EPITA

Final Project (Quiz Management) ADVANCE JAVA

ABSTRACT-USERGUIDE & TECHNICAL SPECIFICATION

BHRIGU MAHAJAN (ME-SDM) RITU SHIKHA (MSC-SE)

22th June ‘2018

Table of Contents

Introduction

This document will propose all features and technical specification of the project. It contains details about objectives, scope limitation, primary requirements, possible project risks, database design, and application flow.

The goal of this project is to develop an application (API oriented, Web-based) that deals with quiz assessments.

The usual problem while preparing an evaluation, are to:

- Constitute an appropriate evaluation regarding of the required level
- Reuse former questions
- Organize sample evaluations
- Correct automatically the MCQ questions.

The implementation of this project has been done by using Frontend (HTML, Angular JS), Backend (REST, Java 8), Database (Derby), Business Logic Framework (Hibernate, Spring) and Server (Tomcat).

Objectives and concentrations:

This Quiz Management Application handles 2 Identities

-Student

-Professor/Admin

Students are having privilege to create the account, choose the relevant quiz type and quiz name, attend the quiz, view the result at the same time.

Professor on the other hand is able to create his/her account, edit/delete/view the registered student's details with their quiz score. Apart, from managing the student's details they are able to create the quiz specifying name, and type.

Scope and limitations:

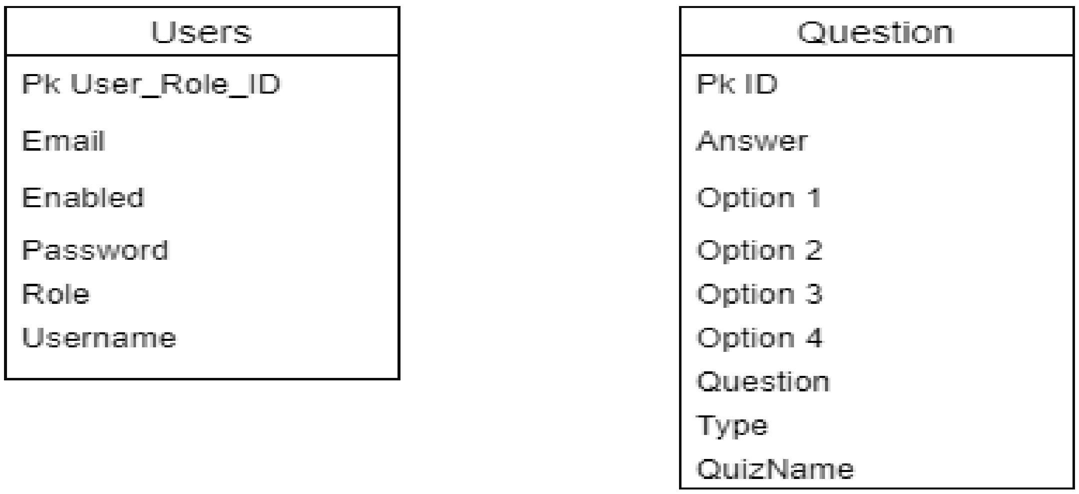
- The system handles all the operations, and generates reports as soon as the test is finish, that includes name, mark, time spent to solve the exam.
- Allow students to see or display his answers after the exam is finish.
- The type of questions is only multiple choice right.
- Manage the students, the admin/professors can add/edit/delete student and review their scores

Software Requirements:

Technology Used

BACKEND/WEBSERVER	FRONTEND	DATABASE	VERSION CONTROL
JAVA - version 8	HTML5	DERBY	GIT
SPRING – version 5.0.4	CSS3	DBeaver –version 5.1.1	
JSP – version 2.2	JAVASCRIPT	HIBERNATE– version 5.2.14	
TOMCAT – version 8.5	SERVLET		

Data Model:



We kept the data model simple with 2 tables called USERS and QUESTION.

Testing

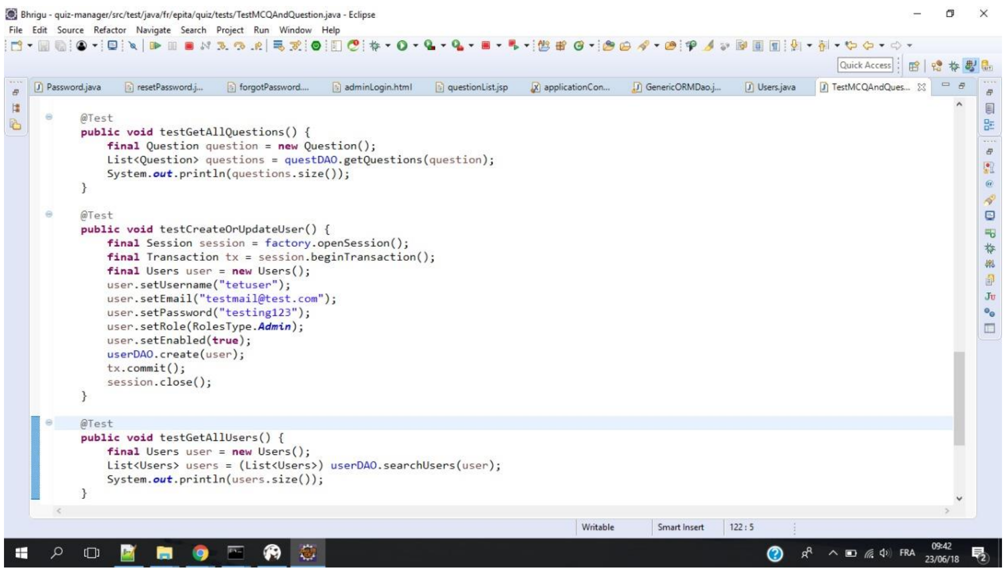
To satisfy the requirements and check whether every build services in the application are executed as expected, one of the earliest, testing efforts performed on the code is Unit Testing. To quickly test any new code or changes to existing code without the overhead and additional time involved in tasks such as server configuration, services setup and application deployment, we integrated the popular testing framework JUnit Framework to execute the unit test on every small code module.

Our objectives are in writing and executing the tests

- We want to code and run the tests without leaving the IDE (Eclipse).

- There should be no special deployment of the code required

- We should be able to exploit other code analysis tools such as Metrics and FindBugs right from within the IDE so we can find any bugs right away and fix those issues.


Bhrigu - quiz-manager/src/test/java/fr/epita/quiz/tests/TestMCQAndQuestion.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Password.java resetPassword.j... forgotPassword.... adminLogin.html questionList.jsp applicationCon... GenericORMDao.j... Users.java TestMCQAndQues...
Writable Smart Insert 122 : 5
FRA 09:42 23/06/18

```
@Test
public void testGetAllQuestions() {
    final Question question = new Question();
    List<Question> questions = questDAO.getQuestions(question);
    System.out.println(questions.size());
}

@Test
public void testCreateOrUpdateUser() {
    final Session session = factory.openSession();
    final Transaction tx = session.beginTransaction();
    final Users user = new Users();
    user.setUsername("tetuser");
    user.setEmail("testmail@test.com");
    user.setPassword("testing123");
    user.setRole(RolesType.Admin);
    user.setEnabled(true);
    userDAO.create(user);
    tx.commit();
    session.close();
}

@Test
public void testGetAllUsers() {
    final Users user = new Users();
    List<Users> users = (List<Users>) userDAO.searchUsers(user);
    System.out.println(users.size());
}
```

Spring Integration with Hibernate

We integrated hibernate framework with this spring application.

No.	Method	Description
1)	void persist(Object entity)	persists the given object.
2)	Serializable save(Object entity)	persists the given object and returns id.
3)	void saveOrUpdate(Object entity)	persists or updates the given object. If id is found, it updates the record otherwise saves the record.
4)	void update(Object entity)	Updates the given object.
5)	void delete(Object entity)	deletes the given object on the basis of id.

6)	Object get(ClassentityClass, Serializable id)	returns the persistent object on the basis of given id.
7)	Object load(ClassentityClass, Serializable id)	returns the persistent object on the basis of given id.
8)	List loadAll(Class entityClass)	returns the all the persistent objects.

Application Module

This section gives a functional requirement that applicable to the Quiz Application.

There are two sub modules in this phase.

- ✓ Admin module.
- ✓ User module.

The functionality of each module is as follows:

- ✓ **User module**: The candidate will logon to the application and take his quiz. He can check the list of question type, name and then attend the quiz as per his preference. The candidate will get result immediately after the completion of the examination.
- ✓ **Admin module:** The professor/admin will logon to the application and take his/her quiz. They can manage the student identities, create quiz – MCQ type, ASSOCIATIVE type, OPEN type, and view the marks of every student.

Major Features

- ✓ **Login/SignUp Authentication**

- ✓ **Manage Identities**
  Add Student
  Update Student
  Delete Student
  Search Student

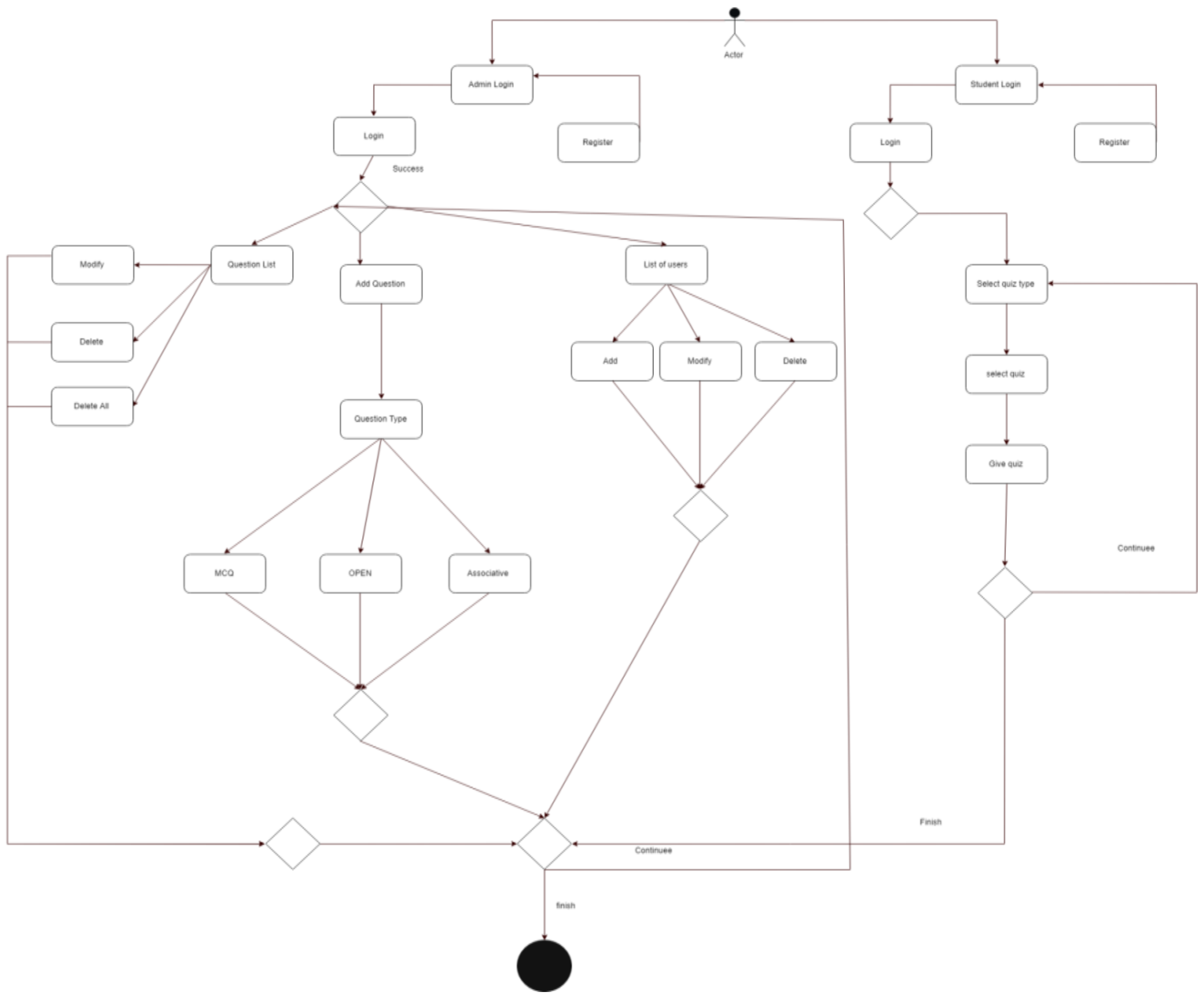
- ✓ **Manage Questions**
  Add Questions

Update Questions
Delete Questions
Search Questions

- ✓ **Take Quiz**

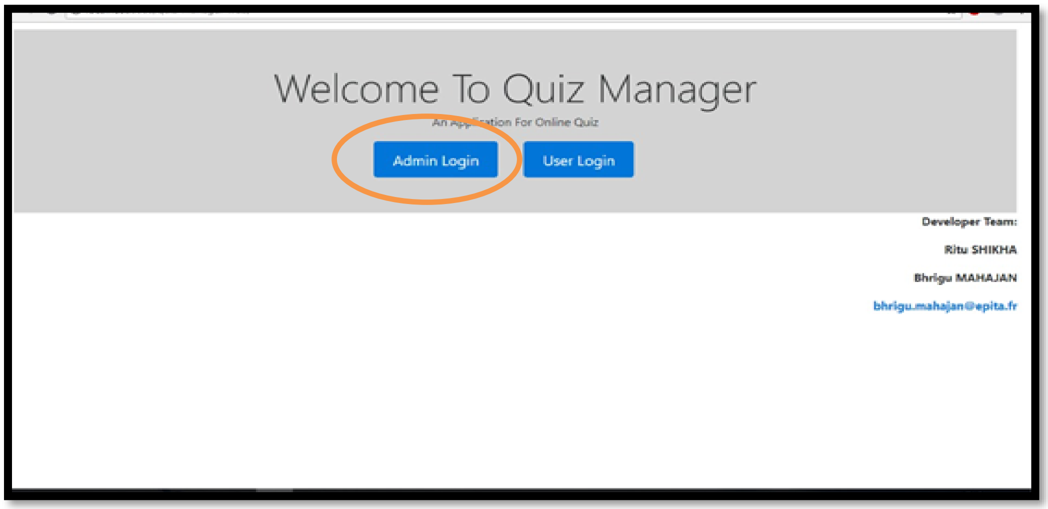
  View Questions
  Give Answers

Global Application Flow:



Screenshots and Workflow

The Hone page of the application


Welcome To Quiz Manager

An Application For Online Quiz

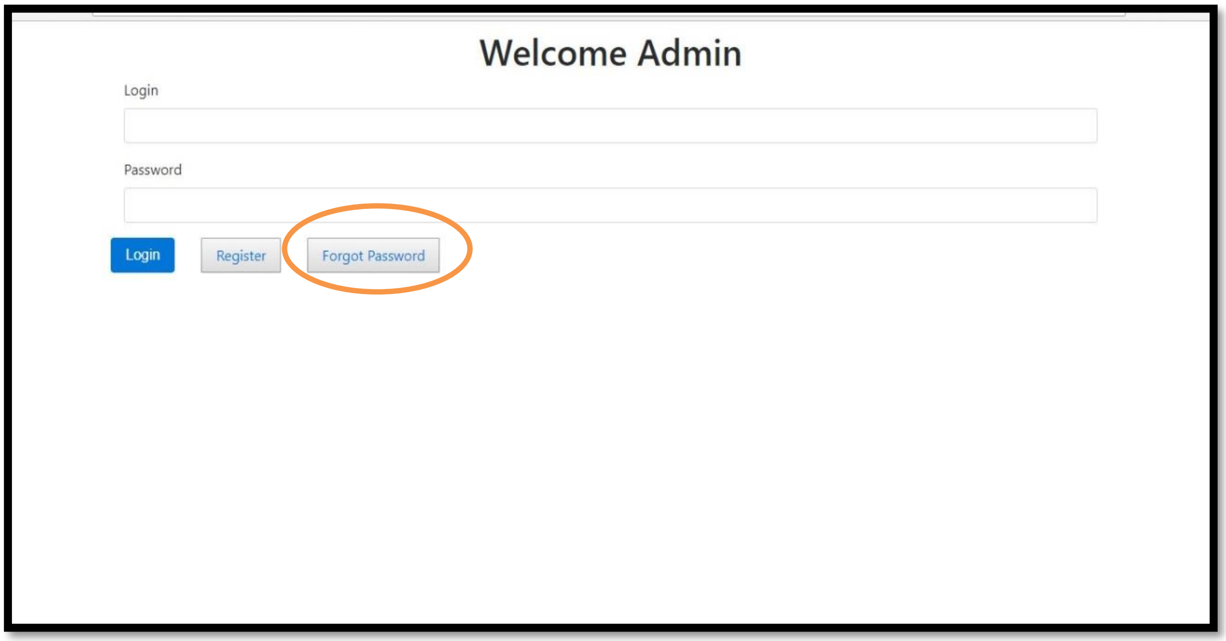
Developer Team:

Ritu SHIKHA

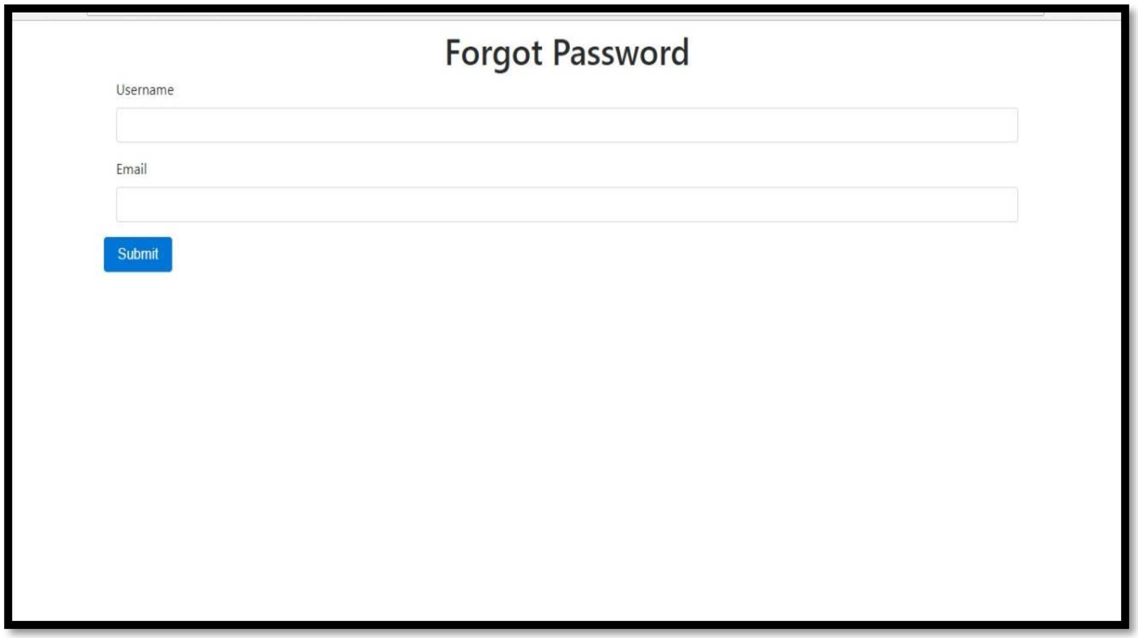
Bhrigu MAHAJAN

bhrigu.mahajan@epita.fr

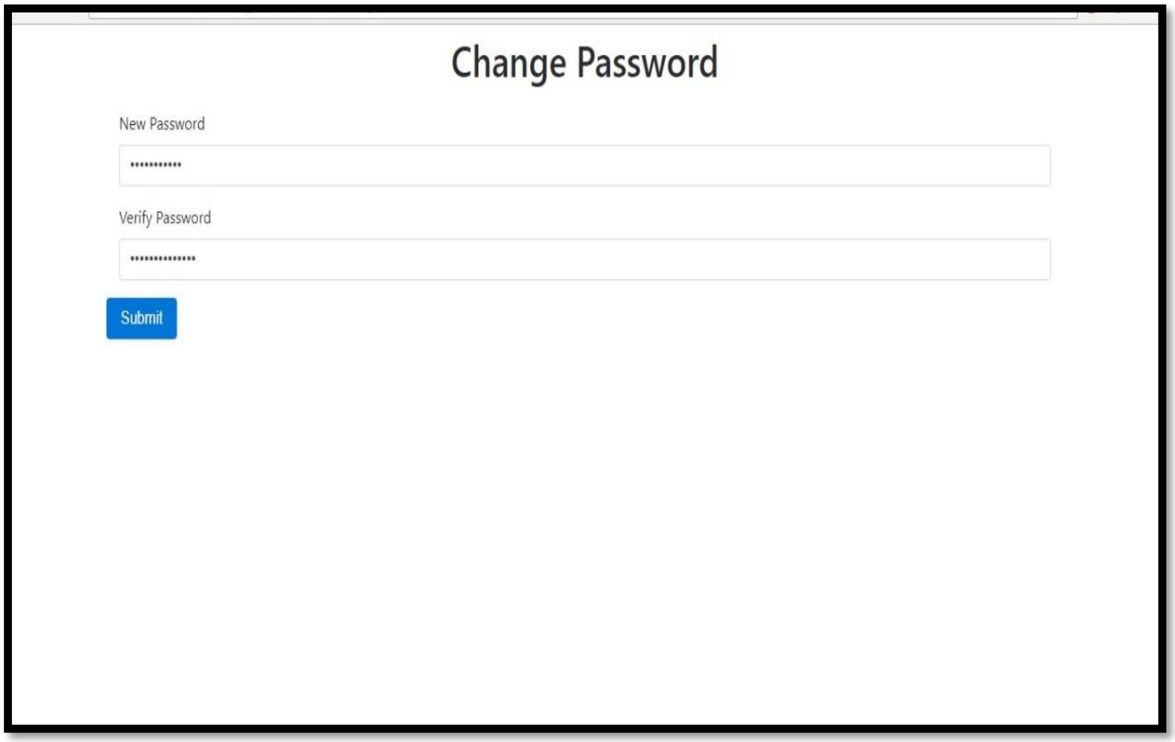
Admin login where the admin has the option to login, Register and recover password.


Welcome Admin

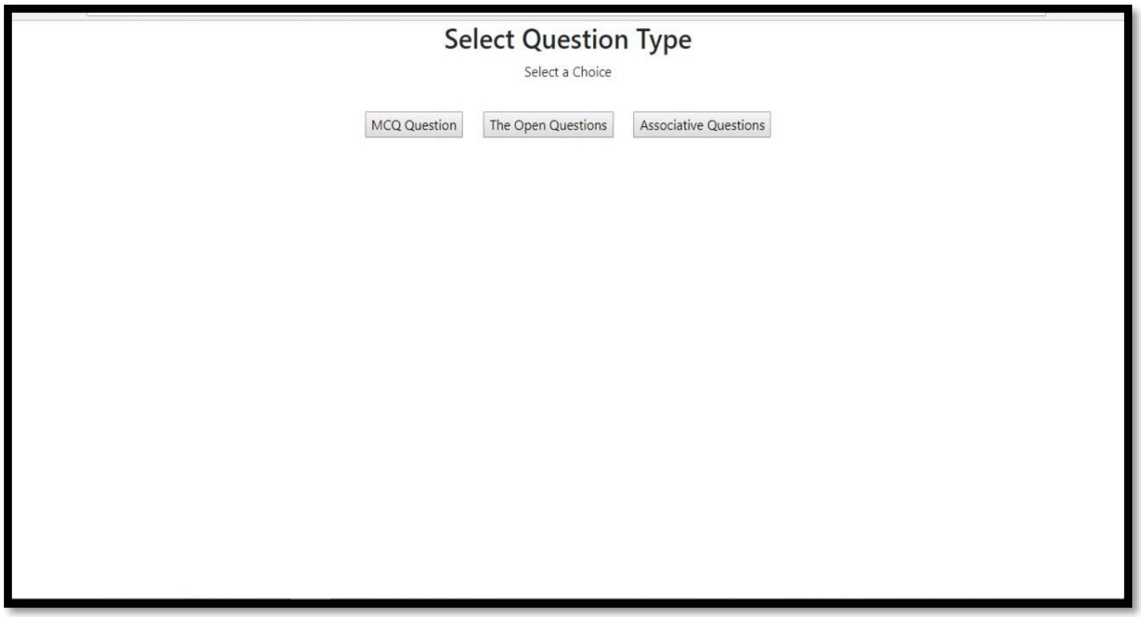
Page which is displayed when the admin clicks forgot password where authentication is required



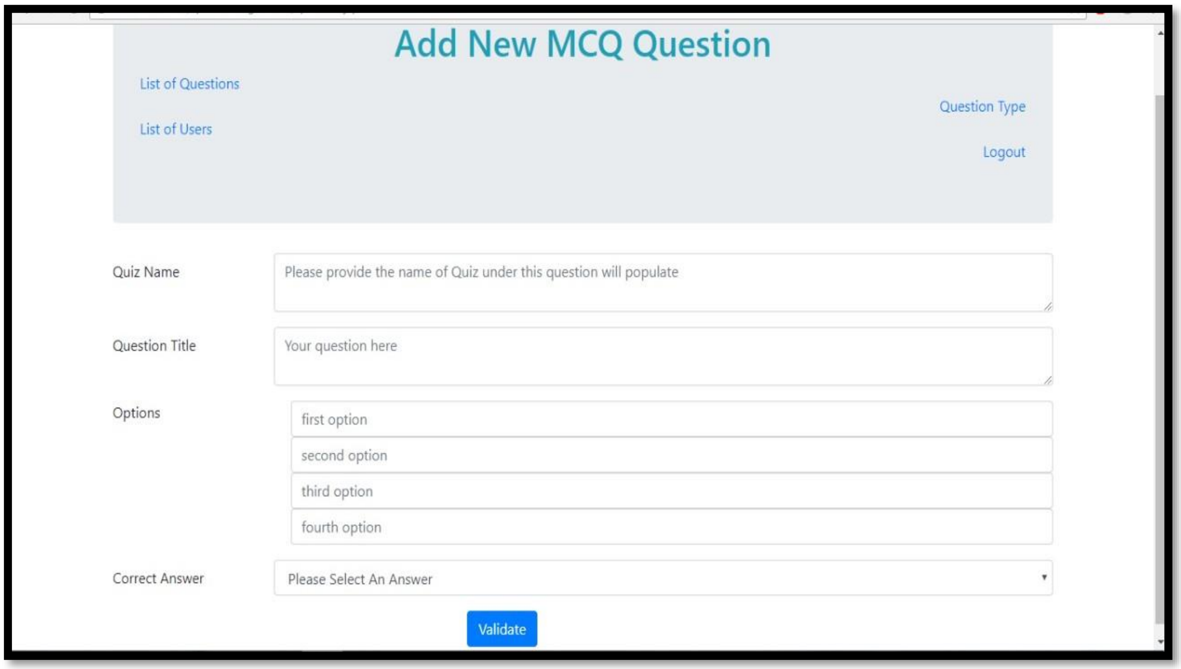
After authentication the admin arrives at this page where he can update the password. After changing the password the user goes to the Admin Login page



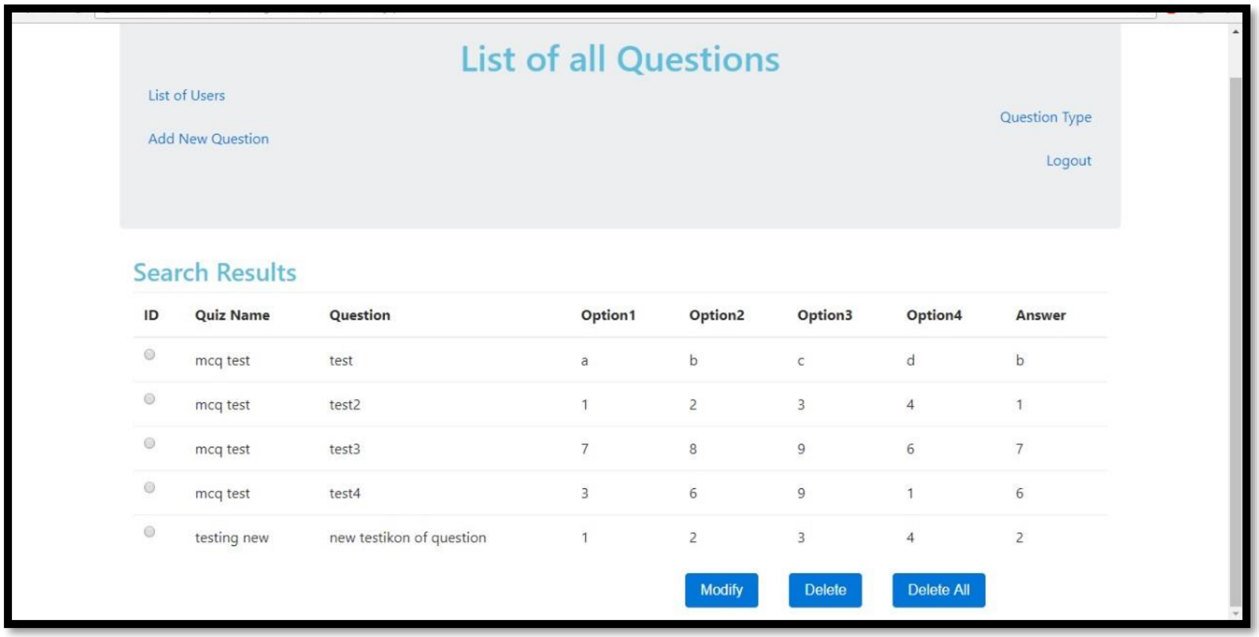
After login the admin can select the type of questions he want to create.



This page is displayed when the admin clicks on the Mcq question. The admin can add different options for a question and specify the right answer. The Admin can also go to the list of questions, list of usersm back to the question type page and also logout



This is the list of all the questions where the admin can modify, delete and delete all the questions.



List of all Questions

List of Users

Add New Question

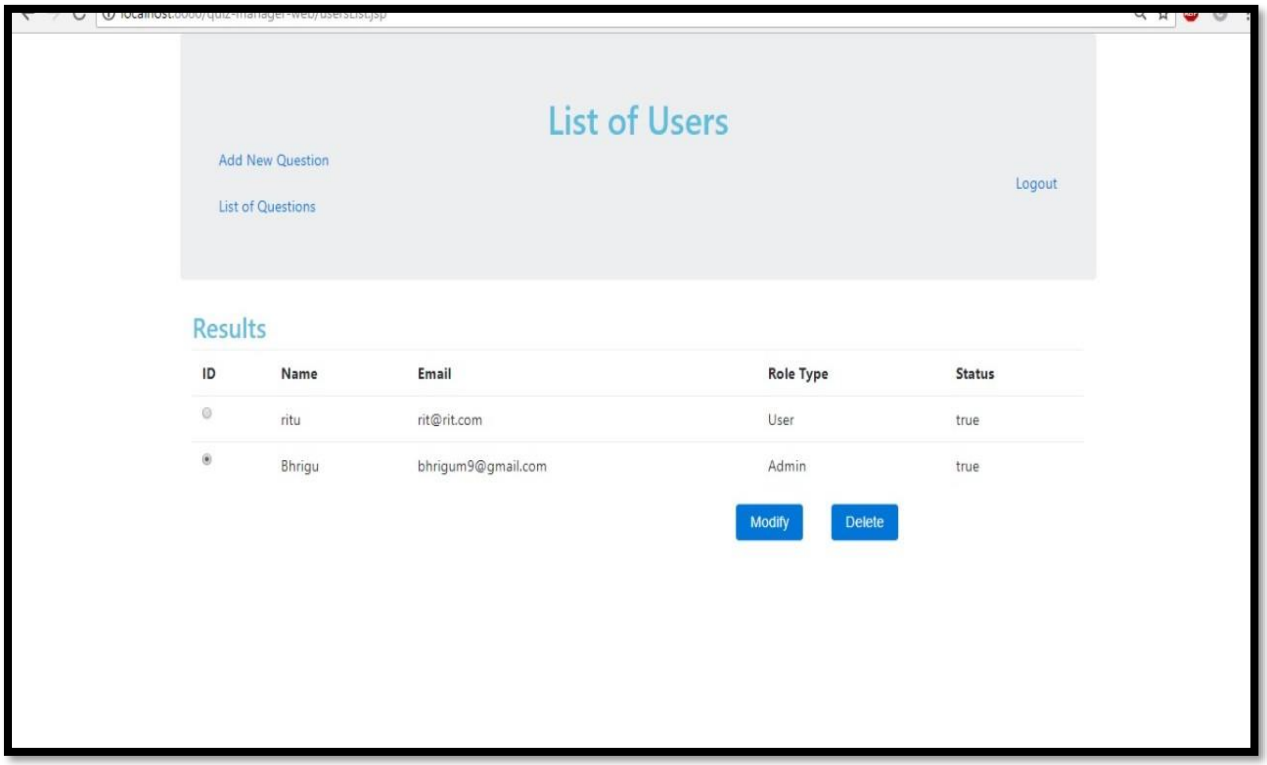
Question Type

Logout

Search Results

ID	Quiz Name	Question	Option1	Option2	Option3	Option4	Answer
	mcq test	test	a	b	c	d	b
	mcq test	test2	1	2	3	4	1
	mcq test	test3	7	8	9	6	7
	mcq test	test4	3	6	9	1	6
	testing new	new testikon of question	1	2	3	4	2

This is the list of all the users where the admin can modify, delete and delete the users.



List of Users

Add New Question

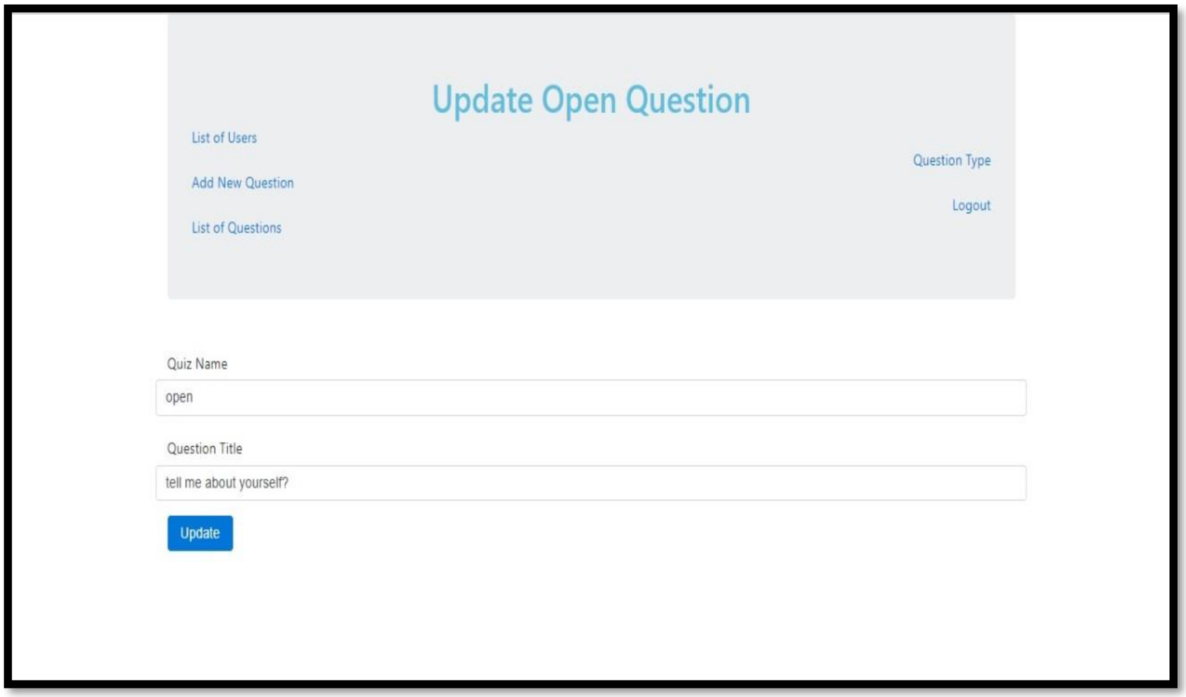
List of Questions

Logout

Results

ID	Name	Email	Role Type	Status
	ritu	rit@rit.com	User	true
●	Bhrigu	bhrigum9@gmail.com	Admin	true

This is the view for updating the open questions



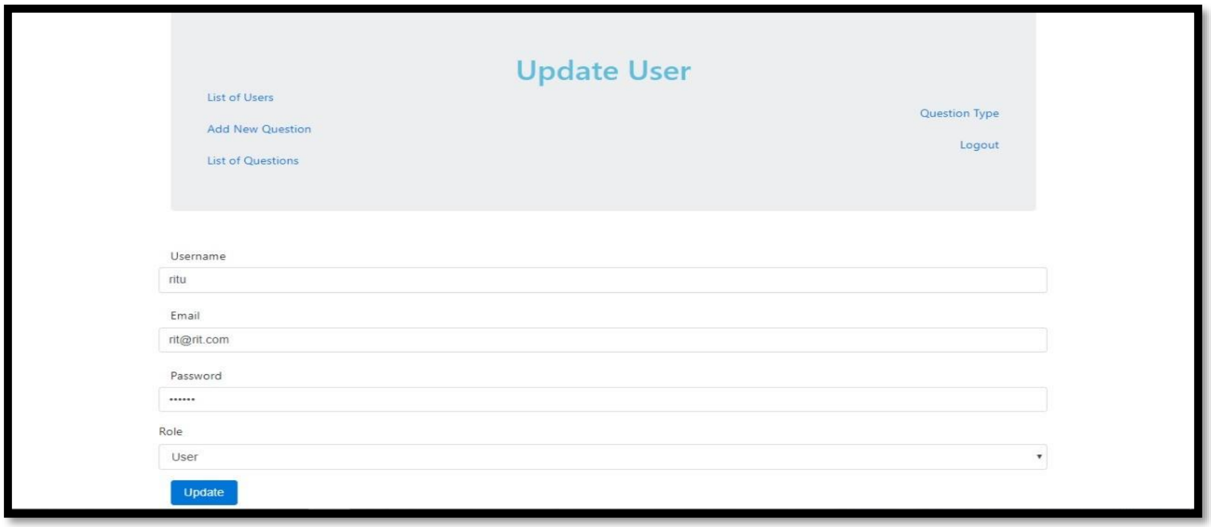
Quiz Name

open

Question Title

tell me about yourself?

This is the page for updating the user who are not admin.



Username

ritu

Email

rit@rit.com

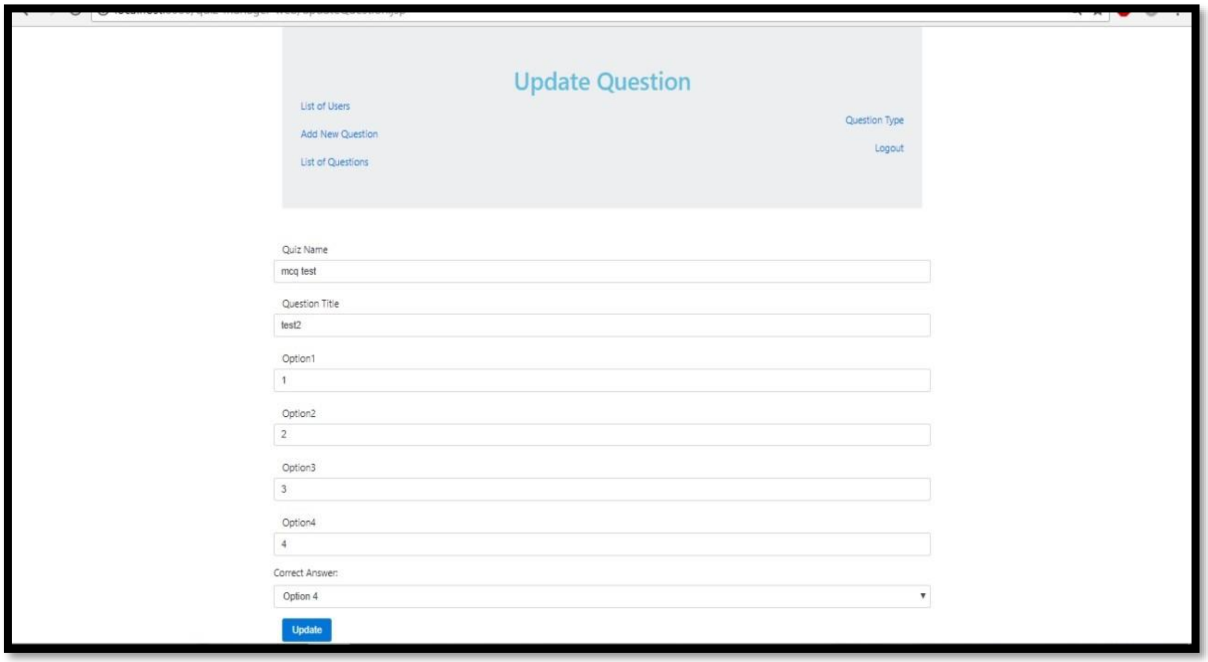
Password

••••••

Role

User

The admin is having the privilege to update the quiz question including its name, title, and the corresponding answer.



Quiz Name
mcq test

Question Title
test2

Option1
1

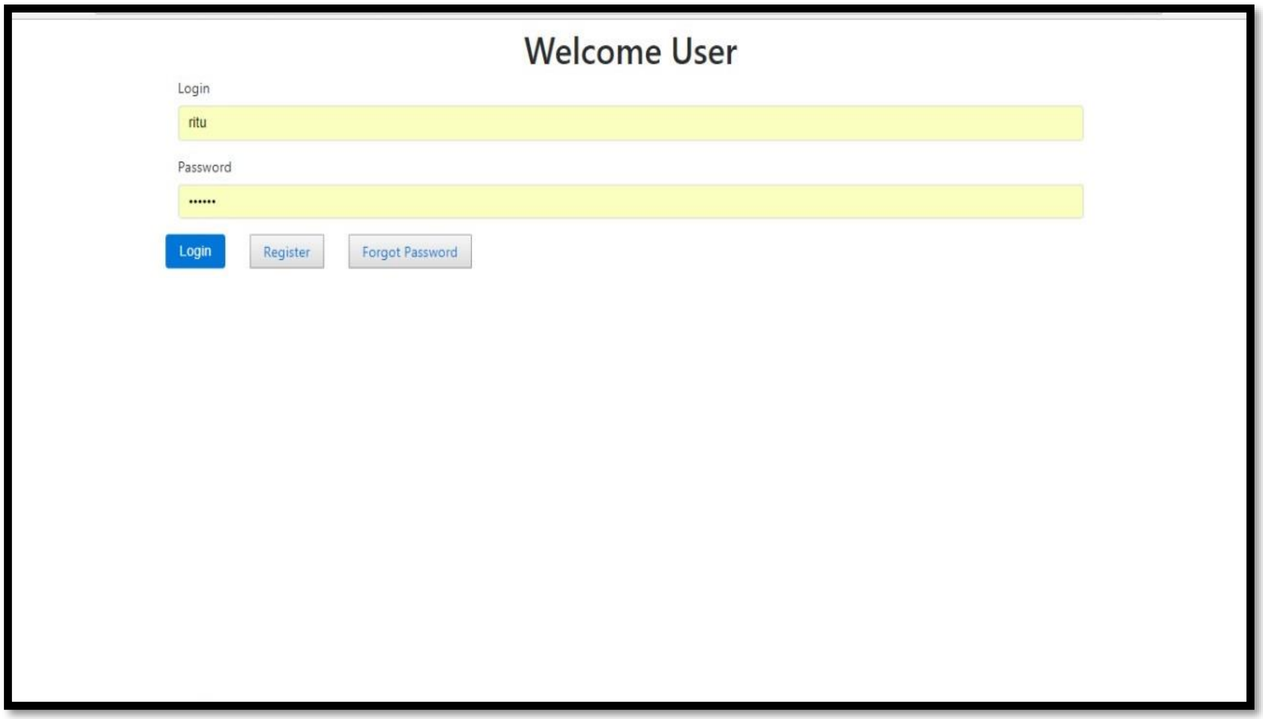
Option2
2

Option3
3

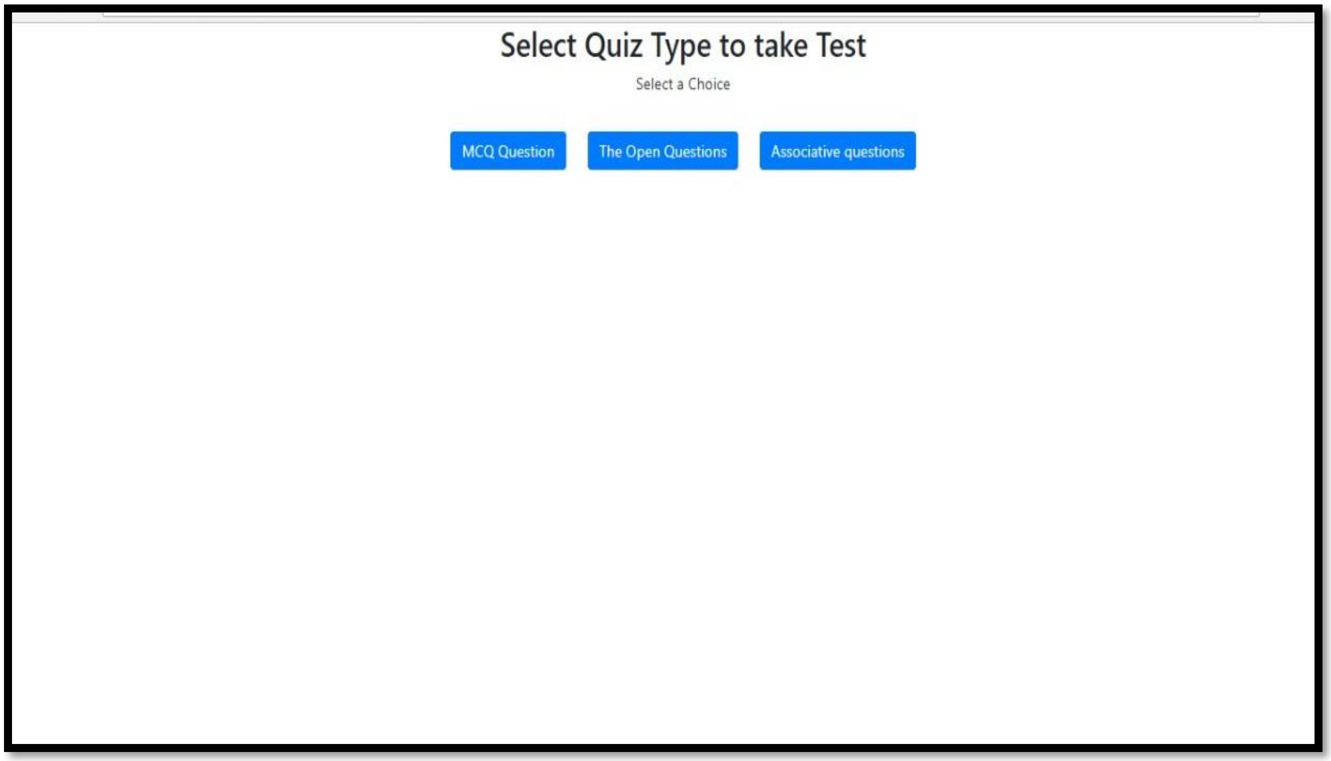
Option4
4

Correct Answer:
Option 4

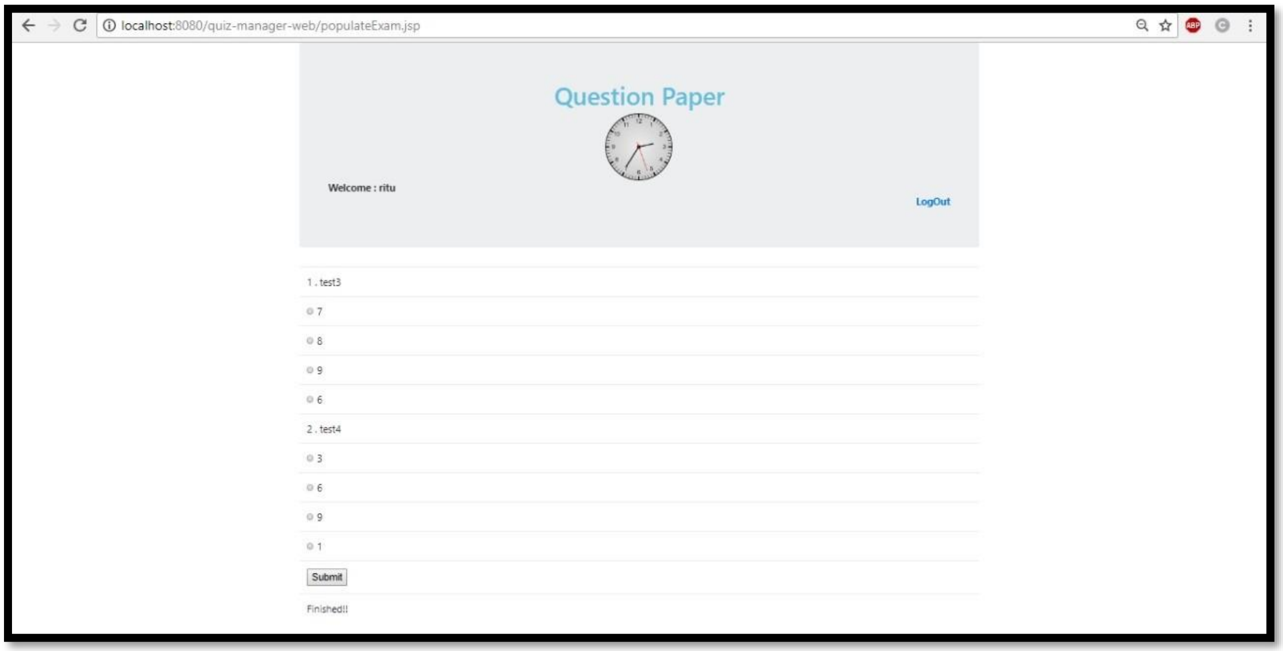
Here is the Welcome page for User with Login, Password, register and the forgot password option



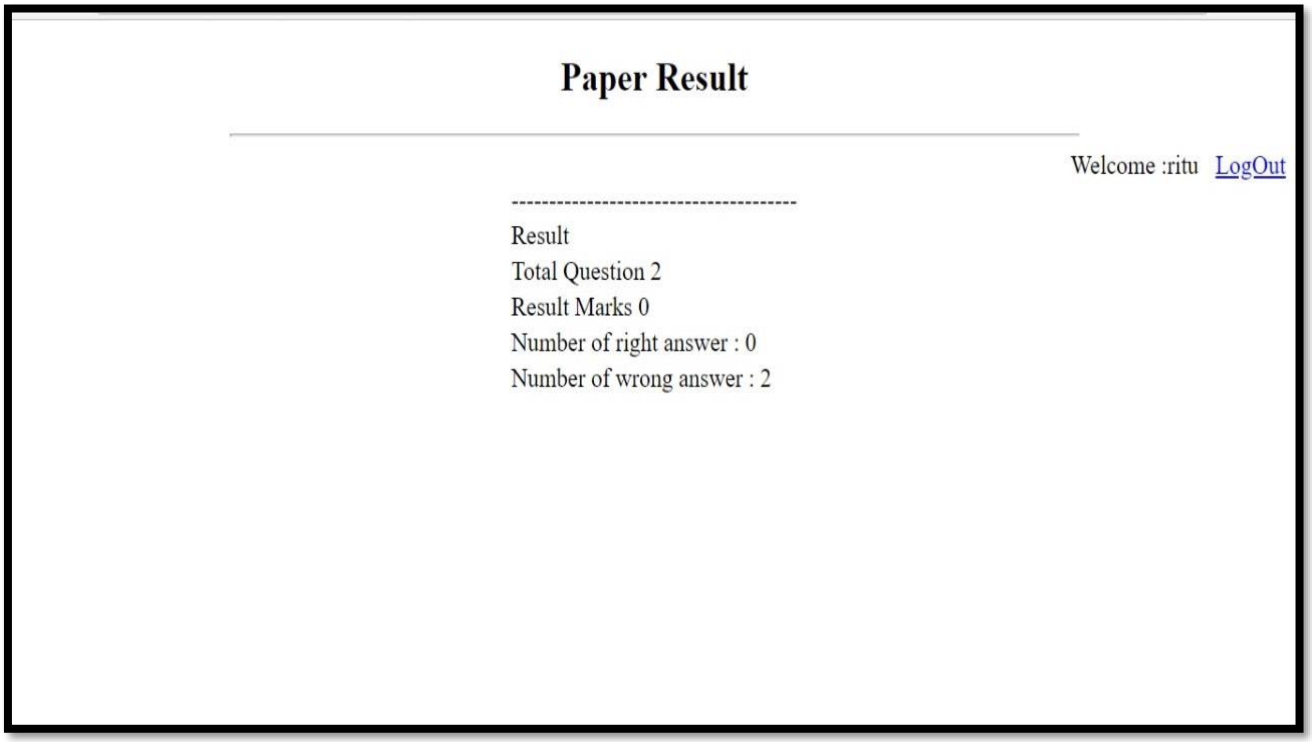
User can select the quiz type – from among the option displayed and then play the quiz



Below is the quiz page where the user can take test attempting the multiple choice question.

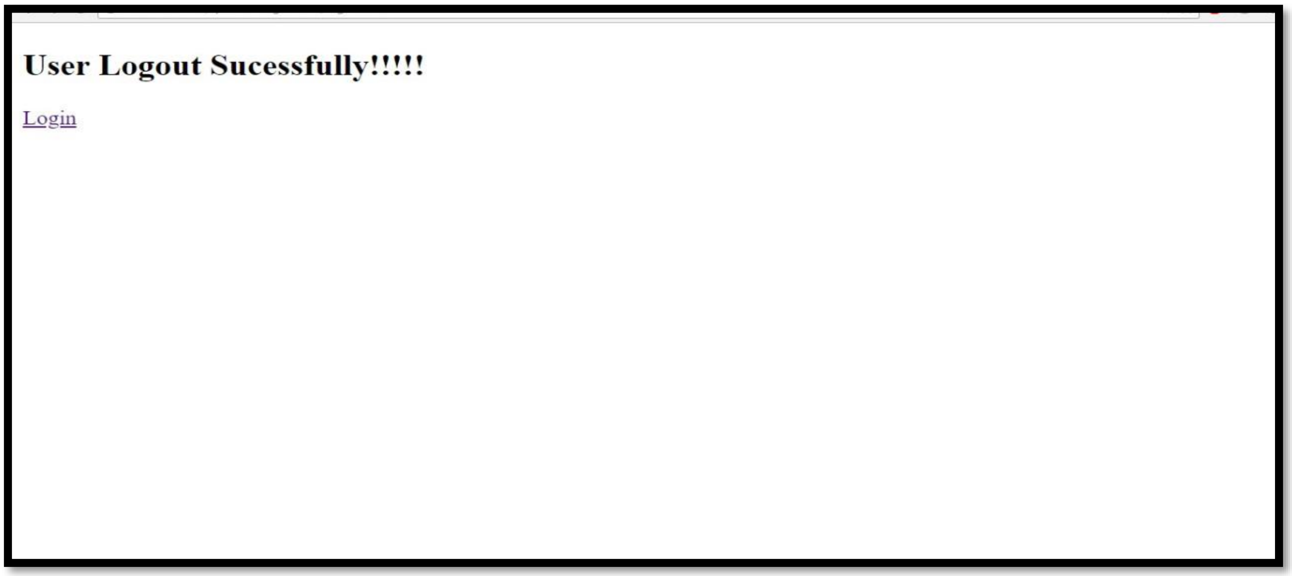


After submitting the quiz the user will be navigated to the result page- that provides the detail analysis of the quiz i.e Total Question, Result Marks, Number of right answers, Number of wrong answers


Paper Result

Welcome :ritu LogOut

```
----------------------------------------
Result
Total Question 2
Result Marks 0
Number of right answer : 0
Number of wrong answer : 2
```


User Logout Sucessfully!!!!!

Login

If the user needs to end his session, here is the logout page available to move the user out from his logged In account

Software Quality Attributes

The Quality of the System is maintained in such a way so that it can be very user friendly to all the users.

The software quality attributes are assumed as under:

- ✓ Accurate and hence reliable.
- ✓ Secured.
- ✓ Fast speed.
- ✓ Compatibility.

System Interfaces:

This section describes how the software interfaces with other software products or users for input or output.

User Interface

Application will be accessed through a Browser Interface. The interface would be viewed best using 1024 x 768 and 800 x 600 pixels resolution setting. The software would be fully compatible with Microsoft Internet Explorer for version 6 and above. No user would be able to access any part of the application without logging on to the system.

Hardware Interfaces

Server Side:

- ✓ Operating System: Windows 9x/xp ,Windows ME
- ✓ Processor: Pentium 3.0 GHz or higher
- ✓ RAM: 256 Mb or more
- ✓ Hard Drive: 10 GB or more

Client side:

- ✓ Operating System: Windows 9x or above, MAC or UNIX.
- ✓ Processor: Pentium III or 2.0 GHz or higher.
- ✓ RAM: 256 Mb or more

Future Scope

- ✓ Variation in question type- Associative and Open
- ✓ Implementing Security features.
- ✓ Better UI.
- ✓ Implement the Enabled Flag
- ✓ Timer – Timeout Functionality

Bibliography:

http://thomas-broussard.fr/work/java/courses/project/advanced.xhtml