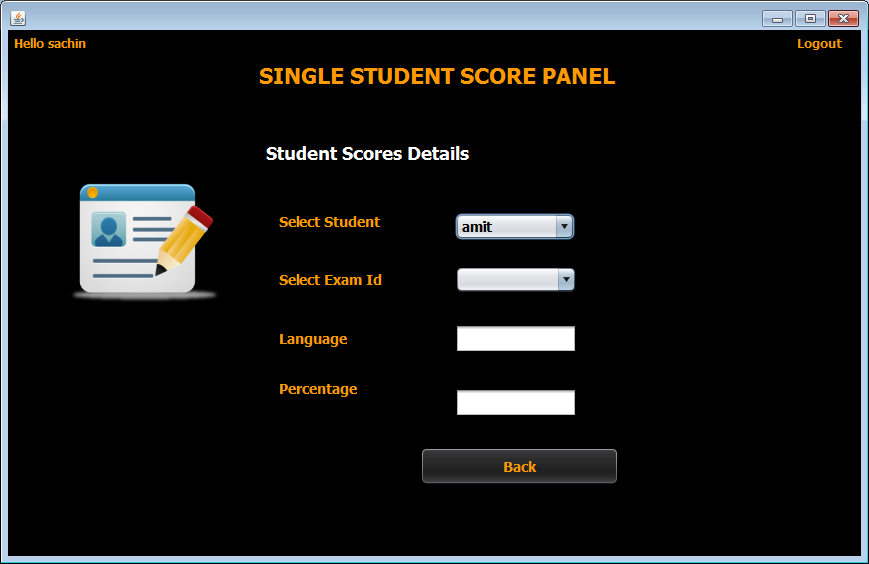
**Designing The ViewSingleScoreFrame**

****

**STEPS TO BE DONE IN** **ViewSingleScoreFrame**

In the **ViewSingleScoreFrame** we need to do following steps:

1. Display **username** on the top left

2. Allow the user to **logout**

3. Handle the constructor so that it loads id's of all the students who have appeared for test from the database. To do this we need to call the method **getAllStudentId( )** of the class **PerformanceDAO**.

4. Write code for displaying exam id as soon as the **user id** is selected.

5. Write code for displaying scores as soon as the **exam id** is selected.

**THE TABLES USED IN ViewSingleScoreFrame**

**1.PERFORMANCE**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| **USERID** | **Varchar2(10)** | **Contains Admin Provided User ID** |
| **EXAMID** | **Varchar2(10)** | **Contains id of the paper** |
| **RIGHT** | **Number(4)** | **Number of correct answers** |
| **WRONG** | **Number(4)** | **Number of wrong answers** |
| **UNATTEMPTED** | **Number(5)** | **Number of unattempted questions** |
| **PER** | **Number(5,2)** | **Percentage obtained** |
| **LANGUAGE** | **Varchar2(10)** | **Programming language name** |

**THE POJO CLASSES USED IN ViewSingleScoreFrame**

1. The **StudentScore** POJO

***public class StudentScore {***

***private String language;***

***private double per;***

***public String getLanguage() {***

***return language;***

***}***

***public void setLanguage(String language) {***

***this.language = language;***

***}***

***public double getPer() {***

***return per;***

***}***

***public void setPer(double per) {***

***this.per = per;***

***}***

***}***

**THE DAO CLASSES USED IN ViewSingleScoreFrame**

1. The **PerformanceDAO**

**HOW TO LOAD AND SHOW THE Student Id:**

To do this we need to create a method called **getAllStudentId( )** in the class **PerformanceDAO**.This method should fetch and return all the student ids from the table **PERFORMANCE.** The prototype of this method will be:

***public static ArrayList<String> getAllStudentId()throws SQLException***

Following is it's code:

***public static ArrayList<String> getAllStudentId()throws SQLException***

***{***

***String qry="select distinct userid from performance";***

***ArrayList<String> studentIdList=new ArrayList<>();***

***Connection conn=DBConnection.getConnection();***

***Statement st=conn.createStatement();***

***ResultSet rs=st.executeQuery(qry);***

***while(rs.next()){***

***String id=rs.getString(1);***

***studentIdList.add(id);***

***}***

***return studentIdList;***

***}***

**DISPLAYING STUDENT ID IN JCOMBOBOX:**

To display the student id in **JComboBox** we design a method called **loadStudentId()** in the **ViewSingleScoreFrame** which does the following:

**a. Calls the method getAllStudentId( ) of the class PerformanceDAO.**

**b. If no id is returned , it displays the message "No student has yet appeared for the exam"**

**c. Otherwise it iterates over the student id list and displays it in the JComboBox**

**d. It also handles SQLException which the method getAllStudentId( ) of the class PerformanceDAO can throw**.

Following is it's code:

***Write Code***

**HANDLING THE CONSTRUCTOR:**

From the body of constructor we would call the above method so as to show the student ids ass soon ass the frame opens

***public ViewSingleScoreFrame() {***

***initComponents();***

***this.setLocationRelativeTo(null);***

***lblUsername.setText("Hello "+UserProfile.getUSername());***

***loadStudentId();***

***}***

**HOW TO LOAD AND SHOW THE Exam Id:**

To do this we need to create a method called **getAllExamId( )** in the class **PerformanceDAO**.This method should accept a student id as argument and fetch and return all the exam ids for which this student has appeared from the table **PERFORMANCE.** The prototype of this method will be:

***public static ArrayList<String> getAllExamId(String studentId)throws SQLException***

Following is it's code:

**Write Code**

**DISPLAYING EXAM ID IN JCOMBOBOX:**

To display the exam id of the selected student in **JComboBox** we have to handle the **ItemEvent** for the **JComboBox** called **jcStudentId** . This is because as soon as we select an item from the **JComboBox** , java executes the **Item Event**.

So we need to create an ***event handler*** for this event which should do the following:

**a. Fetch the selected student id**

**b. Calls the method getAllExamId( ) of the class PerformanceDAO passing it the selected student id as argument**

**c. Iterate over the exam id list and display it in the jcExamId JComboBox**

**d. It also handles SQLException which the method getAllExamId( ) of the class PerformanceDAO can throw**.

Following is it's code:

***Write Code***

**HOW TO LOAD AND SHOW THE SCORE:**

To do this we need to create a method called **getScore( )** in the class **PerformanceDAO**.This method should accept a **student id** and **exam id** as argument and fetch and return it's **percentage** and **language** name as an object of the class **StudentScore** from the table **PERFORMANCE.** The prototype of this method will be:

***public static StudentScore getScore(String studentId,String examId)throws SQLException***

Following is the code:

***Write Code***

**DISPLAYING SCORE:**

To display the score details i.e. the percentage and language of the selected exam of selected student we have to handle the **Item Event** for the **JComboBox** called **jcExamId** . This is because as soon as we select an item from the **JComboBox** , java executes the **Item Event**.

So we need to create an ***event handler*** for this event which should do the following:

**a. Fetch the selected student id and exam id.**

**b. Call the method getScore( ) of the class PerformanceDAO passing it the selected student id and exam id as argument**

**c. Display the language name and it in the jcExamId JComboBox**

**d. It also handles SQLException which the method getScore( ) of the class PerformanceDAO can throw**.

Following is it's code:

***Write Code***

**WRITING THE CODE FOR BUTTON Button.png IN EditQuestionsFrame**

***private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {***

***ViewScoresFrame viewScoresFrame =new ViewScoresFrame();***

***viewScoresFrame.setVisible(true);***

***this.dispose();***

***}***