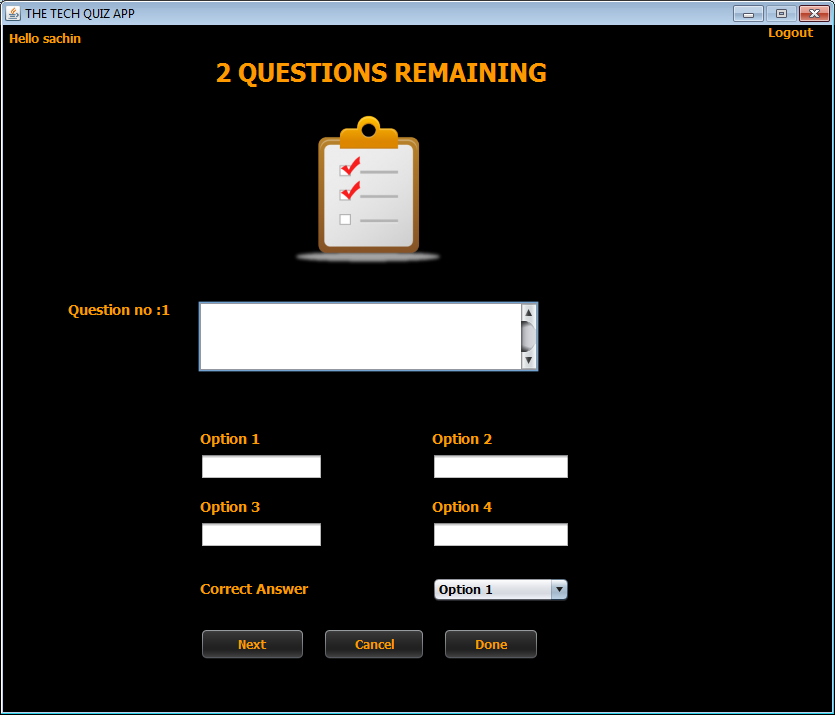
**Designing The SetQuestionsFrame**

****

**STEPS TO BE DONE IN** **SetQuestionsFrame**

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

1. Handle the constructor for proper initialization of class members

2. Allow the user to **logout**

3. Write code for the "**Next**" Button . When this button is clicked it should:

**a. Validate the inputs. If validation fails it should print the Error Message and return**

**b. It should fill all the details about the question in Question POJO and add the Question POJO object in the qstore.**

**c. Update the remaining questions label**

**d. Stop the user from adding more questions**

4. Write code for the "**Done**" Button . When this button is clicked it should:

**a. Verify if all the questions have been set .If not, then it should ask the user to set all the questions and return.**

**b. If all questions have been set then it should make an entry in the EXAM table and add all the questions to the QUESTIONS table.**

**c. Dispose the current frame and open the AdminOptionsFrame**

5. Write code for the "**Cancel**" Button . When this button is clicked it should:

a**. Dispose the current frame and open the SetPaperFrame**

**THE TABLES USED IN SetQuestionsFrame**

**1. EXAM**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| **EXAMID** | **Varchar2(10)** | **Contains id of the paper** |
| **LANGUAGE** | **Varchar2(10)** | **Contains the programming language name** |
| **TOTAL\_QUESTION** | **Number(4)** | **Contains total number of question present** |

**2.QUESTIONS**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| **EXAMID** | **Varchar2(10)** | **Contains id of the paper** |
| **QNO** | **Number(5)** | **Represents the Question No.** |
| **QUESTION** | **Varchar2(100)** | **Represents Question Description** |
| **ANSWER1** | **Varchar2(30)** | **Option 1** |
| **ANSWER2** | **Varchar2(30)** | **Option 2** |
| **ANSWER3** | **Varchar2(30)** | **Option 3** |
| **ANSWER4** | **Varchar2(30)** | **Option 4** |
| **CORRECT\_ANSWER** | **Varchar2(10)** | **The actual correct answer** |
| **LANGUAGE** | **Varchar2(10)** | **Programming language name** |

**THE POJO CLASSES USED IN SetQuestionsFrame**

1. The **Exam** POJO

2. The **QuestionPojo** POJO

3. The **QuestionStore** POJO

***public class QuestionPojo {***

***private String examId;***

***private int qno;***

***private String language;***

***private String answer1,answer2,answer3,answer4;***

***private String correctAnswer;***

***private String question;***

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

***return language;***

***}***

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

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

***}***

***@Override***

***public String toString() {***

***return "Question{" + "examId=" + examId + ", qno=" + qno + ", question=" + question + ", answer1=" + answer1 + ", answer2=" + answer2 + ", answer3=" + answer3 + ", answer4=" + answer4 + ", correctAnswer=" + correctAnswer + '}';***

***}***

***public String getExamId() {***

***return examId;***

***}***

***public void setExamId(String examId) {***

***this.examId = examId;***

***}***

***public int getQno() {***

***return qno;***

***}***

***public void setQno(int qno) {***

***this.qno = qno;***

***}***

***public String getQuestion() {***

***return question;***

***}***

***public void setQuestion(String question) {***

***this.question = question;***

***}***

***public String getAnswer1() {***

***return answer1;***

***}***

***public void setAnswer1(String answer1) {***

***this.answer1 = answer1;***

***}***

***public String getAnswer2() {***

***return answer2;***

***}***

***public void setAnswer2(String answer2) {***

***this.answer2 = answer2;***

***}***

***public String getAnswer3() {***

***return answer3;***

***}***

***public void setAnswer3(String answer3) {***

***this.answer3 = answer3;***

***}***

***public String getAnswer4() {***

***return answer4;***

***}***

***public void setAnswer4(String answer4) {***

***this.answer4 = answer4;***

***}***

***public String getCorrectAnswer() {***

***return correctAnswer;***

***}***

***public void setCorrectAnswer(String correctAnswer) {***

***this.correctAnswer = correctAnswer;***

***}***

***public QuestionPojo(String examId, String language,int qno, String question, String answer1, String answer2, String answer3, String answer4, String correctAnswer) {***

***this.examId = examId;***

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

***this.qno = qno;***

***this.question = question;***

***this.answer1 = answer1;***

***this.answer2 = answer2;***

***this.answer3 = answer3;***

***this.answer4 = answer4;***

***this.correctAnswer = correctAnswer;***

***}***

***@Override***

***public boolean equals(Object obj) {***

***Question other = (Question) obj;***

***if (this.examId.equals(other.examId)==false) {***

***return false;***

***}***

***if (this.qno != other.qno) {***

***return false;***

***}***

***if (this.question.equals(other.question)==false) {***

***return false;***

***}***

***if (this.answer1.equals(other.answer1)==false) {***

***return false;***

***}***

***if (this.answer2.equals(other.answer2)==false) {***

***return false;***

***}***

***if (this.answer3.equals(other.answer3)==false) {***

***return false;***

***}***

***if (this.answer4.equals(other.answer4)==false) {***

***return false;***

***}***

***if (this.correctAnswer.equals(other.correctAnswer)==false) {***

***return false;***

***}***

***return true;***

***}***

***}***

***public class QuestionStore {***

***ArrayList <QuestionPojo> questionList;***

***public QuestionStore()***

***{***

***questionList=new ArrayList<>();***

***}***

***public void addQuestion(QuestionPojo q){***

***questionList.add(q);***

***}***

***public QuestionPojo getQuestion(int pos){***

***return questionList.get(pos);***

***}***

***public void removeQuestion(int pos){***

***questionList.remove(pos);***

***}***

***public void setQuestionAt(int pos,QuestionPojo q){***

***questionList.add(pos, q);***

***}***

***public ArrayList<QuestionPojo> getAllQuestions(){***

***return questionList;***

***}***

***public int getCount(){***

***return questionList.size();***

***}***

***}***

**THE DAO CLASSES USED IN SetQuestionsFrame**

1. The **ExamDAO**

2. The **QuestionDAO**

We will call the method **addExam( )** of **ExamDAO** to add the **Exam** object in the **EXAM** table.

***public static boolean addExam(ExamPojo newExam)throws SQLException{***

***String qry="Insert into Exam values(?,?,?)";***

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

***PreparedStatement ps=conn.prepareStatement(qry);***

***ps.setString(1, newExam.getExamId());***

***ps.setString(2, newExam.getLanguage());***

***ps.setInt(3,newExam.getTotalQuestions());***

***int ans=ps.executeUpdate();***

***return ans==1;***

***}***

**We will call the addQuestions( ) method of QuestionDAO to add Question objects in the QUESTIONS table**

***public static void addQuestions(QuestionStore qstore)throws SQLException{***

***String qry="Insert into questions values(?,?,?,?,?,?,?,?,?)";***

***ArrayList <Question> questionList=qstore.getAllQuestions();***

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

***PreparedStatement ps=conn.prepareStatement(qry);***

***for(Question obj:questionList){***

***ps.setString(1, obj.getExamId());***

***ps.setInt(2, obj.getQno());***

***ps.setString(3,obj.getQuestion());***

***ps.setString(4,obj.getAnswer1());***

***ps.setString(5,obj.getAnswer2());***

***ps.setString(6,obj.getAnswer3());***

***ps.setString(7,obj.getAnswer4());***

***ps.setString(8,obj.getCorrectAnswer());***

***ps.setString(9, obj.getLanguage());***

***ps.executeUpdate();***

***}***

***}***

**Handling the constructor:**

The constructor should do the following:

a. Display the username

b. Create a **QuestionStore** object

c. Create a **HashMap** object

d. Set the number of questions in a label

e. Save the **Exam** object passed by **SetPaperFrame** for further use

***public class SetQuestionsFrame extends javax.swing.JFrame {***

***private Exam newExam;***

***private QuestionStore qstore;***

***private HashMap <String,String> options;***

***private int qno;***

***private String question,option1,option2,option3,option4,correctOption;***

***public SetQuestionsFrame() {***

***initComponents();***

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

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

***qstore=new QuestionStore();***

***options=new HashMap<>();***

***options.put("Option 1","Answer1");***

***options.put("Option 2","Answer2");***

***options.put("Option 3","Answer3");***

***options.put("Option 4","Answer4");***

***qno=1;***

***lblQno.setText(lblQno.getText()+qno);***

***}***

***public SetQuestionsFrame(Exam newExam) {***

***this();***

***lblTitle.setText(newExam.getTotalQuestions()+ " QUESTIONS REMAINING");***

***this.newExam=newExam;***

***}***

**Allowing The User To Logout**

Same as AdminOptionsFrame

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

***public boolean validateInput(){***

***question=txtQuestion.getText().trim();***

***option1=txtOption1.getText().trim();***

***option2=txtOption2.getText().trim();***

***option3=txtOption3.getText().trim();***

***option4=txtOption4.getText().trim();***

***correctOption=jcCorrectOption.getSelectedItem().toString();***

***if(question.isEmpty()||option1.isEmpty()||option2.isEmpty()||option3.isEmpty()||option4.isEmpty()||correctOption.isEmpty())***

***return false;***

***else***

***return true;***

***}***

***public void clearAll(){***

***txtQuestion.setText("");***

***txtOption1.setText("");***

***txtOption2.setText("");***

***txtOption3.setText("");***

***txtOption4.setText("");***

***jcCorrectOption.setSelectedIndex(0);***

***}***

***public void disableAll(){***

***txtQuestion.setEnabled(false);***

***txtOption1.setEnabled(false);***

***txtOption2.setEnabled(false);***

***txtOption3.setEnabled(false);***

***txtOption4.setEnabled(false);***

***jcCorrectOption.setEnabled(false);***

***btnNext.setEnabled(false);***

***}***

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

***if(validateInput()==false)***

***{***

***JOptionPane.showMessageDialog(null, "Please fill all the input fields","Cannot Add Question!",JOptionPane.ERROR\_MESSAGE);***

***return;***

***}***

***String optionName=options.get(correctOption);***

***QuestionPojo obj;***

***obj=new QuestionPojo(newExam.getExamId(),newExam.getLanguage(),***

***qno,question,option1,option2,option3,option4,optionName);***

***// System.out.println(obj);***

***qstore.addQuestion(obj);***

***clearAll();***

***lblTitle.setText((newExam.getTotalQuestions()-qno)+ " QUESTIONS REMAINING");***

***qno++;***

***if(qno>newExam.getTotalQuestions())***

***{***

***disableAll();***

***JOptionPane.showMessageDialog(null, "Your question set has been successfully created.\nPress the DONE button to add it to the database ","Exam Added!",JOptionPane.INFORMATION\_MESSAGE);***

***}***

***else***

***lblQno.setText("Question no:"+qno);***

***}***

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

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

***if(qno<=newExam.getTotalQuestions()){***

***int remainingQuestions=newExam.getTotalQuestions()-qno+1;***

***JOptionPane.showMessageDialog(null, "You still have "+remainingQuestions+" left","Cannot Add Exam!",JOptionPane.ERROR\_MESSAGE);***

***return;***

***}***

***try{***

***ExamDAO.addExam(newExam);***

***QuestionDAO.addQuestions(qstore);***

***JOptionPane.showMessageDialog(null, "Your question set has been successfully added to the database ","Exam Added!",JOptionPane.INFORMATION\_MESSAGE);***

***AdminOptionsFrame adminFrame=new AdminOptionsFrame();***

***adminFrame.setVisible(true);***

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

***}***

***catch(SQLException ex){***

***JOptionPane.showMessageDialog(null, "Error while connecting to DB!","Exception!",JOptionPane.ERROR\_MESSAGE);***

***ex.printStackTrace();***

***}***

***}***

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

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

***SetPaperFrame paperFrame=new SetPaperFrame();***

***paperFrame.setVisible(true);***

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

***}***