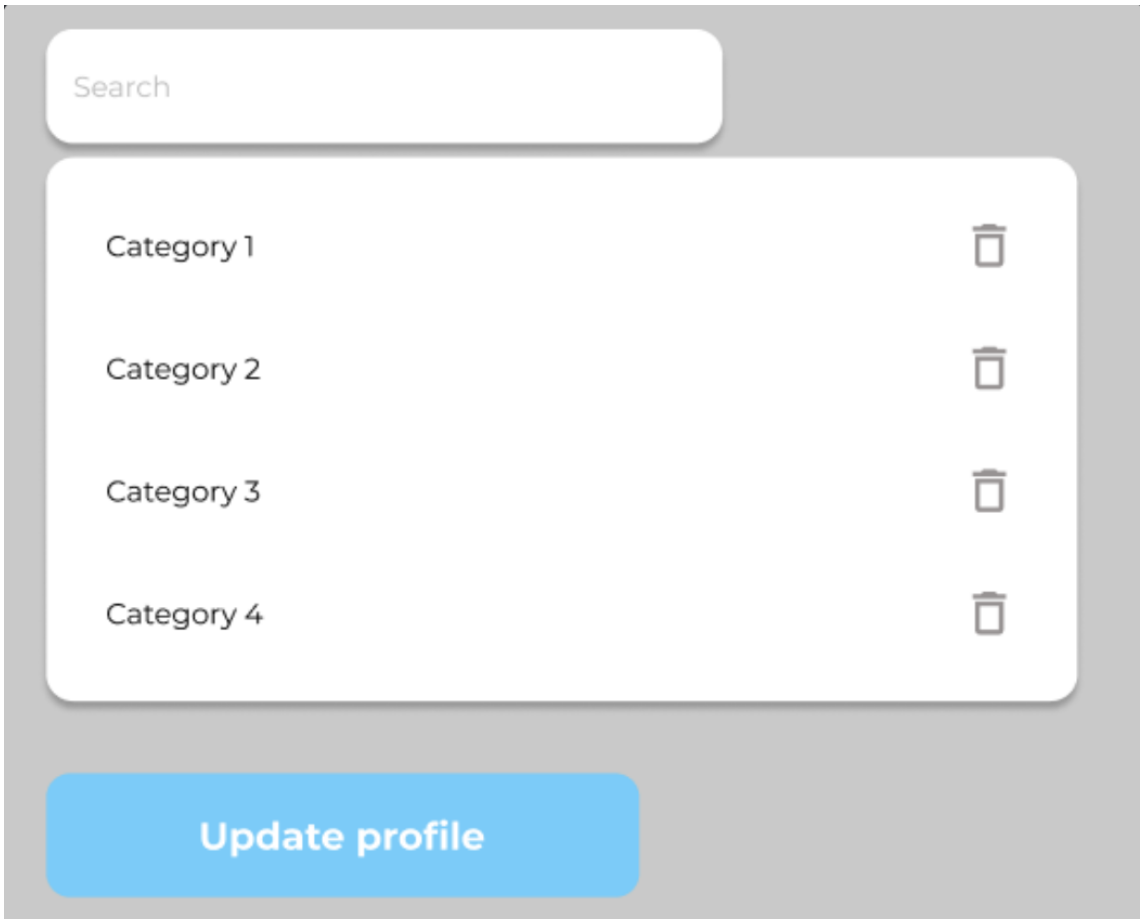


Figure B.4: Original design of selecting page

Appendix C

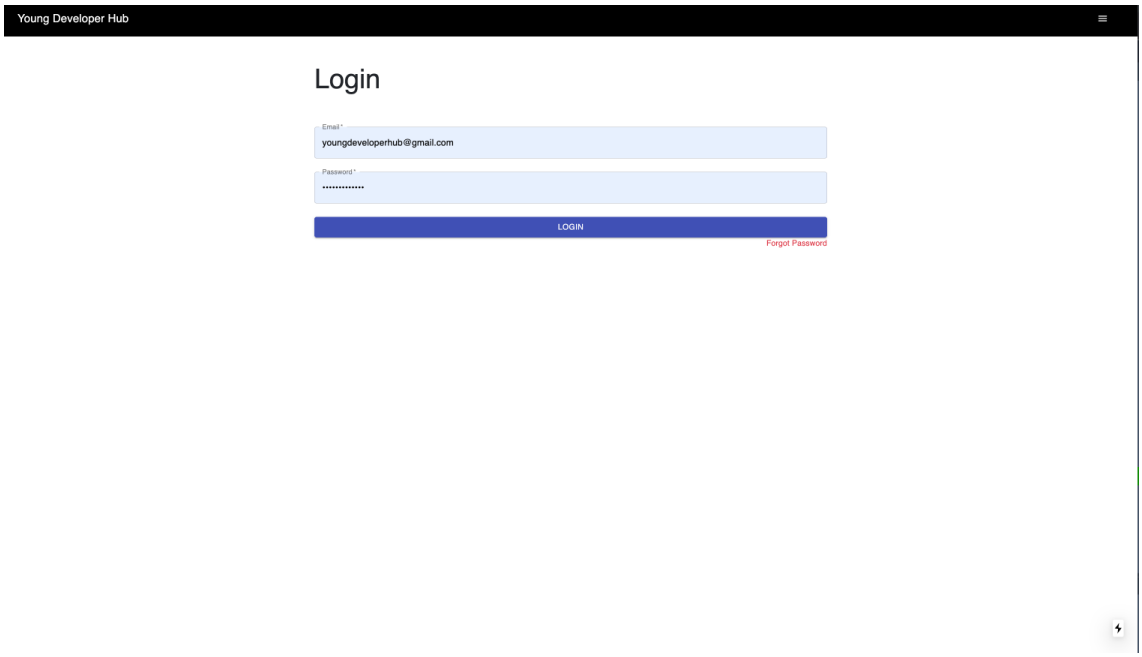


Figure C.1: Screenshot of login feature

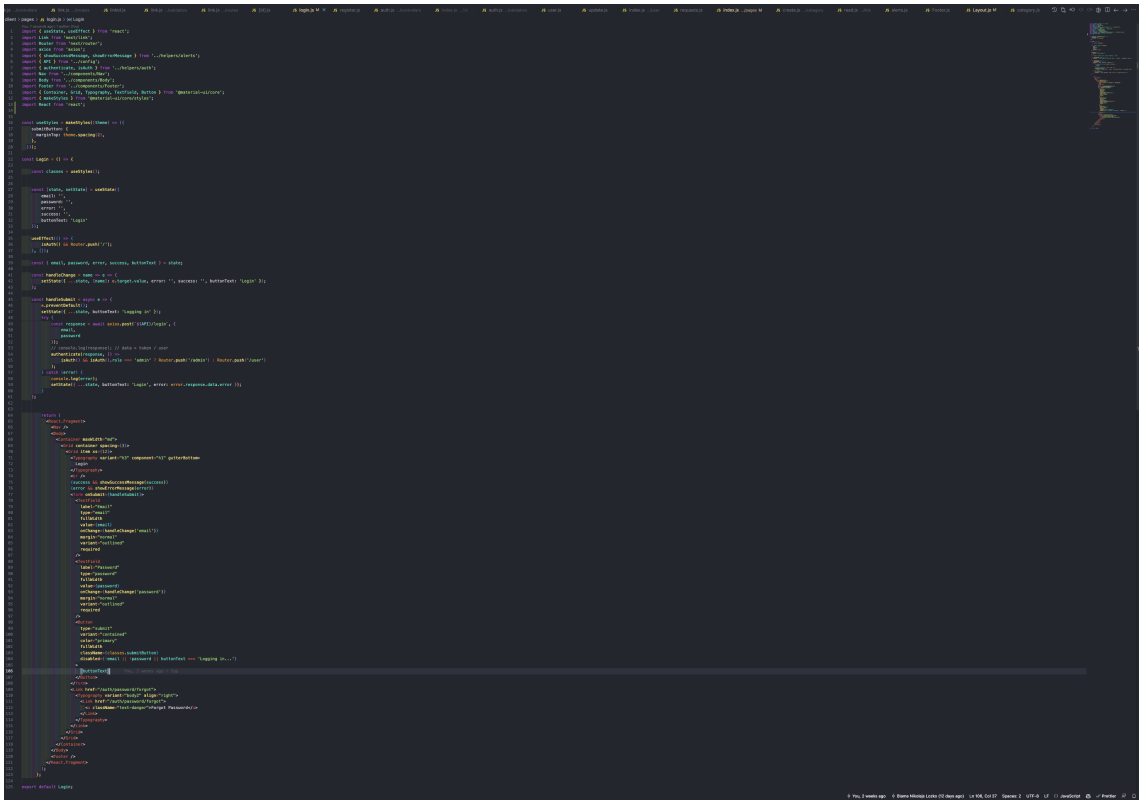


Figure C.2: Login feature code snippet

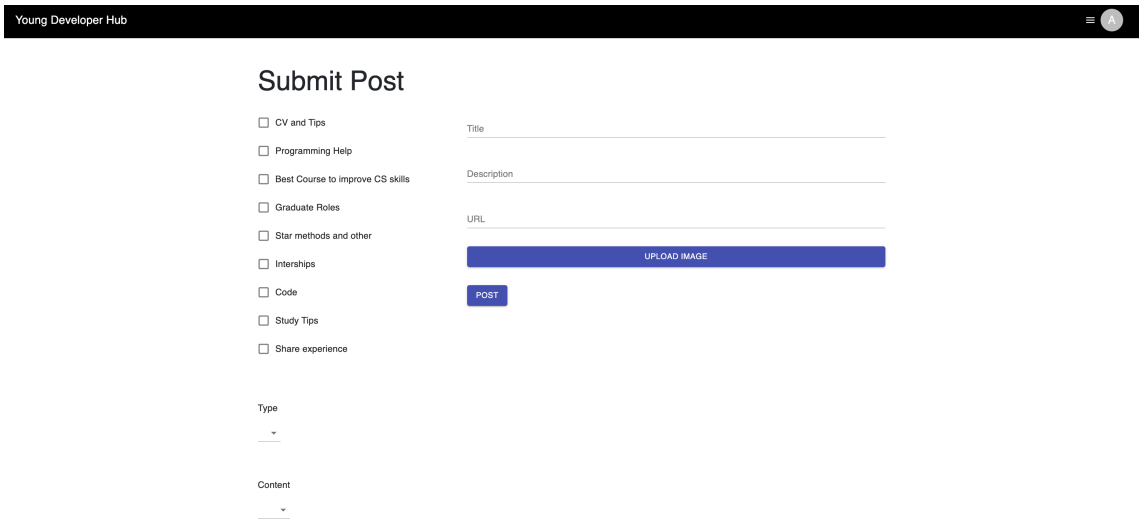


Figure C.3: Screenshot of post feature

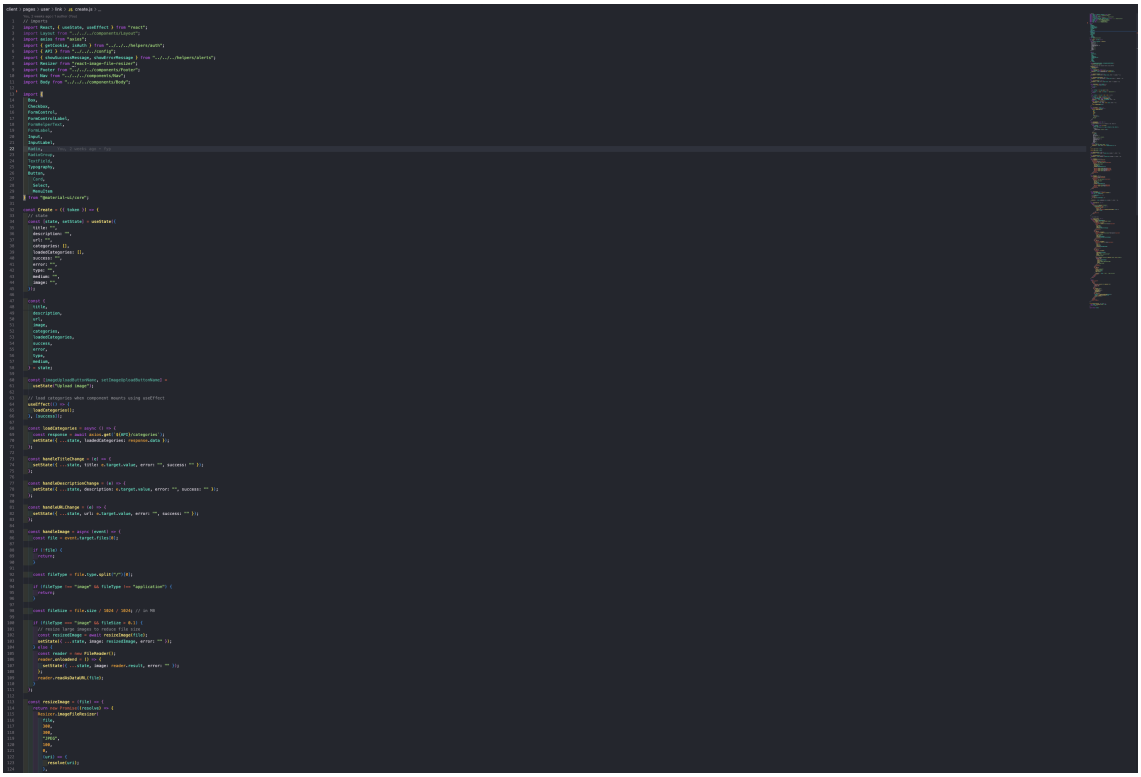


Figure C.4: Post feature code snippet

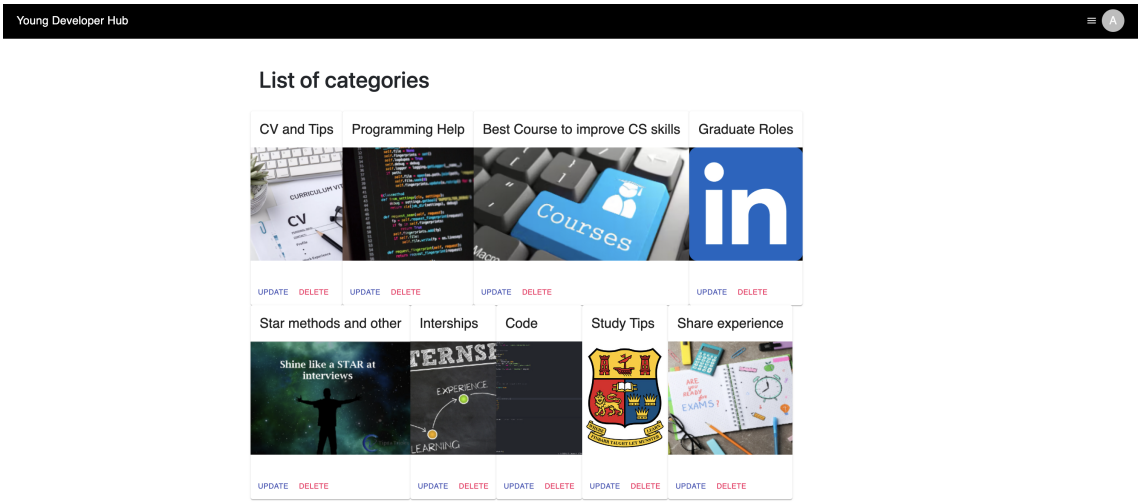


Figure C.5: Dashboard of admin ”update” and ”delete” feature

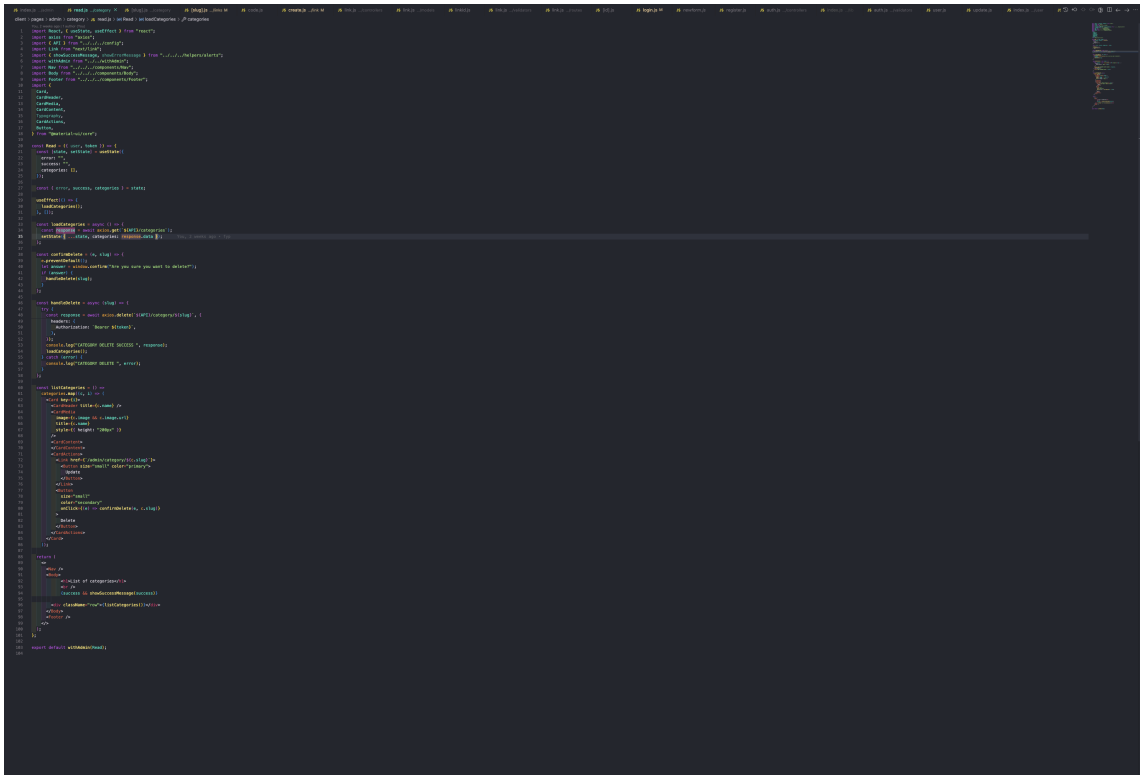


Figure C.6: Code snippet of this dashboard above

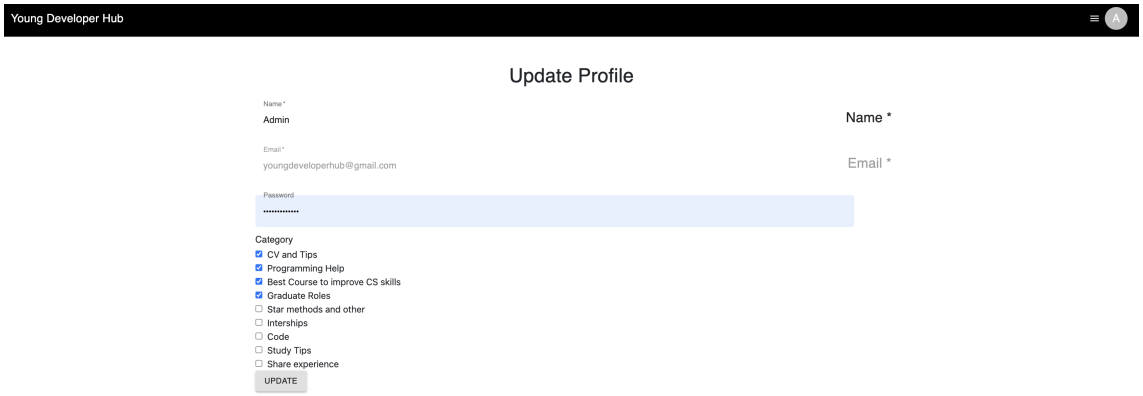


Figure C.7: Profile update for users

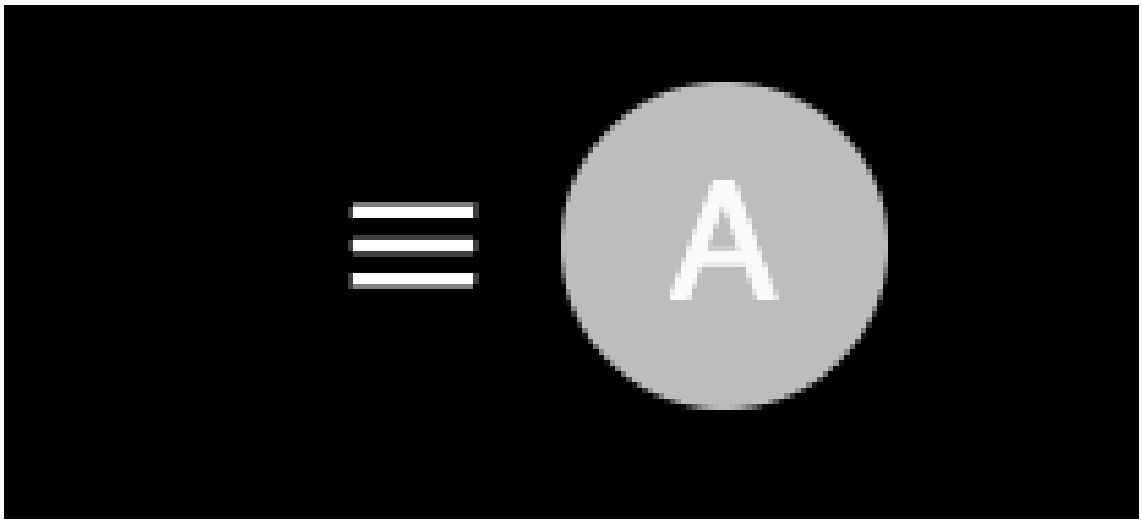


Figure C.8: Dynamic letter name change

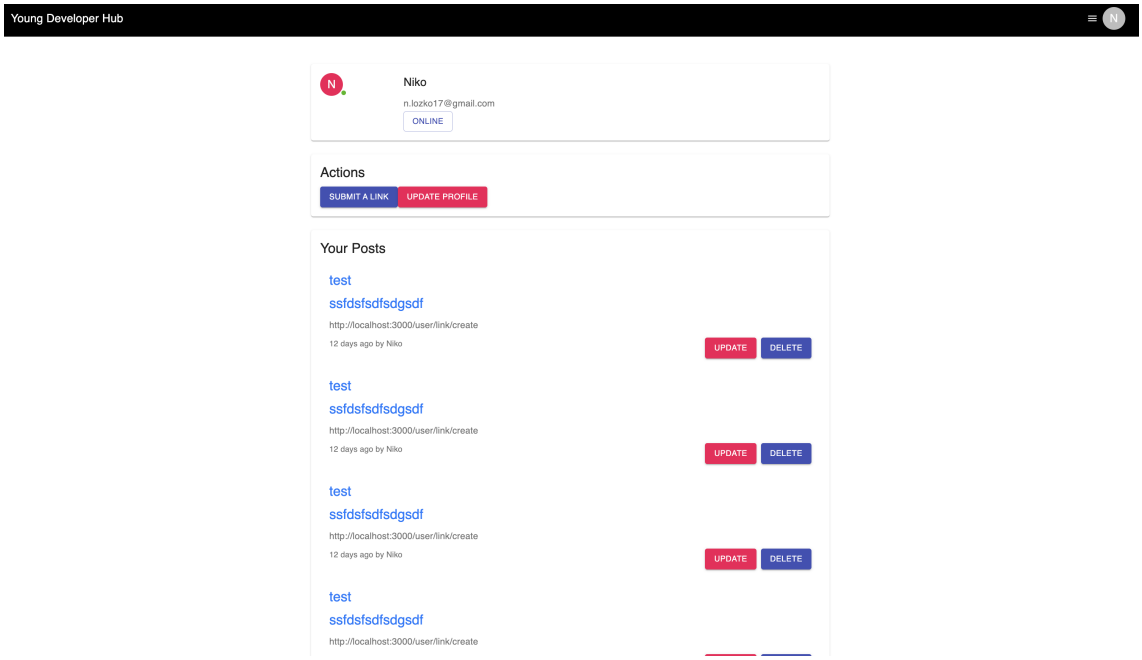


Figure C.9: User view their posts feature

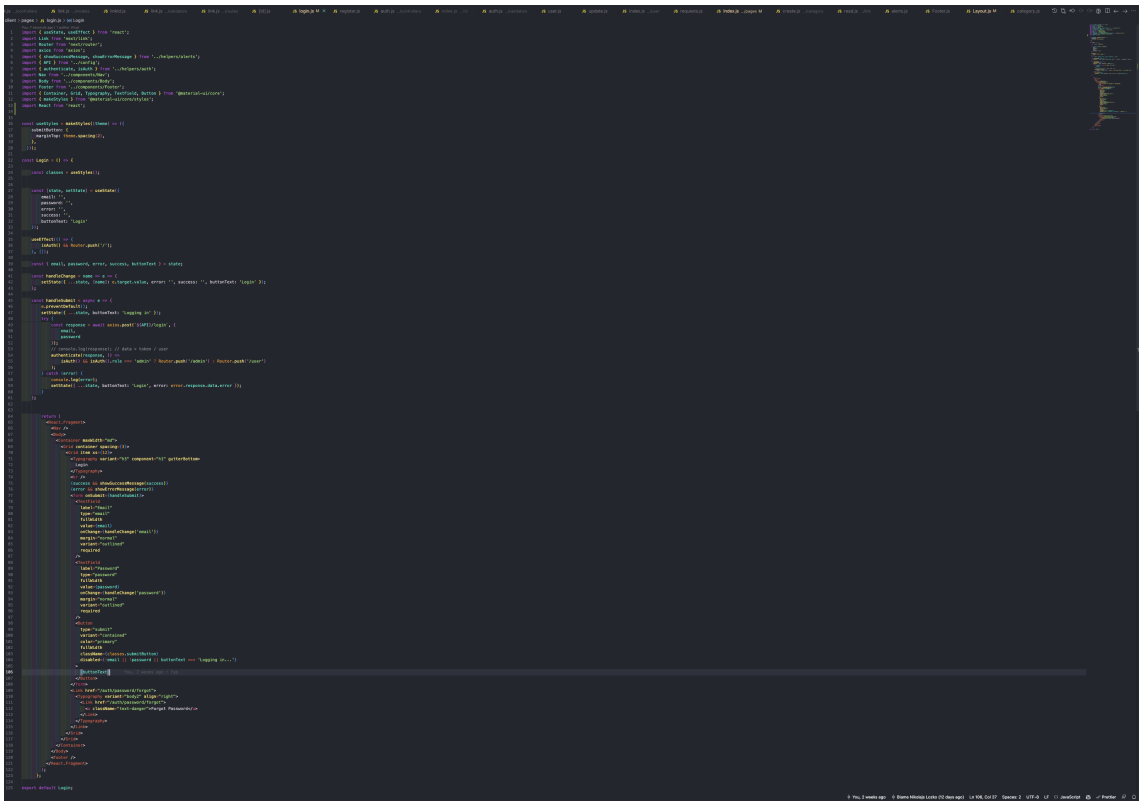


Figure C.10: Code snippet of the two figures above

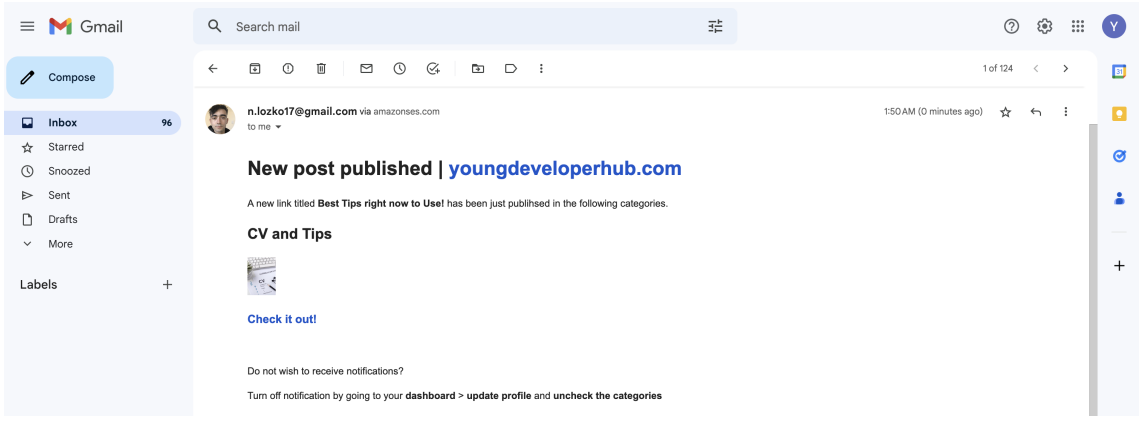


Figure C.11: SES email notification system for new ”post alert”

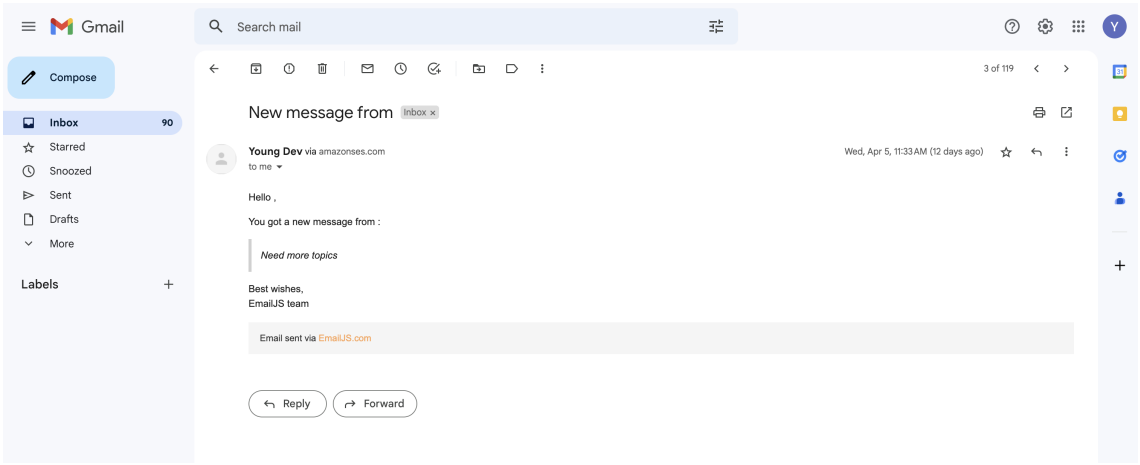


Figure C.12: Email.js message to Admin

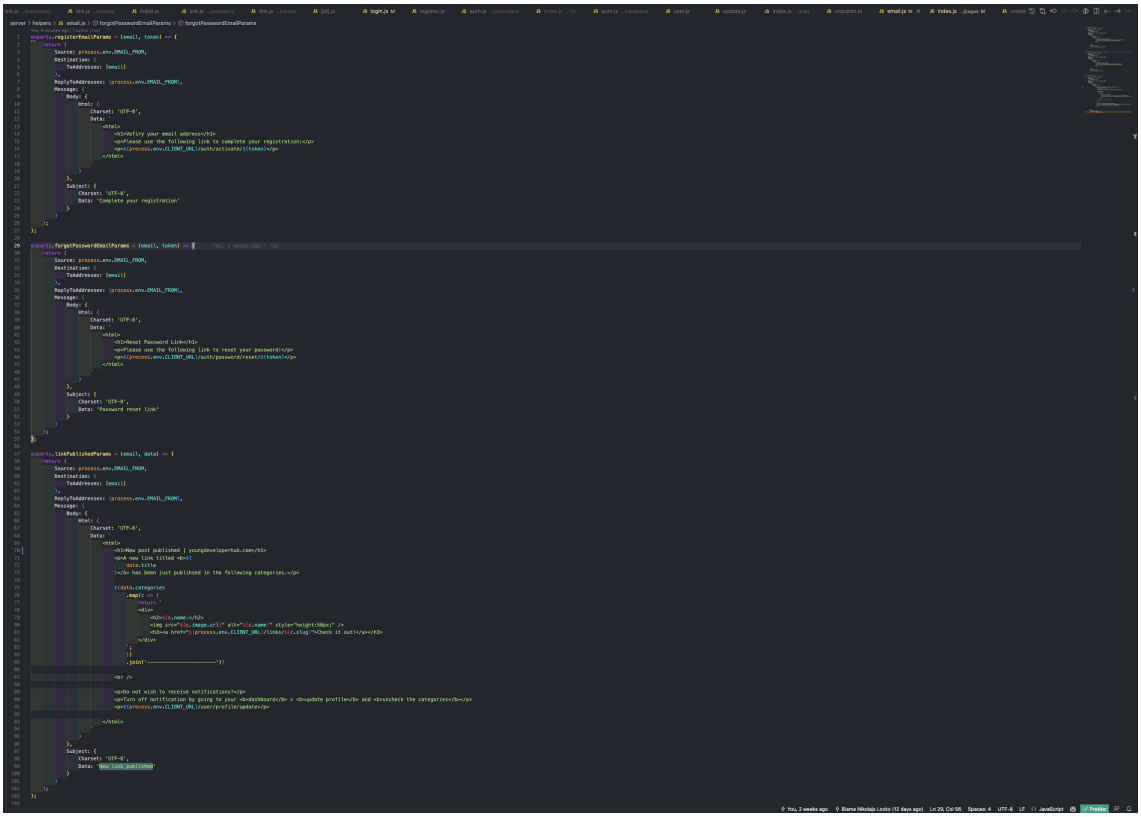


Figure C.13: Code snippet of the messaging and alert systems

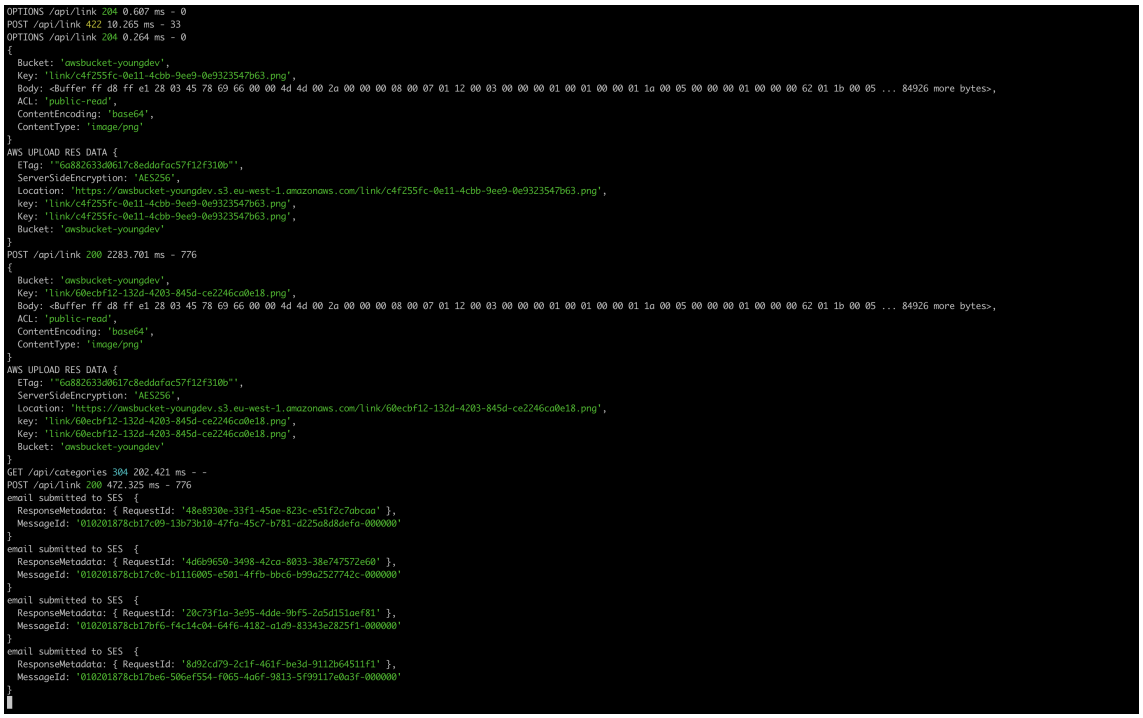


Figure C.14: Screenshot of server successfully working and delivering the messages and notifications

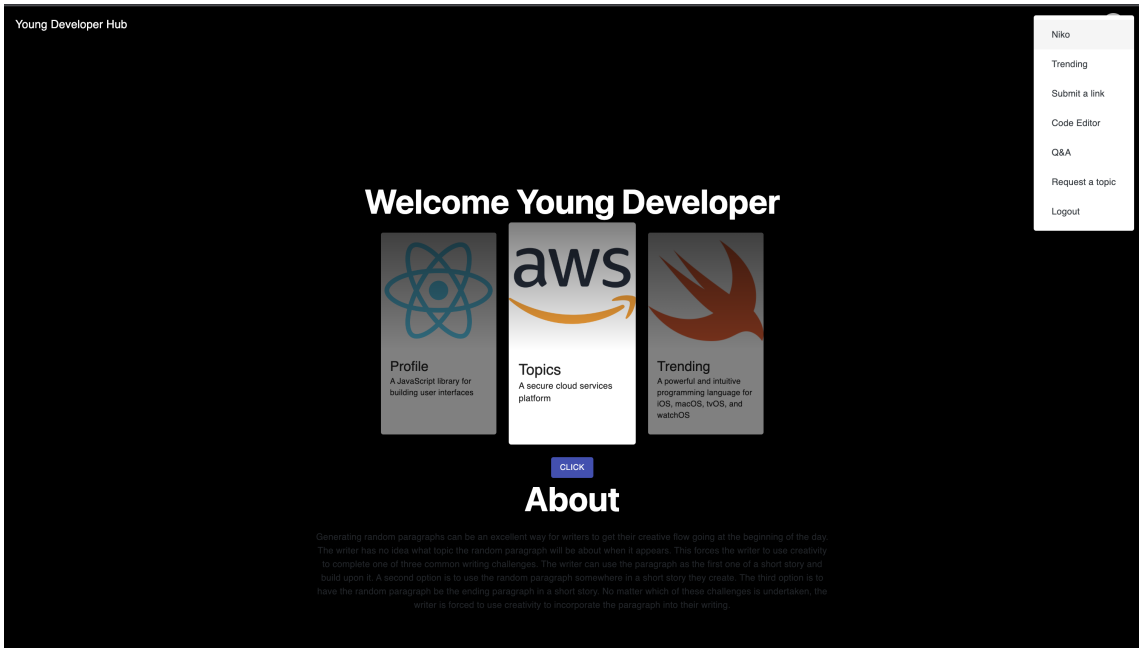


Figure C.15: New improved UI homepage

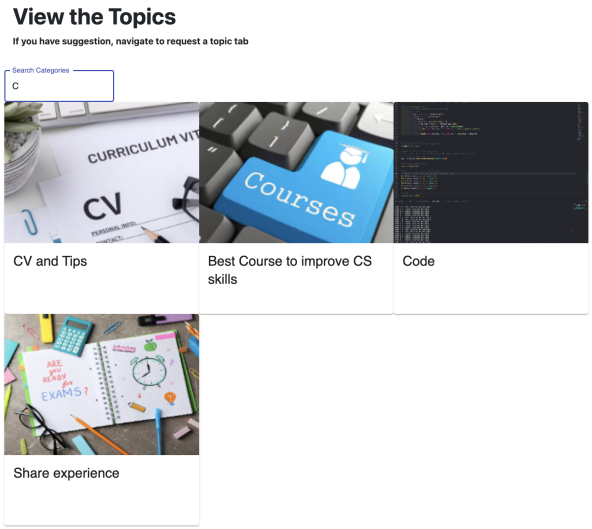
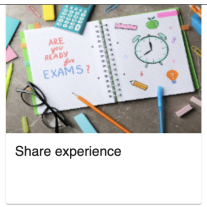


Figure C.16: Improved UI for topics page



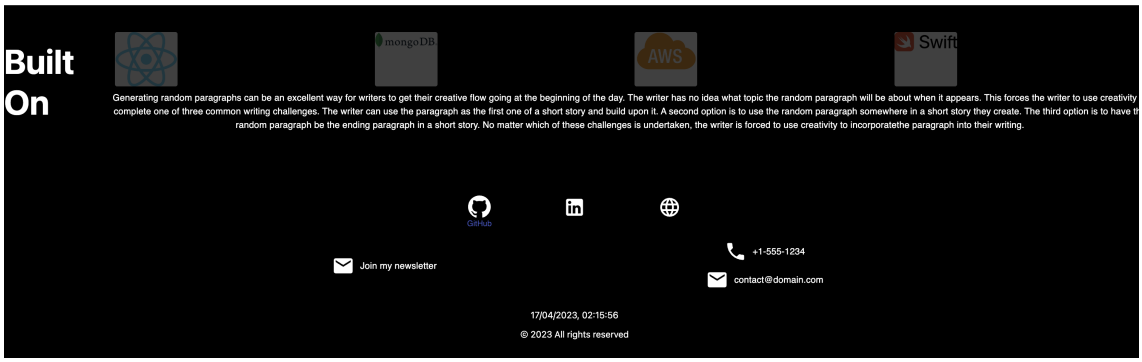


Figure C.17: Footer UI

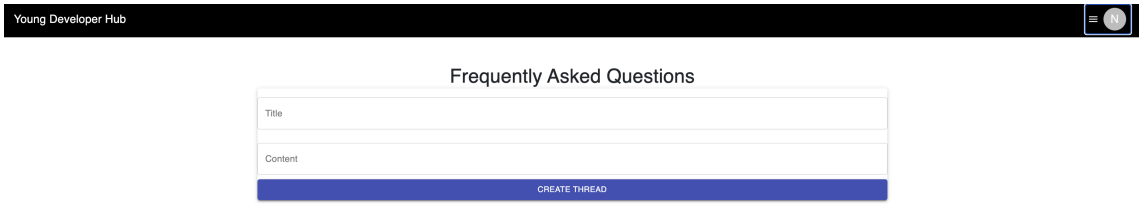


Figure C.18: FQA page for users

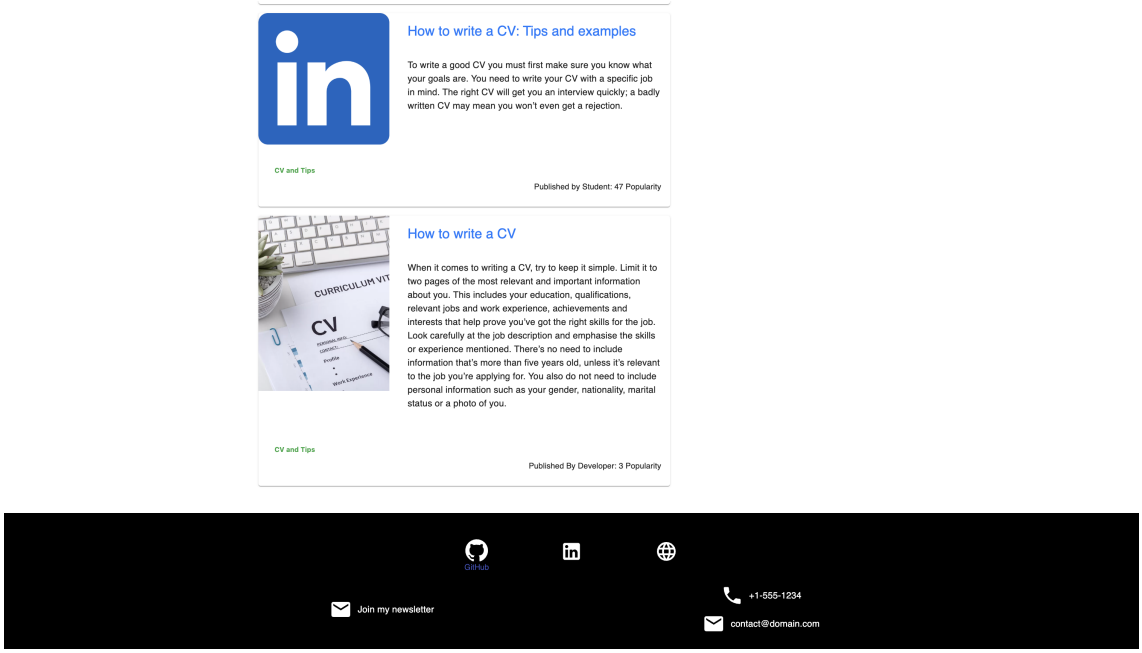


Figure C.19: Roles such ”developer” or ”student” post created feature

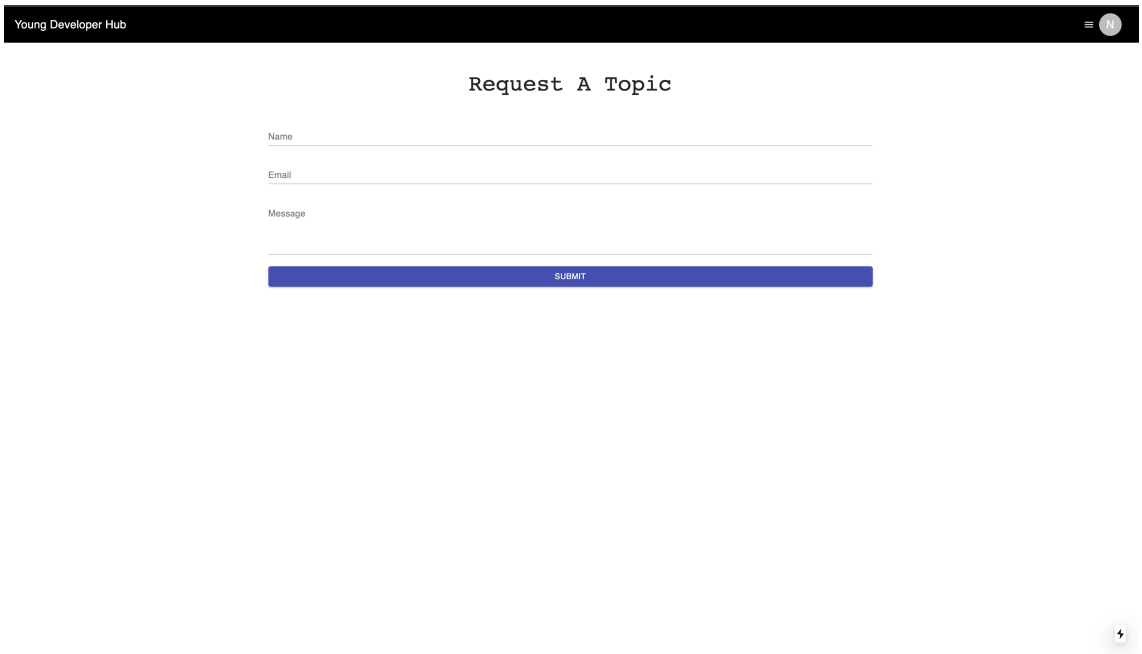


Figure C.20: Request topic to the administrator or leave feedback

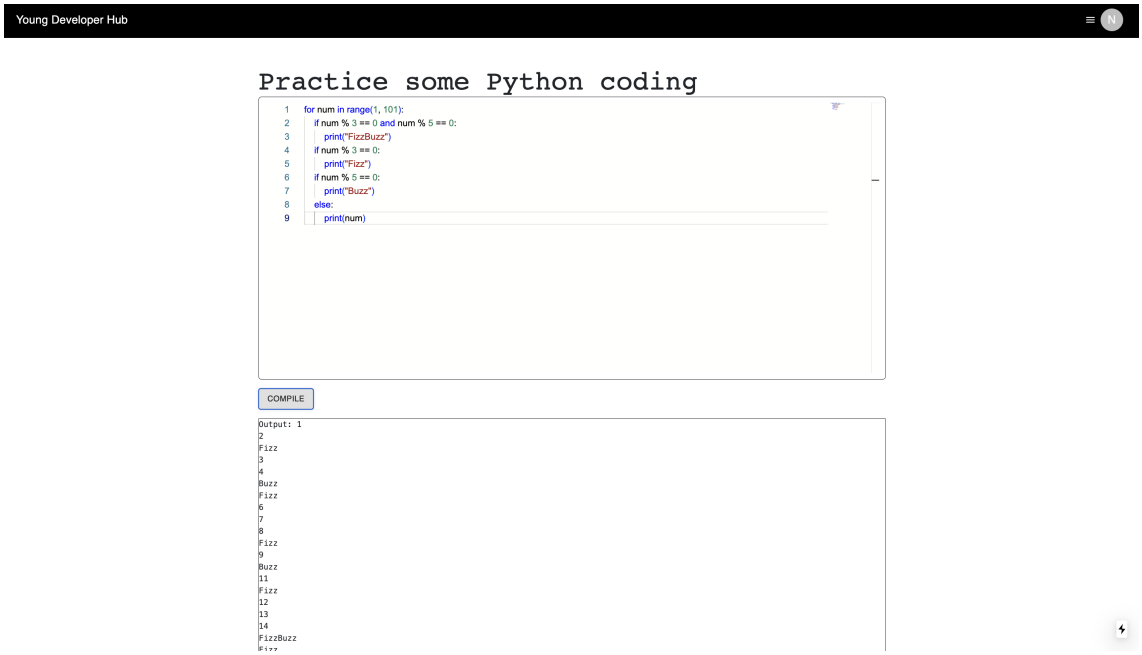


Figure C.21: Python online code editor

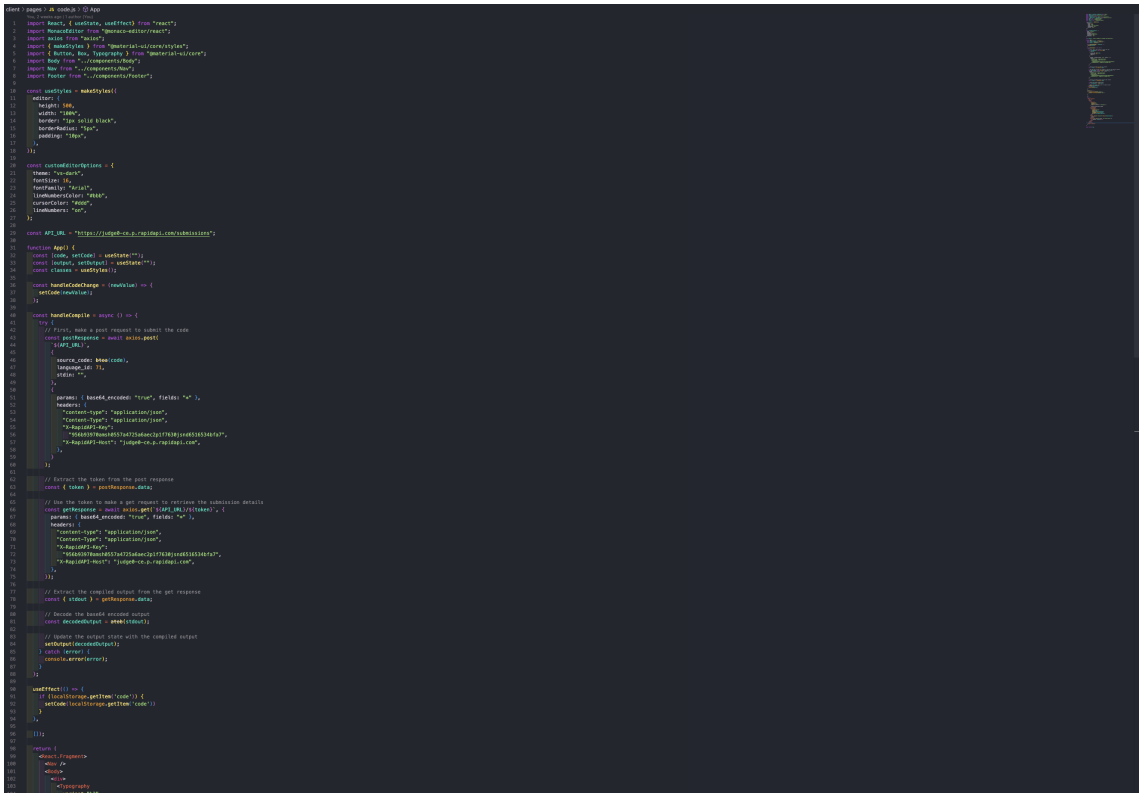


Figure C.22: Code snippet of the online code editor from above figure

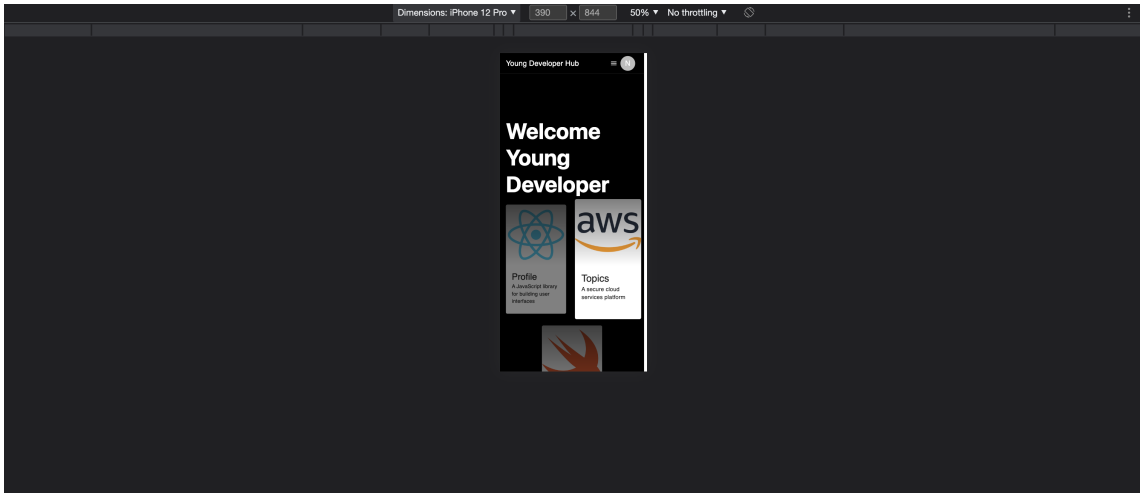


Figure C.23: Figure of iPhone 12 model test

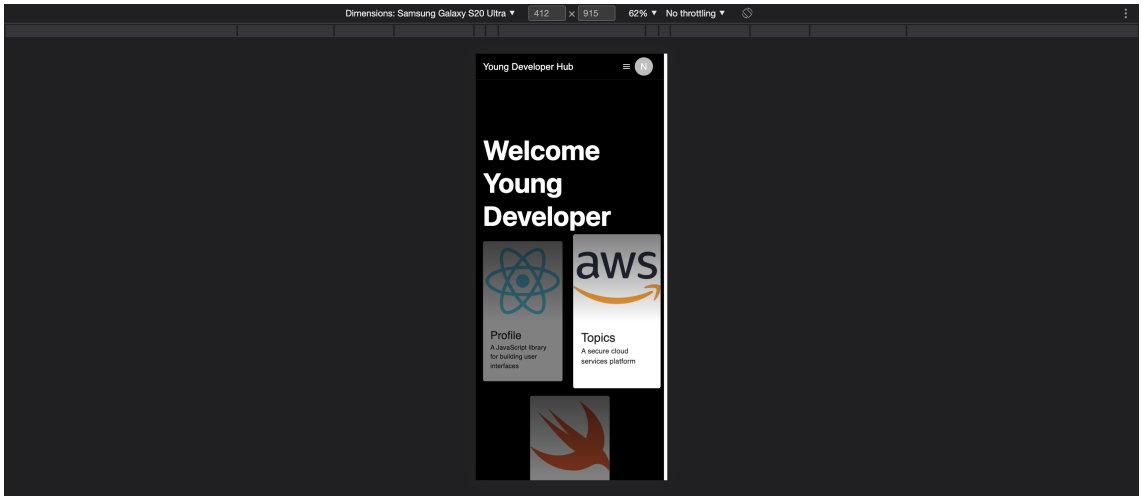


Figure C.24: Figure of Android S20 model test

C.1 AWS deployment

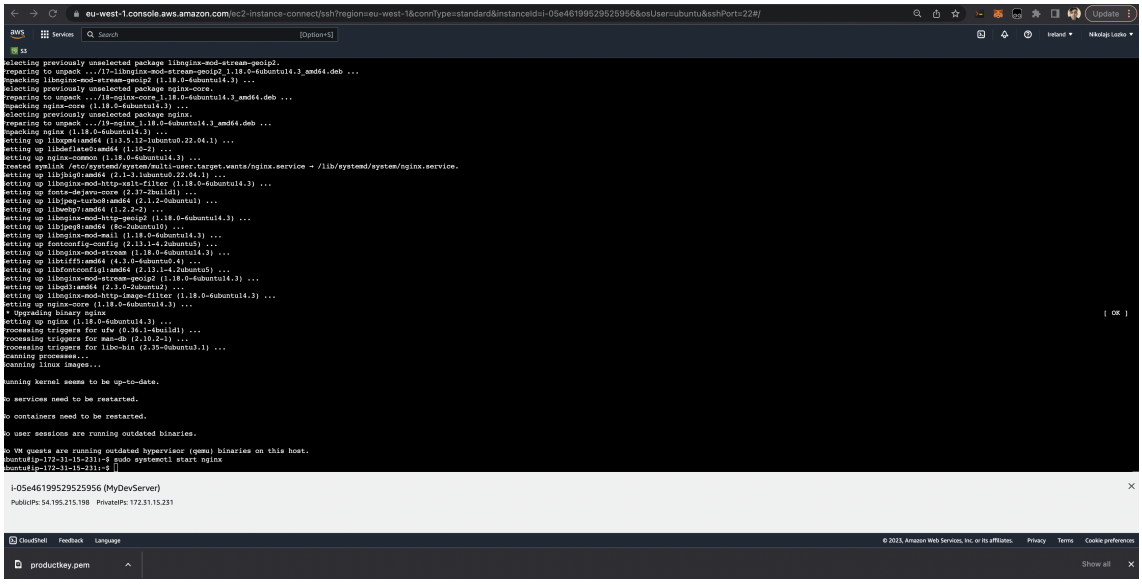


Figure C.25: Running EC2 instance Ubuntu version

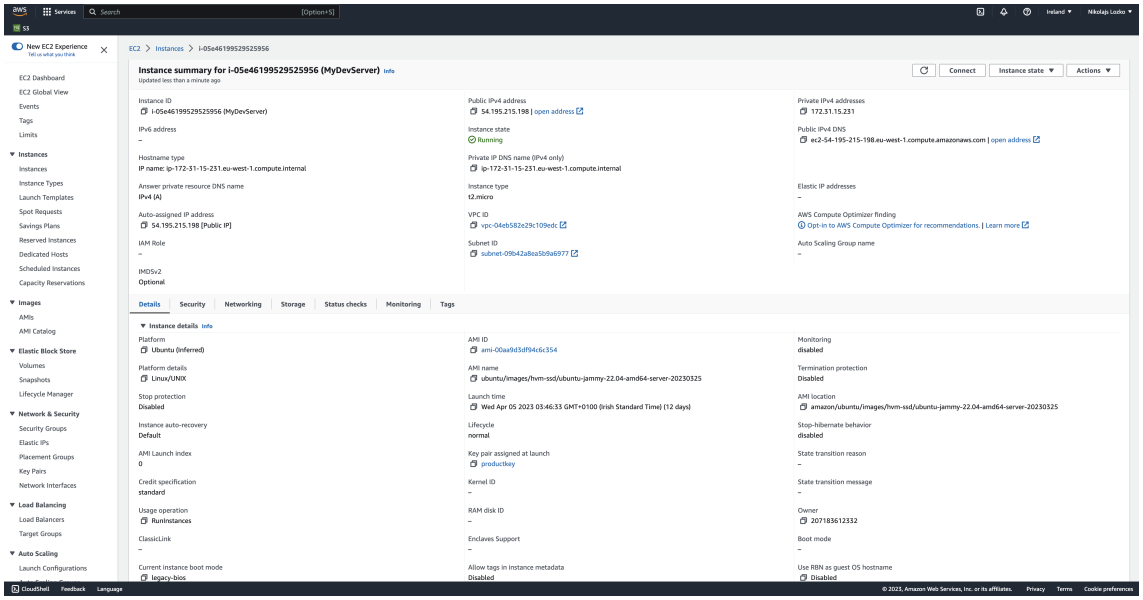


Figure C.26: EC2 Dashboard

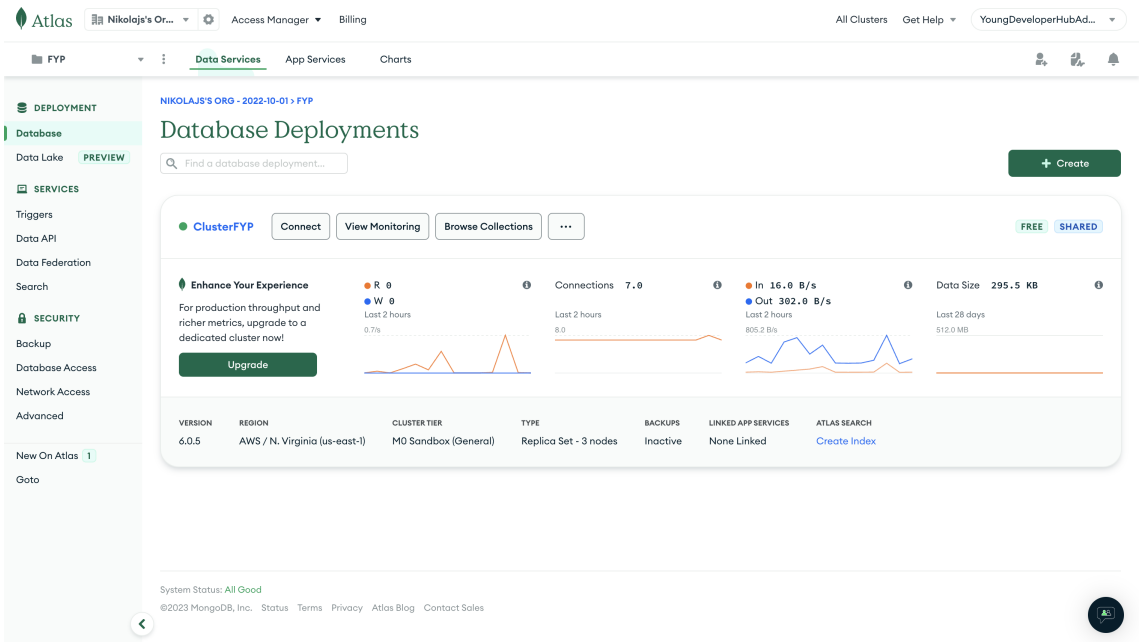


Figure C.27: MangoDB atlas cluster active

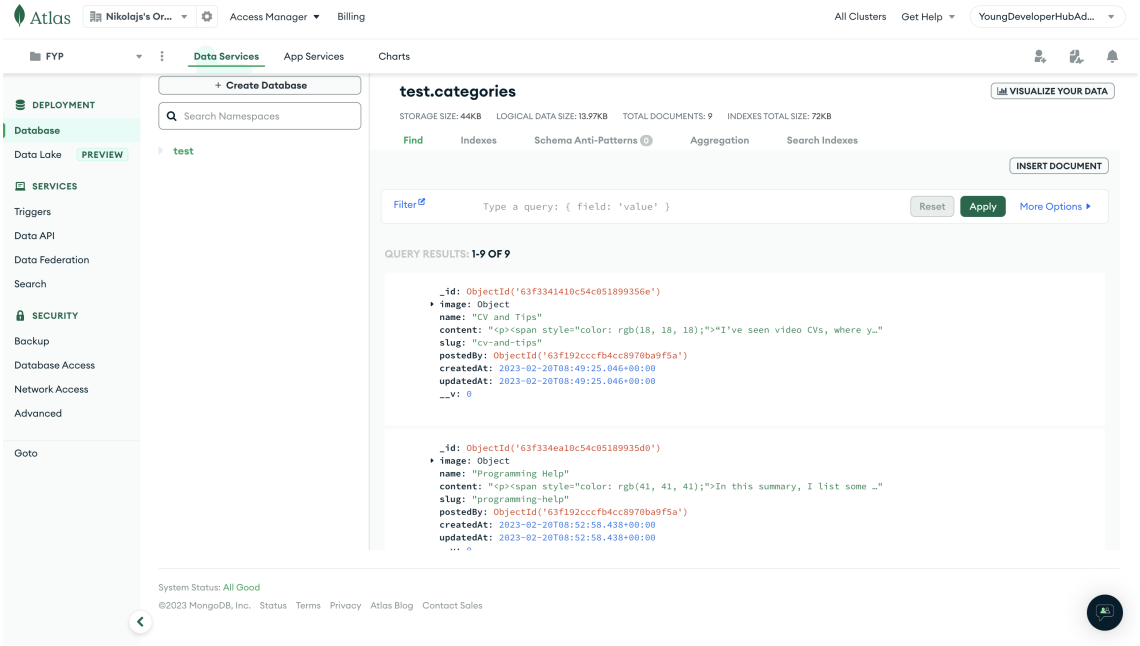


Figure C.28: Database storing the different topics made by administrator

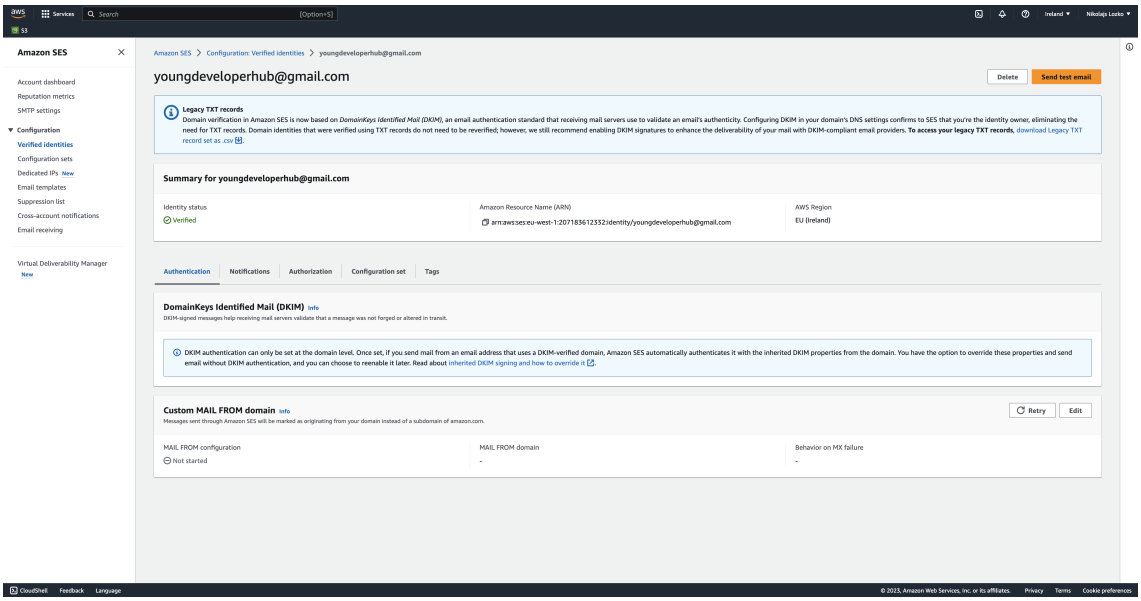


Figure C.29: SES dashboard with verified users

Appendix D

D.1 Interview Questions - Graduate

1. What kind of technical or non-technical skills can I learn on YoungDeveloperHub?

   The platform offers opportunities to develop non-technical skills such as teamwork, communication, and problem-solving. You can also find useful content for third-year students such as content on topics such as internships, graduate roles, project ideas, etc. The platform's user-generated posts allow students to learn from each other and build a diverse skill set.

2. Can I find job or internship opportunities on YoungDeveloperHub?

   Yes, you can find job or internship opportunities on YoungDeveloperHub. The platform provides a place for students and recent graduates, to have access to a variety of learning resources, including job and internship postings. You can browse through the available jobs or internship listings to find opportunities that match your interests.

3. Can I request specific topics to be covered on the platform?

   Yes, you can request specific topics to be covered on YoungDeveloperHub. The platform offers a social community where students can interact with the admin to request relevant topics to be covered on the platform. If you have a specific area of interest that you would like to learn more about or a particular skill you would like to develop, you can request it through the platform's community. The admin can then consider your request and add relevant content or resources to the platform to meet your needs.

4. Can I collaborate with other students on group projects through YoungDeveloperHub?

   Currently, YoungDeveloperHub offers a social community where students can interact with each other and leave comments and feedback on each other's posts. However, the platform is still developing its more advanced collaboration features, and in the future, hoping there will be more progress in development to make groups and enable more collaboration among students with direct messages or user-created groups.

Appendix E

E.1 QA Feedback - Student

1. Overall, the YoungDeveloperHub application was well-designed and user-friendly.
2. Appreciate the popularity rank that tracks students' most popular posts.
3. The application had a good range of learning materials and tools available for students to use, such as the online coding tool, but can be improved with multiple languages support
4. Identified a few minor bugs during testing, such as occasional slow loading times and some issues with post-formatting.
5. Additionally, there were a few areas where the application could benefit from additional features or improvements, such as more advanced search options and the ability to filter posts by topic or popularity level.
6. Suggest adding more interactive elements to the platform, such as quizzes or interactive coding challenges, to further engage students and enhance their learning experience.

   Overall, the QA tester felt that YoungDeveloperHub had great potential and could become a valuable resource for students looking to improve their collaborative skills.

Appendix F

F.1 Code Repository

The code repository for the Hub platform is available on GitHub (username: devnikolz project: FYP-Cloud). The repository contains the code used to develop the platform, including the front-end and back-end code, as well as the dependencies.

The front-end code is built using React, a popular JavaScript library for building user interfaces. The back-end code is built using Node.js and Express, for the server-side. The database used for the platform is MongoDB Atlas, a cloud-based database service.

The repository contains separate folders for each component of the platform. Each folder contains the necessary files and code for that component. To access the code repository, simply follow the information provided, to download it. From there, you can clone the repository or download the code directly.

Appendix G

End

In conclusion, the YoungDeveloperHub platform is a cloud-based web application designed to help young students and developers in learning and enhancing their skills. The platform offers a wide range of learning materials and tools, a personalized dashboard to view posts and offer suggestions for improvement. A social community for collaboration and feedback. The use of cloud services allows the platform to provide a scalable and accessible learning environment.

Providing a functional and user-friendly platform was a success. However, there is still room for improvement, such as, integration with Canvas, machine learning to personalize the learning experience. Also adding more social features such direct messages and user created own groups.

The completion of this dissertation would not have been possible without the guidance, support, and contributions of many individuals. I would like to express my heartfelt gratitude to everyone, my friends and family, especially my grandparents, the participants in the user research and quality testing.

Finally, I hope that this dissertation contributes to the field of education and inspires others to develop innovative solutions that enhance the learning experience for students and young developers across the world.

Thank you for reading.

Date: April 24, 2023

Bibliography

[1] Amazon vs Azure. Aws vs azure – which one gives a promising future? https://blog.trainingbasket.in/aws-vs-azure-which-one-gives-a-promising-future/.

[2] Cloud Research. Article for this comparison used. ://www.dataquest.io/blog/tutorial-azure-sql-database/.

[3] Cloud Comparison research. Article for this comparison used. https://www.bmck.au/posts/2023/comparing-azure-vs-aws-which-cloud-platform-is-right-for-you/.

[4] Comparison of computing tech. Article for this comparison used. https://www.akginfotech.com/2023/02/amazon-web-services-everything-you-want-to-know/.

[5] Cloud Computing. Article for this comparison used. https://www.webnexs.com/blog/headless-ecommerce-solutions-right-for-your-business/.

[6] Cloud Comparison attacks. Article for this comparison used. https://www.edureka.co/blog/aws-vs-azure/.

[7] IDE. Vscode docs. https://code.visualstudio.com/docs.

[8] AC and Metrics. Acceptance criteria for user stories: Purposes, formats, examples, and best practices. https://www.altexsoft.com/blog/business/acceptance-criteria-purposes-formats-and-best-practices/.

[9] Docs. Bdd docs. https://cucumber.io/docs/bdd/.

[10] Docs. Tdd docs. https://www.guru99.com/test-driven-development.html.

[11] Figma. Documentation for developers. https://www.figma.com/best-practices/.

[12] React: A javascript library for building user interfaces documentation. https://legacy.reactjs.org/docs/getting-started.html.

[13] Material-ui: Documentation for compoments. https://mui.com/x/introduction/.

[14] Node. Documentation for developers on node.js. https://nodejs.org/en/docs.

[15] MongoDB Atlas. Documentation for developers on mongodb atlas. https://www.mongodb.com/docs/.

[16] AWS Services. Documentation for developers on aws web services. https://docs.aws.amazon.com/.

[17] LaunchDarkly. Documentation for developers ci/cd tool. https://docs.launchdarkly.com/home.

[18] TEAMWORK. Documentation for management tool. https://support.teamwork.com/desk/getting-started.

[19] Components system. Article research on interactions. https://www.usability.gov/what-and-why/user-interface-design.html.

[20] Rapid API. Documentation for judge0 extra. https://rapidapi.com/judge0-official/api/judge0-extra-ce/.

[21] Design trades. Article for trade offs. https://www.usability.gov/what-and-why/user-interface-design.html.

[22] Design UI. Article for design ui. https://www.linkedin.com/advice/0/how-do-you-evaluate-communicate-trade-offs.

[23] MVC. Documentation for mvc. https://developer.mozilla.org/en-US/docs/Glossary/MVC.

[24] Google. Guidlines by google. https://m2.material.io/design/guidelines-overview.

[25] AWS EC2. Ec2 instances docs. https://docs.aws.amazon.com/pdfs/AWSEC2/latest/UserGuide/ec2-ug.pdf#concepts.

[26] AWS S3. S3 buckets docs. https://aws.amazon.com/s3/?p=pm&c=s3&z=4.

[27] AWS SES. Ses buckets docs. https://docs.aws.amazon.com/pdfs/ses/latest/dg/ses-dg.pdf#Welcome.

[28] Bitrise. Ses buckets docs. https://devcenter.bitrise.io/.

[29] LaunchDarkly More Info. More supported documentation. https://docs.launchdarkly.com/guides/infrastructure/deployment-strategies.

[30] Agile Dev. Documentation agile dev process. https://www.techtarget.com/searchsoftwarequality/definition/agile-software-development.

[31] AWS Load. Documentation load balancing. https://aws.amazon.com/what-is/load-balancing/.

[32] SQL attacks. Documentation for security attacks. https://owasp.org/www-community/attacks/xss/.

[33] Article. Documentation on load test. https://loadninja.com/load-testing/.

[34] Book. Clean code. https://thixalongmy.haugiang.gov.vn/media/1175/clean_code.pdf.