A

PROJECT REPORT ON EXAM PORTAL APPLICATION

Submitted in Partial Fulfillment of the Requirement
for the Award of the Degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

BY

ANJANI KUMAR SHUKLA

(ID. No :20BTCSE013)

UNDER THE SUPERVISION OF

Er. SANJAY T. SINGH

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
VAUGH INSTITUTE OF AGRICULTURAL ENGINEERING
AND TECHNOLOGY
SAM HIGGINBOTTOM UNIVERSITY OF AGRICULTURE,
TECHNOLOGY AND SCIENCES
NAINI, PRAYAGRAJ-211007
2024

DECLARATION

I, **Anjani Kumar Shukla,** declare that the work presented in this Project report entitled "**EXAM PORTAL APPLICATION**" submitted to the Department of Computer Science and Information Technology in Faculty of Engineering and Technology, **Sam Higginbottom University of Agriculture, Technology and Sciences, Naini, Prayagraj,** for the award of the degree of Bachelor of Technology in Computer Science & Engineering is an original work. I have neither plagiarized nor submitted the same work for the award of any other degree. In case this undertaking is found incorrect, my degree may be withdrawn unconditionally by the University.

Date: Anjani Kumar Shukla

Place:Prayagraj ID:20BTCSE013

CERTIFICATE

Certified that the project report entitled **'EXAM PORTAL APPLICATION'** submitted to Department of Computer Science and Information Technology in Faculty of Engineering and Technology, **Sam Higginbottom University of Agriculture, Technology and Sciences, Naini, Prayagraj** in partial fulfillment of the requirement for award of **Bachelor of Technology** in Computer Science and Engineering is a bonafide record of project report carried out by **Mr. Anjani Kumar Shukla (ID No:20BTCSE013)** under my supervision and guidance.

Date:

Place:Prayagraj

Er. Sanjay T. Singh

Project Advisor

CERTIFICATE OF ACCEPTANCE OF EVALUATION COMMITTEE

This is to certify that project report entitled '**EXAM PORTAL APPLICATION'** has been prepared and submitted by **Anjani Kumar Shukla** in partial fulfillment of the requirement for the award of the degree of 'Bachelor of Technology in Computer Science & Engineering' in Department of Computer Science and Information Technology, Vaugh Institute of Agricultural Engineering and Technology, Sam Higginbottom University of Agriculture, Technology and Sciences, Prayagraj (U.P.) INDIA.

Name	Evaluation	Signature
Er. Sanjay T. Singh **Project Advisor** Department of Computer Science & IT SHUATS, Prayagraj	**Satisfactory / Unsatisfactory**	
Dr. Tulika **Project Coordinator** **Assistant Professor** Department of Computer Science & IT SHUATS, Prayagraj	**Satisfactory / Unsatisfactory**	
Er. R. Dileep Kumar **Project Coordinator** **Assistant Professor** Department of Computer Science & IT SHUATS, Prayagraj	**Satisfactory / Unsatisfactory**	

The evaluation committee has thoroughly examined this project and has found it acceptable.

Prof.(Dr.) W. Jeberson

Chairman

HoD

Department of Computer Science and IT

SHUATS, Prayagraj

ABSTRACT

Online examination systems have become increasingly popular in recent years due to their numerous benefits. These systems allow students to take tests from any location with internet connectivity, eliminating the need for physical infrastructure such as classrooms and examination halls. Students can also take tests at their convenience, allowing them to balance their academic commitments with other responsibilities. The system's immediate feedback feature enables students to identify their weak areas and work on them, leading to improved academic performance. This feature also allows administrators to monitor students' progress and identify areas where they may need additional support. and also cost-effective, particularly for institutions that have to rent or maintain physical spaces for testing purposes. This makes the system accessible to a wider range of students, including those from low-income families or remote areas, promoting inclusive and reducing the education gap between students from different socioeconomic backgrounds. However, online examination systems also have several challenges that administrators and students must overcome. Cheating is a significant challenge, and the system must include features such as randomizing questions, setting time limits, and restricting access to external materials during the test to mitigate this challenge. Technical glitches such as server downtime, slow internet connection, and software errors can also impact the student's test-taking experience and their performance. Despite these challenges, online examination systems have a significant impact on student learning and performance. With the continued advancement in technology, online examination systems will continue to revolutionize the education sector, promoting inclusive and improving student learning and performance.

ACKNOWLEDGEMENT

In the completion of this project, I extend my heartfelt appreciation to those individuals and entities who have played pivotal roles in bringing this project to completion. My sincere thanks go to **Er. Sanjay T. Singh**, whose guidance and insightful contributions have been invaluable throughout the developmental stages. I also express gratitude to Project Coordinators for their collaborative efforts and constructive feedback, enhancing the overall quality of this work. The support provided by **Dr.Tulika** and **Er. Dileep Kumar** has been instrumental, and I acknowledge their contribution to the successful completion of this project. This project stands as a testament to the collective efforts and support received, and I am profoundly grateful to all who have contributed to the efforts that have shaped its completion.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

Introduction

A technology called an online test website enables students to take exams remotely from any place with internet access. The student interface and the administrator interaction are the two main parts of the system. While administrators can manage and edit tests and student profiles via the administrator interface, students are able to sign in and take exams through the student user interface. Accessibility, flexibility, efficiency, and cost-effectiveness are just a few advantages that online testing portals can provide. Tests can be taken anywhere, hence there is no longer a need for physical infrastructure like classrooms and exam rooms. Additionally, this lowers the system's cost, which is beneficial for institutions that must rent or maintain physical spaces for testing. Students can take exams at their convenience, provided they meet the deadline set by the administrator, enabling them to balance their academic commitments with other responsibilities such as work and family. The system's immediate feedback feature provides students with instant feedback on their performance, allowing them to identify their weak areas and work on them before taking subsequent exams, leading to improved academic performance. The system's efficiency makes it easy for administrators to manage and edit student profiles and exams. The administrator can add or remove students from the system, create and edit exams, and monitor students' progress, making it particularly useful for institutions with large numbers of students. However, online examination portals also have several challenges that administrators and students must overcome. Cheating is a significant challenge, and the system must include features such as randomizing questions, setting time limits, and restricting access to external materials during the exam to mitigate this challenge. Technical glitches such as server downtime, slow internet connection, and software errors can significantly impact the student's exam-taking experience and their performance. To mitigate 2 this challenge, the system must have a robust technical infrastructure, including backup servers, high-speed internet connectivity, and reliable software. In conclusion, online examination portals have numerous benefits for both students and administrators. They provide accessibility, flexibility, efficiency, and cost-effectiveness, making them an ideal testing platform for modern-day learners. However, the system's challenges, such as cheating and technical glitches, must be adequately addressed to ensure its success. With the continued advancement in technology, online examination portals will continue to revolutionize the education sector, promoting inclusivity and improving student learning and performance.

1.2 Objectives

The primary objective of the EAXM PORTAL APPLICATION is to provide users with a digital platform for efficient and organized the test. This addresses the need for users to transition from traditional paper-based taking-test to a digital format.

1.3 Project Scope

The scope of the project is to develop a website which shows the latest posts and announcements. One can add, edit and delete new quizzes and people can taking the test . The users of the project can register, edit his/her profile pic etc. Categories can also be added. The system is used only by student and admin. It is not integrated with any external system.

1.4 Users of the system

The users of the system are students and admin . They can add categories, manage account, taking test, delete quizzes . They can ask questions and anybody can answer.

1.5 Life cycle model used

The life cycle model used to develop this project is a prototype model. The prototype model is a software development model where a prototype (a working model of the system) is created, tested, and refined before the final product is developed. Prototypes allow for early user involvement and feedback. This is crucial for a exam-portal application where user experience and engagement are key factors. Gathering feedback from potential users during the prototype stage helps in understanding their needs and preferences. The prototype model supports an iterative and incremental development approach. This is beneficial for a exam portal application project as it allows developers to refine and enhance features based on ongoing feedback. Any issues related to usability, navigation, or functionality can be identified and addressed early in the development cycle.

The Prototyping Model is one of the most popularly used Software Development Life Cycle Models (SDLC models). This model is used when the customers do not know the exact project requirements beforehand. In this model, a prototype of the end product is first developed, tested, and refined as per customer feedback repeatedly till a final acceptable prototype is achieved which forms the basis for developing the final product. Figure 1.1 shows the phases of the prototype model.

1. **Requirement Gathering and Analysis:** This is the initial step in designing a prototype model. In this phase, users are asked about what they expect or what they want from the system.
2. **Quick Design:** This is the second step in Prototyping Model. This model covers the basic design of the requirement through which a quick overview can be easily described.
3. **Build a Prototype:** This step helps in building an actual prototype from the knowledge gained from prototype design.
4. **Initial User Evaluation:** This step describes the preliminary testing where the investigation of the performance model occurs, as the customer will tell the strength and weaknesses of the design, which was sent to the developer.
5. **Refining Prototype:** If any feedback is given by the user, then improving the client's response to feedback and suggestions, the final system is approved.
6. **Implement Product and Maintain:** This is the final step in the phase of the Prototyping Model where the final system is tested and distributed to production, here the program is run regularly to prevent failures.

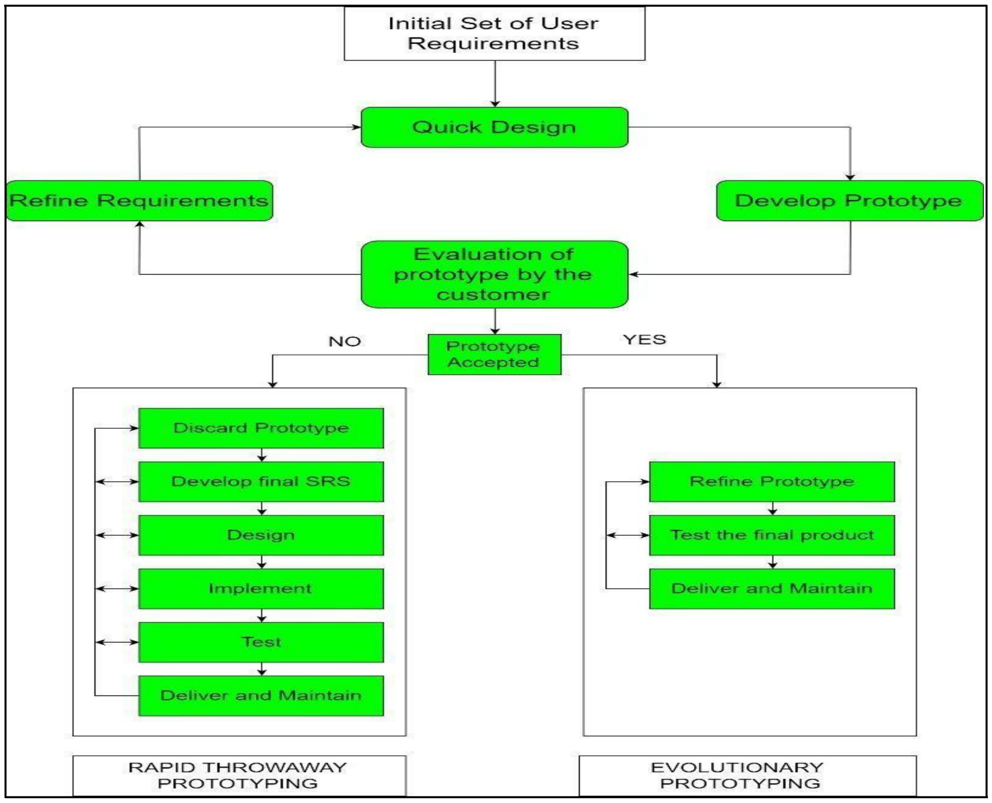


Figure 1.5 Prototype model

CHAPTER 2 REQUIREMENT SPECIFICATION

REQUIREMENT SPECIFICATION

A Requirement Specification provides a comprehensive outline of the functional and nonfunctional requirements. This document serves as a blueprint for the project, guiding developers, designers, and stakeholders in understanding the scope, features, and characteristics of the web app. This chapter presents all the requirements for developing the exam-portal website "**EXAM PORTAL APPLICATION**".

2.1 Functional Requirements

The functional requirements of the system are listed below.

1. **User Management:**

   User Registration:
   - Users can create accounts with a valid email address and password.
   - Include email verification during the registration process.
2. **User Interface**:

   - Design an intuitive and responsive user interface for seamless navigation and optimal user experience.

3. **Authentication:**
   - Users must be able to log in securely using their credentials.
4. **Find Category:**
   - Provide functionality for users to find the category, and format digital take test with various text formatting options.

5. **Show Result:**
   - Implement tools for users to categorize, tag, and organize their test efficiently, supporting a structured and easily accessible system.

2.2 Non-Functional Requirements:

The non-functional requirements of the system are listed below.

1. **Response Time:** The website should load pages and content within 2 seconds.

2. **Scalability:** The system must be scalable to handle an increasing number of users and content without significant degradation in performance.
3. **Availability:** The system must aim for 99.9% up time.
4. **Fault Tolerance:** The system should gracefully handle errors and minimize service disruptions.
5. **Authorization:** The system must implement access controls to ensure users can only access appropriate information.
6. **UI:** The system should have an intuitive user interface with clear navigation.
7. **Compatibility:** The system should be compatible with major web browsers.
8. **Modularity:** Code base should be modular for easy updates and maintenance.

2.3 Hardware Requirements

The hardware requirements for a Exam Portal Application web project developed in Java can vary based on factors such as the scale of the project, expected traffic, and the complexity of the application. However, I can provide you with a general guideline for the hardware requirements. Keep in mind that these are just recommendations, and you may need to adjust them based on your specific project needs.

1. Web Server:

- A robust web server is essential to handle HTTP requests and serve the web application. Common choices include Apache Tomcat.

- Recommended: At least 4 GB of RAM, multi - core processor.

2. Database Server:

- Java web applications often use relational databases like MySQL, PostgreSQL, or Oracle.

- Recommended: Sufficient disk space for the database, at least 4 GB of RAM, multi-core processor.

3. Network Infrastructure:

- Ensure a reliable network infrastructure with sufficient bandwidth to handle incoming and outgoing traffic.

- Recommended: High-speed internet connection, reliable networking equipment.

4. Backup System:

- Implement a robust backup system for both the database and the application code to prevent data loss.

- Recommended: Regular automated backups stored in a secure location.

2.4 Software Requirements

The software requirements for a EXAM PORTAL APPLICATION web project developed in Java include various components and tools that are necessary for the development, deployment, and maintenance of the application. Here are the essential software requirements:

1. Java Development Kit (JDK):

- Java is the primary programming language for your web project. Install the latest version of the Java Development Kit (JDK) on your development machine.

- Recommended: JDK 8 or later, depending on your project's compatibility and requirements.

2. Integrated Development Environment (IDE):

- Choose a Java IDE to streamline the development process. Popular choices include Eclipse, IntelliJ IDEA, and NetBeans.

- Recommended: IntelliJ IDEA is widely used in the Java development community for its features and ease of use.

3. Web Framework:

- Select a Java web framework to simplify web application development. Popular choices include Spring MVC, Spring Boot and Angular.
- Recommended: Spring Boot is a popular choice for its simplicity and extensive features.

4. Database Management System (DBMS):

- Choose a relational database management system (RDBMS) to store and manage data. MySQL, PostgreSQL, and Oracle Database are common choices.
- Recommended: The choice of the DBMS may depend on your organization's preferences and requirements.

5. Version Control System:

- Implement version control to manage source code changes efficiently. Git is the most widely used version control system.
- Recommended: Use Git for version control, and platforms like GitHub or GitLab for collaboration and code hosting.

6. Front-end Technologies:

- Depending on your project requirements, incorporate front-end technologies such as HTML, CSS, Boot Strap . Consider using a front-end framework like React, Angular, or Vue.js if needed.

7. Security Libraries:

- Integrate security(JWT Authentication) libraries to protect your web application from common vulnerabilities.
- Spring Security is a widely used security framework for Java applications.

8. Testing Frameworks:

- Use testing frameworks to ensure the quality of your code. J Unit is a popular choice for unit testing in Java, and tools like Selenium can be used for automated testing of web interfaces.

CHAPTER 3
SYSTEM DESIGN

Exam Portal Application Architecture

Front-end Interface:

- HTML for structure.
- Angular are used to fronted
- CSS for styling.
- Bootstrap for responsive design and UI components.
- Includes a text editor component for user input.
- Displays analysis results using HTML/CSS.
- Handles user authentication and authorization forms using HTML forms.

Back end Server:

- Implemented using Sprint Boot for handling HTTP requests and responses.
- Utilizes Spring boot for dynamic content generation.
- JDBC (Java Database Connectivity) for interacting with the MySQL database.
- Implements business logic and integrates with external services.
- Provides REST full endpoints for front-end - back-end communication.

Text Analysis Service Integration:

- Utilizes Java libraries for making HTTP requests to the Architecture API or similar service.
- Sends text data to the analysis service for processing.
- Receives and processes the analyzed results.
- Implements error handling and retry mechanisms for reliable integration.

Database:

- MySQL database for storing user data, including notes, analysis results, and user profiles.
- JDBC for connecting Angular to the MySQL database.
- Stores data in tables designed to support the application's functionality.

Authentication and Authorization:

- Manages user accounts, authentication, and authorization using Spring JWT.
- Utilizes session management for maintaining user sessions.
- Implements secure password storage and authentication mechanisms.

Additional Features:

- **Collaboration tools**: Allows users to collaborate on notes or share analysis results with others.
- **Search functionality**: Enables users to search through their notes or analysis results.
- **Export/import functionality**: Supports exporting notes in various formats (e.g., PDF, CSV) and importing text documents for analysis.

- **Notifications:** Sends notifications to users for various events, such as completed analysis or shared notes.

3.3 Use Case Diagram

For a Exam-Portal Application project developed in Java, a use case diagram can illustrate the various interactions between external entities (actors) and the system. Here's a potential actors and use cases:

User: Manages user accounts, oversees content moderation, and has access to administrative features.

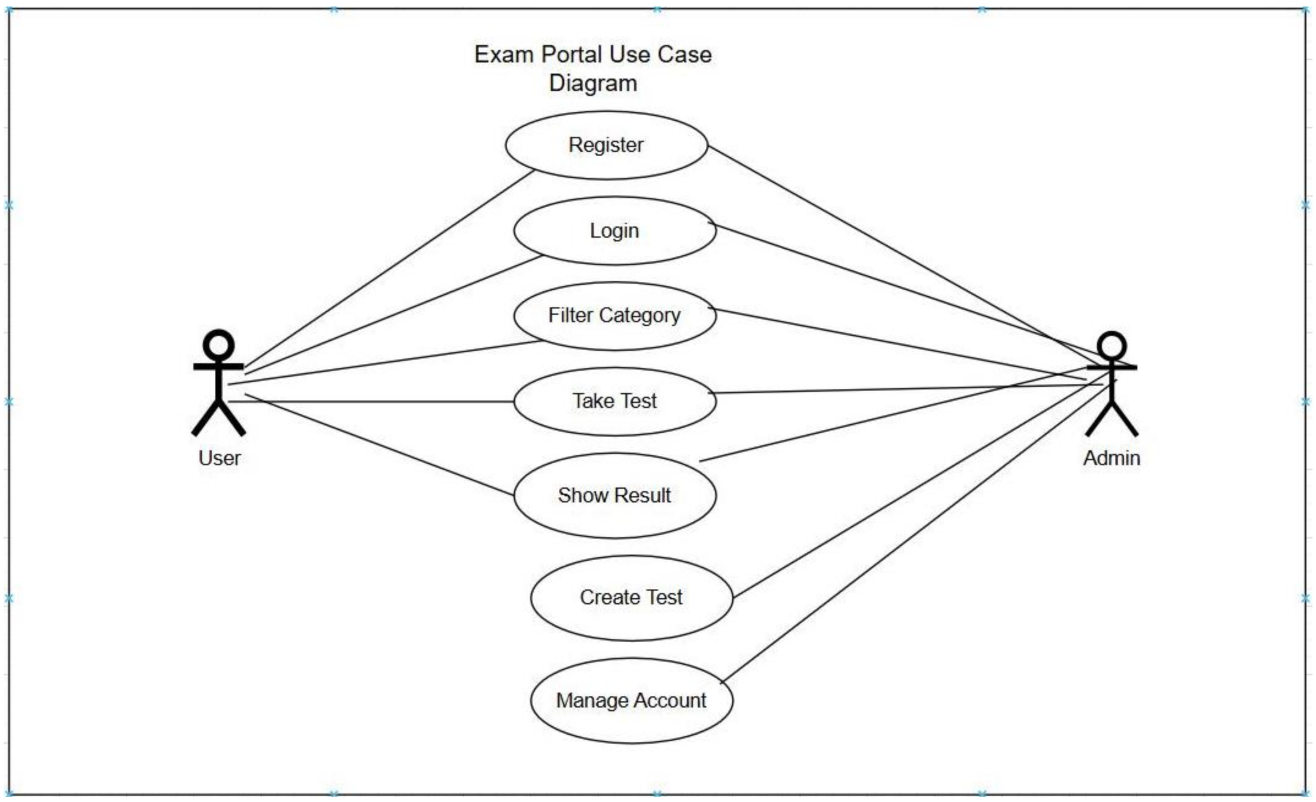


Figure 3.3 Use case diagram of Exam-Portal APPLICATION

3.4 Class Diagram

A class diagram for the "Exam-Portal Application" web project visually represents the system's structure by illustrating the classes (objects), their attributes, and the relationships between them. Here's a key classes and their associations:

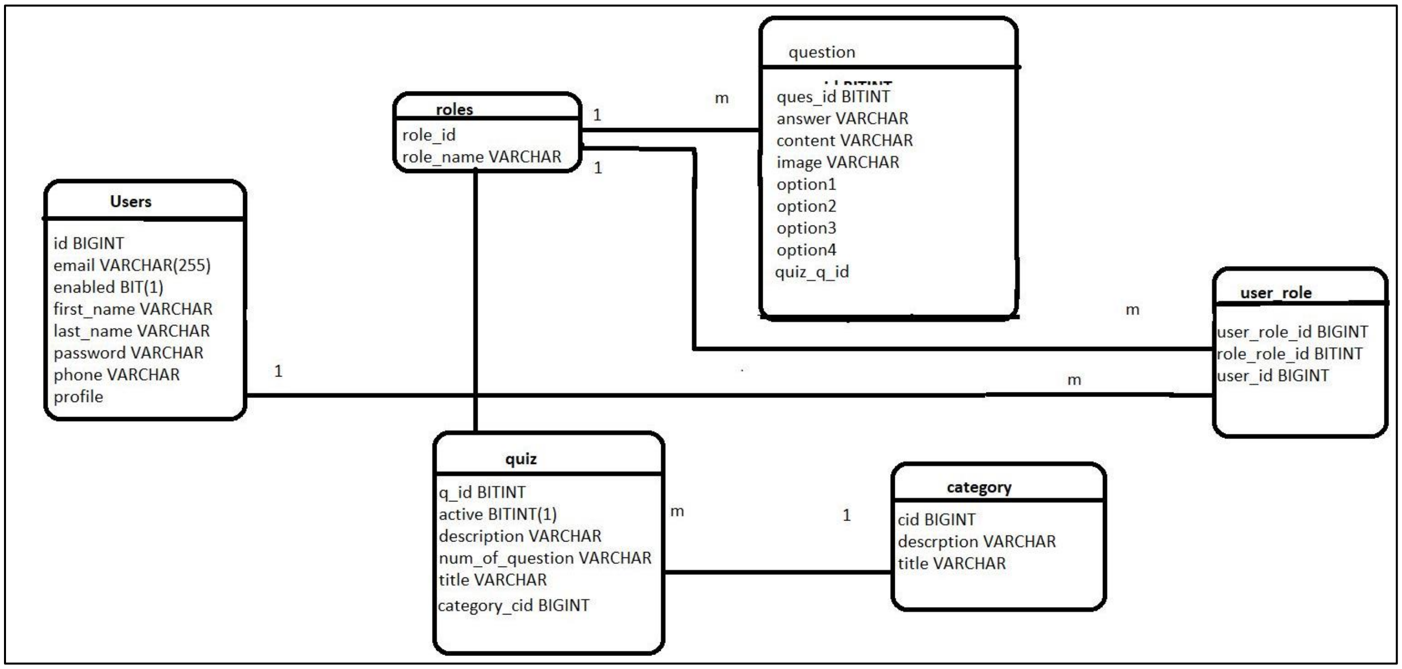


Figure 3.4 Class Diagram of Exam-Portal APPLICATION

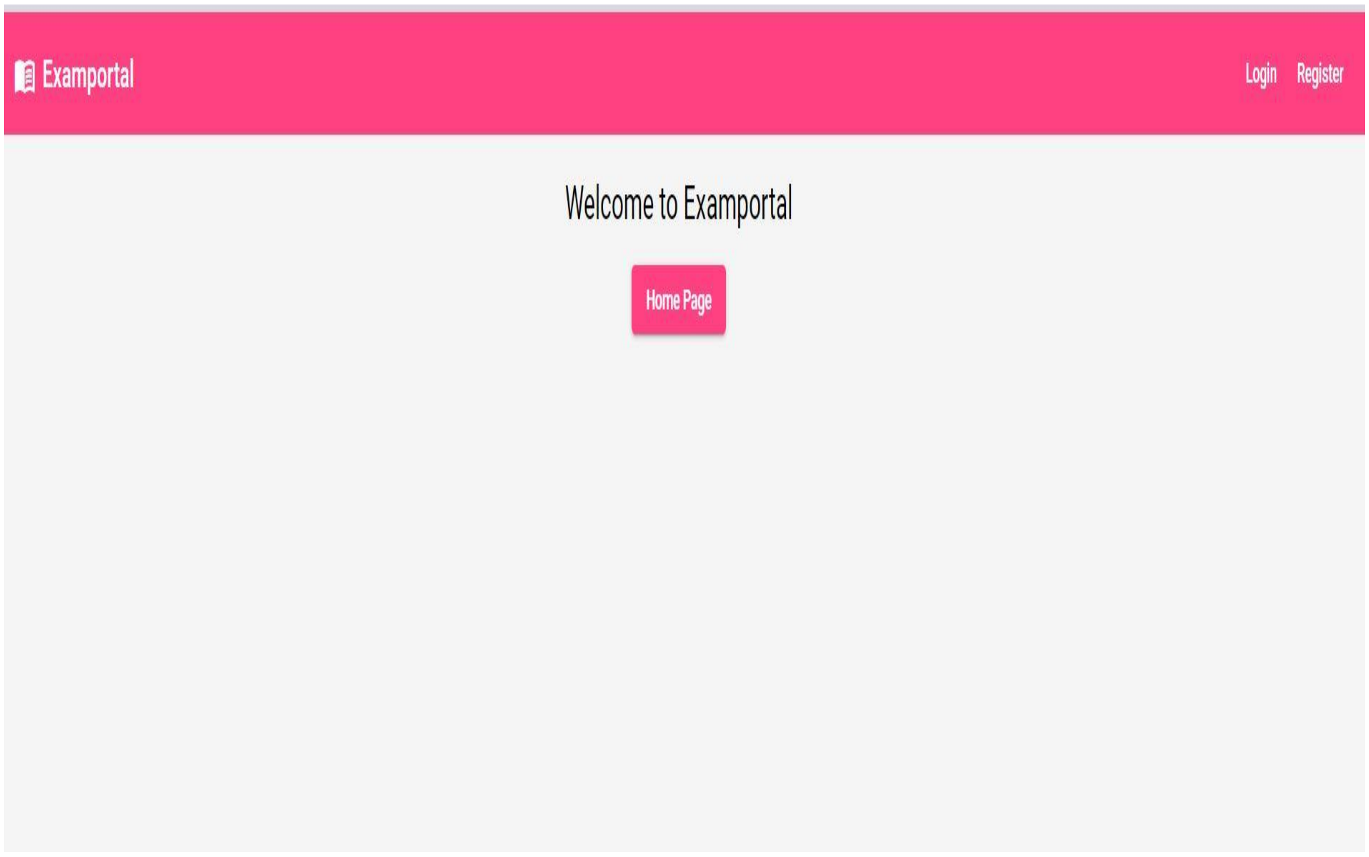
CHAPTER 4 IMPLEMENTATION AND TESTING

Implementation

The implementation phase involves turning the system design into a functioning "Exam Portal Application" web application. This phase encompasses both front-end and back-end development, database creation, and the integration of various components.

4.1 Front-end Development

As already discussed in chapter 2, for front end development, Angular, HTML, CSS and Bootstrap are used for the basic structure and styling. Its JSP is used for a dynamic and responsive user interface.

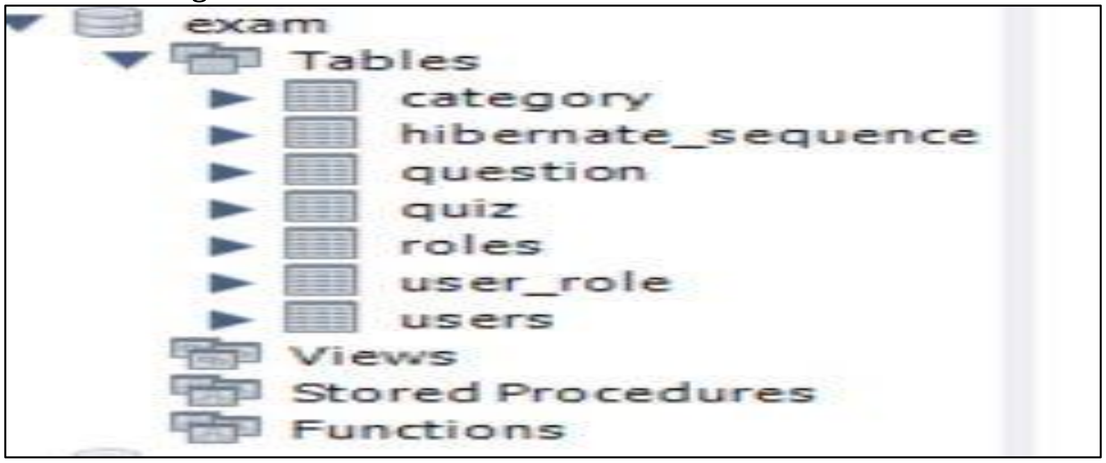


4.2 Back-end Development

1. **Technology Stack:**
   - Choose Java as the primary back-end language, utilizing a web framework like Spring Boot.
   - Develop Restful APIs to handle communication between the front-end and back-end.

2. **Server Setup:**
   - Set up a robust server architecture to handle incoming requests.
   - Implement load balancing to distribute traffic efficiently.

3. **Database Integration:**

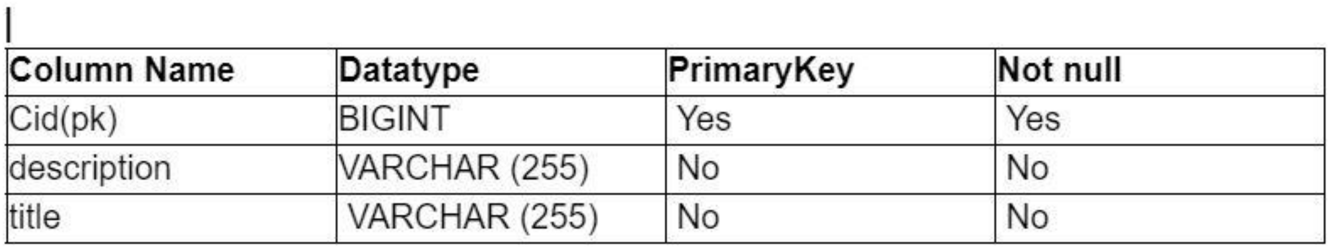


```
exam
  Tables
    category
    hibernate_sequence
    question
    quiz
    roles
    user_role
    users
  Views
  Stored Procedures
  Functions
```

Database Name

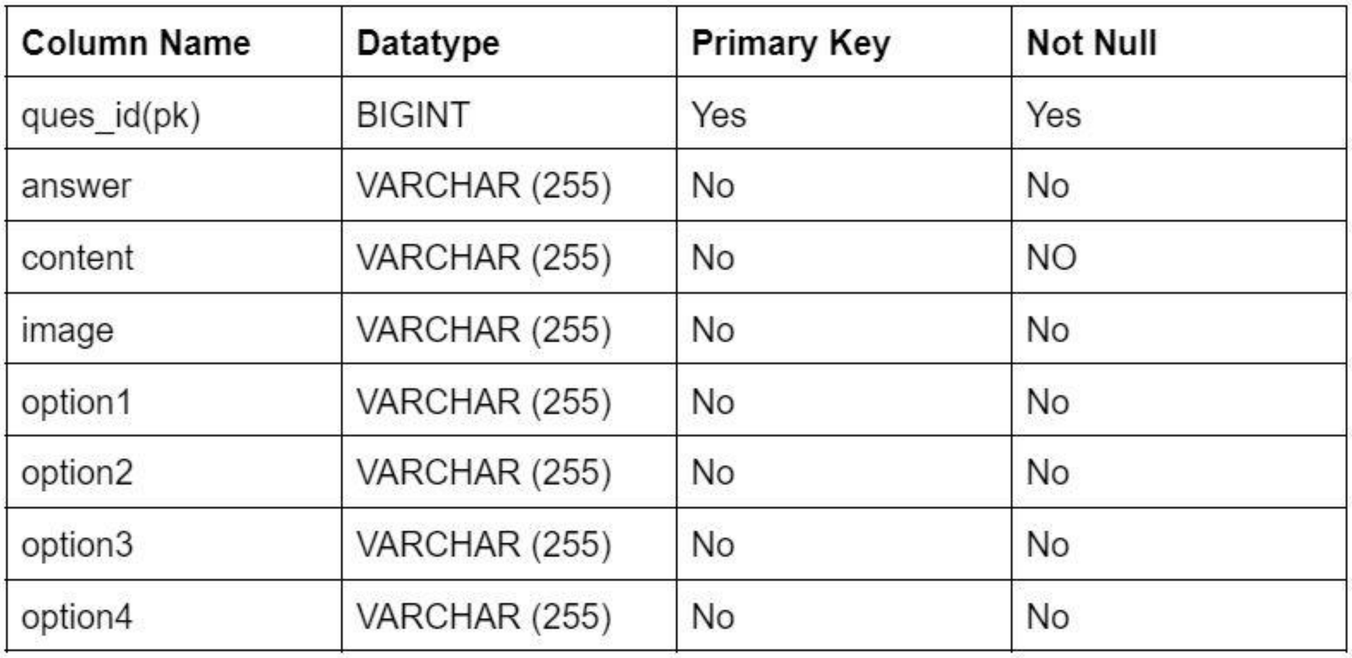
   - Connect the back-end to the MySQL database.
   - Implement data access and manipulation logic using database queries.

4. **Database Schema:**
   - Create a well-designed relational database schema that includes tables for users, blog posts, interactions, etc.
   - Establish relationships between tables to maintain data integrity.



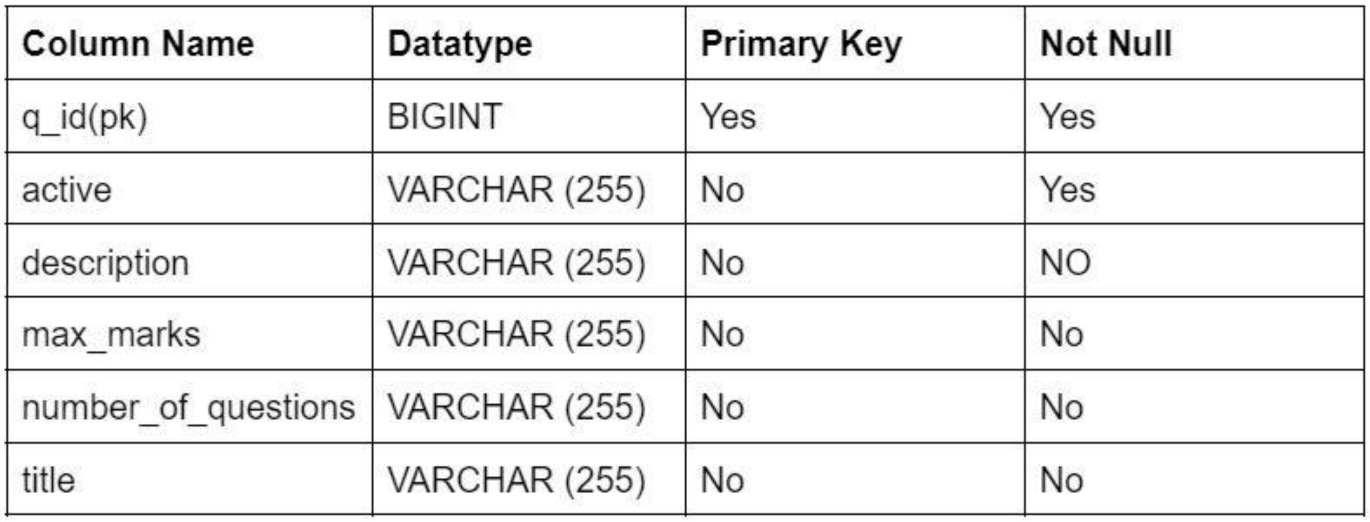
Column Name	Datatype	PrimaryKey	Not null
Cid(pk)	BIGINT	Yes	Yes
description	VARCHAR (255)	No	No
title	VARCHAR (255)	No	No

Table Name- Category



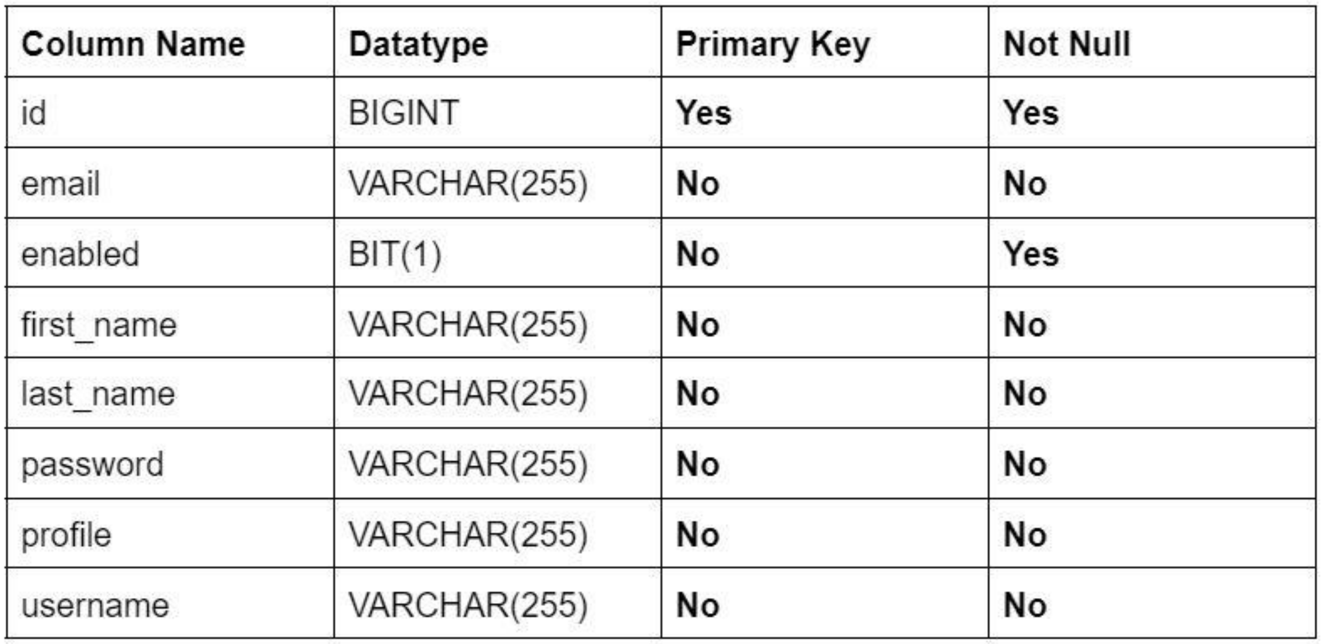
Column Name	Datatype	Primary Key	Not Null
ques_id(pk)	BIGINT	Yes	Yes
answer	VARCHAR (255)	No	No
content	VARCHAR (255)	No	NO
image	VARCHAR (255)	No	No
option1	VARCHAR (255)	No	No
option2	VARCHAR (255)	No	No
option3	VARCHAR (255)	No	No
option4	VARCHAR (255)	No	No

Table Name- Question



Column Name	Datatype	Primary Key	Not Null
q_id(pk)	BIGINT	Yes	Yes
active	VARCHAR (255)	No	Yes
description	VARCHAR (255)	No	NO
max_marks	VARCHAR (255)	No	No
number_of_questions	VARCHAR (255)	No	No
title	VARCHAR (255)	No	No

Table Name- Quiz



Column Name	Datatype	Primary Key	Not Null
id	BIGINT	**Yes**	**Yes**
email	VARCHAR(255)	**No**	**No**
enabled	BIT(1)	**No**	**Yes**
first_name	VARCHAR(255)	**No**	**No**
last_name	VARCHAR(255)	**No**	**No**
password	VARCHAR(255)	**No**	**No**
profile	VARCHAR(255)	**No**	**No**
username	VARCHAR(255)	**No**	**No**

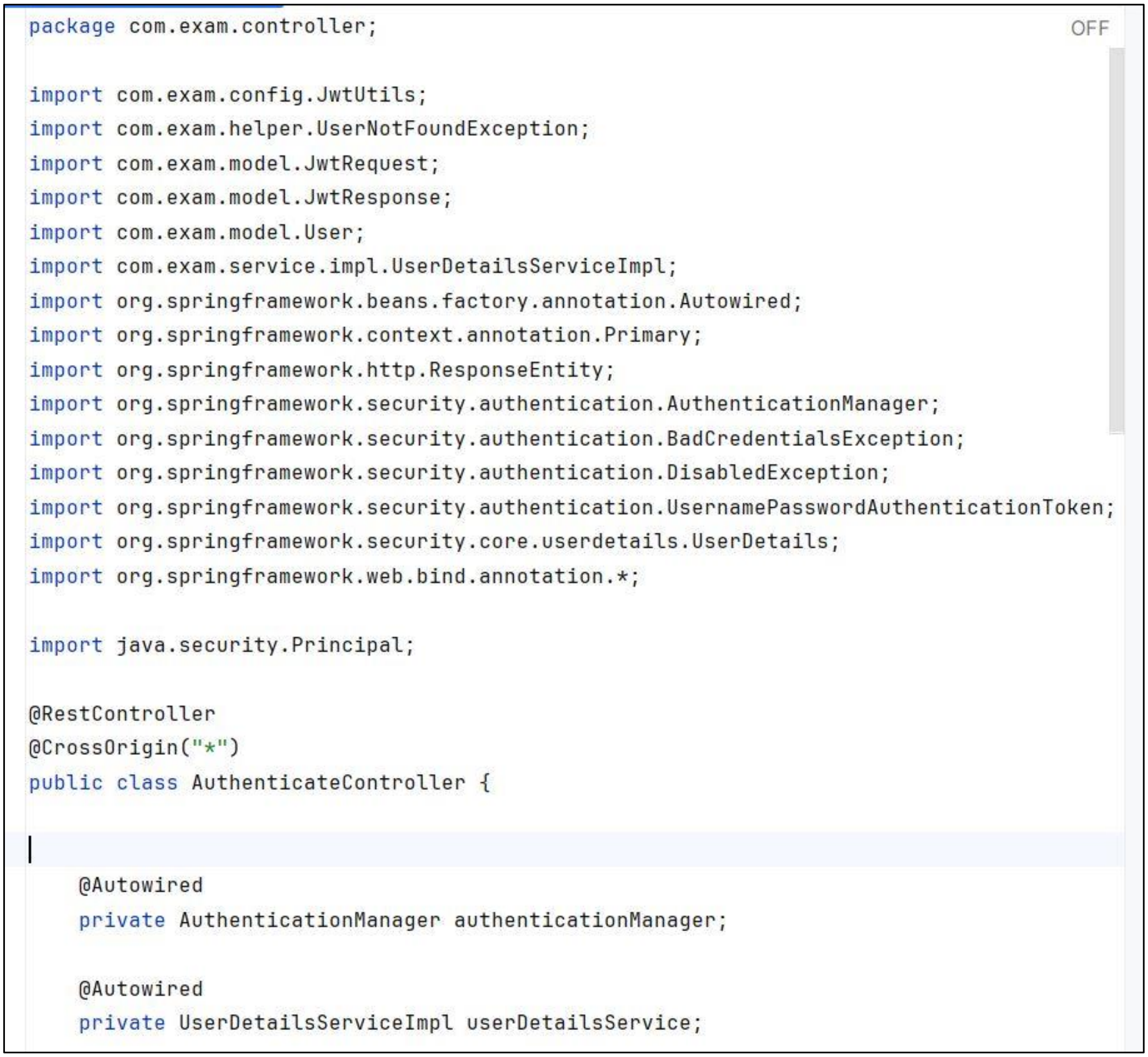
Table Name- Users

5. **Data Security Measures:**

- Implement encryption for sensitive user data stored in the database.
- Set up proper access controls to restrict unauthorized access to the database.

6. **User Authentication:**

- Implement secure user authentication with features like password hashing and token-based authentication.
- Integrate multi - factor authentication options for enhanced security.



```
package com.exam.controller;

import com.exam.config.JwtUtils;
import com.exam.helper.UserNotFoundException;
import com.exam.model.JwtRequest;
import com.exam.model.JwtResponse;
import com.exam.model.User;
import com.exam.service.impl.UserDetailsServiceImpl;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Primary;
import org.springframework.http.ResponseEntity;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.authentication.BadCredentialsException;
import org.springframework.security.authentication.DisabledException;
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.web.bind.annotation.*;

import java.security.Principal;

@RestController
@CrossOrigin("*")
public class AuthenticateController {


    @Autowired
    private AuthenticationManager authenticationManager;

    @Autowired
    private UserDetailsServiceImpl userDetailsService;
```

Code Authentication

4.3. User Experience (UX) Refinement:

1. **Responsive Design:**
   - Optimize the user interface for responsiveness on various devices.
   - Conduct thorough testing to ensure a seamless user experience across browsers and screen sizes.

2. **Feedback Mechanisms:**
   - Implement visual feedback for user actions, such as successful content uploads or interaction notifications.
   - Ensure clear error messages and guidance for users in case of issues.

4.4 Testing:

4.4.1 Unit Testing:

- Conduct unit tests for individual components, functions, and modules.
- Use testing frameworks such as J Unit for Java to verify the correctness.

4.4.2 Integration Testing:

- Test the interaction between different components, APIs, and modules.
- Ensure that data flows seamlessly between the front-end and back-end components.

4.4.2 User Acceptance Testing (UAT):

- Invite a group of users to test the website in a real-world environment.

- Gather feedback on the user interface, functionality, and overall user experience.

4.4.5 Security Testing:

- Perform security audits and vulnerability assessments.

- Identify and address potential security threats, including SQL injection, cross-site scripting (XSS), and data breaches.

4.4.6 Performance Testing:

- Test the website's performance under various conditions, including normal and peak loads.

- Identify and address any bottlenecks or performance issues.

4.4.7 Compatibility Testing:

- Test the website on different web browsers and devices to ensure cross-browser and cross-device compatibility.

- Resolve any layout or functionality issues specific to certain browsers or devices

4.4.8 Accessibility Testing:

- Verify that the website adheres to accessibility standards (e.g., WCAG).

- Test the website with assertive technologies to ensure exclusivity for users with disabilities.

4.4.9 Documentation Review:

- Review and update technical documentation to reflect any changes made during the implementation and testing phases.

- Ensure that documentation is comprehensive for both developers and administrators.

CHAPTER 5 CONCLUSION AND FUTURE SCOPE

Conclusion

Discussion on the Results Achieved: Examination portal can make it easier for people to access education and training programs, regardless of their location or schedule. This can lead to increased participation and higher completion rates. It will reduce errors and inconsistencies in the grading process, resulting in more accurate and reliable assessments of student performance. Online examinations can save time and money by automating tasks such as exam creation, administration, and grading. This can free up resources that can be used for other educational or business purposes and also incorporate security measures such as biometric authentication, anti-cheating features, and data encryption to ensure the integrity of the exam results. provide instant feedback to students, helping them to identify areas where they need improvement and to adjust their learning strategies accordingly. This can lead to better learning outcomes and higher levels of student achievement.

5.1 User-Centric Experience:

The platform offers a unique and intuitive experience, allowing users to transcend traditional online taking test by seamlessly integrating visual elements into their narratives. The responsive design and interactive features contribute to a positive and engaging user journey.

5.2 Community Building:

The incorporation of social interaction features fosters a sense of community among users. The ability to like, share, and comment on collages enhances the collaborative spirit of the platform, encouraging connections and conversations among creators.

5.3 Security and Reliability:

- Stringent security measures, including user authentication, authorization protocols, and data encryption, have been implemented to safeguard user information. The platform's reliability is underscored by its robust server architecture and fault-tolerant design.

5.4 Future Scope:

1.1 Mobile compatibility: Develop a mobile app or optimize the existing portal for mobile devices to increase accessibility and convenience for users who prefer to take exams on their smart phone or tablets.

1.2 Advanced analytic: Implement advanced analytic tools such as machine learning algorithms and predictive models to analyze exam results and provide insights that can improve the effectiveness of the online examination portal.

1.3 Collaborative exams: Allow students to take collaborative exams in which they can work together in real-time to solve problems and answer questions. This can promote teamwork and enhance the learning experience.

1.4 Integration with learning management systems: Integrate the online examination portal with learning management systems (LMS) such as **Noodle**, Blackboard, or Canvas to streamline the exam creation, delivery, and reporting process.

1.5 Accessibility features: Ensure that the online examination portal meets accessibility standards such as WCAG 2.0 to accommodate users with disabilities such as visual impairments or motor disabilities. Overall, incorporating mobile compatibility, advanced analytic, collaborative exams, integration with LMS, gamification, accessibility features, and block-chain technology can help to improve the functionality and effectiveness of an online examination portal.

REFERENCES

YouTubeLink

(https://youtu.be/Yb688IQjPnc?si=IiiDCdWq3qldXww1)

ChatGpt

(https://chat.openai.com/)

Tutorials Point

(https://www.tutorialspoint.com/servlets-and-jsp-tutorial
for-beginners/index.asp

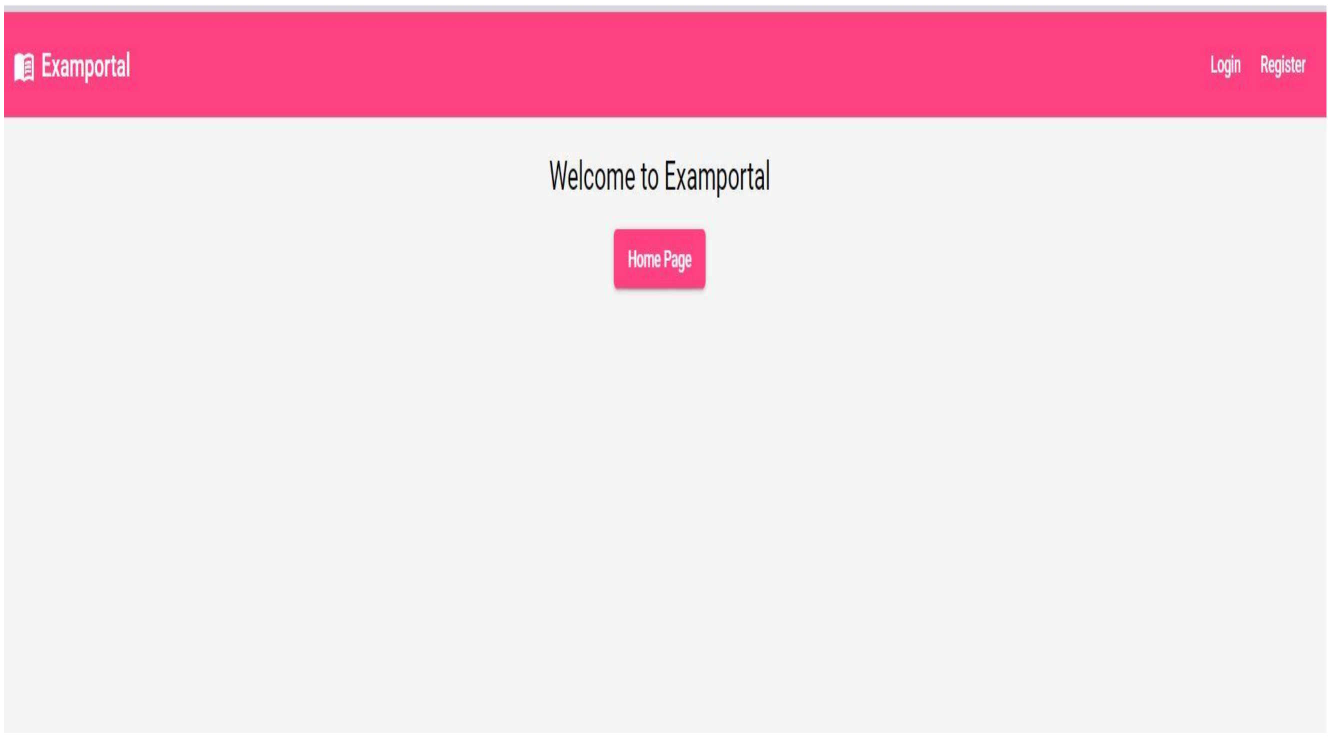
APPENDICES

Sample Code

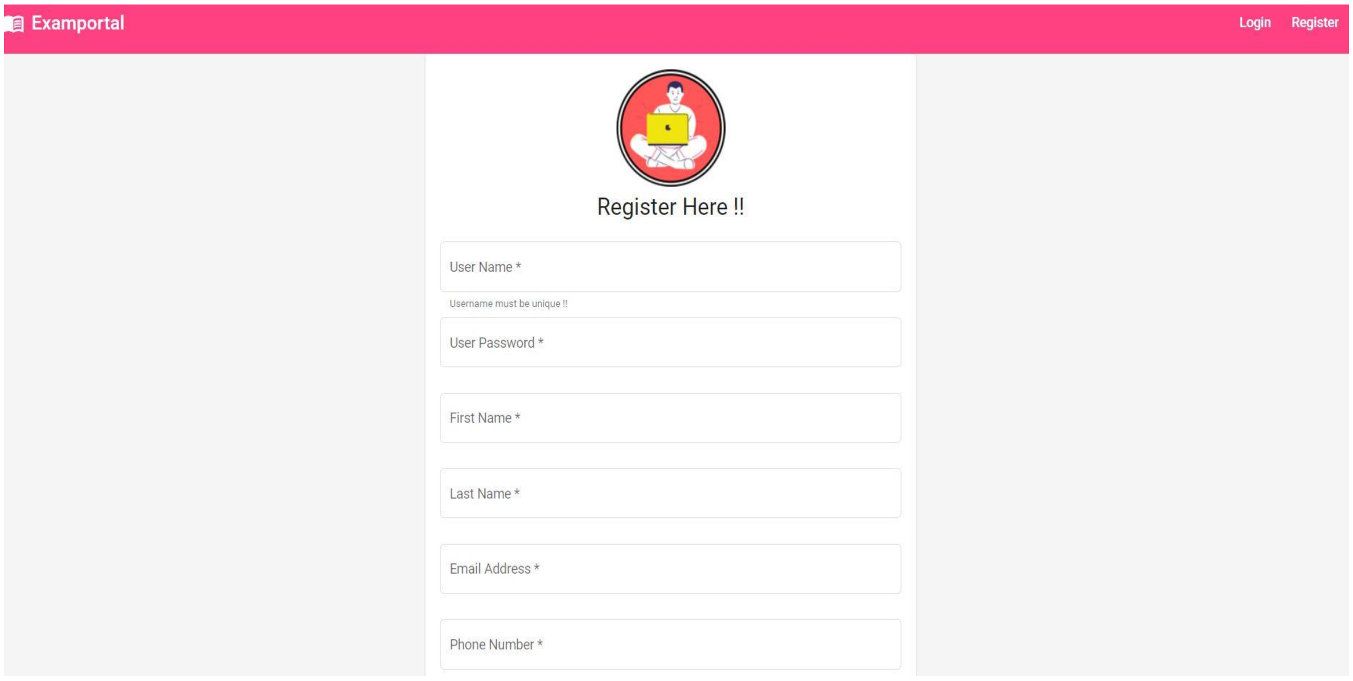
```
package com.exam.controller;


import com.exam.config.JwtUtils;
import com.exam.helper.UserNotFoundException;
import com.exam.model.JwtRequest;
import com.exam.model.JwtResponse;
import com.exam.model.User;
import com.exam.service.impl.UserDetailsServiceImpl;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Primary;
import org.springframework.http.ResponseEntity;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.authentication.BadCredentialsException;
import org.springframework.security.authentication.DisabledException;
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.web.bind.annotation.*;


import java.security.Principal;


@RestController
@CrossOrigin("*")
public class AuthenticateController {
    @Autowired
    private AuthenticationManager authenticationManager;


    @Autowired
    private UserDetailsServiceImpl userDetailsService;
    @Autowired
    private JwtUtils jwtUtils
   //generate token
 @PostMapping("/generate-token")
    public ResponseEntity<?> generateToken(@RequestBody JwtRequest jwtRequest) throws Exception {


        try {
            authenticate(jwtRequest.getUsername(), jwtRequest.getPassword());
        } catch (UserNotFoundException e) {
            e.printStackTrace();
            throw new Exception("User not found ");
        }


        ////////////authenticate
        UserDetails userDetails = this.userDetailsService.loadUserByUsername(jwtRequest.getUsername());
        String token = this.jwtUtils.generateToken(userDetails);
        return ResponseEntity.ok(new JwtResponse(token));
    }
    private void authenticate(String username, String password) throws Exception {
        try {
            authenticationManager.authenticate(new UsernamePasswordAuthenticationToken(username,
password));
        } catch (DisabledException e) {
            throw new Exception("USER DISABLED " + e.getMessage());
        } catch (BadCredentialsException e) {
            throw new Exception("Invalid Credentials " + e.getMessage());
        }
    }package com.exam.controller;
import com.exam.config.JwtUtils;
import com.exam.helper.UserNotFoundException;
import com.exam.model.JwtRequest;
import com.exam.model.JwtResponse;
import com.exam.model.User;
import com.exam.service.impl.UserDetailsServiceImpl;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Primary;
import org.springframework.http.ResponseEntity;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.authentication.BadCredentialsException;
import org.springframework.security.authentication.DisabledException;
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.web.bind.annotation.*;


import java.security.Principal;
@RestController
@CrossOrigin("*")
public class AuthenticateController {
    @Autowired
    private AuthenticationManager authenticationManager;
    @Autowired
    private UserDetailsServiceImpl userDetailsService;
    @Autowired
    private JwtUtils jwtUtils;

    //generate token
    @PostMapping("/generate-token")
    public ResponseEntity<?> generateToken(@RequestBody JwtRequest jwtRequest) throws Exception {
      try {
        authenticate(jwtRequest.getUsername(), jwtRequest.getPassword())


      } catch (UserNotFoundException e) {
        e.printStackTrace();
        throw new Exception("User not found ");
      }
      /////////////authenticate
      UserDetails userDetails = this.userDetailsService.loadUserByUsername(jwtRequest.getUsername());
      String token = this.jwtUtils.generateToken(userDetails);
      return ResponseEntity.ok(new JwtResponse(token));
    }
    private void authenticate(String username, String password) throws Exception {
      try {
        authenticationManager.authenticate(new UsernamePasswordAuthenticationToken(username,
  password));
      } catch (DisabledException e) {
        throw new Exception("USER DISABLED " + e.getMessage());
      } catch (BadCredentialsException e) {
        throw new Exception("Invalid Credentials " + e.getMessage());
      }
    }
```

```
import { Injectable } from '@angular/core';
import baseUrl from './helper';
@Injectable({
  providedIn: 'root',
})
export class UserService {
  constructor(private http: HttpClient) {}
 //add user
  public addUser(user: any) {
    return this.http.post(`${baseUrl}/user/`, user);
  }
import { TestBed } from '@angular/core/testing';
import { UserService } from './user.service';
describe('UserService', () => {
  let service: UserService;
  beforeEach(() => {
    TestBed.configureTestingModule({});
    service = TestBed.inject(UserService);
  });
  it('should be created', () => {
    expect(service).toBeTruthy();
  });
});
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AddCategoryComponent } from './pages/admin/add-category/add-category.component';
import { AddQuestionComponent } from './pages/admin/add-question/add-question.component';
import { AddQuizComponent } from './pages/admin/add-quiz/add-quiz.component';
import { DashboardComponent } from './pages/admin/dashboard/dashboard.component';
import { UpdateQuizComponent } from './pages/admin/update-quiz/update-quiz.component';
import { ViewCategoriesComponent } from './pages/admin/view-categories/view-
categories.component';
import { ViewQuizQuestionsComponent } from './pages/admin/view-quiz-questions/view-quiz-
questions.component';
import { ViewQuizzesComponent } from './pages/admin/view-quizzes/view-quizzes.component';
import { WelcomeComponent } from './pages/admin/welcome/welcome.component';
import { HomeComponent } from './pages/home/home.component';
import { LoginComponent } from './pages/login/login.component';
import { ProfileComponent } from './pages/profile/profile.component';
import { SignupComponent } from './pages/signup/signup.component';
import { InstructionsComponent } from './pages/user/instructions/instructions.component';
import { LoadQuizComponent } from './pages/user/load-quiz/load-quiz.component';
import { StartComponent } from './pages/user/start/start.component';
import { UserDashboardComponent } from './pages/user/user-dashboard/user-
dashboard.component';
import { AdminGuard } from './services/admin.guard';
import { NormalGuard } from './services/normal.guard';
const routes: Routes = [

    path: '',
    component: HomeComponent,
    pathMatch: 'full',

    path: 'signup',
    component: SignupComponent,
    pathMatch: 'full',
    path: 'login',
    component: LoginComponent,
    pathMatch: 'full'
    path: 'admin',
    component: DashboardComponent,
    canActivate: [AdminGuard],
    children:
        path: '',
        component: WelcomeComponent
        path: 'profile',
        component: ProfileComponent
        path: 'categories',
        component: ViewCategoriesComponent,
      },
      {
        path: 'add-category',
        component: AddCategoryComponent,
        path: 'quizzes',
        component: ViewQuizzesComponent,

        path: 'add-quiz',
        component: AddQuizComponent,

        path: 'quiz/:qid',
        component: UpdateQuizComponent,

        path: 'view-questions/:qid/:title',
        component: ViewQuizQuestionsComponent,
      },
      {
        path: 'add-question/:qid/:title',
        component: AddQuestionComponent,
      },
    ],
  },
  {
    path: 'user-dashboard',
    component: UserDashboardComponent,
    canActivate: [NormalGuard],
    children: [
      {
        path: ':catId',
        component: LoadQuizComponent,
      },
      {
        path: 'instructions/:qid',
        component: InstructionsComponent,
      }
    path: 'start/:qid',
    component: StartComponent,
    canActivate: [NormalGuard],
@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule],
})
export class AppRoutingModule
import { CUSTOM_ELEMENTS_SCHEMA, NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { MatButtonModule } from '@angular/material/button';
import { NavbarComponent } from './components/navbar/navbar.component';
import { FooterComponent } from './components/footer/footer.component';
import { SignupComponent } from './pages/signup/signup.component';
import { LoginComponent } from './pages/login/login.component';
import { MatInputModule } from '@angular/material/input';
import { MatFormFieldModule } from '@angular/material/form-field';
import { FormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';
import { MatSnackBarModule } from '@angular/material/snack-bar';
import { HomeComponent } from './pages/home/home.component';
import { MatCardModule } from '@angular/material/card';
import { MatToolbarModule } from '@angular/material/toolbar';
import { MatIconModule } from '@angular/material/icon';
import { authInterceptorProviders } from './services/auth.interceptor';
import { DashboardComponent } from './pages/admin/dashboard/dashboard.component';
import { UserDashboardComponent } from './pages/user/user-dashboard/user-
dashboard.component';
import { ProfileComponent } from './pages/profile/profile.component';
import { MatListModule } from '@angular/material/list';
import { SidebarComponent } from './pages/admin/sidebar/sidebar.component';
import { WelcomeComponent } from './pages/admin/welcome/welcome.component';
import { ViewCategoriesComponent } from './pages/admin/view-categories/view-
categories.component';
import { AddCategoryComponent } from './pages/admin/add-category/add-category.component';
import { ViewQuizzesComponent } from './pages/admin/view-quizzes/view-quizzes.component';
import { AddQuizComponent } from './pages/admin/add-quiz/add-quiz.component';
import { MatSlideToggleModule } from '@angular/material/slide-toggle';
import { MatSelectModule } from '@angular/material/select';
import { UpdateQuizComponent } from './pages/admin/update-quiz/update-quiz.component';
import { ViewQuizQuestionsComponent } from './pages/admin/view-quiz-questions/view-quiz-
questions.component';
import { AddQuestionComponent } from './pages/admin/add-question/add-question.component';
import { CKEditorModule } from '@ckeditor/ckeditor5-angular';
```

Screenshots:



User Interface



Register Here !!

User Name *

Username must be unique !!

User Password *

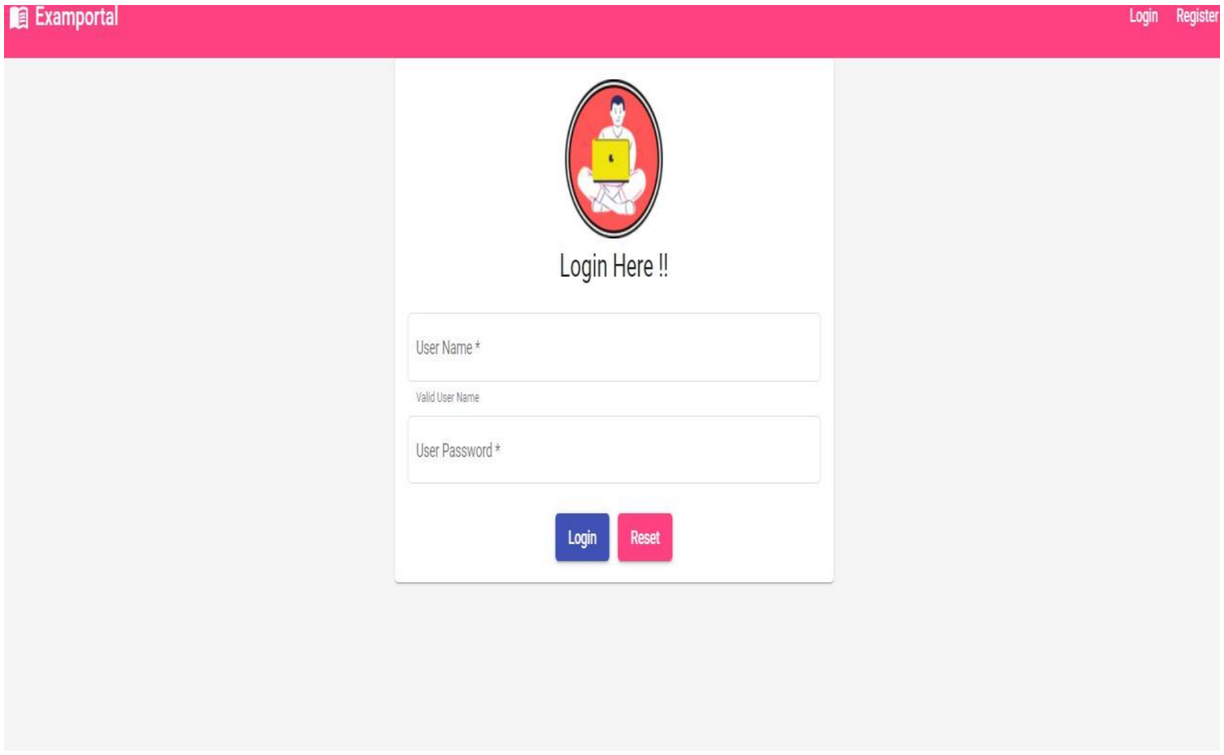
First Name *

Last Name *

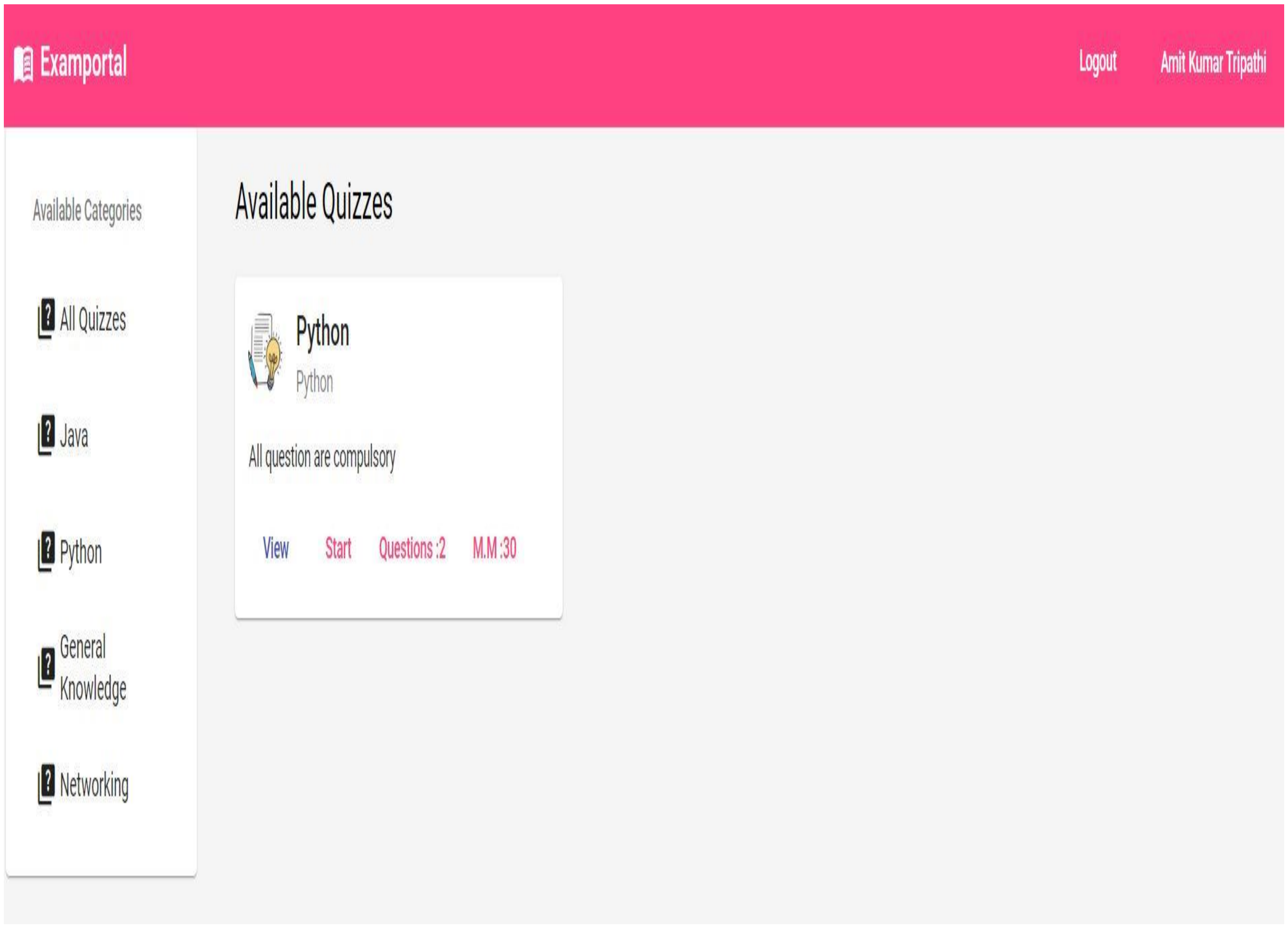
Email Address *

Phone Number *

Registration Page


Examportal
Login
Register
Login Here !!
User Name *
Valid User Name
User Password *
Login
Reset

Login Page



Show Quizzes