**CampusQ**

***A project report submitted to***

**Rajiv Gandhi University of Knowledge Technologies**

**SRIKAKULAM**

***In Partial fulfillment of the requirements for the Award of the degree of***

**BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING**

**Submitted by**

***3rd year B.Tech 2nd Semester***

**P. Kushal Vardhan (S180177)**

**Y.Divya (S180389)**

**E.Rajesh (S180155)**

**Under the Estimated Guidance of**

Mr.Y.Kumar Sekhar,

**Asst. Professor M.Tech(Phd),**

Department of CSE, Srikakulam.



**Department of Computer Science and Engineering,**

**RGUKT-SRIKAKULAM, ETCHERLA.**

**July 2023**



## **CERTIFICATE**

This is to certify that this project work titled **“CampusQ”** was carried out by **P.Kushal Vardhan (S180177), Y.Divya (S180389), E.Rajesh (S180155)**. In partial fulfillment of requirements for the award of Bachelor of Technology in Computer Science and Engineering is a bona fide work carried out by them under my supervision and guidance.

The report has not been submitted previously in part or in full to any other University or Institution for the award of any degree or diploma.

**Mr.Y.Kumar Sekhar, Mr.N.Sesha Kumar,**

**Asst. Professor M.Tech(Phd) Asst. Professor M.Tech**

**Project Guide, Head of the Department,**

**Department of CSE, Department of CSE,**

**RGUKT-SRIKAKULAM. RGUKT-SRIKAKULAM**



## **BONAFIDE CERTIFICATE**

Certified that this project work titled "CampusQ" is the bona fide work of **E.Rajesh (S180155), P.Kushal Vardhan (S180177), Y.Divya (S180389)** who carried out the work under my supervision, and submitted in partial fulfillment of the requirements for the award of the degree, BACHELOR OF TECHNOLOGY, during the year 2022-2023.

**Mr.Y.Kumar Sekhar, Mr.N.Sesha Kumar,**

**Project Guide, Head of the Department,**

**Asst. Professor M.Tech(Phd) Asst. Professor M.Tech**

**Department of CSE, Department of CSE,**

**RGUKT-SRIKAKULAM. RGUKT-SRIKAKULAM**

## **DECLARATION**

We hereby declare that this thesis entitled "**CampusQ**" is carried out by me during the academic year 2022-2023 in fulfillment of the requirements for the Mini Project in Computer Science and Engineering.

We further declare that this dissertation has not been submitted elsewhere for any Degree. The matter embodied in this dissertation report has not been submitted elsewhere for any other degree. Furthermore, the technical details furnished in various chapters of this thesis are purely relevant to the mentioned project and there is no deviation from the theoretical point of view for design, development, and implementation.

**Date: 17-07-2023 P. Kushal Vardhan(S180177) Place: RGUKT-SKLM Y. Divya (S180389)**

**E. Rajesh (S180155)**

## 

## 

## 

## 

## 

## 

## 

## **ACKNOWLEDGEMENT**

We would like to articulate my profound gratitude and indebtedness to our project guide **Asst. Prof. Sri Y.Kumar Sekhar sir**, Assistant Professor who has always been a constant motivation and guiding factor throughout the project time. It has been a great pleasure for us to get an opportunity to work under his guidance and complete the thesis work successfully. We wish to extend our sincere thanks to **Asst. Prof. Sri N.Sesha Kumar sir**, Head of the CSE {Computer Science and Engineering} Department, for his constant encouragement throughout the project.

We are also grateful to other members of the department without their support our work would have been carried out so successfully.

We thank one and all who have rendered help to us directly or indirectly in the completion of our thesis work.

**PROJECT ASSOCIATE BY**

**P. Kushal Vardhan (S180177)**

**Y. Divya (S180389)**

**E. Rajesh (S180155)**

## 

## **ABSTRACT**

The proposed platform is a “Students for Students” platform aimed at facilitating student engagement and knowledge sharing. The platform is college-specific as the name explains itself. Students will have an account to access the platform’s functionalities. The platform allows students to post their doubts, which can be answered by other students. The platform has a feature that allows everyone to view the questions asked and answers provided by other students. It is very user friendly and allows you to filter the questions based on time and department. Users also have the ability to add tags to questions for easy navigation. Overall, the platform seeks to promote student engagement and knowledge sharing by providing a platform where students can ask questions and get answers from their peers.

**Keywords:** Students for Students, College-Specific, Posting Doubts, Providing Answers, Filtering Questions, Tags.

## 

## 

## Table of Contents

Table of Contents

[**CERTIFICATE** ii](#_Toc140505272)

[**BONAFIDE CERTIFICATE** iii](#_Toc140505273)

[**DECLARATION** iv](#_Toc140505274)

[**ACKNOWLEDGEMENT** v](#_Toc140505275)

[**ABSTRACT** vi](#_Toc140505276)

[Table of Contents vii](#_Toc140505277)

[**CHAPTER- 1** 1](#_Toc140505278)

[**INTRODUCTION** 1](#_Toc140505279)

[**1.1. Introduction** 1](#_Toc140505280)

[**1.2. Problem Statement** 1](#_Toc140505281)

[**1.3. Objective** 2](#_Toc140505282)

[**1.4. Goals** 2](#_Toc140505283)

[**1.5. Project Scope** 2](#_Toc140505284)

[**1.6. Applications** 3](#_Toc140505285)

[**1.7. Limitations** 3](#_Toc140505286)

[**CHAPTER -2** 4](#_Toc140505287)

[**BASIC METHODOLOGIES** 4](#_Toc140505288)

[**2.1. Existing System** 4](#_Toc140505289)

[**2.2. Proposed system** 5](#_Toc140505290)

[**2.3. Comparison between Existing and Proposed System** 6](#_Toc140505291)

[**CHAPTER- 3** 8](#_Toc140505292)

[**REQUIREMENT SPECIFICATION** 8](#_Toc140505293)

[**3.1. Functional Requirements** 8](#_Toc140505294)

[**3.2. Non-functional requirements** 10](#_Toc140505295)

[**3.3 Other Non-Functionalities** 11](#_Toc140505296)

[**3.4 System Requirements** 11](#_Toc140505297)

[**CHAPTER -4** 13](#_Toc140505298)

[**SYSTEM DESIGNS** 13](#_Toc140505299)

[**4.1 Design of The System** 13](#_Toc140505300)

[**CHAPTER- 5** 20](#_Toc140505301)

[**SYSTEM IMPLEMENTATION** 20](#_Toc140505302)

[**5.1. User Register** 20](#_Toc140505303)

[**5.2 User Login** 20](#_Toc140505304)

[**5.3 Home Page** 21](#_Toc140505305)

[**5.4 Profile** 22](#_Toc140505306)

[**5.5 Ask Page** 23](#_Toc140505307)

[**5.6. Tagging** 23](#_Toc140505308)

[**5.7 Board** 24](#_Toc140505309)

[**5.8 View Answers for a particular Question** 25](#_Toc140505310)

[**5.9 Answer a Question** 25](#_Toc140505311)

[**5.10 Admin** 26](#_Toc140505312)

[**CHAPTER -6** 28](#_Toc140505313)

[**SOURCE CODE** 28](#_Toc140505314)

[**6.1 Annotations (Package)** 28](#_Toc140505315)

[**6.2 Audit (Package)** 29](#_Toc140505316)

[**6.3 Configurations (Package)** 29](#_Toc140505317)

[**6.4 Controller (Package)** 30](#_Toc140505318)

[**6.5 Model (Package)** 43](#_Toc140505319)

[**6.6 Repository (Package)** 48](#_Toc140505320)

[**6.7 Security (Package)** 50](#_Toc140505321)

[**6.8 Service (Package)** 51](#_Toc140505322)

[**6.9 Validations (Package)** 55](#_Toc140505323)

[**6.10 Static CSS** 56](#_Toc140505324)

[**6.11 HTML files** 98](#_Toc140505325)

[**6.12 pom.xml** 115](#_Toc140505326)

[**CHAPTER- 7** 118](#_Toc140505327)

[**TESTING** 118](#_Toc140505328)

[**7.1 Introduction to Testing** 118](#_Toc140505329)

[**7.2 Types of tests performed** 118](#_Toc140505330)

[**7.2.1 Unit testing** 118](#_Toc140505331)

[**CHAPTER- 8** 121](#_Toc140505332)

[**CONCLUSION** 121](#_Toc140505333)

[**CHAPTER- 9** 122](#_Toc140505334)

[**FUTURE ENHANCEMENT** 122](#_Toc140505335)

[**APPENDICES** 123](#_Toc140505336)

## **CHAPTER- 1**

## **INTRODUCTION**

## **1.1. Introduction**

In today's rapidly evolving educational landscape, student engagement and knowledge sharing play pivotal roles in fostering a vibrant and dynamic learning environment. With its college-specific focus, the platform serves as a virtual gathering place where students can connect, ask questions, and receive answers from their peers. The "Students For Students" platform prioritizes user-friendliness, ensuring that students can easily navigate and utilize its various features. Upon creating an account, students gain access to a range of functionalities that promote engagement and knowledge sharing. Transparency is a fundamental principle of the platform, as all questions and corresponding answers are available for everyone to view. To enhance usability further, the platform offers a tagging feature, enabling students to categorize questions based on relevant subject areas or keywords.

Through this platform, students can actively participate in their own learning journey while benefiting from the collective wisdom and insights of their peers.

### **1.2. Problem Statement**

We often find it easy to tackle our doubts together with our friends. It is actually a good practice to discuss with like-minded people and our friends can exactly understand our point. But it is tedious to approach everyone as we may not easily find someone in our friends who easily clarifies our doubts. We are creating an online student engagement and knowledge sharing platform. This makes the collaboration part really easy. You can reach hundreds of people at your fingertips. You can get your doubts cleared and you can hear various people’s views with just a click on submit. You can also help your peers by giving answers.

### **1.3. Objective**

The objective of the "Students For Students" platform is to facilitate student engagement, foster collaborative learning, and promote knowledge sharing within a college-specific context. It aims to enhance student support by providing a space for students to ask questions and receive answers from their peers. The platform strives to promote active student engagement, community building, and student ownership of learning. It facilitates easy information retrieval through filtering and tagging features. Ultimately, the objective is to create a dynamic and inclusive online community that empowers students to actively participate in their education and exchange knowledge with their peers.

### **1.4. Goals**

* User Friendly
* Simple and Fast
* Low cost and Effective
* Easy to use and Understand
* Collaborative learning
* Diverse viewpoints

### **1.5. Project Scope**

* Students should be able to create a new account for the first time.
* Students should be able to register through already existing accounts.
* Students should be able to share questions in the form of text.
* Students should be able to answer to other's questions if they want to.
* Students should be able to view other’s questions and their answers.

### **1.6. Applications**

* Can be used as a great collaborative platform in a college among students.

### **1.7. Limitations**

* Limited to only a particular college like RGUKT SKLM.

## 

## **CHAPTER -2**

## **BASIC METHODOLOGIES**

### **2.1. Existing System**

* Quora is a question and answer website where users can ask questions on a wide range of topics and receive answers from the users.
* Quora allows users to follow topics and other users, up vote and down vote answers and leave comments on questions and answers.
* In addition to education-related content Quora covers a wide range of topics from technology and politics for entertainment and personal development.

#### **2.1.1. Technologies Used in Existing System**

* **Frontend Technologies**
* JavaScript
* HTML
* CSS
* jQuery
* **Backend Technologies**
* Python
* C++
* PostgreSQL
* Memcached
* RabbitMQ
* Cassandra
* Hadoop
* Hive
* Pig
* Scribe
* **Web Servers**
* Nginx
* Apache

#### **2.1.2. Drawbacks of Existing System**

* Quora is not just confined to education purposes only it has a wider range of topics and discussions.
* **Trolls and harassment:** Like any online platform Quora can attract trolls and individuals who engage in harassment or bullying.
* **Lack of privacy:** Quora requires users to create a profile and disclose personal information, which may be a concern for those who value their privacy.
* **Time-wasting:** Quora can be addictive, and users may spend too much time on the website

### **2.2. Proposed system**

* “CampusQ” product innovates one great platform for students to have a quality collaborative learning environment.
* Students from all years and branches can use this effectively to excel in both academics and career.
* It is confined to a specific university, where students know each other very well and share a common learning environment so that they can easily understand our questions and answer them.
* Our application is only dedicated for education purposes.

#### **2.2.1. Technologies Used in Proposed System**

* **Mother Language**
* Java
* **Backend script**
* Spring boot and its dependencies
* **Frontend**
* HTML, CSS, Java Script
* **Online version control system**
* Github
* **Integrated Development Environment**
* IntelliJ

#### **2.2.2. Advantages of Proposed System**

* **Easy access:** An online student engagement and knowledge sharing platform provides easy access to a large community of learners from different backgrounds and areas of expertise.
* **Convenience:** Users can access the platform anytime and from anywhere, making it convenient for them to get their doubts cleared and share their knowledge.
* **Collaborative learning:** The platform fosters a collaborative learning environment where users can learn from each other's experiences and perspectives.
* **Diverse viewpoints:** Users can hear various people's views on a topic, which can increase their understanding and provide them with new insights.
* **Recognition:** We are always up to help our community and get noticed for the knowledge we share. We can even build a good network of people.

### **2.3. Comparison between Existing and Proposed System**

The proposed system addresses many of the drawbacks associated with the existing system. It focuses specifically on education, providing easy access to a community of learners and promoting collaborative learning. The platform offers convenience, diverse viewpoints, and opportunities for recognition and networking. However, it is important to implement measures to address potential issues such as online harassment and user privacy concerns. Overall, the proposed system appears to be a more specialized and beneficial platform for educational engagement and knowledge sharing compared to the existing system.

## 

## **CHAPTER- 3**

## **REQUIREMENT SPECIFICATION**

### **3.1. Functional Requirements**

#### **3.1.1 Student Registration**

**Description:** Firstly students have to register in order to access the Ask RGUKTian software. Student details stored in the database.

**Input:** Student Registration details

**Output:** Registration Successful message

#### **3.1.2 Student Login**

**Description:** whenever Student wants to use Ask an RGUKTian software must have to Login

**Input:** Student mail id and password

**Output:** Redirect to the software

#### **3.1.3 Manage their own questions**

**3.1.3.1 Ask a new question**

**Description:** Question must be in the form of text. This question is Stored in the database and a unique serial number or id is generated. A person can ask a question only if he is registered. A question can be assigned tags.

**Input:** Question

**Output:** Question asked. Anyone can view it.

**3.1.3.2 View/Display Questions (“My Questions”)**

**Description:** A person can view the questions he has asked in the past. With every question, the owner’s id is also stored.

**Input:** when “view my questions” is clicked, the user's id is queried in the questions database.

**Output:** All the questions asked by the user.

**3.1.3.3 Delete Question**

**Description:** If a person wants to delete the question he has asked, he can do that.

**Input:** Question id

**Output:** Question deleted from the database.

**3.1.3.4 Filter questions by tags**

**Description:** A CS student can filter the CS related questions and choose to answer them. As every question has a tag stored in the database, it is easy to filter questions by tags.

**Input:** tag

**Output:** Display All the Questions with the specified tag, sorted by date and time.

#### **3.1.4 Manage their own answers**

**3.1.4.1 Answer a question**

**Description:** A person can answer a question. The answer must be in the form of text.

(Hyperlinks to references can be provided). Every answer is stored in a database with a

Unique id or serial number.

**Input:** Text depicting answer to a question.

**Output:** Answer is stored. Anyone can view it.

**3.1.4.2 View/Display the answered (“My Answers”)**

**Description:** A person can view all the questions he has answered in the past.

This is possible because, with every answer stored, we also assign the owner's id.

**Input:** User id

**Output:** All the questions he has answered

**3.1.3.3 Delete Answer**

**Description:** If a person wants to delete the answer he has posted, he can do that.

**Input:** Answer id

**Output:** Answer deleted from the database.

### **3.2. Non-functional requirements**

#### **3.2.1 User Interface**

* Users need a modern web browser with JavaScript enabled.

#### **3.2.1 Hardware Interfaces**

● Processor: 1 GHz or higher

● RAM: 512 MB or higher

● Storage: 1 GB or higher

● Network: Ethernet or Wi-Fi

#### **3.2.1 Software Interfaces**

● JDK

● an IDE(IntelliJ)

● a build tool

● a web server(Tomcat)

● a database(MYSQL)

### **3.3 Other Non-Functionalities**

#### **3.3.1 Performance**

● The landing page supporting 3,000 users per hour must provide 5 seconds or less response time in a chrome desktop browser including the rendering of the text, images.

#### **3.3.2 Scalability**

● The software must be freely scalable to serve 25000 users and maintain optimal performance.

#### **3.3.3 Portability**

● This software supports mobile and desktop environments.

#### **3.3.4 Reliability**

● The system should perform with 95% accuracy.

#### **3.3.5 Availability**

● This software is available for 22 hours a day.

### **3.4 System Requirements**

#### **3.4.1 Hardware Requirements**

* For smooth run, user should possess
* Processor: 1 GHz or higher
* RAM: 512 MB or higher
* Storage: 1 GB or higher
* Network: Ethernet or Wi-Fi
* For Production environment
* Processor: 2 GHz or higher
* RAM: 4 GB or higher
* Storage: 10 GB or higher (SSD recommended)

#### **3.4.2 Software Requirements**

* User’s side
* Users need a modern web browser with JavaScript enabled.
* Developer’s side
* JDK
* An IDE(IntelliJ)
* A build tool
* A web server(Tomcat)
* A database(MYSQL)
* Any additional dependencies.

## 

## **CHAPTER -4**

## **SYSTEM DESIGNS**

### **4.1 Design of The System**

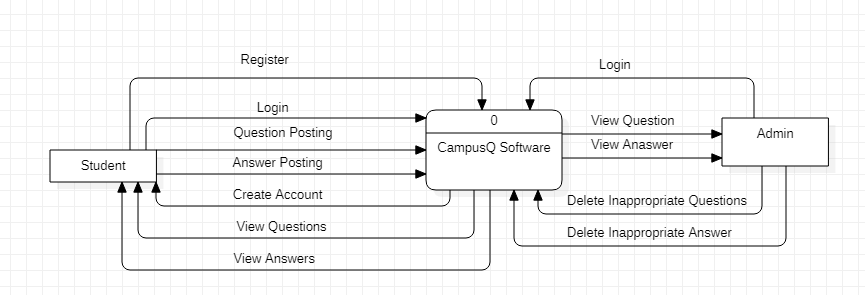
#### **4.1.1 DFD Diagrams**

A data flow diagram or bubble chart (DFD) is a graphical representation of the "flow" of data. Through an information system, modeling its process aspects. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

**4.1.1.1. Level 0 DFD Diagram**

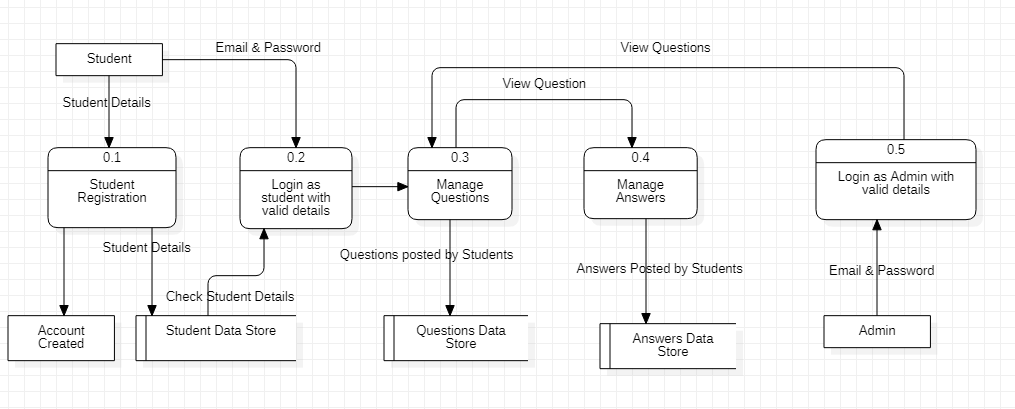
This is the highest-level DFD, which provides an overview of the entire system. It shows the major processes, data flows, and data stores in the system, without providing any details about the internal workings of these processes.



**Fig. 4.1.1.1. Level 0 DFD Diagram**

**4.1.1.2. Level 1 DFD Diagram**

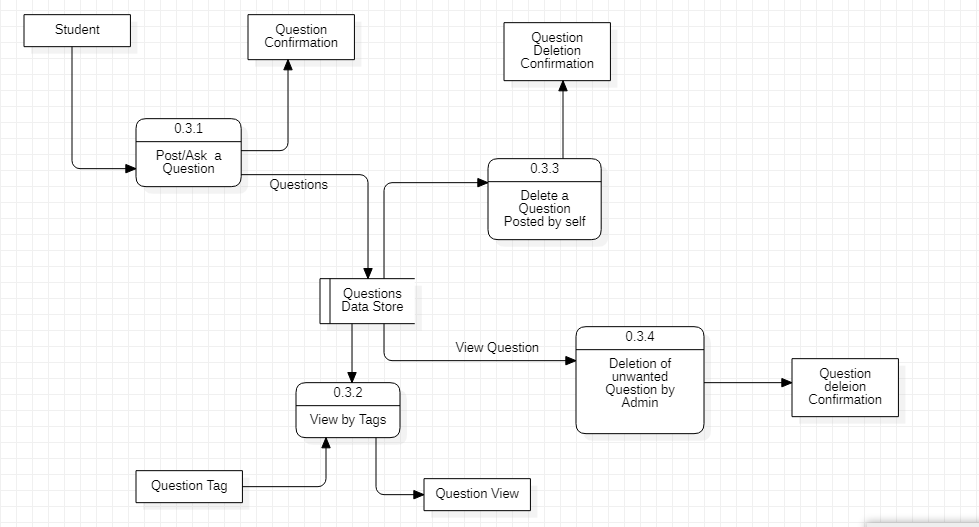
This level provides a more detailed view of the system by breaking down the major processes identified in the level 0 DFD into sub-processes. Each sub-process is depicted as a separate process on the level 1 DFD. The data flows and data stores associated with each sub-process are also shown.



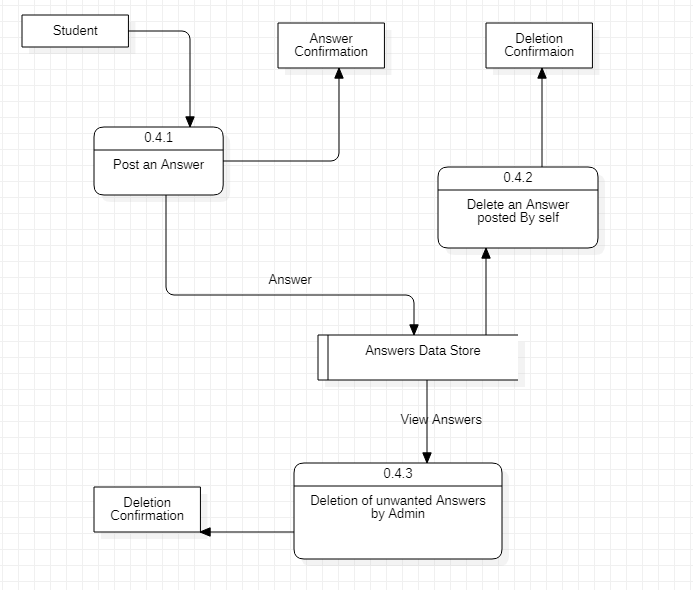
##### **Fig. 4.1.1.2. Level 1 DFD Diagram**

**4.1.1.3. Level 2 DFD Diagram**

This level provides an even more detailed view of the system by breaking down the sub-processes identified in the level 1 DFD into further sub-processes. Each sub-process is depicted as a separate process on the level 2 DFD. The data flows and data stores associated with each sub-process are also shown.



**Fig. 4.1.1.3. Level 2 DFD Diagram Manage Question Module**

****

**Fig. 4.1.1.3. Level 2 DFD Diagram Manage Answer Module**

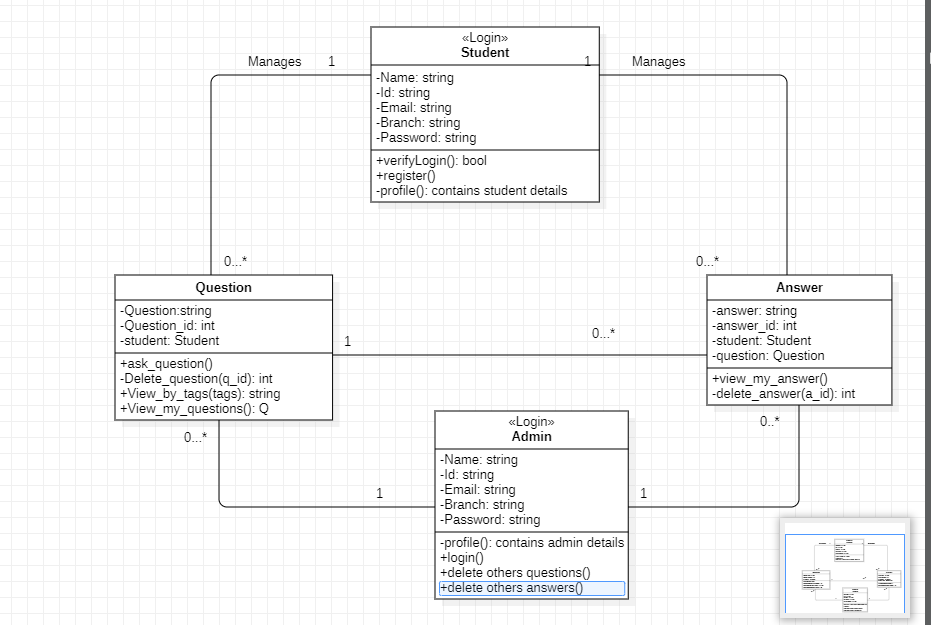
#### 

#### **4.1.2. UML Diagrams**

Unified Modeling Language (UML) was created in 1995 by merging diagramming conventions used by three application development methodologies: OMT by James Rumbaing, Objector y by Ivor Jacobson and the Brooch procedure by using Grady Brooch. Previous to this time, these three amigos, together with a few dozen other practitioners had promoted competing methodologies for systematic program development, each and every with it possessing a system of diagramming conventions. The methodologies adopted a sort of cookbook sort of pushing an application task via a succession of life cycle stages, culminating with delivered and documented software. One purpose of UML was once to slash the proliferation of diagramming techniques by way of standardizing on an original modeling language, as a result facilitating verbal exchange between builders.

**4.1.2.1. Class Diagram**

Class diagram in the Unified Modeling Language (UML), is a kind of static structure diagram that describes the constitution of a process through showing the system's classes, their attributes, and the relationships between the classes. The motive of a class diagram is to depict the classes within a model. In object-oriented software, classes have attributes (Member variables), operations (member capabilities) and relation**.**

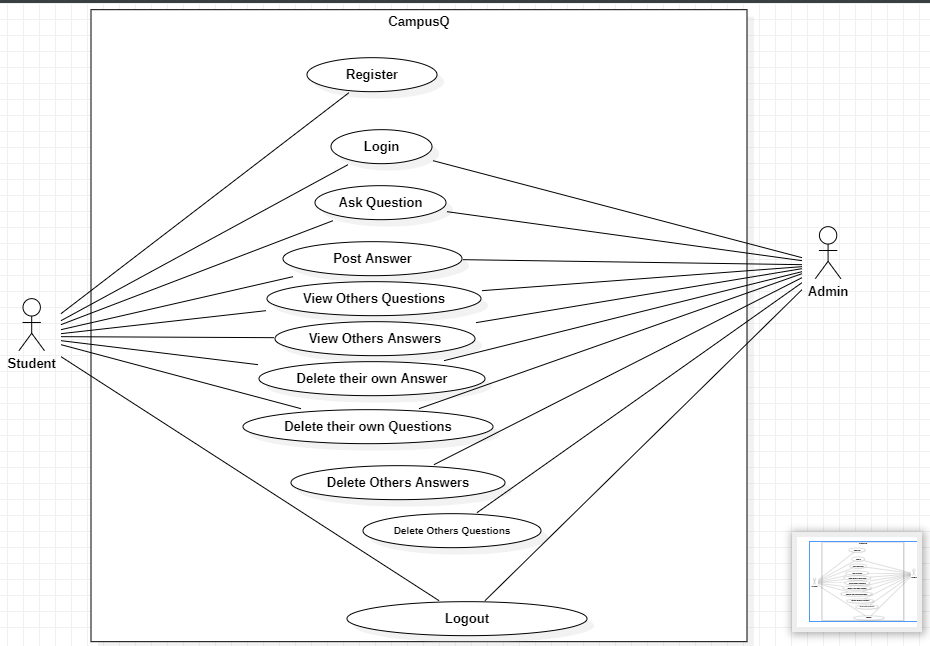


**Fig. 4.1.2.1. Class Diagram**

**4.1.2.2. Use Case Diagram**

It is a visual representation of what happens when an actor interacts with a system. A use case diagram captures the functional aspects of a system.

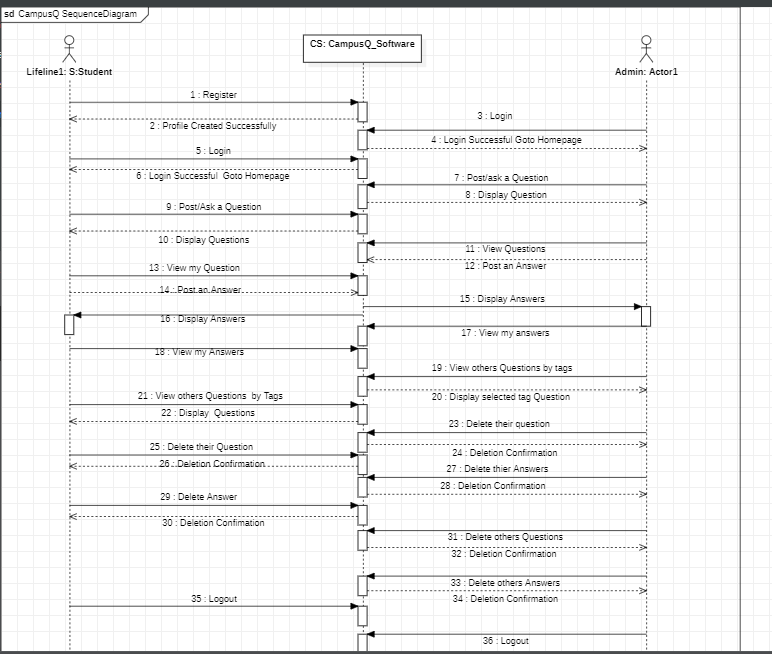
The system is shown as a rectangle with the name of the system inside, the actor is shown as stick figures, the use case is shown as solid bordered ovals labeled with the name of the use case and relationships are lines or arrows between actors and use cases. Symbols used in Use case are as follows-



**Fig. 4.1.2.2. Use Case Diagram**

**4.1.2.3. Sequence Diagram**

A sequence diagram in Unified Modeling Language (UML) is one variety of interaction diagram that suggests how methods operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are quite often referred to as event-hint diagrams, event situations, and timing diagrams. A sequence diagram suggests, as parallel vertical traces (lifelines), special systems or objects that are residing at the same time, and, as horizontal arrows, the messages exchanged between them, within the order the place they occur.



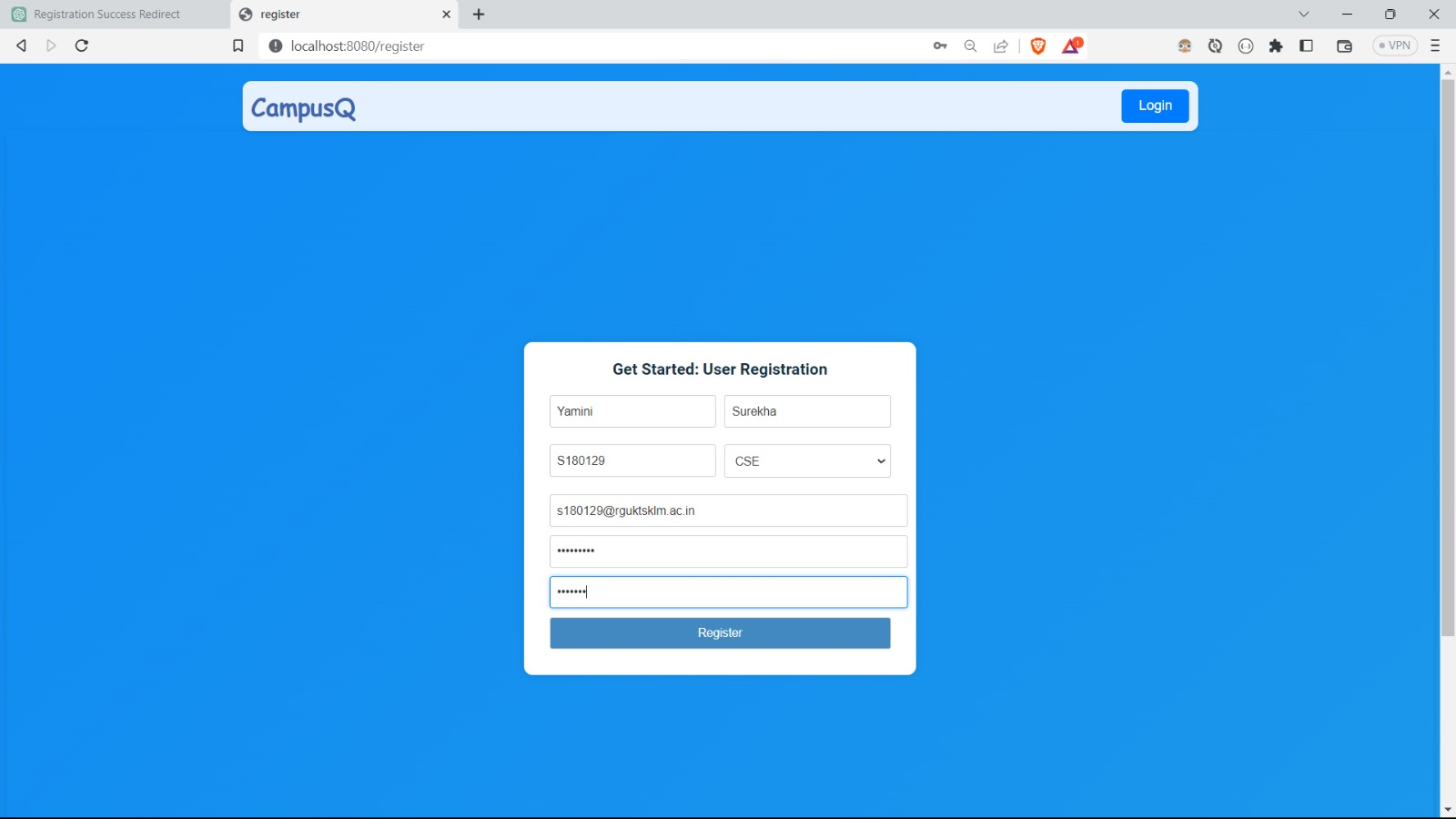
**Fig. 4.1.2.3. Sequence Diagram**

## **CHAPTER- 5**

## **SYSTEM IMPLEMENTATION**

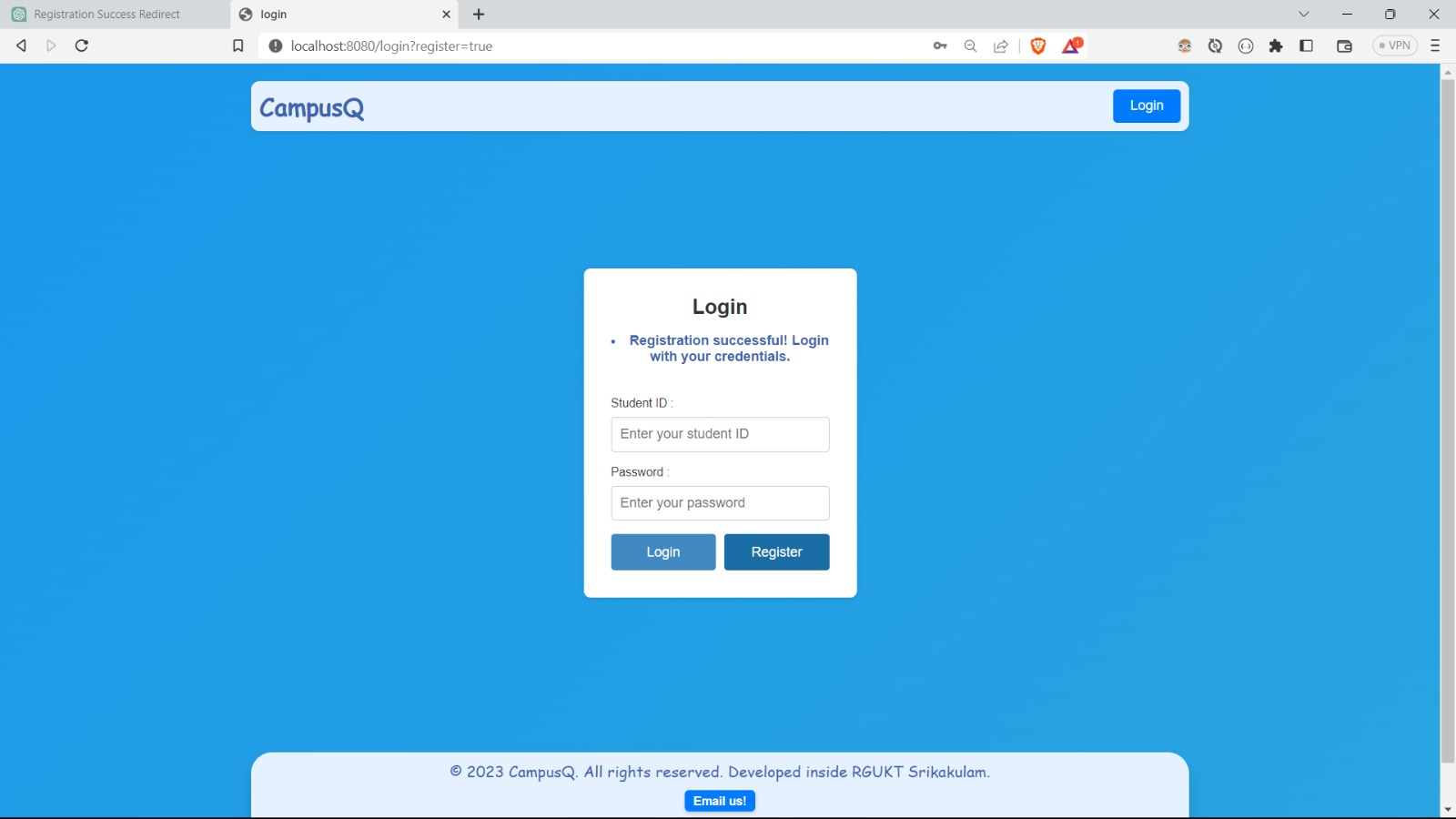
### **5.1. User Register**

* Here a user can register using his full name, id number, RGUKT mail id.
* There will be an option to create a password for their account.
* After filling every detail the registration will be completed.

**Fig .5.1. User Register**

### **5.2 User Login**

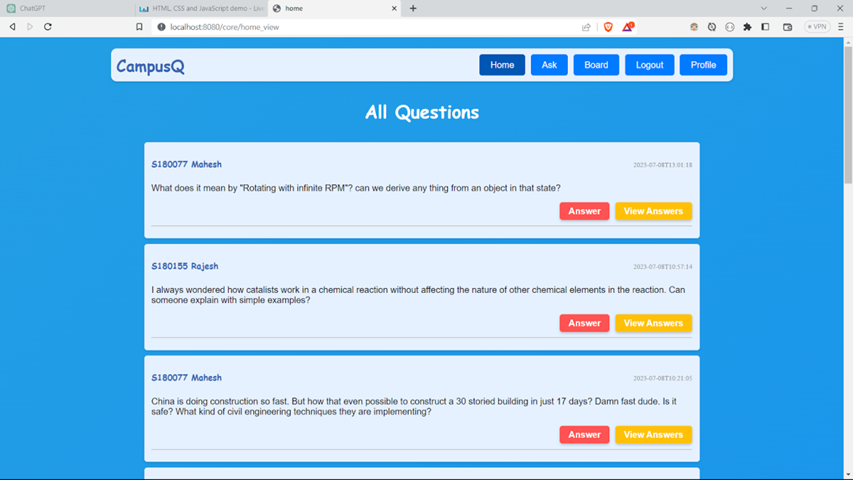
* A user can login to the application using his/her credentials. Here if the user will be authenticated.
* If the credentials are not valid then it will then generate an error message and ask to re-enter valid details.
* If the user doesn’t have an account then it asks to register as a new user.



**Fig. 5.2 User Login**

### **5.3 Home Page**

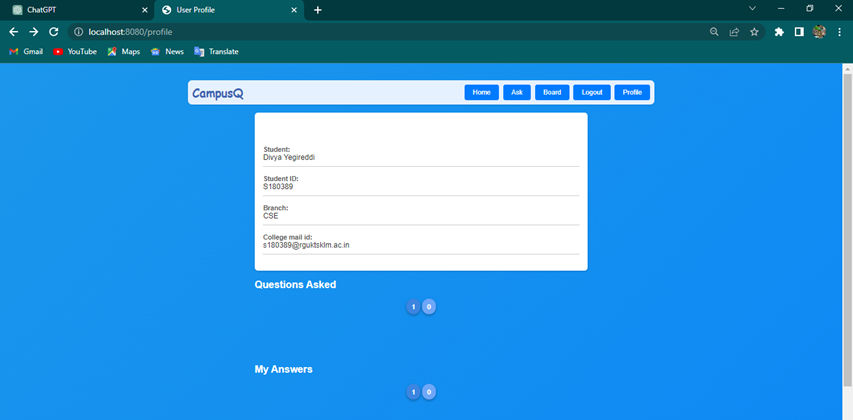
* After login successfully then it is redirected to the home page.
* It will be showing every question that has been asked up to that point of time.
* If the user wants to see the answers, he/she can click on the view answers button.
* If the user wants to answer others' questions, he/she can click on the answer button.
* The questions will be labeled with the id and name of the person who posted it.



**Fig. 5.3 Home Page**

### **5.4 Profile**

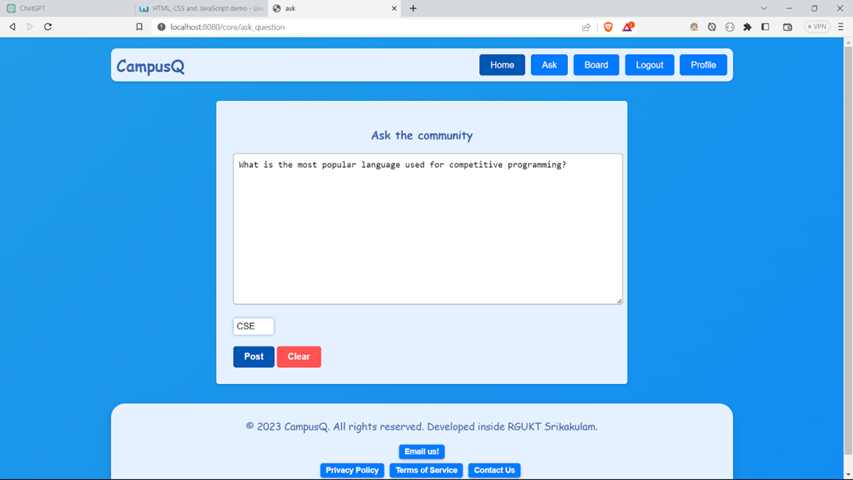
* After registering successfully it will generate a profile to the user.
* This function makes the user's registration-related information available.
* In addition, users can access the questions they've posted and the questions they've answered in their profiles.
* This profile button will appear on the right end of the home page navigator.



**Fig. 5.4 Profile**

### **5.5 Ask Page**

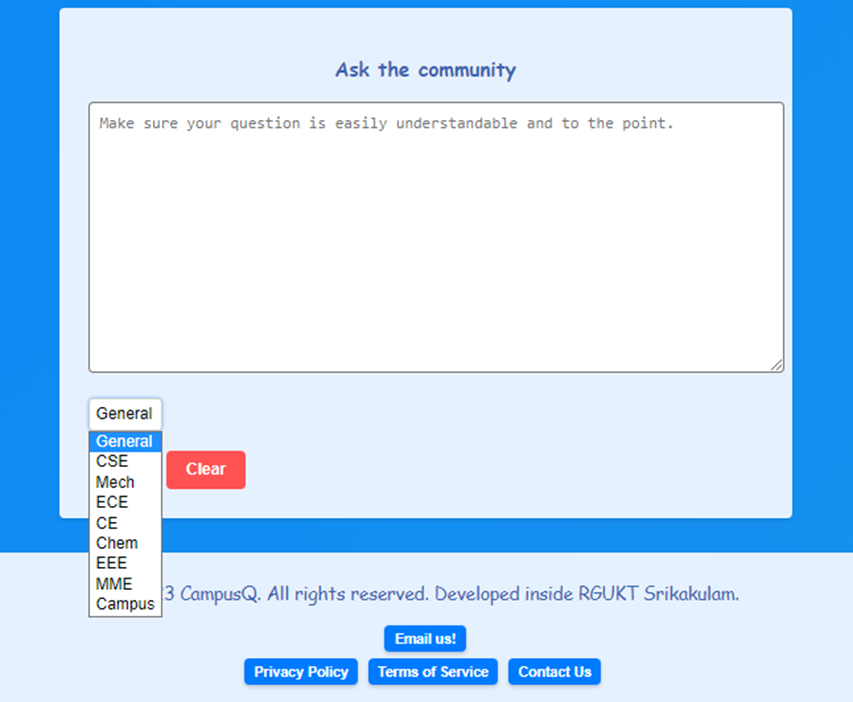
* In the home page navigator we can observe an ask button, if a user wants to ask a question then he clicks the button and then posts a question by typing in the space provided.
* Additionally, the user must choose a tag to indicate the sort of question he is asking; otherwise, it will be interpreted as a general question.
* Then the users have to click on the post button to post the question.
* It will get posted with his name.



**Fig 5.5 Ask Page**

### **5.6. Tagging**

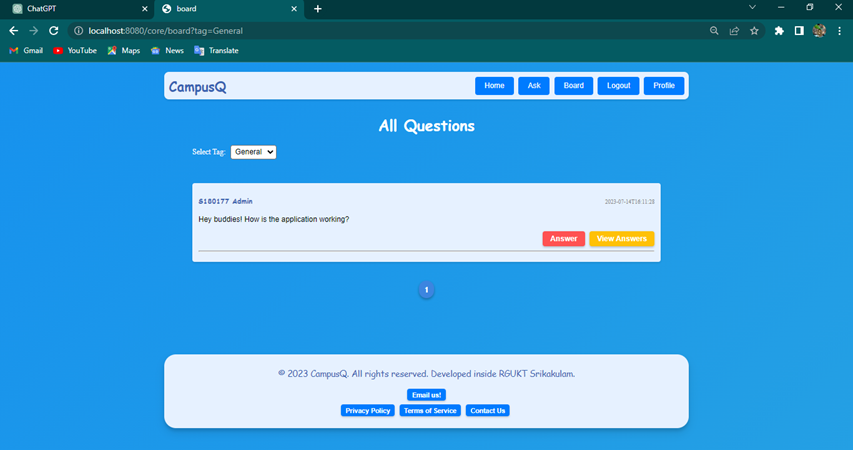
* It would be helpful for the user to filter the questions which he/she wanted.



**Fig 5.6. Tagging**

### **5.7 Board**

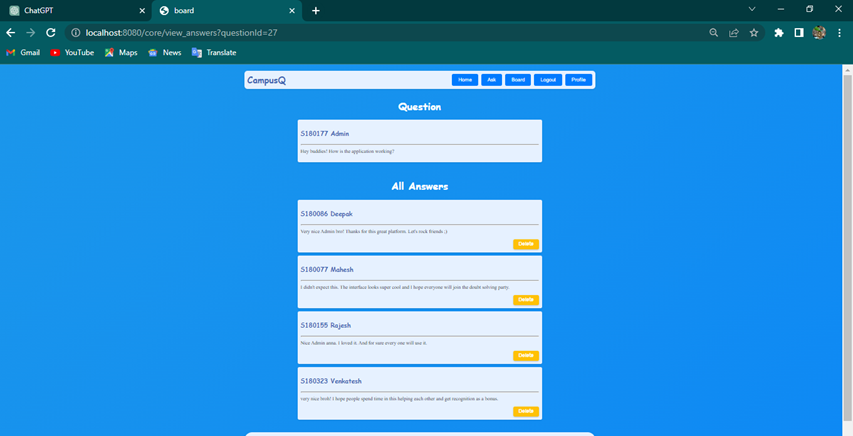
* The questions on the home page cannot be filtered by the user's interests.
* Filtering board functionality offers the user the choice to separate the questions in accordance with their needs.



**Fig. 5.7 Board**

### **5.8 View Answers for a particular Question**

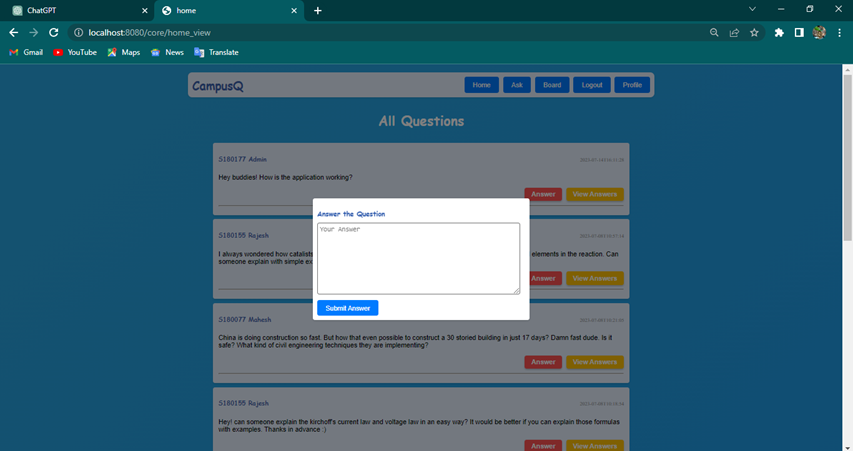
* If the user wants to see the answers, he/she can click on the view answers button.
* Then it will show the interface as shown in below:



**Fig. 5.8 View Answers for a particular Question**

### **5.9 Answer a Question**

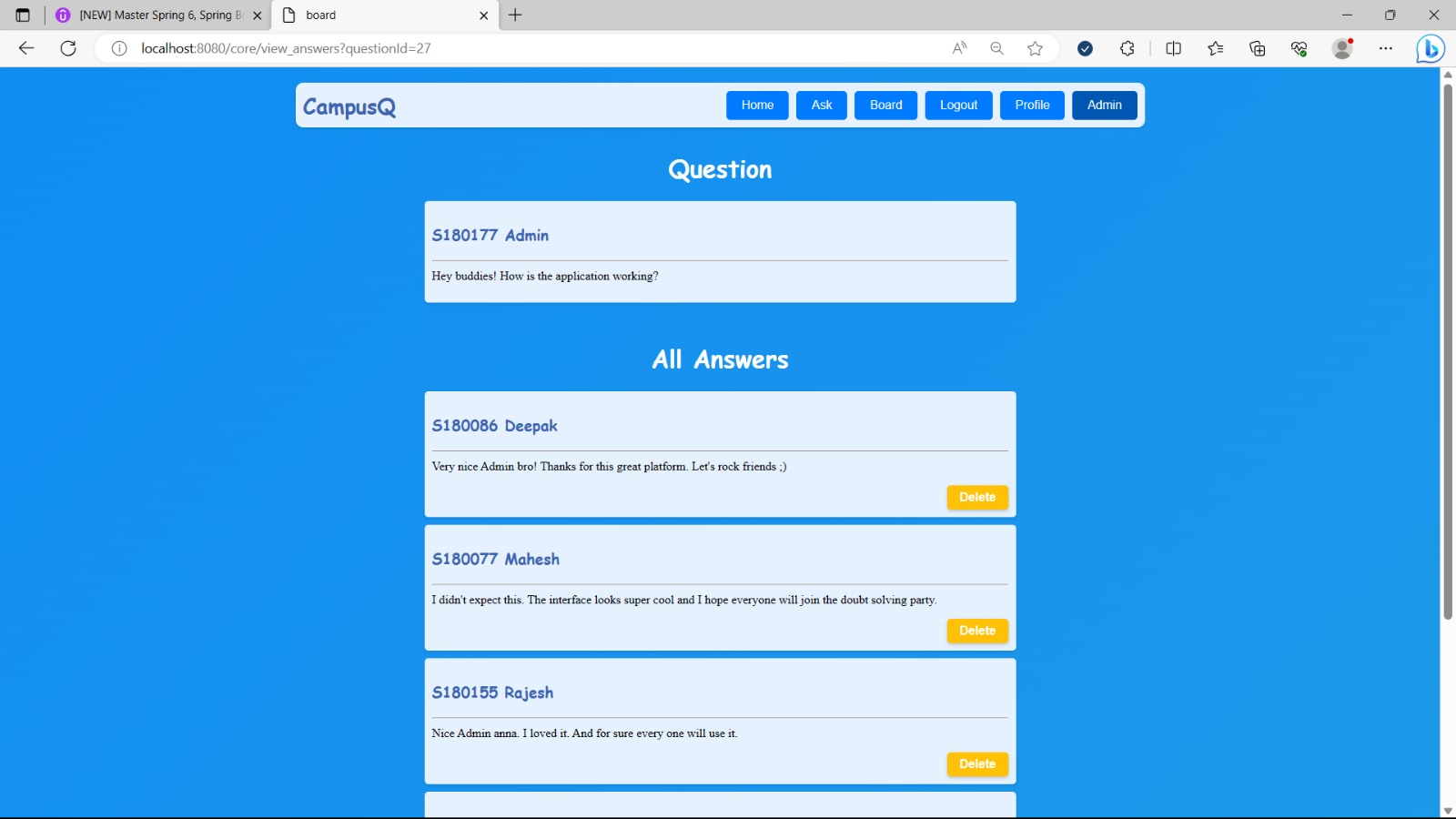
* In order to answer a question they have to click on the answer button below the questions.
* The user will then be provided space to input the response.
* After completing the answer, users have to click on the submit button to post the answer.



**Fig. 5.9 Answer a Question**

### **5.10 Admin**

* Admin can directly login to the application and he/she can perform all the operations performed by a student and additionally admin can delete others questions and answers operation.



**Fig. 5.10 Admin**

## 

## **CHAPTER -6**

## **SOURCE CODE**

### **6.1 Annotations (Package)**

#### **6.1.1 FieldsValueMatch.java**

package com.campusq.mini.annotation;

import com.campusq.mini.validations.FieldsValueMatchValidator;

import jakarta.validation.Constraint;

import jakarta.validation.Payload;

import java.lang.annotation.ElementType;

import java.lang.annotation.Retention;

import java.lang.annotation.RetentionPolicy;

import java.lang.annotation.Target;

@Constraint(validatedBy = FieldsValueMatchValidator.class)

@Target({ ElementType.TYPE })

@Retention(RetentionPolicy.RUNTIME)

public @interface FieldsValueMatch {

Class<?>[] groups() default {};

Class<? extends Payload>[] payload() default {};

String message() default "Fields values don't match!";

String field();

String fieldMatch();

@Target({ ElementType.TYPE })

@Retention(RetentionPolicy.RUNTIME)

@interface List {

FieldsValueMatch[] value();

}

}

### **6.2 Audit (Package)**

#### **6.2.1 AuditAwareClass.java**

package com.campusq.mini.audit;

import org.springframework.data.domain.AuditorAware;

import org.springframework.security.core.context.SecurityContextHolder;

import org.springframework.stereotype.Component;

import java.util.Optional;

@Component("auditAwareClass")

public class AuditAwareClass implements AuditorAware<String> {

@Override

public Optional<String> getCurrentAuditor(){

return Optional.ofNullable(

SecurityContextHolder

.getContext()

.getAuthentication()

.getName());

}

}

### **6.3 Configurations (Package)**

#### **6.3.1 ProjectSecurityConfig.java**

package com.campusq.mini.config;

import lombok.extern.slf4j.Slf4j;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.core.userdetails.User;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

import org.springframework.security.web.SecurityFilterChain;

@Configuration

public class ProjectSecurityConfig {

@Bean

SecurityFilterChain securityCustomConfig(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeHttpRequests()

.requestMatchers("/home").permitAll()

.requestMatchers("/core/\*\*").authenticated()

.requestMatchers("/profile/\*\*").authenticated()

.requestMatchers("/register").permitAll()

.requestMatchers("register\_user").permitAll()

.requestMatchers("/assets/\*\*").permitAll()

.and().formLogin().loginPage("/login")

.defaultSuccessUrl("/core/home\_view").failureUrl("/login?error=true").permitAll()

.and().logout().logoutSuccessUrl("/login?logout=true").invalidateHttpSession(true).permitAll()

.and().httpBasic();

return http.build();

}

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

}

### **6.4 Controller (Package)**

#### **6.4.1 AdminController.java**

package com.campusq.mini.controller;

import com.campusq.mini.model.Answer;

import com.campusq.mini.model.Question;

import com.campusq.mini.repository.AnswerRepository;

import com.campusq.mini.repository.QuestionRepository;

import com.campusq.mini.repository.StudentRepository;

import com.campusq.mini.service.QuestionService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.core.Authentication;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import java.util.List;

@Controller

@RequestMapping("admin")

public class AdminController {

@Autowired

QuestionRepository questionRepository;

@Autowired

QuestionService questionService;

@Autowired

StudentRepository studentRepository;

@Autowired

AnswerRepository answerRepository;

@RequestMapping(value={"/admin\_delete\_question\_home"}, method = {RequestMethod.GET, RequestMethod.POST})

public String adminDeleteQuestion(@RequestParam(value = "questionId", required = true) int questionId, Authentication authentication){

boolean isAdmin = authentication.getAuthorities().stream()

.anyMatch(role -> role.getAuthority().equals("ROLE\_ADMIN"));

if (isAdmin) {

Question question = questionRepository.findByQuestionId(questionId);

List<Answer> answers = answerRepository.findByQuestionQuestionId(questionId);

for (Answer answer : answers) {

answerRepository.delete(answer);

}

questionRepository.delete(question);

return "redirect:/core/home\_view";

} else {

throw new SecurityException("You are not allowed to perform this operation. Admin access required.");

}

}

@RequestMapping(value={"/admin\_delete\_answer"}, method = {RequestMethod.GET, RequestMethod.POST})

public String adminDeleteAnswer(@RequestParam(value = "questionId", required = true) int questionId,

@RequestParam(value = "answerId", required = true) int answerId, Authentication authentication){

boolean isAdmin = authentication.getAuthorities().stream()

.anyMatch(role -> role.getAuthority().equals("ROLE\_ADMIN"));

if (isAdmin) {

Answer answer = answerRepository.findByAnswerId(answerId);

answerRepository.delete(answer);

return "redirect:/core/view\_answers?questionId=" + questionId;

} else {

throw new SecurityException("You are not allowed to perform this operation. Admin access required.");

}

}

}

#### **6.4.2 AskController.java**

package com.campusq.mini.controller;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

@Controller

public class AskController {

@RequestMapping(value={"/ask"})

public String askquestion(){

return "ask.html";

}

}

#### **6.4.3 CoreController.java**

package com.campusq.mini.controller;

import com.campusq.mini.model.Answer;

import com.campusq.mini.model.Question;

import com.campusq.mini.model.Student;

import com.campusq.mini.repository.AnswerRepository;

import com.campusq.mini.repository.QuestionRepository;

import com.campusq.mini.repository.StudentRepository;

import com.campusq.mini.service.QuestionService;

import jakarta.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.security.core.Authentication;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.Errors;

import org.springframework.web.bind.annotation.\*;

import org.springframework.web.servlet.ModelAndView;

import java.util.ArrayList;

import java.util.List;

import java.util.Optional;

@Controller

@RequestMapping("core")

public class CoreController {

@Autowired

QuestionRepository questionRepository;

@Autowired

QuestionService questionService;

@Autowired

StudentRepository studentRepository;

@Autowired

AnswerRepository answerRepository;

@RequestMapping(value = {"/ask\_question"}, method = {RequestMethod.GET, RequestMethod.POST})

public ModelAndView askQuestion(@RequestParam(value = "post", required = false) String post,

@RequestParam(value = "fail", required = false) String fail){

ModelAndView modelAndView = new ModelAndView("ask.html");

modelAndView.addObject("question", new Question());

String message = null;

if(post != null) {

message = "Question Posted! Wait for the Answers.";

}else if(fail != null){

message = "Failed to post! Please check and try again.";

}

modelAndView.addObject("message", message);

return modelAndView;

}

@RequestMapping(value={"/post\_question"}, method = {RequestMethod.GET, RequestMethod.POST})

public String postNewQuestion(@Valid @ModelAttribute("question") Question question, Errors errors, Authentication authentication){

if(errors.hasErrors()){

return "redirect:/core/ask\_question?fail=true";

}

Student student = studentRepository.findByStudentId(authentication.getName());

question.setStudent(student);

questionRepository.save(question);

return "redirect:/core/ask\_question?post=true";

}

@RequestMapping(value={"/board"}, method = {RequestMethod.GET, RequestMethod.POST})

public String viewBoard(@RequestParam(value = "tag", defaultValue = "Campus") String tag,

@RequestParam(value = "page", defaultValue = "1") int page, Model model){

Page<Question> questions = questionService.findBoardQuestions(tag, page);

model.addAttribute("questions", questions.getContent());

model.addAttribute("selectedTag", tag);

model.addAttribute("currentPage", questions.getNumber()+1);

model.addAttribute("totalPages", questions.getTotalPages());

return "board";

}

@RequestMapping(value={"/home\_view"}, method = {RequestMethod.GET, RequestMethod.POST})

public ModelAndView viewHome(@RequestParam(defaultValue = "1") int page){

Page<Question> questions = questionService.findQuestions(page);

ModelAndView modelAndView = new ModelAndView("home.html");

modelAndView.addObject("questions", questions.getContent());

modelAndView.addObject("currentPage", questions.getNumber()+1);

modelAndView.addObject("totalPages", questions.getTotalPages());

return modelAndView;

}

@RequestMapping(value={"/save\_answer"}, method = {RequestMethod.GET, RequestMethod.POST})

public String saveAnswer(@RequestParam(value = "questionId", required = true) int questionId,

@RequestParam(value = "answerText", required = true, defaultValue = "") String answerText,

Authentication authentication) {

Answer answer = new Answer();

answer.setDescription(answerText);

if(answer.getDescription().isBlank()){

return "redirect:/core/home\_view";

}

Question question = questionRepository.findByQuestionId(questionId);

answer.setQuestion(question);

Student student = studentRepository.findByStudentId(authentication.getName());

answer.setStudent(student);

answerRepository.save(answer);

return "redirect:/core/home\_view"; // Redirect to the board page after submitting the answer

}

@RequestMapping(value={"/save\_answer\_from\_Board"}, method = {RequestMethod.GET, RequestMethod.POST})

public String saveAnswerFromBoard(@RequestParam(value = "questionId", required = true) int questionId,

@RequestParam(value = "answerText", required = true, defaultValue = "") String answerText,

@RequestParam(value = "selectedTag", required = true, defaultValue = "Campus") String selectedTag,

Authentication authentication) {

Answer answer = new Answer();

answer.setDescription(answerText);

if(answer.getDescription().isBlank()){

return "redirect:/core/board?tag=" + selectedTag;

}

Question question = questionRepository.findByQuestionId(questionId);

answer.setQuestion(question);

Student student = studentRepository.findByStudentId(authentication.getName());

answer.setStudent(student);

answerRepository.save(answer);

return "redirect:/core/board?tag=" + selectedTag; // Redirect to the board page after submitting the answer

}

@RequestMapping(value={"/view\_answers"}, method = {RequestMethod.GET, RequestMethod.POST})

public ModelAndView viewAnswers(@RequestParam(value = "questionId", required = true) int questionId){

ModelAndView modelAndView = new ModelAndView("answers.html");

Question question = questionRepository.findByQuestionId(questionId);

List<Answer> answers = answerRepository.findByQuestionQuestionId(questionId);

modelAndView.addObject("question", question);

modelAndView.addObject("answers", answers);

return modelAndView;

}

@RequestMapping(value={"/more"}, method = {RequestMethod.GET, RequestMethod.POST})

public ModelAndView moreOptions(@RequestParam(value = "questionId", required = true) int questionId){

ModelAndView modelAndView = new ModelAndView("more.html");

Question question = questionRepository.findByQuestionId(questionId);

modelAndView.addObject("question", question);

return modelAndView;

}

@RequestMapping(value={"/delete\_question"}, method = {RequestMethod.GET, RequestMethod.POST})

public String deleteQuestion(@RequestParam(value = "questionId", required = true) int questionId, Authentication authentication){

Question question = questionRepository.findByQuestionId(questionId);

if (!authentication.getName().equals(question.getStudent().getStudentId())) {

throw new SecurityException("You are not allowed to perform this operation! Current User: " + authentication.getName() + ", Author ID: " + question.getStudent().getStudentId());

}

List<Answer> answers = answerRepository.findByQuestionQuestionId(questionId);

for (Answer answer : answers) {

answerRepository.delete(answer);

}

questionRepository.delete(question);

return "redirect:/profile";

}

@RequestMapping(value={"/delete\_answer"}, method = {RequestMethod.GET, RequestMethod.POST})

public String deleteAnswer(@RequestParam(value = "answerId", required = true) int answerId, Authentication authentication){

Answer answer = answerRepository.findByAnswerId(answerId);

if (!authentication.getName().equals(answer.getStudent().getStudentId())) {

throw new SecurityException("You are not allowed to perform this operation! Current User: " + authentication.getName() + ", Author ID: " + answer.getStudent().getStudentId());

}

answerRepository.delete(answer);

return "redirect:/profile";

}

}

#### **6.4.4 BoardController.java**

package com.campusq.mini.controller;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

@Controller

public class FeedController {

@RequestMapping(value={"/board"})

public String viewQuestions(){

return "board.html";

}

}

#### **6.4.5 GlobalExceptionController.java**

/\*

package com.campusq.mini.controller;

import lombok.extern.slf4j.Slf4j;

import org.springframework.web.bind.annotation.ControllerAdvice;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.servlet.ModelAndView;

@Slf4j

@ControllerAdvice

public class GlobalExceptionController {

@ExceptionHandler(Exception.class)

public ModelAndView exceptionHandler(Exception exception){

ModelAndView errorPage = new ModelAndView();

errorPage.setViewName("error");

errorPage.addObject("errorMessage", exception.getMessage());

return errorPage;

}

}

\*/

**6.4.6 HomeController.java**

package com.campusq.mini.controller;

import com.campusq.mini.model.Question;

import com.campusq.mini.repository.QuestionRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.servlet.ModelAndView;

@Controller

public class HomeController {

@Autowired

QuestionRepository questionRepository;

@RequestMapping(value={"", "/", "home"})

public String displayHome(Model model){

model.addAttribute("username", "RGUKTian");

return "home.html";

}

}

#### **6.4.7 LoginController.java**

#### 

package com.campusq.mini.controller;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import org.springframework.security.core.Authentication;

import org.springframework.security.core.context.SecurityContextHolder;

import org.springframework.security.web.authentication.logout.SecurityContextLogoutHandler;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

@Controller

public class LoginController {

@RequestMapping(value="/login", method = {RequestMethod.GET, RequestMethod.POST})

public String displayLogin(@RequestParam(value = "error", required = false) String error,

@RequestParam(value = "logout", required = false) String logout,

Model model) {

String errorMessage = null;

if(error != null){

errorMessage = "Username and Password Mismatch!";

}else if(logout != null){

errorMessage = "You have been successfully logged out!";

}

model.addAttribute("errorMessage", errorMessage);

return "login.html";

}

@RequestMapping(value="/logout", method = RequestMethod.GET)

public String logoutAction(HttpServletRequest request, HttpServletResponse response){

Authentication auth = SecurityContextHolder.getContext().getAuthentication();

if(auth != null){

new SecurityContextLogoutHandler().logout(request, response, auth);

}

return "redirect:/login?logout=true";

}

}

#### **6.4.8 StudentController.java**

package com.campusq.mini.controller;

import com.campusq.mini.model.Answer;

import com.campusq.mini.model.Question;

import com.campusq.mini.model.Student;

import com.campusq.mini.repository.QuestionRepository;

import com.campusq.mini.repository.StudentRepository;

import com.campusq.mini.service.AnswerService;

import com.campusq.mini.service.QuestionService;

import com.campusq.mini.service.StudentService;

import jakarta.validation.Valid;

import lombok.extern.slf4j.Slf4j;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.security.core.Authentication;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.Errors;

import org.springframework.validation.ObjectError;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.servlet.ModelAndView;

import java.util.List;

import static org.springframework.web.bind.annotation.RequestMethod.POST;

@Slf4j

@Controller

public class StudentController {

private final StudentService studentService;

@Autowired

public StudentController(StudentService studentService){

this.studentService = studentService;

}

@Autowired

StudentRepository studentRepository;

@Autowired

QuestionRepository questionRepository;

@Autowired

QuestionService questionService;

@Autowired

AnswerService answerService;

@RequestMapping(value={"/register"}, method = {RequestMethod.GET, RequestMethod.POST})

public String newUser(@RequestParam(value = "success", required = false) String success,

@RequestParam(value = "failed", required = false) String failed,

Model model){

String message = null;

if(success != null){

message = "Registration Successful! Login with your credentials.";

}else if(failed != null){

message = "Registration Failed! Try again.";

}

model.addAttribute("message", message);

model.addAttribute("student", new Student());

//throw new RuntimeException("OOPS!! something went wrong!");

return "register.html";

}

@RequestMapping(value = {"/register\_user"}, method = {RequestMethod.GET, RequestMethod.POST})

public String registerStudent(@Valid @ModelAttribute("student") Student student, Errors errors){

if(errors.hasErrors()){

return "register.html";

}

boolean isSaved = studentService.saveStudentDetails(student);

if(isSaved){

return "redirect:/login?register=true";

} else {

return "redirect:/register?failed=true";

}

}

@RequestMapping(value = {"/profile"}, method = {RequestMethod.GET, RequestMethod.POST})

public ModelAndView studentProfile(@RequestParam(defaultValue = "1") int page, Authentication authentication){

ModelAndView modelAndView = new ModelAndView("profile.html");

Student student = studentRepository.findByStudentId(authentication.getName());

modelAndView.addObject("user", student);

Page<Question> questions = questionService.findProfileQuestions(student, page);

Page<Answer> answers = answerService.findMyAnswers(student, page);

modelAndView.addObject("userQuestions", questions.getContent());

modelAndView.addObject("userAnswers", answers.getContent());

modelAndView.addObject("QueCurrentPage", questions.getNumber()+1);

modelAndView.addObject("QueTotalPages", questions.getTotalPages());

modelAndView.addObject("AnsCurrentPage", answers.getNumber()+1);

modelAndView.addObject("AnsTotalPages", answers.getTotalPages());

return modelAndView;

}

}

### **6.5 Model (Package)**

#### **6.5.1 Answer.java**

package com.campusq.mini.model;

import jakarta.persistence.\*;

import jakarta.validation.constraints.NotBlank;

import lombok.Data;

import org.hibernate.annotations.GenericGenerator;

@Data

@Entity

@Table(name = "answers")

public class Answer extends BaseClass{

@Id

@GeneratedValue(strategy = GenerationType.AUTO, generator="native")

@GenericGenerator(name = "native", strategy = "native")

private int answerId;

@NotBlank(message = "Cant submit empty block!")

private String description;

@OneToOne(fetch = FetchType.LAZY, cascade = CascadeType.PERSIST, targetEntity = Student.class)

@JoinColumn(name = "created\_by", referencedColumnName = "studentId", nullable = false)

private Student student;

@OneToOne(fetch = FetchType.LAZY, cascade = CascadeType.PERSIST, targetEntity = Question.class)

@JoinColumn(name = "mother\_id", referencedColumnName = "questionId", nullable = false)

private Question question;

}

#### **6.5.2 BaseClass.java**

package com.campusq.mini.model;

import jakarta.persistence.Column;

import jakarta.persistence.EntityListeners;

import jakarta.persistence.MappedSuperclass;

import lombok.Data;

import org.springframework.data.annotation.CreatedBy;

import org.springframework.data.annotation.CreatedDate;

import org.springframework.data.jpa.domain.support.AuditingEntityListener;

import java.time.LocalDateTime;

import java.time.format.DateTimeFormatter;

@Data

@MappedSuperclass

@EntityListeners(AuditingEntityListener.class)

public class BaseClass {

@CreatedDate

@Column(updatable=false)

private LocalDateTime createdAt;

public String getCreatedAtFormatted() {

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");

return createdAt.format(formatter);

}

}

#### **6.5.3 Question.java**

package com.campusq.mini.model;

import jakarta.persistence.\*;

import jakarta.validation.constraints.NotBlank;

import lombok.Data;

import org.hibernate.annotations.GenericGenerator;

import java.time.format.DateTimeFormatter;

@Data

@Entity

@Table(name = "questions")

public class Question extends BaseClass{

@Id

@GeneratedValue(strategy = GenerationType.AUTO, generator="native")

@GenericGenerator(name = "native", strategy = "native")

private int questionId;

@NotBlank(message = "Cant submit empty block!")

private String description;

@NotBlank(message = "Cant submit with out tag selected!")

private String tag;

@OneToOne(fetch = FetchType.LAZY, cascade = CascadeType.PERSIST, targetEntity = Student.class)

@JoinColumn(name = "created\_by", referencedColumnName = "studentId", nullable = false)

private Student student;

@Override

public String toString() {

return "qId=" + questionId + "]";

}

}

#### **6.5.4 Roles.java**

package com.campusq.mini.model;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import lombok.Data;

import org.hibernate.annotations.GenericGenerator;

@Data

@Entity

public class Roles {

@Id

@GeneratedValue(strategy= GenerationType.AUTO,generator="native")

@GenericGenerator(name = "native",strategy = "native")

private int roleId;

private String roleName;

}

#### **6.5.5 Student.java**

package com.campusq.mini.model;

import com.campusq.mini.annotation.FieldsValueMatch;

import com.fasterxml.jackson.annotation.JsonIgnore;

import jakarta.persistence.\*;

import jakarta.validation.constraints.NotBlank;

import jakarta.validation.constraints.Pattern;

import jakarta.validation.constraints.Size;

import lombok.Data;

import lombok.Getter;

import lombok.Setter;

import java.util.Set;

@Getter

@Setter

@Entity

@FieldsValueMatch.List({

@FieldsValueMatch(

field = "password",

fieldMatch = "confirmPwd",

message = "Passwords do not match!"

)

})

public class Student {

@Id

@NotBlank(message="ID Number must not be blank!")

@Pattern(regexp = "^S18(0[0-9]{3}|1[01][0-9]{2}|1200)$", message = "Invalid student ID!")

private String studentId;

@NotBlank(message="First Name must not be blank!")

@Size(min=3, message="First Name must be at least 3 characters long!")

private String firstName;

@NotBlank(message="Last Name must not be blank!")

@Size(min=3, message="Last Name must be at least 3 characters long!")

private String lastName;

private String branch;

@NotBlank(message="Email must not be blank!")

@Pattern(regexp = "^s18(0[0-9]{3}|1[01][0-9]{2}|1200)@rguktsklm\\.ac\\.in$", message = "Invalid student email")

private String email;

@NotBlank(message="Password must be 8 to 12 characters long!")

@Pattern(regexp = "^(?=.\*[A-Z])(?=.\*[a-z])(?=.\*[0-9])(?=.\*[@#$%^&+=\_])(?=\\S+$).{8,12}$", message = "Invalid password")

private String password;

@NotBlank(message="Password must be 8 to 12 characters long!")

@Pattern(regexp = "^(?=.\*[A-Z])(?=.\*[a-z])(?=.\*[0-9])(?=.\*[@#$%^&+=\_])(?=\\S+$).{8,12}$", message = "Invalid Confirm password")

@Transient

private String confirmPwd;

@OneToOne(fetch = FetchType.EAGER, cascade = CascadeType.PERSIST, targetEntity = Roles.class)

@JoinColumn(name = "role\_id", referencedColumnName = "roleId", nullable = false)

private Roles roles;

@Override

public String toString() {

return studentId + " " + firstName;

}

}

### **6.6 Repository (Package)**

#### **6.6.1 AnswerRepository.java**

package com.campusq.mini.repository;

import com.campusq.mini.model.Answer;

import com.campusq.mini.model.Question;

import com.campusq.mini.model.Student;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.CrudRepository;

import org.springframework.stereotype.Repository;

import java.util.List;

@Repository

public interface AnswerRepository extends JpaRepository<Answer, Integer> {

List<Answer> findByQuestionQuestionId(int questionId);

void deleteByAnswerId(int id);

Page<Answer> findByStudent(Student student, Pageable pageable);

Answer findByAnswerId(int aId);

}

#### **6.6.2 QuestionRepository.java**

package com.campusq.mini.repository;

import com.campusq.mini.model.Roles;

import com.campusq.mini.model.Student;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.repository.CrudRepository;

import org.springframework.stereotype.Repository;

import com.campusq.mini.model.Question;

import java.util.List;

@Repository

public interface QuestionRepository extends JpaRepository<Question, Integer> {

Question findByQuestionId(int qId);

Page<Question> findByStudent(Student student, Pageable pageable);

List<Question> findAllByOrderByCreatedAtDesc();

Page<Question> findAllByTag(String tag, Pageable pageable);

Page<Question> findAll(Pageable pageable);

void deleteByQuestionId(Integer id);

}

#### **6.6.3 RolesRepository.java**

package com.campusq.mini.repository;

import com.campusq.mini.model.Roles;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface RolesRepository extends JpaRepository<Roles, Integer> {

Roles getByRoleName(String roleName);

}

#### **6.6.4 StudentRepository.java**

package com.campusq.mini.repository;

import com.campusq.mini.model.Student;

import org.springframework.data.repository.CrudRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface StudentRepository extends CrudRepository<Student, String> {

Student findByStudentId(String studentId);

}

### **6.7 Security (Package)**

#### **6.7.1CampusqAuthenticationProvider.java**

package com.campusq.mini.security;

import com.campusq.mini.model.Roles;

import com.campusq.mini.model.Student;

import com.campusq.mini.repository.StudentRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.authentication.AuthenticationProvider;

import org.springframework.security.authentication.BadCredentialsException;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.Authentication;

import org.springframework.security.core.AuthenticationException;

import org.springframework.security.core.GrantedAuthority;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.stereotype.Component;

import java.util.ArrayList;

import java.util.List;

@Component

public class CampusqAuthenticationProvider implements AuthenticationProvider {

@Autowired

private StudentRepository studentRepository;

@Autowired

private PasswordEncoder passwordEncoder;

@Override

public Authentication authenticate(Authentication authentication)

throws AuthenticationException {

String sId = authentication.getName();

String pwd = authentication.getCredentials().toString();

Student student = studentRepository.findByStudentId(sId);

if(student != null && passwordEncoder.matches(pwd, student.getPassword())){

return new UsernamePasswordAuthenticationToken(student.getStudentId(), pwd, getGrantedAuthorities(student.getRoles()));

}else{

throw new BadCredentialsException("Invalid Credentials!");

}

}

private List<GrantedAuthority> getGrantedAuthorities(Roles roles) {

List<GrantedAuthority> grantedAuthorities = new ArrayList<>();

grantedAuthorities.add(new SimpleGrantedAuthority("ROLE\_"+roles.getRoleName()));

return grantedAuthorities;

}

@Override

public boolean supports(Class<?> authentication) {

return authentication.equals(UsernamePasswordAuthenticationToken.class);

}

}

### **6.8 Service (Package)**

#### **6.8.1 AnswerService.java**

package com.campusq.mini.service;

import com.campusq.mini.model.Answer;

import com.campusq.mini.model.Question;

import com.campusq.mini.model.Student;

import com.campusq.mini.repository.AnswerRepository;

import com.campusq.mini.repository.QuestionRepository;

import com.campusq.mini.repository.StudentRepository;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

@Slf4j

@Service

public class AnswerService {

@Autowired

private QuestionRepository questionRepository;

@Autowired

private StudentRepository studentRepository;

@Autowired

private AnswerRepository answerRepository;

public Page<Answer> findMyAnswers(Student student, int pageNum){

int pageSize = 10;

Pageable pageable = PageRequest.of(pageNum-1, pageSize, Sort.by("createdAt").descending());

Page<Answer> answers = answerRepository.findByStudent(student, pageable);

return answers;

}

}

#### **6.8.2 QuestionService.java**

package com.campusq.mini.service;

import com.campusq.mini.model.Answer;

import com.campusq.mini.model.Question;

import com.campusq.mini.model.Student;

import com.campusq.mini.repository.AnswerRepository;

import com.campusq.mini.repository.QuestionRepository;

import com.campusq.mini.repository.StudentRepository;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

@Slf4j

@Service

public class AnswerService {

@Autowired

private QuestionRepository questionRepository;

@Autowired

private StudentRepository studentRepository;

@Autowired

private AnswerRepository answerRepository;

public Page<Answer> findMyAnswers(Student student, int pageNum){

int pageSize = 10;

Pageable pageable = PageRequest.of(pageNum-1, pageSize, Sort.by("createdAt").descending());

Page<Answer> answers = answerRepository.findByStudent(student, pageable);

return answers;

}

}

#### **6.8.3 StudentService.java**

package com.campusq.mini.service;

import com.campusq.mini.model.Roles;

import com.campusq.mini.model.Student;

import com.campusq.mini.repository.RolesRepository;

import com.campusq.mini.repository.StudentRepository;

import lombok.extern.slf4j.Slf4j;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.stereotype.Service;

@Slf4j

@Service

public class StudentService {

@Autowired

private StudentRepository studentRepository;

@Autowired

private RolesRepository rolesRepository;

@Autowired

private PasswordEncoder passwordEncoder;

public boolean saveStudentDetails(Student student){

boolean isSaved = false;

Roles role = rolesRepository.getByRoleName("STUDENT");

student.setRoles(role);

student.setPassword(passwordEncoder.encode(student.getPassword()));

student = studentRepository.save(student);

if(student != null && student.getStudentId() != null){

isSaved = true;

}

return isSaved;

}

}

#### **6.8.4 MiniApplication.java**

package com.campusq.mini;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.data.jpa.repository.config.EnableJpaAuditing;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

@SpringBootApplication

@EnableJpaRepositories("com.campusq.mini.repository")

@EntityScan("com.campusq.mini.model")

@EnableJpaAuditing(auditorAwareRef = "auditAwareClass")

public class MiniApplication {

public static void main(String[] args) {

SpringApplication.run(MiniApplication.class, args);

}

}

### **6.9 Validations (Package)**

#### **6.9.1 FieldsValueMatchValidator.java**

package com.campusq.mini.validations;

import com.campusq.mini.annotation.FieldsValueMatch;

import jakarta.validation.ConstraintValidator;

import jakarta.validation.ConstraintValidatorContext;

import org.springframework.beans.BeanWrapperImpl;

public class FieldsValueMatchValidator

implements ConstraintValidator<FieldsValueMatch, Object> {

private String field;

private String fieldMatch;

@Override

public void initialize(FieldsValueMatch constraintAnnotation) {

this.field = constraintAnnotation.field();

this.fieldMatch = constraintAnnotation.fieldMatch();

}

@Override

public boolean isValid(Object value, ConstraintValidatorContext context) {

Object fieldValue = new BeanWrapperImpl(value)

.getPropertyValue(field);

Object fieldMatchValue = new BeanWrapperImpl(value)

.getPropertyValue(fieldMatch);

if (fieldValue != null) {

if(fieldValue.toString().startsWith("$2a")){

return true;

}else {

return fieldValue.equals(fieldMatchValue);

}

} else {

return fieldMatchValue == null;

}

}

}

### **6.10 Static CSS**

#### **6.10.1 Answers.css**

/\* styles.css \*/

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

}

@keyframes gradientAnimation {

0% {

background-position: 0% 50%;

}

50% {

background-position: 100% 50%;

}

100% {

background-position: 0% 50%;

}

}

.container {

width: 100%;

max-width: 800px;

margin: 0 auto;

padding: 10px;

box-sizing: border-box;

}

h1 {

text-align: center;

color: #ffffff;

font-family: 'Comic Sans MS', cursive, sans-serif;

}

.statement-list {

margin-top: 10px;

}

.statement {

background-color: #E6F1FF;

border-radius: 5px;

padding: 10px;

margin-bottom: 10px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}

.statement h3 {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 20px;

}

.statement p {

margin-top: 10px;

font-size: 16px;

}

.button-container {

display: flex;

justify-content: flex-end;

margin-top: 10px;

}

.button-container button,

.button-container a {

margin-left: 10px;

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

color: #fff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

}

.button-container button:first-child,

.button-container a:first-child {

margin-left: 0;

}

.button-container .btn-danger {

background-color: #FF5252;

}

.button-container .btn-warning {

background-color: #FFC107;

}

.button-container button:hover,

.button-container a:hover {

opacity: 0.8;

}

.statement hr {

margin-top: 10px;

}

/\* board.css \*/

.popup-overlay {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

display: none;

justify-content: center;

align-items: center;

z-index: 9999;

}

.popup {

background-color: #fff;

padding: 10px;

border-radius: 5px;

max-width: 500px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

width: 100%;

}

.popup h3 {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 16px;

margin-bottom: 10px;

}

.popup textarea {

width: 100%;

height: 160px;

padding: 10px;

font-size: 16px;

border-radius: 5px;

resize: vertical;

margin-bottom: 10px;

}

.popup .submit-answer-btn {

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

color: #fff;

background-color: #007bff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

}

.popup .submit-answer-btn:hover {

background-color: #0056b3;

}

#### **6.10.2 Ask.css**

/\* ask.css \*/

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

}

@keyframes gradientAnimation {

0% { background-position: 0% 50%; }

50% { background-position: 100% 50%; }

100% { background-position: 0% 50%; }

}

.form-container {

max-width: 800px;

margin: 35px auto; /\* Add margin for gap above and below the container \*/

padding: 20px;

background-color: #E6F1FF; /\* Set container background color \*/

border-radius: 5px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

width: 700px;

}

.form {

padding: 10px;

}

.form h2 {

text-align: center;

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 20px;

margin-bottom: 20px;

}

.form-group {

margin-top: 10px;

}

.form-group textarea {

width: 100%;

height: 250px;

padding: 10px;

font-size: 16px;

border-radius: 5px;

resize: vertical;

margin-bottom: 10px;

}

.form-group select {

width: auto; /\* Set the width to auto \*/

padding: 6px;

font-size: 16px;

border-radius: 5px;

margin-bottom: 10px;

appearance: none;

background-color: #fff;

border: 1px solid #ccc;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}

.form-group select:focus {

outline: none;

box-shadow: 0 0 5px rgba(0, 123, 255, 0.5);

}

.form-group button {

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

font-weight: bold;

color: #fff;

background-color: #007bff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

}

.form-group button[type="reset"] {

background-color: #FF5252;

}

.form-group button:hover {

background-color: #0056b3;

}

.alert {

margin-top: 10px;

padding: 10px;

border-radius: 5px;

}

.alert p {

color: #fff;

font-size: 16px;

text-align: center;

}

.alert .post {

background-color: #4CAF50;

}

.alert .fail {

background-color: #FF5252;

}

#### **6.10.3 Board.css**

/\*

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

}

@keyframes gradientAnimation {

0% { background-position: 0% 50%; }

50% { background-position: 100% 50%; }

100% { background-position: 0% 50%; }

}

.container {

width: 100%;

max-width: 1000px;

margin: 0 auto;

padding: 10px;

display: flex;

flex-direction: column;

align-items: stretch;

}

h1 {

text-align: center;

color: #ffffff;

font-family: 'Comic Sans MS', cursive, sans-serif;

}

.statement-list {

margin-top: 10px;

}

.statement {

background-color: #E6F1FF;

border-radius: 5px;

padding: 10px;

margin-bottom: 10px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}

.question-header {

display: flex;

justify-content: space-between;

align-items: baseline;

}

.question-header h3 {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 20px;

margin-right: 10px;

}

.question-header .created-at {

color: #888888;

font-size: 12px;

}

.question-text {

margin-top: 10px;

font-size: 16px;

}

.button-container {

display: flex;

justify-content: flex-end;

margin-top: 10px;

}

.button-container button,

.button-container a {

margin-left: 10px;

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

color: #fff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

}

.button-container button:first-child,

.button-container a:first-child {

margin-left: 0;

}

.button-container .btn-danger {

background-color: #FF5252;

}

.button-container .btn-warning {

background-color: #FFC107;

}

.button-container button:hover,

.button-container a:hover {

opacity: 0.8;

}

.statement hr {

margin-top: 10px;

}

.popup-overlay {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

display: none;

justify-content: center;

align-items: center;

z-index: 9999;

}

.popup {

background-color: #fff;

padding: 10px;

border-radius: 5px;

max-width: 500px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

width: 100%;

}

.popup h3 {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 16px;

margin-bottom: 10px;

}

.popup textarea {

width: 95%;

height: 160px;

padding: 5px;

font-size: 16px;

border-radius: 5px;

resize: vertical;

margin-bottom: 10px;

}

.popup .submit-answer-btn {

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

color: #fff;

background-color: #007bff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

}

.popup .submit-answer-btn:hover {

background-color: #0056b3;

}

.expandable-textarea {

overflow: hidden;

transition: height 0.3s ease;

}

.expandable-textarea.expanded {

height: 100px;

}

.tag-dropdown {

margin-bottom: 20px;

display: flex;

align-items: center;

}

.tag-dropdown label {

color: #fff;

font-size: 16px;

margin-right: 10px;

}

.tag-dropdown select {

padding: 5px;

font-size: 16px;

border-radius: 5px;

}

.statement-list {

margin-top: 30px;

}

.pagination {

display: flex;

justify-content: center;

margin-top: 20px;

margin-bottom: 120px;

}

.pagination a {

padding: 8px 12px;

text-decoration: none;

color: #fff;

background-color: #6ea6f8;

border: none;

border-radius: 20px;

margin: 0 2px;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

font-size: 16px;

font-weight: bold;

transition: background-color 0.3s;

box-shadow: 0 3px 6px rgba(0, 0, 0, 0.3);

}

.pagination a:hover {

background-color: #3c85e0;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);

}

.pagination .prev:before {

content: '<<';

font-weight: bold;

}

.pagination .next:after {

content: '>>';

font-weight: bold;

}

.pagination .prev:before {

margin-right: 5px;

}

.pagination .next:after {

margin-left: 5px;

}

.pagination .page {

font-weight: bold;

}

.pagination .active {

background-color: #3c85e0;

}

\*/

/\* board.css \*/

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

}

@keyframes gradientAnimation {

0% { background-position: 0% 50%; }

50% { background-position: 100% 50%; }

100% { background-position: 0% 50%; }

}

.container {

width: 100%;

max-width: 1000px;

margin: 0 auto;

padding: 10px;

display: flex;

flex-direction: column;

align-items: stretch;

}

h1 {

text-align: center;

color: #ffffff;

font-family: 'Comic Sans MS', cursive, sans-serif;

}

.statement-list {

margin-top: 10px;

}

.statement {

background-color: #E6F1FF;

border-radius: 5px;

padding: 13px;

margin-bottom: 10px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}

.question-header {

display: flex;

justify-content: space-between;

align-items: baseline;

}

.question-header h3 {

color: #3B64B1;

font-family: 'Roboto', cursive, sans-serif;

font-size: 15px;

margin-right: 10px;

}

.question-header .created-at {

color: #888888;

font-size: 12px;

}

.question-text {

margin-top: 5px;

font-size: 16px;

font-family: 'Roboto', sans-serif;

}

.button-container {

display: flex;

justify-content: flex-end;

margin-top: 10px;

}

.button-container button,

.button-container a {

margin-left: 10px;

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

font-weight: bold;

font-family: 'Roboto', sans-serif;

color: #fff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

text-decoration: none;

display: inline-block;

text-align: center;

line-height: 1;

vertical-align: middle;

}

.button-container button:first-child,

.button-container a:first-child {

margin-left: 0;

}

.button-container .btn-danger {

background-color: #FF5252;

}

.button-container .btn-warning {

background-color: #FFC107;

}

.button-container button:hover,

.button-container a:hover {

opacity: 0.8;

}

.button-container .btn-warning:hover {

text-decoration: none;

}

.button-container .btn-danger,

.button-container .btn-warning {

padding: 8px 16px;

}

.button-container .btn-danger {

background-color: #FF5252;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);

}

.button-container .btn-warning {

background-color: #FFC107;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);

}

.button-container .btn-danger:hover,

.button-container .btn-warning:hover {

opacity: 0.8;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

}

.statement hr {

margin-top: 10px;

}

.popup-overlay {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

display: none;

justify-content: center;

align-items: center;

z-index: 9999;

}

.popup {

background-color: #fff;

padding: 10px;

border-radius: 5px;

max-width: 500px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

width: 100%;

}

.popup h3 {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 16px;

margin-bottom: 10px;

}

.popup textarea {

width: 95%;

height: 160px;

padding: 5px;

font-size: 16px;

border-radius: 5px;

resize: vertical;

margin-bottom: 10px;

}

.popup .submit-answer-btn {

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

color: #fff;

background-color: #007bff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

}

.popup .submit-answer-btn:hover {

background-color: #0056b3;

}

.expandable-textarea {

overflow: hidden;

transition: height 0.3s ease;

}

.expandable-textarea.expanded {

height: 100px;

}

/\* board.css \*/

.tag-dropdown {

margin-bottom: 20px;

display: flex;

align-items: center;

}

.tag-dropdown label {

color: #fff;

font-size: 16px;

margin-right: 10px;

}

.tag-dropdown select {

padding: 5px;

font-size: 16px;

border-radius: 5px;

}

.statement-list {

margin-top: 30px;

}

.pagination {

display: flex;

justify-content: center;

margin-top: 20px;

margin-bottom: 120px;

}

.pagination a {

padding: 8px 12px;

text-decoration: none;

color: #fff;

Oops! It seems like the CSS code got cut off at the end. Here's the missing part:

```css

background-color: #6ea6f8;

border: none;

border-radius: 20px;

margin: 0 2px;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

font-size: 16px;

font-weight: bold;

transition: background-color 0.3s;

box-shadow: 0 3px 6px rgba(0, 0, 0, 0.3);

}

.pagination a:hover {

background-color: #3c85e0;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);

}

.pagination .prev:before {

content: '<<';

font-weight: bold;

}

.pagination .next:after {

content: '>>';

font-weight: bold;

}

.pagination .prev:before {

margin-right: 5px;

}

.pagination .next:after {

margin-left: 5px;

}

.pagination .page {

font-weight: bold;

}

.pagination .active {

background-color: #3c85e0;

}

#### **6.10.4 Error.css**

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

background-color: #4e98c6;

}

.container {

max-width: 800px;

margin: 70px auto; /\* Add margin to create a gap \*/

padding: 20px;

background-color: #fff;

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);

border-radius: 5px;

height: 50vh;

}

h1 {

font-size: 24px;

color: #333;

}

.error-message {

background-color: #f8d7da;

color: #721c24;

padding: 10px;

margin-top: 20px;

border-radius: 5px;

animation: neonError 2s linear infinite;

}

.error-message p {

margin: 0;

}

@keyframes neonError {

0% {

background-color: #f8d7da;

}

50% {

background-color: #ff394f;

}

100% {

background-color: #f8d7da;

}

}

#### **6.10.5 Footer.css**

/\* footer.css \*/

.footer {

background-color: #E6F1FF;

padding: 10px;

display: flex;

justify-content: center;

align-items: center;

box-shadow: 0 10px 20px rgba(0, 0, 0, 0.1), 0 6px 6px rgba(0, 0, 0, 0.1);

border-radius: 25px;

max-width: 1200px;

margin: 20px auto 0;

width: 1100px;

}

.footer .footer-content {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 18px;

margin: 0;

margin-bottom: 15px;

text-align: center;

}

.footer p {

margin-bottom: 0.5rem;

}

.footer a {

margin-left: 10px;

padding: 5px 10px;

border-radius: 5px;

font-size: 14px;

font-weight: bold;

color: #fff;

background-color: #007bff;

text-decoration: none;

transition: background-color 0.3s ease;

font-family: 'Arial', sans-serif;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);

}

.footer a:first-child {

margin-left: 0;

}

.footer a:hover {

opacity: 0.8;

}

.footer nav {

display: flex;

justify-content: center;

align-items: center;

margin-top: 10px;

}

.footer nav a {

margin: 0 5px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);

}

#### **6.10.6 Header.css**

/\* Reset styles

\* {

box-sizing: border-box;

margin: 0;

padding: 0;

}

body {

font-family: Arial, sans-serif;

font-size: 16px;

line-height: 1.5;

background-color: #4e98c6;

}

header {

background-color: #102f45;

padding: 1rem;

color: #ffffff;

}

header h2 {

font-size: 24px;

margin-bottom: 0.5rem;

font-family: "Comic Sans MS", cursive, sans-serif;

}

header p {

font-size: 16px;

margin-bottom: 1rem;

}

.navbar {

margin-top: 1rem;

display: flex;

justify-content: center;

}

\*/

/\* header.css \*/

/\* header.css \*/

header {

background-color: #E6F1FF;

padding: 10px;

display: flex;

justify-content: space-between;

align-items: center;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

border-radius: 10px; /\* Add border radius for a curvy look \*/

max-width: 1200px; /\* Set a maximum width for the container \*/

margin: 20px auto 0; /\* Center the container horizontally \*/

width: 1100px;

}

header h2 {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 28px;

margin: 0;

}

nav {

display: flex;

}

nav a {

margin-left: 10px;

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

color: #fff;

background-color: #007bff;

text-decoration: none;

transition: background-color 0.3s ease;

font-family: 'Arial', sans-serif; /\* Replace with the desired stylish font \*/

}

nav a.active {

background-color: #0056b3;

}

nav a:hover {

opacity: 0.8;

}

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

}

@keyframes gradientAnimation {

0% { background-position: 0% 50%; }

50% { background-position: 100% 50%; }

100% { background-position: 0% 50%; }

}

#### **6.10.7 Home.css**

/\* home.css \*/

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

}

@keyframes gradientAnimation {

0% { background-position: 0% 50%; }

50% { background-position: 100% 50%; }

100% { background-position: 0% 50%; }

}

.container {

width: 100%;

max-width: 1000px;

margin: 0 auto;

padding: 10px;

display: flex;

flex-direction: column;

align-items: stretch;

}

h1 {

text-align: center;

color: #ffffff;

font-family: 'Comic Sans MS', cursive, sans-serif;

}

.statement-list {

margin-top: 12px;

}

.statement {

background-color: #E6F1FF;

border-radius: 5px;

padding: 13px;

margin-bottom: 10px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}

.question-header {

display: flex;

justify-content: space-between;

align-items: baseline;

}

.question-header h3 {

color: #3B64B1;

font-family: 'Roboto', cursive, sans-serif;

font-size: 15px;

margin-right: 10px;

}

.question-header .created-at {

color: #888888;

font-size: 12px;

}

.question-text {

margin-top: 10px;

font-size: 16px;

font-family: 'Roboto', sans-serif;

}

.button-container {

display: flex;

justify-content: flex-end;

margin-top: 10px;

}

/\* home.css \*/

/\* home.css \*/

.button-container button,

.button-container a {

margin-left: 10px;

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

font-weight: bold;

font-family: 'Roboto', sans-serif;

color: #fff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

text-decoration: none;

display: inline-block;

text-align: center;

line-height: 1;

vertical-align: middle;

}

.button-container button:first-child,

.button-container a:first-child {

margin-left: 0;

}

.button-container .btn-danger {

background-color: #FF5252;

}

.button-container .btn-warning {

background-color: #FFC107;

}

.button-container button:hover,

.button-container a:hover {

opacity: 0.8;

}

.button-container .btn-warning:hover {

text-decoration: none;

}

.button-container .btn-danger,

.button-container .btn-warning {

padding: 8px 16px;

}

.button-container .btn-danger {

background-color: #FF5252;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);

}

.button-container .btn-warning {

background-color: #FFC107;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);

}

.button-container .btn-danger:hover,

.button-container .btn-warning:hover {

opacity: 0.8;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

}

.statement hr {

margin-top: 10px;

}

.popup-overlay {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

display: none;

justify-content: center;

align-items: center;

z-index: 9999;

}

.popup {

background-color: #fff;

padding: 10px;

border-radius: 5px;

max-width: 500px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

width: 100%;

}

.popup h3 {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 16px;

margin-bottom: 10px;

}

.popup textarea {

width: 95%;

height: 160px;

padding: 5px;

font-size: 16px;

border-radius: 5px;

resize: vertical;

margin-bottom: 10px;

}

.popup .submit-answer-btn {

padding: 10px 20px;

border-radius: 5px;

font-size: 16px;

color: #fff;

background-color: #007bff;

border: none;

cursor: pointer;

transition: background-color 0.3s ease;

}

.popup .submit-answer-btn:hover {

background-color: #0056b3;

}

.expandable-textarea {

overflow: hidden;

transition: height 0.3s ease;

}

.expandable-textarea.expanded {

height: 100px;

}

.pagination {

display: flex;

justify-content: center;

margin-top: 20px;

margin-bottom: 120px;

}

.pagination a {

padding: 8px 12px;

text-decoration: none;

color: #fff;

background-color: #6ea6f8;

border: none;

border-radius: 20px;

margin: 0 2px;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

font-size: 16px;

font-weight: bold;

transition: background-color 0.3s;

box-shadow: 0 3px 6px rgba(0, 0, 0, 0.3);

}

.pagination a:hover {

background-color: #3c85e0;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);

}

.pagination .prev:before {

content: '<<';

font-weight: bold;

}

.pagination .next:after {

content: '>>';

font-weight: bold;

}

.pagination .prev:before {

margin-right: 5px;

}

.pagination .next:after {

margin-left: 5px;

}

.pagination .page {

font-weight: bold;

}

.pagination .active {

background-color: #3c85e0;

}

#### 

#### **6.10.8 Login.css**

\* {

box-sizing: border-box;

margin: 0;

padding: 0;

}

body {

background-color: #4e98c6;

font-family: Arial, sans-serif;

}

.login-container {

display: flex;

justify-content: center;

align-items: center;

height: 80vh;

}

.login-form {

background-color: #ffffff;

border-radius: 8px;

box-shadow: 0 2px 6px rgba(0, 0, 0, 0.1);

padding: 2rem;

text-align: center;

width: 320px;

}

.login-form h2 {

font-size: 24px;

color: #333333;

margin-bottom: 1rem;

}

.form-group {

margin-bottom: 1rem;

text-align: left;

}

.form-group label {

display: block;

font-size: 14px;

color: #333333;

margin-bottom: 0.5rem;

}

.form-group input {

width: 100%;

padding: 10px;

border: 1px solid #cccccc;

border-radius: 4px;

font-size: 16px;

}

button {

width: 100%;

padding: 12px;

background-color: #4389C1;

color: #ffffff;

border: none;

border-radius: 4px;

font-size: 16px;

cursor: pointer;

transition: background-color 0.3s ease;

}

button:hover {

background-color: #1A6CA4;

}

.login-alerts {

color: red;

font-weight: bold;

}

.login-alerts-cool{

color: #3B64B1;

font-weight: bold;

}

.button-container {

display: flex;

justify-content: space-between;

margin-top: 1rem;

}

.login-button {

width: calc(50% - 5px);

padding: 12px;

background-color: #4389C1;

color: #ffffff;

border: none;

border-radius: 4px;

font-size: 16px;

cursor: pointer;

transition: background-color 0.3s ease;

}

.register-button {

width: calc(50% - 5px);

padding: 12px;

background-color: #1A6CA4;

color: #ffffff;

border: none;

border-radius: 4px;

font-size: 16px;

cursor: pointer;

text-decoration: none;

text-align: center;

transition: background-color 0.3s ease;

}

.register-button:hover {

background-color: #0E4B73;

}

#### **6.10.9 More.css**

/\* styles.css \*/

/\* Add your custom styles here \*/

.container {

max-width: 800px;

margin: 0 auto;

padding: 20px;

}

.container h1 {

color: #FFF; /\* Changed the color to white \*/

font-size: 24px;

margin-bottom: 15px;

font-family: "Roboto", sans-serif; /\* Added font family \*/

}

.container .statement {

margin-bottom: 20px;

background-color: #E6F1FF;

border-radius: 15px;

padding: 10px;

box-shadow: 0 2px 4px rgba(0.1, 0.1, 0.1, 0.1);

font-family: "Roboto", sans-serif; /\* Added font family \*/

}

.container .question-header {

display: flex;

justify-content: space-between;

align-items: center;

}

.container .created-at {

color: #888;

font-size: 14px;

}

.container .question-text {

margin-top: 10px;

font-size: 16px;

color: #000011;

}

.container hr {

border: none;

border-top: 1px solid #ccc;

margin: 20px 0;

}

.container .tag-dropdown {

margin-bottom: 20px;

display: flex;

align-items: center;

}

.container label {

color: #fff;

font-size: 18px;

margin-right: 10px;

font-weight: bold;

font-family: "Roboto", sans-serif; /\* Added font family \*/

}

.container select {

padding: 5px;

font-size: 16px;

border-radius: 5px;

background-color: #F5F5F5;

border: 1px solid #ccc;

}

.container .button-container {

margin-top: 20px;

display: flex;

flex-direction: column;

align-items: flex-start;

}

.container .btn {

display: inline-block;

padding: 8px 16px;

background-color: #FF5252;

color: #fff;

text-decoration: none;

border-radius: 10px;

transition: background-color 0.3s;

font-size: 16px;

border: none;

cursor: pointer;

font-family: "Roboto", sans-serif;

}

.container .btn:hover {

background-color: #FF0000;

}

.container p {

font-size: 16px;

color: #FFF; /\* Changed the color to white \*/

margin-top: 10px;

text-align: justify;

text-decoration: none;

font-family: "Roboto", sans-serif;

font-weight: bold; /\* Added font-weight property \*/

}

.container .warning-message {

color: #FFF;

font-weight: bold;

font-size: 18px;

text-align: center;

margin-top: 15px;

}

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

}

@keyframes gradientAnimation {

0% { background-position: 0% 50%; }

50% { background-position: 100% 50%; }

100% { background-position: 0% 50%; }

}

#### **6.10.10 Profile.css**

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

font-family: Arial, sans-serif;

margin: 0;

padding: 20px;

}

@keyframes gradientAnimation {

0% {

background-position: 0% 50%;

}

50% {

background-position: 100% 50%;

}

100% {

background-position: 0% 50%;

}

}

.container {

max-width: 800px;

margin: 20px auto;

}

.profile-card {

background-color: #fff;

border-radius: 8px;

padding: 20px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

margin-bottom: 20px;

}

.container h1 {

margin-top: 0;

color: #fff;

}

.container h2 {

margin-top: 0;

color: #fff;

}

.profile-info {

margin-bottom: 20px;

border-bottom: 1px solid #ccc;

padding-bottom: 10px;

}

.profile-info label {

font-weight: bold;

font-size: 16px;

color: #555;

}

.profile-info p {

margin: 0;

font-size: 18px;

color: #333;

}

.profile-info .user-fullname {

font-size: 24px;

color: #3B64B1;

margin-bottom: 10px;

}

.profile-info .user-role {

font-style: italic;

color: #999;

}

.profile-info .user-email {

color: #555;

}

.profile-info .user-social {

margin-top: 10px;

}

.profile-info .user-social a {

display: inline-block;

margin-right: 10px;

color: #555;

text-decoration: none;

transition: color 0.3s;

}

.profile-info .user-social a:hover {

color: #3B64B1;

}

.statement-list {

margin-top: 10px;

}

.statement {

background-color: #E6F1FF;

border-radius: 5px;

padding: 10px;

margin-bottom: 10px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}

.statement h3 {

color: #3B64B1;

font-family: 'Comic Sans MS', cursive, sans-serif;

font-size: 20px;

}

.statement p {

margin-top: 10px;

font-size: 16px;

}

.button-container {

margin-top: 10px;

}

.btn {

display: inline-block;

padding: 5px 10px;

border: none;

border-radius: 4px;

color: #fff;

font-size: 14px;

cursor: pointer;

}

.btn-warning {

background-color: orange;

}

.btn-danger {

background-color: yellow;

}

.btn-orange {

background-color: red;

}

.popup-overlay {

display: none;

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

justify-content: center;

align-items: center;

z-index: 999;

}

.popup {

background-color: #fff;

border-radius: 8px;

padding: 20px;

width: 400px;

}

.btn-danger {

background-color: red;

font-family: Roboto, Arial, sans-serif;

font-weight: bold;

text-decoration: none;

}

.btn-primary {

background-color: #3498db;

}

.popup p {

margin-bottom: 20px;

}

.popup .button-container {

text-align: center;

}

.pagination {

display: flex;

justify-content: center;

margin-top: 20px;

margin-bottom: 120px;

}

.pagination a {

padding: 8px 12px;

text-decoration: none;

color: #fff;

background-color: #6ea6f8;

border: none;

border-radius: 20px;

margin: 0 2px;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

font-size: 16px;

font-weight: bold;

transition: background-color 0.3s;

box-shadow: 0 3px 6px rgba(0, 0, 0, 0.3);

}

.pagination a:hover {

background-color: #3c85e0;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);

}

.pagination .prev:before {

content: '<<';

font-weight: bold;

}

.pagination .next:after {

content: '>>';

font-weight: bold;

}

.pagination .prev:before {

margin-right: 5px;

}

.pagination .next:after {

margin-left: 5px;

}

.pagination .page {

font-weight: bold;

}

.pagination .active {

background-color: #3c85e0;

}

#### **6.10.11 Register.css**

@import url('https://fonts.googleapis.com/css?family=Roboto:400,700&display=swap');

@import url('https://fonts.googleapis.com/css?family=Roboto:400,700&display=swap');

.container {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);

}

body {

background: linear-gradient(135deg, #007bff, #29a5e0, #007bff);

background-size: 600% 600%;

animation: gradientAnimation 10s ease infinite;

}

@keyframes gradientAnimation {

0% { background-position: 0% 50%; }

50% { background-position: 100% 50%; }

100% { background-position: 0% 50%; }

}

.form-wrapper {

width: 90%;

max-width: 400px;

padding: 20px 30px;

background-color: #fff;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

border-radius: 10px;

}

h3 {

text-align: center;

color: #102f45;

margin-top: 0;

font-family: 'Roboto', sans-serif;

}

.form {

margin-top: 20px;

}

.form input,

.form button,

.form select {

width: 100%;

margin-bottom: 10px;

padding: 10px;

border: 1px solid #ccc;

border-radius: 3px;

font-size: 14px;

background-color: #fff;

color: #333;

transition: border-color 0.3s ease;

}

.form button {

background-color: #4389C1;

color: white;

cursor: pointer;

}

.form button:hover {

background-color: #356d8a;

}

.form input::placeholder,

.form select::placeholder {

color: #999;

}

.form input[type="text"],

.form input[type="tel"],

.form input[type="email"],

.form input[type="password"],

.form select {

padding-left: 8px;

transition: border-color 0.3s ease;

}

.form input[type="text"]:focus,

.form input[type="tel"]:focus,

.form input[type="email"]:focus,

.form input[type="password"]:focus,

.form select:focus {

outline: none;

border-color: #007BFF;

box-shadow: 0 0 5px rgba(0, 123, 255, 0.5);

}

.form .name-container {

display: flex;

gap: 10px;

margin-bottom: 10px;

}

.form .name-container input {

flex: 1;

}

.form .row-container {

display: flex;

gap: 10px;

margin-bottom: 10px;

}

.form .row-container input {

flex: 1;

}

.form .studentId-container,

.form .branch-container {

flex: 1;

width: 100%;

}

.form .studentId-container input,

.form .branch-container input {

width: 100%;

box-sizing: border-box;

}

.form .email-container,

.form .password-container {

flex: 1;

}

.form .email-container input,

.form .password-container input {

width: 100%;

box-sizing: border-box;

max-width: calc(50% - 5px);

}

.message {

padding: 10px;

margin-bottom: 10px;

text-align: center;

font-size: 18px;

}

.success {

background-color: #4CAF50;

color: #155724;

border: 1px solid #4CAF50;

font-weight: bold;

}

.failed {

background-color: #FF5252;

color: #721c24;

border: 1px solid #FF5252;

font-weight: bold;

}

.error-list {

color: #a94442;

list-style-type: none;

margin: 0;

padding: 0;

text-align: center;

}

.error-list li {

margin-bottom: 5px;

}

### **6.11 HTML files**

#### **6.11.1 Answers.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>board</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/answers.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="container">

<h1>Question</h1>

<div class="statement-list">

<div class="statement">

<h3 th:text="${question.student.toString()}"></h3>

<hr>

<p th:text="${question.description}"></p>

</div>

</div>

</div>

<div class="container">

<h1>All Answers</h1>

<div class="statement-list">

<div th:each="answer : ${answers}" class="statement">

<h3 th:text="${answer.student.toString()}"></h3>

<hr>

<p th:text="${answer.description}"></p>

</div>

</div>

</div>

<div th:replace="~{footer :: footer}"></div>

</body>

</html>

#### **6.11.2 Ask.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>ask</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/ask.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="form-container">

<form th:action="@{/core/post\_question}" class="form" th:object="${question}">

<h2>Ask the community</h2>

<div th:if="${message != null}" class="alert">

<p th:text="${message}" th:classappend="${post != null} ? 'post' : 'fail'"></p>

</div>

<div class="form-group">

<textarea placeholder="Make sure your question is easily understandable and to the point." rows="10" th:field="\*{description}"></textarea>

</div>

<div class="form-group">

<select th:field="\*{tag}" id="tagSelect">

<option value="General" selected>General</option>

<option value="CSE">CSE</option>

<option value="Mech">Mech</option>

<option value="ECE">ECE</option>

<option value="CE">CE</option>

<option value="Chem">Chem</option>

<option value="EEE">EEE</option>

<option value="MME">MME</option>

<option value="Campus">Campus</option>

</select>

</div>

<div class="form-group">

<button type="submit" id="postButton">Post</button>

<button type="reset">Clear</button>

</div>

</form>

</div>

<div th:replace="~{footer :: footer}"></div>

</body>

</html>

#### **6.11.3 Board.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>board</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/board.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="container">

<h1>All Questions</h1>

<!-- Dropdown for selecting the tag -->

<div class="tag-dropdown">

<label for="tag-select">Select Tag:</label>

<select id="tag-select" onchange="location = this.value;">

<option value="/core/board?tag=General" th:selected="${selectedTag == 'General'}">General</option>

<option value="/core/board?tag=CSE" th:selected="${selectedTag == 'CSE'}">CSE</option>

<option value="/core/board?tag=ECE" th:selected="${selectedTag == 'ECE'}">ECE</option>

<option value="/core/board?tag=CE" th:selected="${selectedTag == 'CE'}">CE</option>

<option value="/core/board?tag=Mech" th:selected="${selectedTag == 'Mech'}">Mech</option>

<option value="/core/board?tag=EEE" th:selected="${selectedTag == 'EEE'}">EEE</option>

<option value="/core/board?tag=Chem" th:selected="${selectedTag == 'Chem'}">Chem</option>

<option value="/core/board?tag=MME" th:selected="${selectedTag == 'MME'}">MME</option>

<option value="/core/board?tag=Campus" th:selected="${selectedTag == 'Campus'}">Campus</option>

</select>

</div>

<div class="statement-list">

<div th:each="question : ${questions}" class="statement">

<div class="question-header">

<h3><span th:text="${question.student.toString()}"></span></h3>

<p class="created-at" th:text="${question.createdAt}"></p>

</div>

<p class="question-text" th:text="${question.description}"></p>

<div class="button-container">

<button class="btn btn-danger answer-btn" th:data-questionid="${question.questionId}">Answer</button>

<a class="btn btn-warning" th:href="@{/core/view\_answers(questionId=${question.questionId})}">View Answers</a>

</div>

<hr>

</div>

</div>

</div>

<div class="popup-overlay">

<div class="popup">

<h3>Answer the Question</h3>

<form th:action="@{/core/save\_answer\_from\_Board}" method="post">

<input type="hidden" name="questionId" class="question-id-input" th:value="${questionId}" />

<input type="hidden" name="selectedTag" th:value="${selectedTag}" />

<textarea class="expandable-textarea" placeholder="Your Answer" name="answerText"></textarea>

<button class="submit-answer-btn btn btn-primary" type="submit">Submit Answer</button>

</form>

</div>

</div>

<div class="pagination">

<a th:if="${currentPage > 1}" th:href="@{/core/board(tag=${selectedTag}, page=${currentPage - 1})}">&lt;&lt;</a>

<a th:each="page : ${#numbers.sequence(1, totalPages)}" th:href="@{/core/board(tag=${selectedTag}, page=${page})}"

th:class="${currentPage == page} ? 'active' : ''" th:text="${page}"></a>

<a th:if="${currentPage < totalPages}" th:href="@{/core/board(tag=${selectedTag}, page=${currentPage + 1})}">&gt;&gt;</a>

</div>

<div th:replace="~{footer :: footer}"></div>

<script>

// JavaScript code to handle the popup functionality

document.addEventListener("DOMContentLoaded", function() {

var answerButtons = document.querySelectorAll(".answer-btn");

var popupOverlay = document.querySelector(".popup-overlay");

var popup = document.querySelector(".popup");

var textarea = document.querySelector(".expandable-textarea");

var questionIdInput = document.querySelector(".question-id-input");

// Open the popup when Answer button is clicked

answerButtons.forEach(function(button) {

button.addEventListener("click", function() {

var questionId = parseInt(button.getAttribute("data-questionid"));

questionIdInput.value = questionId;

popupOverlay.style.display = "flex";

textarea.focus();

});

});

// Close the popup when clicking outside of it

popupOverlay.addEventListener("click", function(event) {

if (event.target === popupOverlay) {

popupOverlay.style.display = "none";

textarea.value = "";

textarea.classList.remove("expanded");

}

});

// Expand the textarea when typing more than 3 lines of text

textarea.addEventListener("input", function() {

var lineHeight = parseInt(getComputedStyle(textarea).lineHeight);

var lines = textarea.scrollHeight / lineHeight;

if (lines > 3) {

textarea.classList.add("expanded");

} else {

textarea.classList.remove("expanded");

}

});

});

</script>

</body>

</html>

#### **6.11.4 Error.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org" xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity6">

<head>

<meta charset="UTF-8">

<title>error</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/error.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="container">

<h1>Looks like you've lost! </h1>

<div class="error-message">

<p th:text="${'Error Message - ' + errorMessage}">{{errorMessage}}</p>

</div>

</div>

<div th:replace="~{footer :: footer}"></div>

</body>

</html>

#### **6.11.5 Footer.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org"

xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity6">

<head>

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

</head>

<body>

<footer class="footer">

<div class="footer-content">

<p>&copy; 2023 CampusQ. All rights reserved. Developed inside RGUKT Srikakulam.</p>

<p><a href="mailto:kushalvardhan3@gmail.com">Email us!</a></p>

<nav>

<a href="#">Privacy Policy</a>

<a href="#">Terms of Service</a>

<a href="#">Contact Us</a>

</nav>

</div>

</footer>

</body>

</html>

#### **6.11.6 Header.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org"

xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity6">

<head>

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

</head>

<body>

<header class="fixed-top">

<h2>CampusQ</h2>

<nav>

<a th:href="@{/core/home\_view}" sec:authorize="isAuthenticated()">Home</a>

<a th:href="@{/core/ask\_question}" sec:authorize="isAuthenticated()">Ask</a>

<a th:href="@{/core/board?tag=General}" sec:authorize="isAuthenticated()">Board</a>

<a th:href="@{/login}" sec:authorize="isAnonymous()">Login</a>

<a th:href="@{/logout}" sec:authorize="isAuthenticated()">Logout</a>

<a th:href="@{/profile}" sec:authorize="isAuthenticated()">Profile</a>

<a th:href="@{/adminControl}" class="active" sec:authorize="hasRole('ROLE\_ADMIN')">Admin</a>

</nav>

</header>

</body>

</html>

#### **6.11.7 Home.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>home</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/home.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="container">

<h1>All Questions</h1>

<div class="statement-list">

<div th:each="question : ${questions}" class="statement">

<div class="question-header">

<h3><span th:text="${question.student.toString()}"></span></h3>

<p class="created-at" th:text="${question.createdAt}"></p>

</div>

<p class="question-text" th:text="${question.description}"></p>

<div class="button-container">

<button class="btn btn-danger answer-btn" th:data-questionid="${question.questionId}">Answer</button>

<a class="btn btn-warning" th:href="@{/core/view\_answers(questionId=${question.questionId})}">View Answers</a>

</div>

<hr>

</div>

</div>

</div>

<div class="popup-overlay">

<div class="popup">

<h3>Answer the Question</h3>

<form th:action="@{/core/save\_answer}" method="post">

<input type="hidden" name="questionId" class="question-id-input" th:value="${questionId}" />

<textarea class="expandable-textarea" placeholder="Your Answer" name="answerText"></textarea>

<button class="submit-answer-btn btn btn-primary" type="submit">Submit Answer</button>

</form>

</div>

</div>

<div class="pagination">

<a th:if="${currentPage > 1}" th:href="@{/core/home\_view(page=${currentPage - 1})}">&lt;&lt;</a>

<a th:each="page : ${#numbers.sequence(1, totalPages)}" th:href="@{/core/home\_view(page=${page})}"

th:class="${currentPage == page} ? 'active' : ''" th:text="${page}"></a>

<a th:if="${currentPage < totalPages}" th:href="@{/core/home\_view(page=${currentPage + 1})}">&gt;&gt;</a>

</div>

<div th:replace="~{footer :: footer}"></div>

<script>

// JavaScript code to handle the popup functionality

document.addEventListener("DOMContentLoaded", function() {

var answerButtons = document.querySelectorAll(".answer-btn");

var popupOverlay = document.querySelector(".popup-overlay");

var popup = document.querySelector(".popup");

var textarea = document.querySelector(".expandable-textarea");

var questionIdInput = document.querySelector(".question-id-input");

// Open the popup when Answer button is clicked

answerButtons.forEach(function(button) {

button.addEventListener("click", function() {

var questionId = parseInt(button.getAttribute("data-questionid"));

questionIdInput.value = questionId;

popupOverlay.style.display = "flex";

textarea.focus();

});

});

// Close the popup when clicking outside of it

popupOverlay.addEventListener("click", function(event) {

if (event.target === popupOverlay) {

popupOverlay.style.display = "none";

textarea.value = "";

textarea.classList.remove("expanded");

}

});

// Expand the textarea when typing more than 3 lines of text

textarea.addEventListener("input", function() {

var lineHeight = parseInt(getComputedStyle(textarea).lineHeight);

var lines = textarea.scrollHeight / lineHeight;

if (lines > 3) {

textarea.classList.add("expanded");

} else {

textarea.classList.remove("expanded");

}

});

});

</script>

</body>

</html>

#### **6.11.8 Login.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org"

xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity6">

<head>

<meta charset="UTF-8">

<title>login</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/login.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="login-container">

<form action="/login" method="post" class="login-form">

<h2>Login</h2>

<li class="login-alerts" th:if="${!#strings.isEmpty(errorMessage)}" th:text="${errorMessage}"></li>

<div class="form-group">

<label for="username">Username:</label>

<input type="text" id="username" name="username" placeholder="Enter your student ID">

</div>

<div class="form-group">

<label for="password">Password:</label>

<input type="password" id="password" name="password" placeholder="Enter your password">

</div>

<div class="button-container">

<button type="submit" class="login-button">Login</button>

<a th:href="@{/register}" class="register-button" sec:authorize="isAnonymous()">Register</a>

</div>

</form>

</div>

<div th:replace="~{footer :: footer}"></div>

</body>

</html>

#### **6.11.9 More.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>board</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/more.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="container">

<h1>Manage Question</h1>

<div class="statement-list">

<div class="statement">

<div class="question-header">

<h3><span th:text="${question.student.toString()}"></span></h3>

<p class="created-at" th:text="${question.createdAt}"></p>

</div>

<p class="question-text" th:text="${question.description}"></p>

<hr>

</div>

</div>

<!-- Dropdown for selecting the tag -->

<div class="tag-dropdown">

<label for="tag-select">Edit Tag:</label>

<select id="tag-select" onchange="location = this.value;">

<option value="/core/editTag?tag=General" th:selected="${question.questionId == 'General'}">General</option>

<option value="/core/editTag?tag=CSE" th:selected="${question.questionId == 'CSE'}">CSE</option>

<option value="/core/editTag?tag=ECE" th:selected="${question.questionId == 'ECE'}">ECE</option>

<option value="/core/editTag?tag=CE" th:selected="${question.questionId == 'CE'}">CE</option>

<option value="/core/editTag?tag=Mech" th:selected="${question.questionId == 'Mech'}">Mech</option>

<option value="/core/editTag?tag=EEE" th:selected="${question.questionId == 'EEE'}">EEE</option>

<option value="/core/editTag?tag=Chem" th:selected="${question.questionId == 'Chem'}">Chem</option>

<option value="/core/editTag?tag=MME" th:selected="${question.questionId == 'MME'}">MME</option>

<option value="/core/editTag?tag=Campus" th:selected="${question.questionId == 'Campus'}">Campus</option>

</select>

</div>

<div class="button-container">

<a class="btn btn-warning" th:href="@{/core/moreDelete(questionId=${question.questionId})}">Delete Question!</a>

<p>Warning!! all the answers for this question will also be deleted. This process cannot be undone.</p>

</div>

</div>

<div th:replace="~{footer :: footer}"></div>

</body>

</html>

#### **6.11.10 Profile.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>User Profile</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/profile.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="container">

<div class="profile-card">

<div class="profile-info-left">

<h1>User Profile</h1>

<div class="profile-info">

<label>Student:</label>

<p th:text="${user.firstName} + ' ' + ${user.lastName}"></p>

</div>

<div class="profile-info">

<label>Student ID:</label>

<p th:text="${user.studentId}"></p>

</div>

<div class="profile-info">

<label>Branch:</label>

<p th:text="${user.branch}"></p>

</div>

<div class="profile-info">

<label>College mail id:</label>

<p th:text="${user.email}"></p>

</div>

</div>

</div>

<h2>Questions Asked</h2>

<div class="statement-list">

<div th:each="question : ${userQuestions}" class="statement">

<p th:text="${question.description}"></p>

<div class="button-container">

<a class="btn btn-warning" th:href="@{/core/view\_answers(questionId=${question.questionId})}">View

Answers</a>

<a class="btn btn-orange" th:href="@{/core/delete\_question(questionId=${question.questionId})}")>Delete</a>

</div>

</div>

</div>

<div class="pagination">

<a th:if="${QueCurrentPage > 1}" th:href="@{/profile(page=${QueCurrentPage - 1})}">&lt;&lt;</a>

<a th:each="page : ${#numbers.sequence(1, QueTotalPages)}" th:href="@{/profile(page=${page})}"

th:class="${QueCurrentPage == page} ? 'active' : ''" th:text="${page}"></a>

<a th:if="${QueCurrentPage < QueTotalPages}" th:href="@{/profile(page=${QueCurrentPage + 1})}">&gt;&gt;</a>

</div>

<h2>My Answers</h2>

<div class="statement-list">

<div th:each="answer : ${userAnswers}" class="statement">

<p th:text="'Q : ' + ${answer.question.description}"></p>

<hr>

<p th:text="'A : ' + ${answer.description}"></p>

<div class="button-container">

<a class="btn btn-orange" th:href="@{/core/delete\_answer(answerId=${answer.answerId})}")>Delete</a>

</div>

</div>

</div>

<div class="pagination">

<a th:if="${AnsCurrentPage > 1}" th:href="@{/profile(page=${AnsCurrentPage - 1})}">&lt;&lt;</a>

<a th:each="page : ${#numbers.sequence(1, AnsTotalPages)}" th:href="@{/profile(page=${page})}"

th:class="${AnsCurrentPage == page} ? 'active' : ''" th:text="${page}"></a>

<a th:if="${AnsCurrentPage < AnsTotalPages}" th:href="@{/profile(page=${AnsCurrentPage + 1})}">&gt;&gt;</a>

</div>

</div>

<div th:replace="~{footer :: footer}"></div>

</body>

</html>

#### **6.11.11 Register.html**

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>register</title>

<link rel="stylesheet" th:href="@{/assets/css/styles.css}" />

<link rel="stylesheet" th:href="@{/assets/css/header.css}" />

<link rel="stylesheet" th:href="@{/assets/css/footer.css}" />

<link rel="stylesheet" th:href="@{/assets/css/register.css}" />

</head>

<body>

<div th:replace="~{header :: header}"></div>

<div class="container">

<div class="form-wrapper">

<h3>Get Started: User Registration</h3>

<ul class="error-list">

<li th:each="error : ${#fields.errors('student.\*')}" th:text="${error}"></li>

</ul>

<div th:if="${message != null}" class="message">

<p th:text="${message}" th:classappend="${success != null} ? 'success' : 'failed'"></p>

</div>

<form th:action="@{/register\_user}" method="post" id="signup-form" class="form" th:object="${student}">

<div class="name-container">

<input type="text" th:field="\*{firstName}" placeholder="First Name">

<input type="text" th:field="\*{lastName}" placeholder="Last Name">

</div>

<div class="row-container">

<div class="studentId-container">

<input type="text" th:field="\*{studentId}" placeholder="Student ID">

</div>

<div class="branch-container">

<select th:field="\*{branch}">

<option value="CSE">CSE</option>

<option value="ECE">ECE</option>

<option value="ME">ME</option>

<option value="CE">CE</option>

<option value="chem">Chem</option>

<option value="EEE">EEE</option>

</select>

</div>

</div>

<input type="text" th:field="\*{email}" placeholder="Email">

<input type="password" th:field="\*{password}" placeholder="Password">

<input type="password" th:field="\*{confirmPwd}" placeholder="Confirm Password">

<button type="submit">Register</button>

</form>

</div>

</div>

<div th:replace="~{footer :: footer}"></div>

</body>

</html>

### **6.12 pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.0.6</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.campusq</groupId>

<artifactId>mini</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>mini</name>

<description>mini project with spring boot</description>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-validation</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

<dependency>

<groupId>org.thymeleaf.extras</groupId>

<artifactId>thymeleaf-extras-springsecurity6</artifactId>

</dependency>

<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

## 

## 

## **CHAPTER- 7**

## **TESTING**

### **7.1 Introduction to Testing**

The cause of testing is to detect mistakes. Making an attempt out is the technique of looking to realize each viable fault or weakness in a piece product. It presents a method to determine the performance of add-ons, sub-assemblies, assemblies and/or a completed product. It is the method of exercising a program with the intent of constructing certain that the application procedure meets its necessities and client expectations and does no longer fail in an unacceptable process. There are rather plenty of forms of scan. Each experiment sort addresses a special trying out requirement.

### **7.2 Types of tests performed**

### **7.2.1 Unit testing**

Unit checking out involves the design of scan circumstances that validate that the internal application's good judgment is functioning safely, and that program inputs produce legitimate outputs. All decision branches and interior code floats must be validated. It's the checking out of character application items of the application. It is achieved after the completion of a person unit earlier than integration. It is a structural checking out that relies on competencies of its construction and is invasive. Unit exams participate in common exams at component level and scan a distinct business approach, utility, and/or process configuration. Unit assessments are certain that every specified course of an industry method performs appropriately to the documented requisites and involves clearly outlined inputs and anticipated results.

#### **7.2.2 Integration testing**

Integration Testing is designed to scan built-in program accessories to determine within the occasion that they evidently run as one software. Trying out is occasion driven and is more concerned with the fundamental final result of screens or fields. Integration assessments reveal that despite the fact that the accessories had been for my part pleasure, as proven through effectively unit checking out, the combo of accessories is correct and regular. Integration checking out is chiefly aimed at exposing the issues that come up from the performance of different components.

#### **7.2.3 Functional testing**

Functional Testing checks provide systematic demonstrations that capabilities established are to be had as particular by means of the business and technical specifications, method documentation, and consumer manuals. Functional testing is working on below mentioned data: Legitimate input: identified lessons of legitimate input ought to be accredited. Invalid enter: recognized lessons of unacceptable effort must be rejected. Capabilities: recognized features ought to be exercised. Output: recognized courses of software outputs have got to be exercised. Systems/Procedures: performance of the system here was invoked. Individual and team work of useful checks is fascinated by specifications, key capabilities, or special scan instances. Moreover, systematic insurance plans concerning establishing business method flows; data fields, predefined processes, and successive strategies have to be regarded for trying out. Before useful trying out is whole, extra checks are recognized and the strong price of present checks be strong minded.

#### **7.2.4 System testing**

Scheme difficulty ensures that the whole included agenda process meets principles. It examines a pattern to make sure an identified and predictable outcome. An illustration of procedure testing is the configuration-oriented approach integration scan. System testing is based on approach descriptions and flows, emphasizing pre-driven system links and integration aspects.

## 

## **CHAPTER- 8**

## **CONCLUSION**

To conclude the description about the project. The project developed using Spring boot and its dependencies and HTML, CSS, Java Script based on the requirement specification of the user and the analysis of the existing system, with flexibility for future enhancement

Overall, the platform seeks to promote student engagement and knowledge sharing by providing a platform where students can ask questions and get answers from their peers.

## 

## **CHAPTER- 9**

## **FUTURE ENHANCEMENT**

We are currently developing a comprehensive system that aims to enhance the sign-in process and provide efficient communication for students. Two key features that we plan to incorporate in the future are Notification System and email verification with OTP (One-Time Password)."

* The Notification System will keep students informed and engaged by delivering important updates, announcements, and reminders directly to their accounts. This feature will ensure that students never miss out on vital information, such as assignment deadlines, class schedule changes, or campus events. By centralizing these notifications, we aim to streamline communication and improve overall student experience.
* In addition, we are implementing email verification with OTP as part of the sign-in process. This added security measure will require students to verify their email addresses by entering a unique OTP sent to their registered email. This verification step will help ensure that the sign-in process is secure and that students have full control over their accounts.

"These future enhancements are intended to optimize student engagement, facilitate effective communication, and enhance the overall security of our project. We are committed to continuously improving the system to provide a seamless and user-friendly experience for all students."

## **APPENDICES**

1. For rectifying errors that we have encountered while implementing the project we have used Chat Gpt and Eazy bytes Spring Pdf.
2. 2. To understand about spring technologies, we used the Telusuko Spring Boot playlist.

Telusuko Playlist-<https://youtube.com/playlist?list=PLsyeobzWxl7oA8QOlMtQsRT_I7Rx2hoX4>