**ONLINE EXAMINATION PORTAL**

GOUTHAMNATH.P - 185001052

JAANHAVI.S – 185001062

**DESCRPTION:**

Online Examination Portal is a java application made to attend a quiz for a particular subject. This portal has two modules: 1) Professor 2) Student.

Implemented using Java swing applications (GUI). MySQL is used as a database to store the questions and answers, the professor id and passwords , student id and passwords and a table to access the present student’s random ques number and the respective score.

**Project Objective:**

* Online Examination System can reduce the hectic job of assessing the answers given by candidates manually.
* Responses or the answers by the candidates can be checked automatically and instantly.
* It will reduce paper work to be an integrated Online Examination System.
* The result can be shown immediately to the students reducing the anxiety.
* The questions are random. Thus , for every login the questions will be of different order and questions will also differ

**Project Modules:**

* 1. PROFESSOR:

Has two applications

* View Scores of the students who attended the test
* Add a new question into the database
  1. STUDENT
* Student will take up the test
* After the test is over , the student can view their performance history and the mark they scored in the test

**Class diagram and Function description:**

1. **StartPage.java :: JFrame**

|  |
| --- |
| **StartPage** |
| --- ProfButtonActionPerformed(): void  --- StudentButtonActionPerformed() : void |

* + **ProfButtonActionPerformed() –** directs you to professor login
  + **StudentButtonActionPerformed() –** directs you to student login

**PROFESSOR MODULE:**

**1)ProfLogin.java :: JFrame**

|  |
| --- |
| **ProfLogin** |
| --- backButtonActionPerformed() : void  --- resetButtonActionPerformed(): void  --- LoginButtonActionPerformed() : void |

* + **backButtonActionPerformed() –** goes back to start page
  + **resetButtonActionPerformed() –** clears the texts that have been typed in the username and password field
  + **LoginButtonActionPerformed() –** if the username and password are correct , it directs you to the professor menu page. Else , “incorrect username/password “ message will be shown

1. **ProfMenu.java :: JFrame**

|  |
| --- |
| **ProfMenu** |
| --- ViewButtonActionPerformed() : void  --- AddQuesButtonActionPerformed() : void  --- StartButtonActionPerformed() : void |

* + **ViewButtonActionPerformed() –** will be able to view the student name who performed the test along with the scores they have scored in the test
  + **AddQuesButtonActionPerformed() –** directs you to a page where you will be able to add a question
  + **StartButtonActionPerformed() –** goes back to start page

|  |
| --- |
| **WrongChoiceException** |
| --- choice:int |
| + WrongChoiceException(int ):  + toString(): String: |

1. **AddQuestion.java :: JFrame**

|  |
| --- |
| **Exception** |
|  |

**<-------------------------**

|  |
| --- |
| **AddQuestion** |
| --- AddButtonActionPerformed(): void |

* + **AddButtonActionPerformed() –** if the details entered are valid , shows “saved” message and adds the question into the database. Else , shows exceptions
    - 1. **WrongChoiceException:** invalid choice entered
      2. **NullPointerException:** any field is left blank
      3. **NumberFormatException:** string entered instead of number

**STUDENT MODULE:**

1. **StudentLogin.java :: JFrame**

|  |
| --- |
| **StudentLogin** |
| --- username : String |
| --- backButtonActionPerformed() : void  --- resetButtonActionPerformed() : void  --- LoginButtonActionPerformed() : void  **---** AddNewButtonActionPerformed() : void  + getUsername() : String |

* + **backButtonActionPerformed() –** goes back to start page
  + **resetButtonActionPerformed() –** clears the texts that have been typed in the username and password field
  + **LoginButtonActionPerformed() –** if the username and password are correct , it directs you to the quiz rule page. Else , “incorrect username/password “ message will be shown
  + **AddNewButtonActionPerformed() –** directs you to the page where you can add new professor details for signing in
  + **getUsername() –** to make the username accessible throughout the course of the quiz

1. **AddNewStudent.java :: JFrame**

|  |
| --- |
| **AddNewStudent** |
| **---** addButtonActionPerformed() : void |

* + **addButtonActionPerformed() -** checks whether the password entered and confirm password text are same and then adds the student into the databse with default score as -1

1. **StudentNext.java :: JFrame**

|  |
| --- |
| **StudentNext** |
| --- nextButtonActionPerformed(): void  --- resetOb(): void  --- resetTable(): void  + getRand(int ques ) :int  + setQuestion(JLabel,int ques): void  + setChoice( JLabel, int ques , int choice): void  + checkAns(int ques,int choice) : void |

* + This page explains the rules of the quiz
  + **nextButtonActionPerformed() –** goes to the timer page and the quiz starts
  + **resetOb() –** creates a new Question form
  + **resetTable() –** resets the view Perfomance table present in the database to null
  + **getRand(int ques ) :int –** returns the random number for the particular question number
  + **setQuestion(JLabel,int ques) –** sets the question(label) for the particular question number
  + **setChoice( JLabel, int ques , int choice) –** sets the choice in the label for that particular question and choice
  + **checkAns(int ques,int choice) -** checks whether the chosen answer is right and updates it in the database

1. **QuizTimer.java :: JFrame**

|  |
| --- |
| **QuizTimer** |
| **+** count : static int = 60 |
| --- startButtonActionPerformed() : void |

* + This page displays the timer for the quiz
  + **startButtonActionPerformed() –** the test starts

1. **Question1.java (common to all other 4 question forms) :: JFrame**

|  |
| --- |
| **Question1** |
| **---** score : int = 0  ~ sn: new StudentNext()  + qob1 = static new Question1() |
| --- backButtonActionPerformed() :void  --- nextButtonActionPerformed() : void  **---** Q1AncestorAdded() : void  --- choice1AncestorAdded() : void  --- choice2AncestorAdded() : void  --- choice3AncestorAdded() : void  --- choice4AncestorAdded() : void  --- choice1ActionPerformed() : void  --- choice2ActionPerformed() : void  --- choice3ActionPerformed() : void  --- choice4ActionPerformed() : void |

* + The question and options are displayed. The student can click the option that are sufficient
  + **backButtonActionPerformed() –** goes to the previous question
  + **nextButtonActionPerformed() –** goes to the next question
  + **AncestorAdded() –** dynamically addes te questions into the label which calls this event
  + **Choice1ActionPerformed() -** if that radio button is clicked the necessary function to check the answer is called

1. **TestEnd.java :: JFrame**

|  |
| --- |
| **TestEnd** |
| --- endButtonActionPerformed(): void  --- backButtonActionPerformed(): void |

* + **endButtonActionPerformed() –** ends the text and goes to the menu page
  + **backButtonActionPerformed() –** goes back to the previous question

1. **ViewPer.java :: JFrame**

|  |
| --- |
| **ViewPer** |
| --- total : int =0 |
| **---** showTable() : void  --- updateScore() : void  --- startButtonActionPerformed() : void |

* + **showTable() -** gets the data from the database which contains the performance of the student and shows it to the student
  + **startButtonActionPerformed() –** goes back to the start page
  + **updateScore() :** calculates the total score and updates it into the database

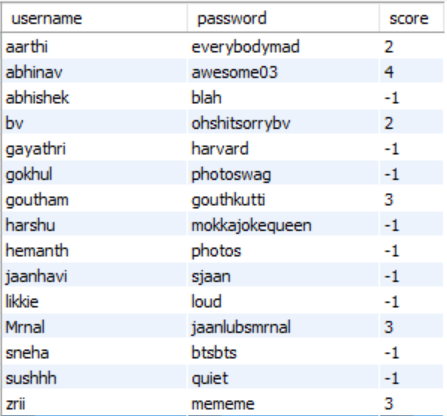
1. **ScoreRand.java**

|  |
| --- |
| **ScoreRand** |
| **---** score[] : static int[10]  --- rand[] : static int[10]  + quesNo : int |
| **+** reset() : void  + createRand() : void  + geTotalQuesNo() :int  + insertAllQuery() : void |

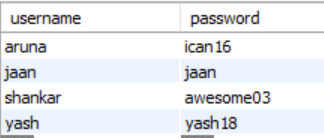
* + **reset() –** resets the score to 0 and the random numbers to 0 for new student
  + **createRand() –** creates a random list of numbers for the available questions and assigns it to the array
  + **getTotalQuesNo(): int –** returns the total number of questions available in databse
  + **insertAllQuery() –** inserts the random number and the score (initial) for the particular student into the database so that the changes can be made in score when the student enters correct answer and the software can get the question of random

**DATABASE CONTENTS**

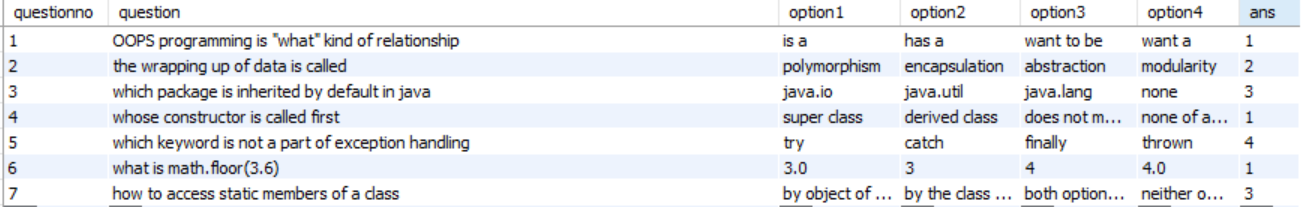
1. **studentlogin**
   * contains username and the passwords of the students along with the score they scored after performing the quiz. If quiz not attended , score is -1

****

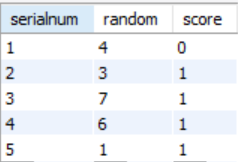
1. **proflogin**
   * contains username and the passwords of the professors

****

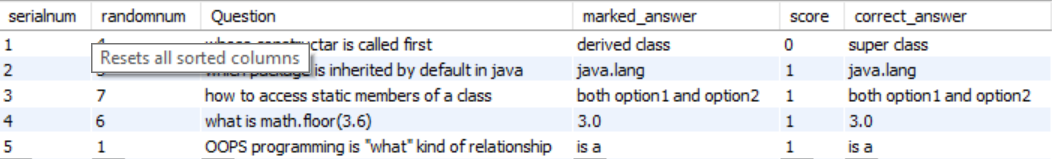
1. **mcq**
   * contains all the questions, the four options and the correct option number

****

1. **scorerand**
   * this table gets updated for each login
   * the random number (the question to be selected) and the score for that question is updated here

****

1. **viewtable**
   * this table contains the performance of the student and gets updated for each login. It contains the information about the question attended , the correct and for it and the score for that particular question

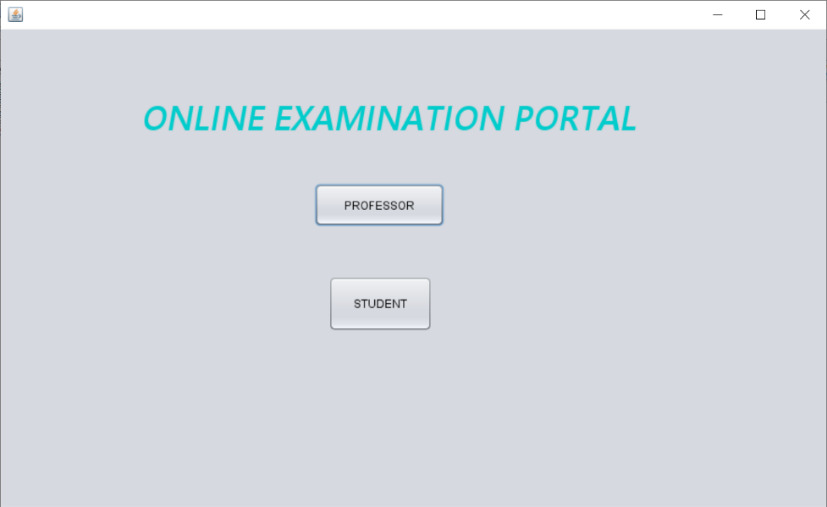
****

**OUTPUT SCREENSHOTS:**

1. **Start Page**

Two modules:

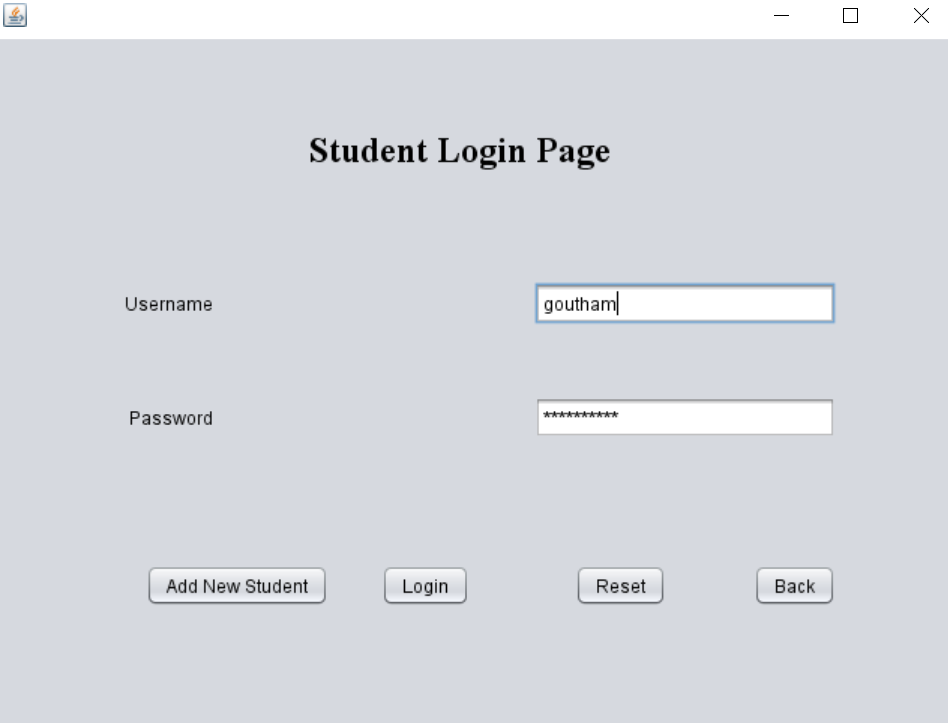
* + 1. Professor
    2. Student



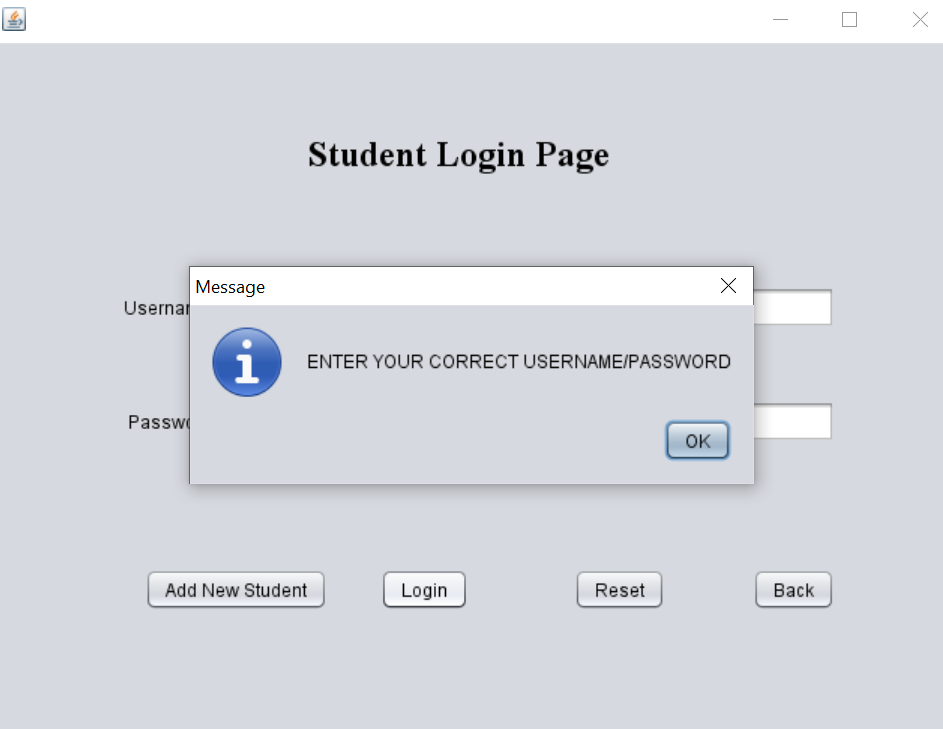
1. **Student Login Page**

The Student will be directed to the rule page for the quiz after entering the correct username and password

A new Entry can also be added

****

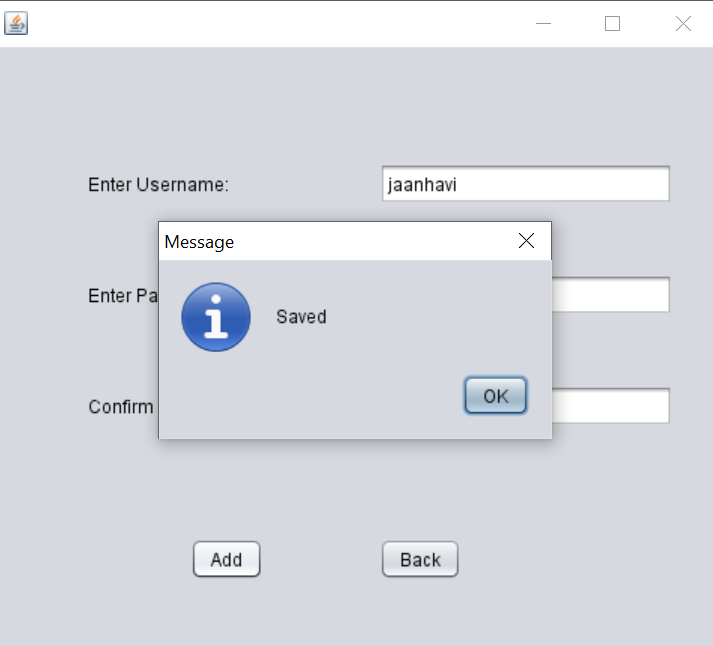
If the student enters the wrong username/password , the below message will be prompted

**s**

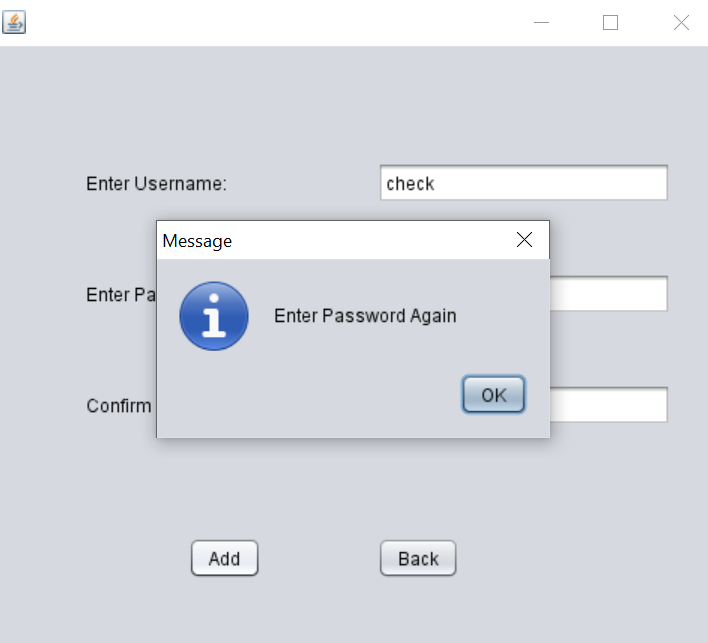
1. **Add new Student**

**A NEW entry can be added which can be then later used to sign in**

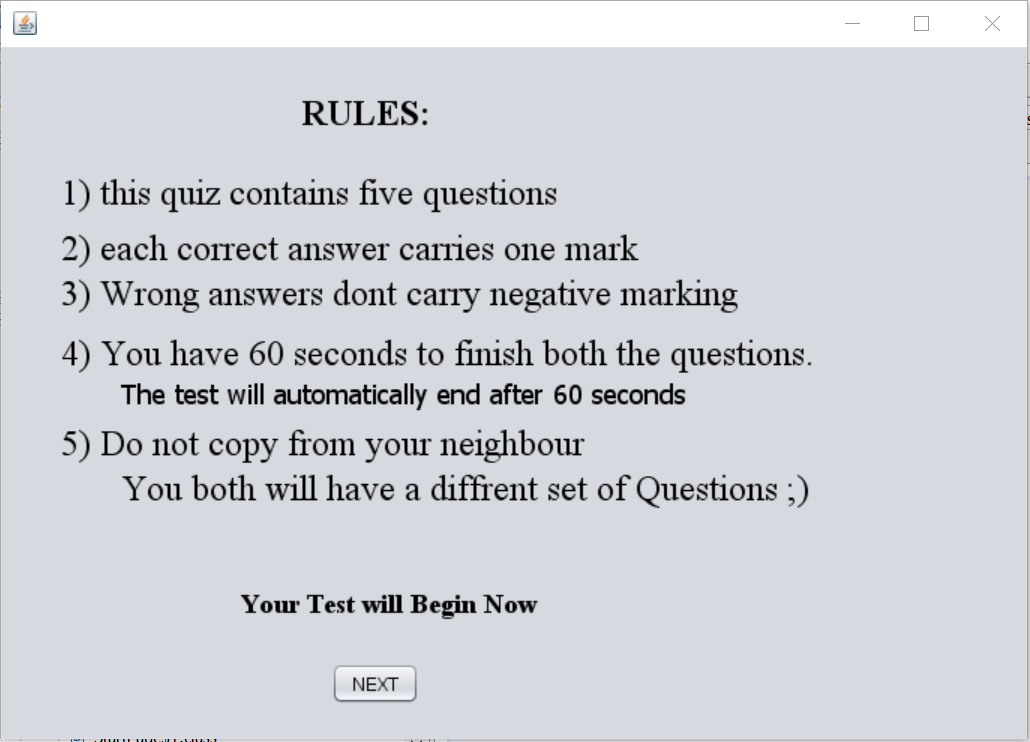
Only when the password matches with the confirm password, then entry will be saved

****

In case the passwords don’t match ,

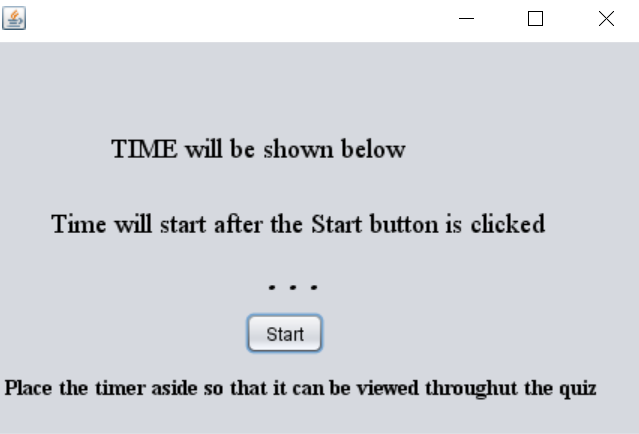


1. **Rule Page:**

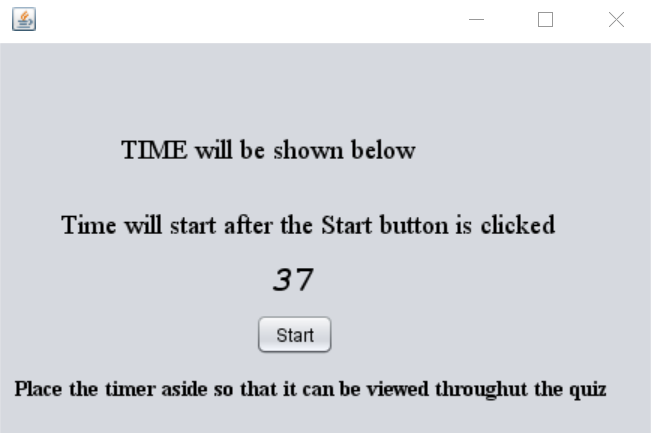


1. **Timer Page**

To be kept aside, as the time left will be shown in this page



When the time is running , the page will look like this

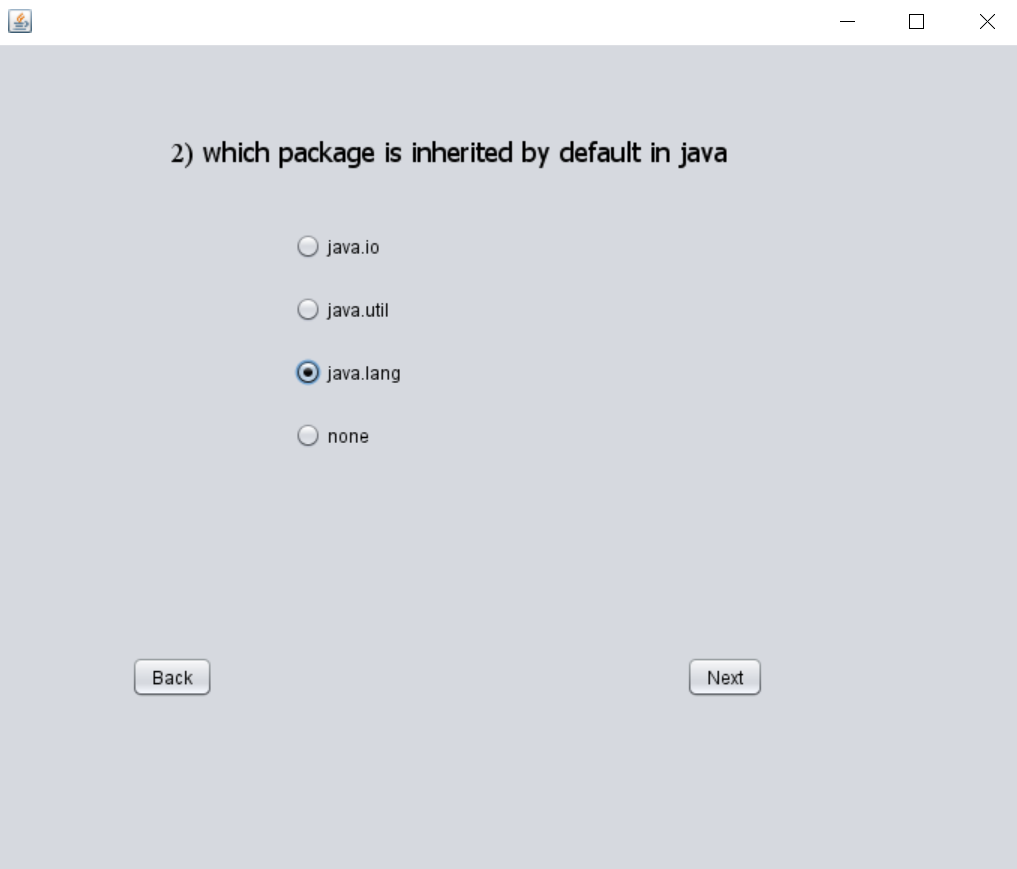


1. **Question Page:**

After start button is clicked , the student will be directed to the questions. There are five test questions to be attended.

If the back button is clicked , the previous question will be shown

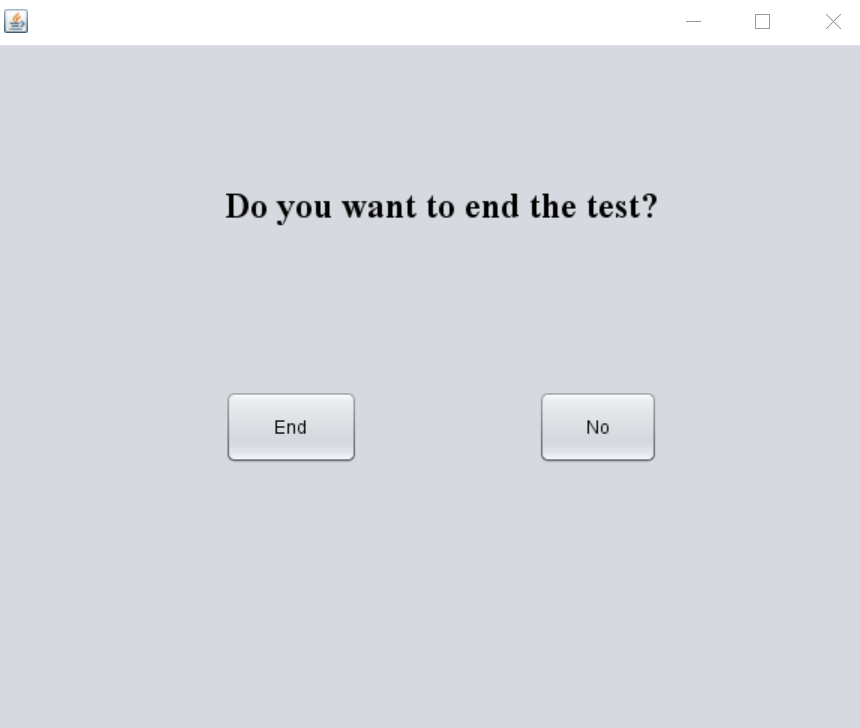
If the next button is clicked , the next question will be shown

****

1. **Prompt to check whether the student wants the end the test**

If End button is clicked , the student will be directed to the performance page

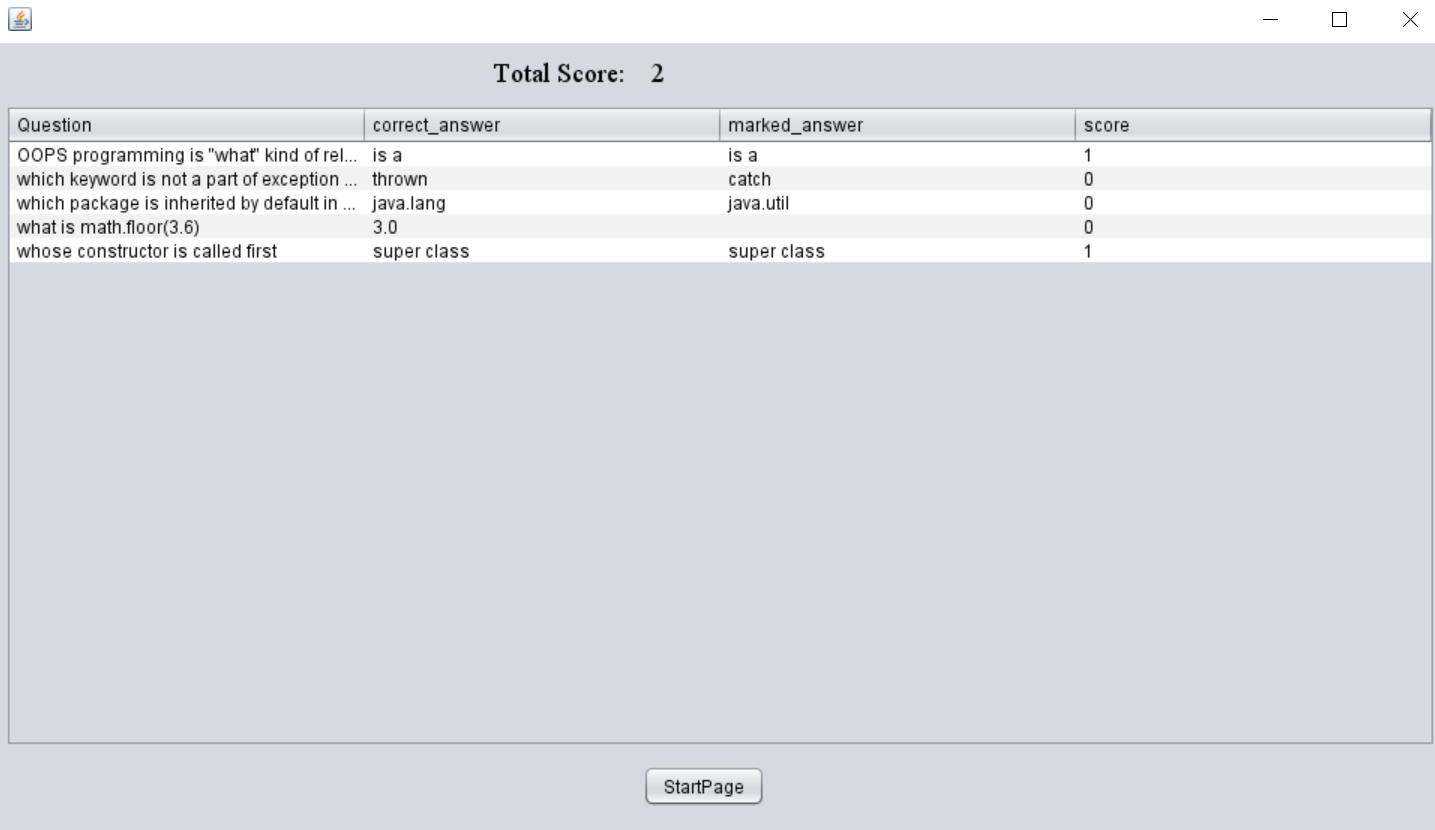
If No button is clicked , the student will be directed to the last question

****

1. **Performance Page:**

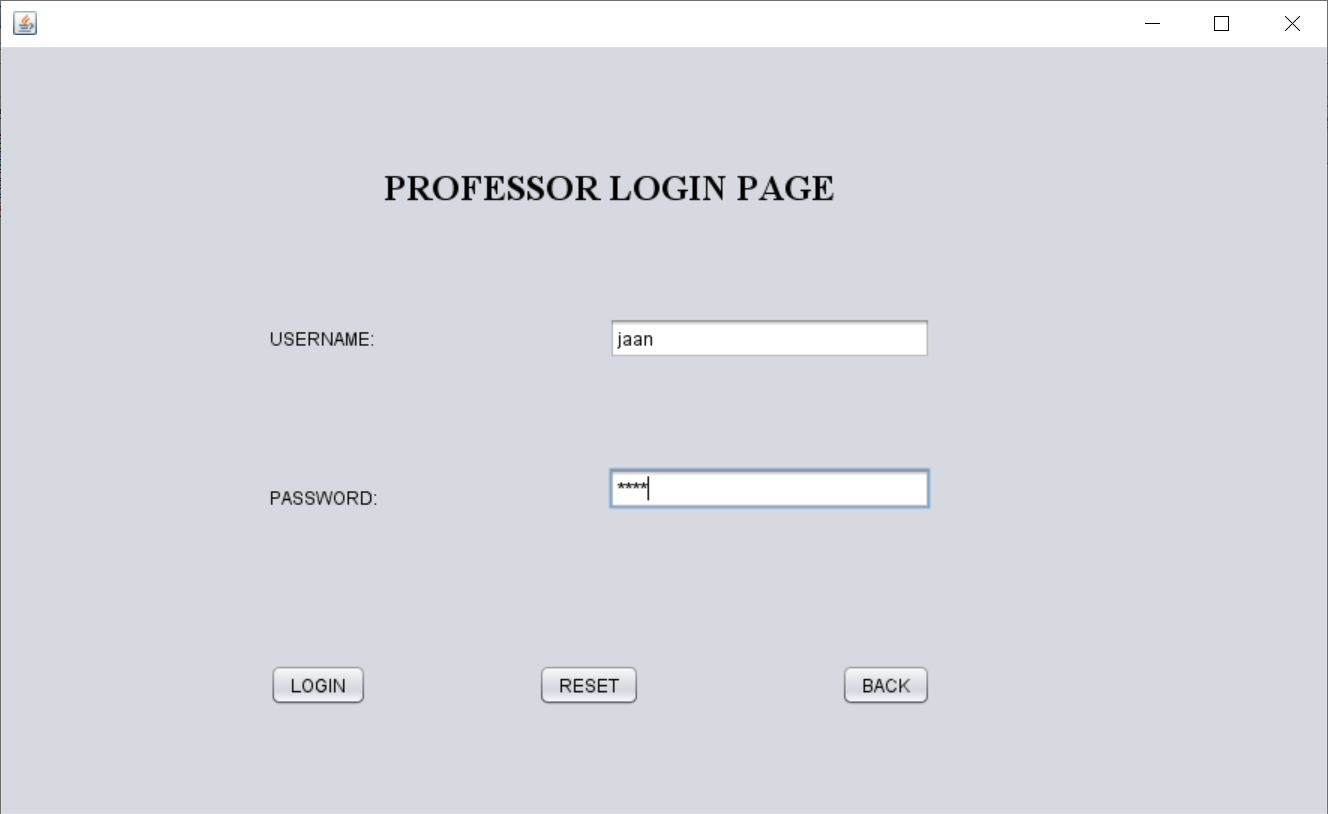
The student can view their performance history and their score

The student will be able to view the questions he attended , the choice he marked , the score for that particular question and the correct answer is displayed even when the correct answer is marked.

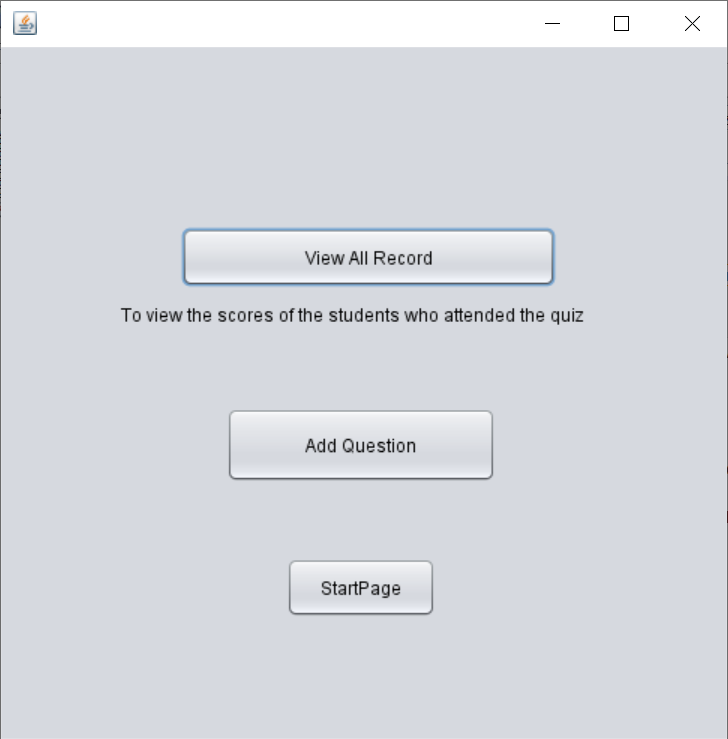
****

1. **Professor Login Page**

The professor will be directed to the next page after entering the correct username and password.

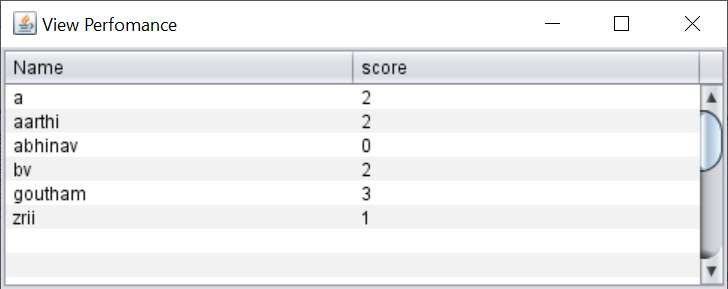


1. **Menu for professor to choose from**
   1. **View All Scores of students who attended the test**
   2. **Add a new Question into the database**
   3. **Go back to start page**



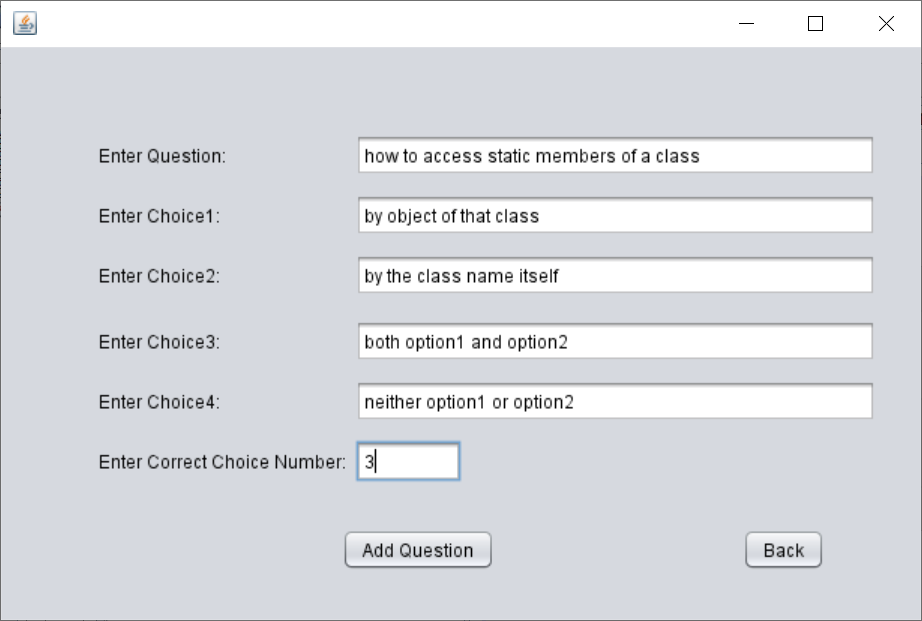
1. **View All Student Score:**

If the “View All Record” button is clicked , the name and scores of the students will be displayed

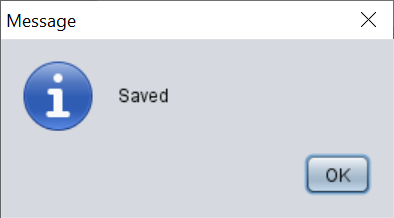


1. **Add Question**

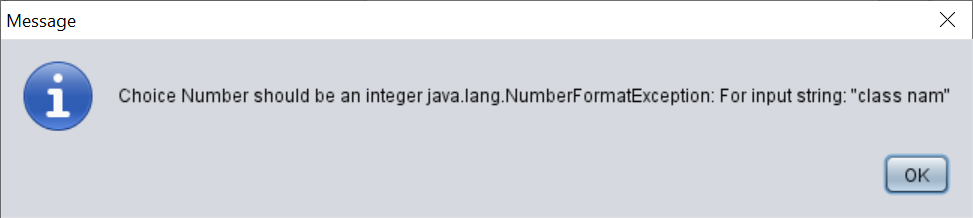
If the “Add Question” button is clicked, then the below screen will be shown. The Professor must enter the question and four choices to go with it. He/She must also enter the correct choice option number.



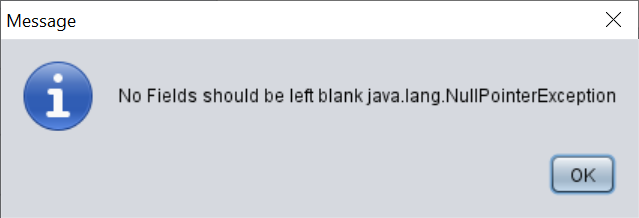
If the details are valid , then the below message will be on prompt.



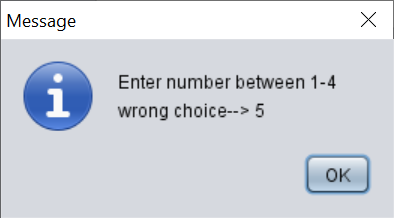
If in the “Correct Choice Number” Field – a string is placed , then **NumberFormatException** will be shown.



If any field is left blank, then **NullPointerException** will be shown



If in the “Correct Choice Number” Field , an incorrect option number is entered then , **WrongChoiceException** will be thrown



**LEARNING OUTCOME:**

* I learnt how to develop an application in java from scratch
* I learnt how to use java swing application in netbeans ide
* I learnt how to incorporate database with java application
* I learnt how to add , delete and use records from database using java syntax
* I learnt how to create user defined exceptions
* I learnt how to create a frame , add a scroll pane into the frame and add a table into the scroll pane without using the ide
* I learnt to how to generate random numbers by using collection algorithms ,interfaces and classes