MOGA MUZAMIL ABDUL WAHAB

S21B23/013

A94166

Background

Google Developer Student Clubs (DSCs) are community groups for university and college students interested in learning about Google technologies and developer tools. These clubs are open to any student, regardless of their academic background or major.

DSCs are organized and supported by Google Developers, who provide resources and guidance to help students learn, grow, and build projects using Google's technologies. DSCs offer a variety of activities, including workshops, study jams, speaker sessions, hackathons, and hands-on projects.

TOOLS USED IN THE WEB APPLICATION

Material UI, React Js, CSS, Javascript, Routes, Sendgrid, Cloudinary.com

DESCRIPTION OF THE APPLICATION:

The DSC_UCU Community Web Application is a platform for managing the members of the Google Developer Student Club chapter at Uganda Christian University. It allows users to register as members by providing basic information such as their name, email address, phone number, and any other relevant details. The application then validates and stores this information in a database or other data storage system. Upon successful registration, the user receives a confirmation message.

In addition, the application allows users to authenticate themselves by logging in with their email and password. Upon successful login, the user is presented with a welcome screen that displays a proper representation of their profile from the details they provided during registration. And, registered members or users are able to edit their profiles such as their names, passwords and telephone numbers, and are able to apply for any Member-only content such as exclusive content, features or benefits provided by the club.

The application also has an admin signup page where someone must provide an admin sign-up token to sign up as an admin. The admin has access to manage the membership of the club, including tracking, modifying and deleting member information. The application helps to streamline the management of the club's membership.

The admin is also in-charge of the management of the members like the application reflects the number of registered members to the admin, their status (details contained in their profile), is able to update, delete or do modifications on a user’s profile.

The activities provided by the club include:

Tech Talk: Monthly talk on trending technologies in the tech industry.

Hackathon: A 24- hour coding competition to build solutions for real-world problems.

Workshop: Hands-on training on a specific technology or tool.

Project showcase: Opportunity to show case personal or team projects built using Google's technologies.

Study Jam: Group study sessions to prepare for Google's certification exams.

Guest Speaker: Inviting experienced developers to speak on various topics in the tech industry.

Code Lab: Hands-on coding sessions to learn new technologies and tools.

Career Fair: Bringing tech companies to campus to meet and recruit potential hires.

Tech Outreach: Organizing workshops and coding sessions for high school students to inspire them to pursue careers in tech.

Community services: Using technology to address social issues in the community.

Google Cloud Bootcamp: Intensive training on Google Cloud technologies.

Design Sprint: A 5-Day process for ideation, prototyping, and testing solutions to real-world problems.

The application consists of an imaginary team of 12 people which include the following members:

The members have a "View Profile" that when clicked on should display information about the individual.

1. **Moga:** Project Lead- Front End Lead responsible for managing the team, setting project goals, and ensuring code quality and adherence to best practices.
2. **Landon Martinez-** UI/UX Designer – responsible for designing the user interface and experience for the application, ensuring that it is intuitive and easy to use.
3. **Isabella Lee:** Frontend Developer- Responsible for writing the code that powers the frontend of the application, bringing the design to life.

4. **Lucas Rodriguez:** Quality Assurance Tester-Responsible for testing the frontend of the application, ensuring that it is bug-free and functioning correctly.
5. **Sophie Patel:** Accessibility Specialist- Responsible for ensuring that the frontend of the application is accessible to users with disabilities, following best practices and guidelines.
6. **Shan Mary:** Frontend Engineer- Responsible for developing and maintaining the frontend codebase of the application, implementing new features and optimizing performances.
7. **Olivia Evans:** Backend Developer- Responsible for developing and maintaining the backend codebase of the application, managing the database, and ensuring that data is properly handled and secured.
8. **Gavin Lee: DevOps Engineer-** Responsible for ensuring that the deployment process runs smoothly, managing server infrastructure, and implementing automated testing and deployment processes to improve development efficiency and reliability.
9. **Zoe Nguyen:** Product Manager- Responsible for driving the product vision and roadmap, conducting user research and analysis, and ensuring that the product meets the needs of users.
10. **Sawn Suzie:** Data Scientist- Responsible for analyzing user data and feedback to make data driven decisions for the application.
11. **Martel Inglesie:** Backend Developer- Responsible for developing and maintaining the backend codebase of the application, managing the database, and ensuring that data is properly handled and secured.
12. **Moris Wonder World:** Quality Assurance Tester- Responsible for testing the frontend of the application, ensuring that it is bug-free and functioning correctly.

Briefly explaining the kind of web architecture am going to use and why?

Web Architecture

A suitable web architecture for this application is a client-server architecture. This architecture separates the application into three parts: the presentation layer (client), the application logic layer (server), and the database layer.

The client, which in this case is the web browser, communicates with the server through HTTP requests and receives responses. The server processes these requests, performs the necessary application logic, and communicates with the database to retrieve or store data.

We use this architecture specifically while sending emails using the SendGrid email API. The email API key is securely stored on the server, and the server-side application logic utilizes the API to send email notifications to users.

The diagrams as part of Analysis and Design phase of the web Application (Included in a separate pdf.)

IMPORTANT NOTES ON THE WEB APPLICATION

1. The application is based on 3 layers, client(frontend), server(backend) and local storage (database) to facilitate the use SendGrid email API to send the notifications to members after registering to the application.
2. The application is able to collect required information from an individual registering such as name, email address, phone number and other included details.
3. The application is able to validate information entered by a user and is able to display error messages when wrong structure of email address has been used such as leaving behind the "@", identifies errors for wrong telephone numbers.
4. The application is able to store the entered details by the user on a local storage.
5. The application is able to display a confirmation messages displaying all information entered by a user into the system.
6. The application is able to authenticate members by logging in with their email and password.

7. A user cannot login using an admin login page and also an admin cannot login using a user login page.
8. Tokens that include the following where used to register an individual as an admin and a wrongly used token is recognized by the system. (128456, 789012, 345678, 901234, 567890, 234567, 890123, 456789).
9. The application is able to send email notifications to users that have registered with the system through the use of the SendGrid API.
10. SendGrid was used send notifications to an email entered by an individual registering on the application.
11. The client(frontend) takes a user's email address and last name and sends to the backend(server) and then the server forwards to the SendGrid API that transmits the notification message to the user's email address.

12. The phone number has enabled to receive a maximum of 10 digits, other than that, it identifies an error message.
13. An admin is able to edit or delete members in the application.
14. A registered member is able to apply for a service or privilege or roles provided by the team.
15. The application has been implemented on Netlify and can be accessed using this link. https://dsc-ucu.netlify.app/