TECHNICAL SPECIFICATION DOCUMENT

Fundamental Java Project – Quiz Manager

By,

Abhigna Doddiganahalli Chandrashekar (MSC-Data Science)

(abhigna.dc@gmail.com)

Krishna Teja KANCHERLA (MSC-Data Science)

(kancherlakrishnateja14@gmail.com)

Table of Contents

1. Introduction

1.1. Project Background

This project is a Quiz management desktop application - preparation and execution. This document is prepared by **Krishna Teja KANCHERLA and Abhigna Doddiganahalli Chandrashekar** for their Fundamental Java Project in the Computer Science Master's Program at L'École Pour l'Informatique et les Techniques Avancées (**EPITA**).

1.2. Project Overview

In this project, there are two modules, Admin and candidate. As a student, a user can take a graded quiz. As an admin user can perform CRUD operations on quiz and questions contained in them.

2. Requirement Specification

2.1. Admin

2.1.1. Perform CRUD operations on quiz and questions
2.1.2. Categorize quiz based on topics
2.1.3. Export quiz as PDF
2.1.4. Search Quiz based on topic
2.1.5. GUI

2.2. Candidate

2.2.1. Choose quiz to participate based on topic and difficulty
2.2.2. Get grade at the end of quiz
2.2.3. Answer each question
2.2.4. GUI

3. Scope definition

3.1. Topics within scope of project

- Create, Read, Update, Delete Quiz with MCQ questions
- Create, Read, Update, Delete Questions in quiz
- Admin and candidate module
- Auto Grading
- Export Quiz in PDF format
- GUI
- Segregation of Quiz based on topics
- Difficulty level Indicator
- Candidate registration
- Executable .jar file

All functionality not explicitly defined in section 3.1 is assumed to be out of scope

4. Acronyms and abbreviations

Acronym	Meaning
CRUD	(Create, Read, Update, Delete) Functions that are implemented in relational database applications
GUI	(Graphical User Interface) Interface presented to the user of an application
DAO	(Data Access Object) Service class for the application to communicate with the H2 database
UML	(Unified Modelling Language) a standard way to visualize the design of a system
MCQ	(Multiple Choice Question) Question with enumerated possible answers, implemented with radial buttons in our application

5. Dependencies

5.1. Setup

5.1.1. Install h2 JDBC driver

5.1.2. Install Java 8

5.1.3. Create tables in database (commands.sql)

5.1.4. Add questions

5.1.5. Configure itextPdf jar

5.1.6. Configure Javadoc.jar in build path

5.1.7. Configure Junit5 for testcases

5.2. Software and hardware requirements

Software Requirements

Front End	JFrames
Backend	Java 8
Database	H2 Console
Version Control	GIT
Testing	JUnits

Hardware Requirements

Hardware	**Requirement**
Operating System	Windows, Linux, Mac
RAM	Min. 256 MB
Disk space	Min. 500 MB
Processor	64 Bit, four core, 2.5 GHz min per core

6. System Specifications

6.1. Functional Requirements

BR #	Implementations

	Main Class – used to Launch the Application
	`/**`
	` *`
	` * @author Krishna, Abhigna`
	` * @Main class for launching application`
	` */`
	`public class Main extends JFrame {`
	`  private static final long serialVersionUID = -2728957700299762075L;`
	`  private StatusBar statusBar;`
	
	`  public Main() {`
	`  JFrame frame = new JFrame("Quiz Manager");`
	`  JButton admn, candidate;`
	`  admn = new JButton("Admin");`
	`  admn.addActionListener(new ActionListener() {`
	`  public void actionPerformed(ActionEvent e) {`
	`  new Admn(); // To Admin Portal`
	`  setVisible(false);`
	`  }`
	`  });`
	`  candidate = new JButton("Candidate");`
	`  candidate.addActionListener(new ActionListener() {`
	`  public void actionPerformed(ActionEvent e) {`
	`  new Candidate(); // To Candidate Portal`
	`  setVisible(false);`
	`  }`
	`  });`

	Admin Login – Based on Authentication
	`JButton login = new JButton("LogIn");`
	`login.addActionListener(new ActionListener() {`
	
	`  @Override`
	`  public void actionPerformed(ActionEvent e) {`
	`  @SuppressWarnings("deprecation")`
	`  boolean isAuth = authenticate(unameFld.getText(), pwdFld.getText())`
	
	`  if(isAuth)`
	`  {`
	`  new AdmnLogin(); // Admin Login for CRUD Operations`
	`  setVisible(false);`
	`  }`
	`  else {`
	`  JOptionPane.showMessageDialog(null, CHCK_MSG);`
	`  }`
	`  }`
	`});`
	`login.setBounds(127, 181, 89, 23);`
	`panel.add(login);`

3	**Adding a Quiz – creating a Question based on the input provided**

```
JButton addBtn = new JButton("ADD");  // To add a Question
addBtn.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            Answer ans = new Answer(fldCrctAns.getText());
            String string = comboBox.getSelectedItem().toString();
            String[] quiz = string.split(" - ");
            id = Integer.parseInt(quiz[0].toString());
            ans.setQuestion(new Question(id, 0, quesFld.getText(), topicFld.getText()
                    Integer.parseInt(diffFld.getText())));
            ans.setMultChce(new MultChoice(fldA.getText(), fldB.getText(), fldC.getTe
            System.out.println("Answers:" + ans);
            boolean isSucc = dao.createQues(ans);
            if (isSucc) {
                JOptionPane.showMessageDialog(null, ADD_MSG);
            }
        } catch (CreateFailedException e1) {
            e1.printStackTrace();
        }
    }
});
```

4 **Admin can export the Quiz with Questions and Answers**

```
public boolean exportQuiz() throws FileNotFoundException, DocumentException {
    boolean isExpSucc = false;
    try (Connection connection = getConnection();
            PreparedStatement pstmt = connection.prepareStatement(EXPORT_QUERY)) {
        ResultSet rs = pstmt.executeQuery();

        Document docRpt = new Document();
        PdfWriter.getInstance(docRpt, new FileOutputStream("Quiz.pdf"));
        docRpt.open();
        PdfPTable col = new PdfPTable(5);
        col.addCell("ID");
        col.addCell("Question");
        col.addCell("Topic");
        col.addCell("Difficulty");
        col.addCell("Answer");

        PdfPCell tblCell;
        while (rs.next()) {

            String qid = rs.getString("QID");
            tblCell = new PdfPCell(new Phrase(qid));
            col.addCell(tblCell);
            String content = rs.getString("CONTENT");
            tblCell = new PdfPCell(new Phrase(content));
            col.addCell(tblCell);
            String topics = rs.getString("TOPICS");
            tblCell = new PdfPCell(new Phrase(topics));
            col.addCell(tblCell);
            String diff = rs.getString("DIFFICULTY");
            tblCell = new PdfPCell(new Phrase(diff));
            col.addCell(tblCell);
            String ans = rs.getString("ANSWER");
            tblCell = new PdfPCell(new Phrase(ans));
            col.addCell(tblCell);
```

5 **Candidate can have the quiz evaluated and displayed with marks**

```
    nxtBtn.setBounds(180, 435, 89, 23);
    mainPane.add(nxtBtn);
    JButton endBtn = new JButton("End"); // Ends the Quiz and generates marks.
    endBtn.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            JOptionPane.showMessageDialog(null, "The Quiz is Completed Successfully \n Your Score is
            new Main();
            setVisible(false);
        }
    });
    endBtn.setBounds(180, 495, 89, 23);
    mainPane.add(endBtn);
```

6 **Candidate can search Questions based on Topic and Difficulty**

```
mainPane.add(titleLbl);
try {
    Connection conn = getConnection();

    PreparedStatement pstmt = conn.prepareStatement(RETQUES_QUERY);
    pstmt.setString(1, topic);
    pstmt.setInt(2, diff);
     ResultSet rs = pstmt.executeQuery();
     while(rs.next())
        {

            qid[i] = rs.getInt("QID");
            ques[i] = rs.getString("CONTENT");
            chcA[i] = rs.getString("CHOICEA");
            chcB[i] = rs.getString("CHOICEB");
            chcC[i] = rs.getString("CHOICEC");
            chcD[i] = rs.getString("CHOICED");
            crctAns[i] = rs.getString("ANSWER");
            i++;
        }
```

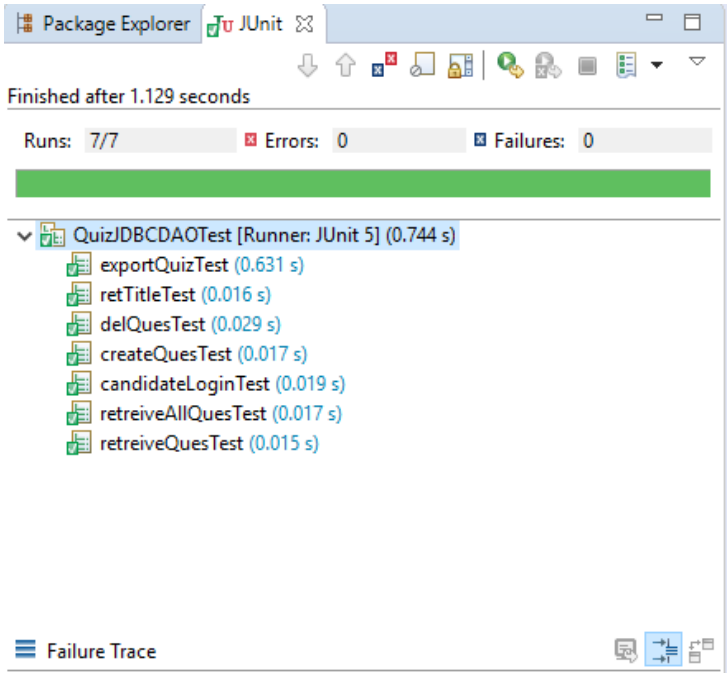
6.2. Non-Functional Requirements

- The program shall be able to run on GUI using JFrame
- The data is stored in h2 databases.

7. Testing

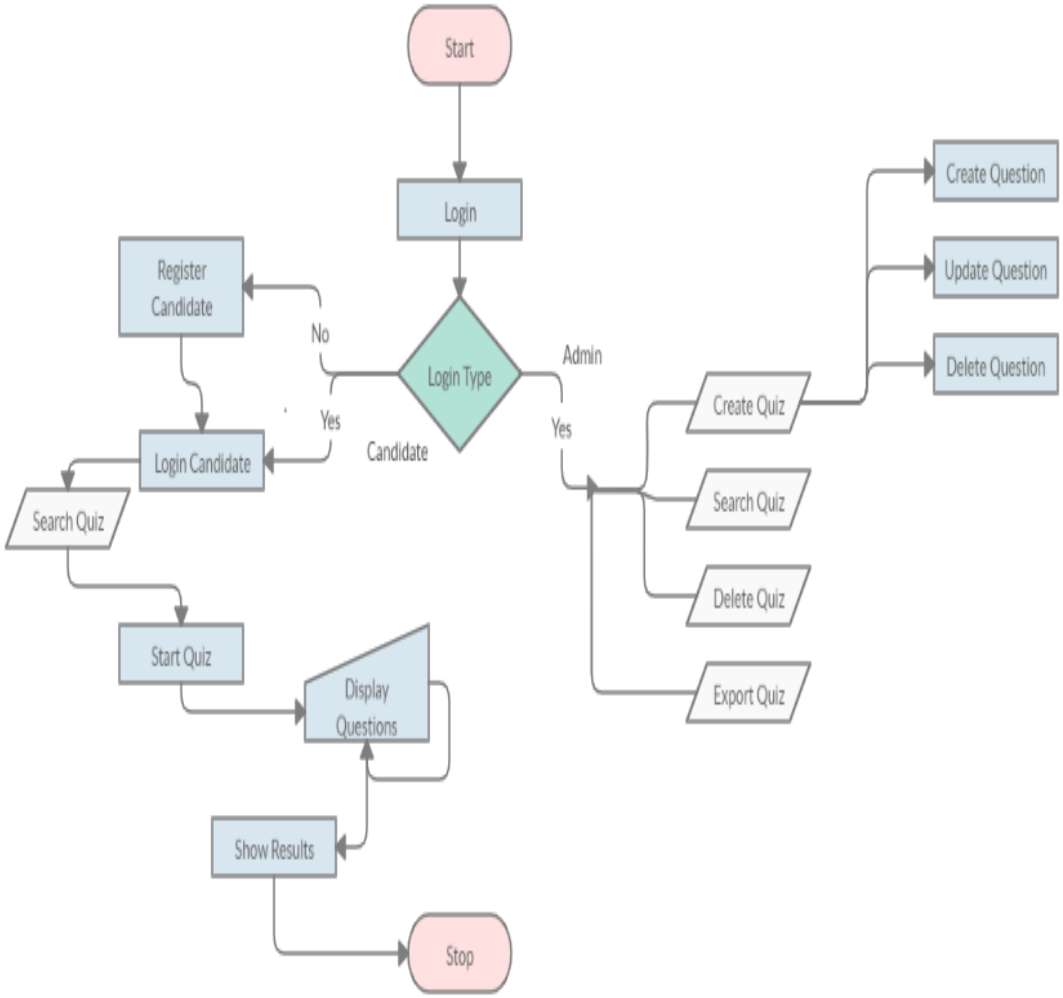
Test Case	Expected result	Status
Login Module – User Should give Credentials and Press Enter	Candidate / Admin should login	PASS
Create Quiz – Admin selects the create quiz and gives inputs	Questions should be created.	PASS
Search Quiz - Admin selects the Search quiz and gives inputs	Questions Searched	PASS
Delete Quiz - Admin selects to Delete based on Topic	Questions Deleted	PASS
Export Quiz - Admin Clicks Export to PDF the Quiz	Exported tp PDF	PASS
Candidate Quiz - User Attempts Quiz to get Grades	Quiz graded Successfully	PASS

Junit Results



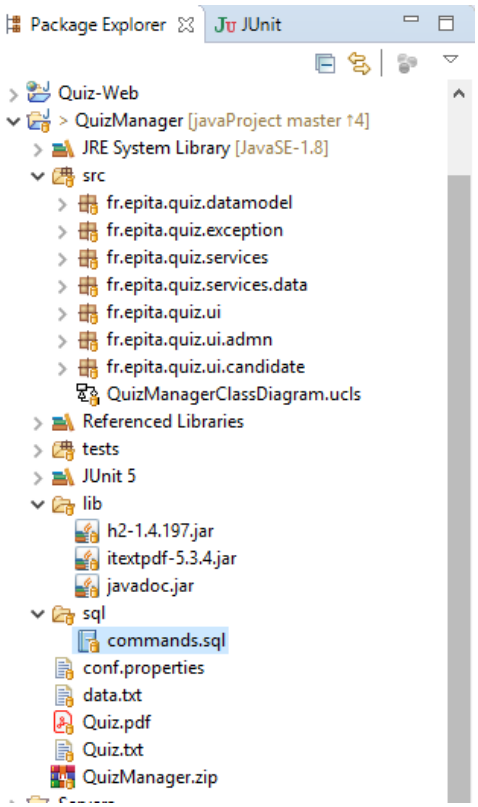
8. Appendix

8.1. Flowchart



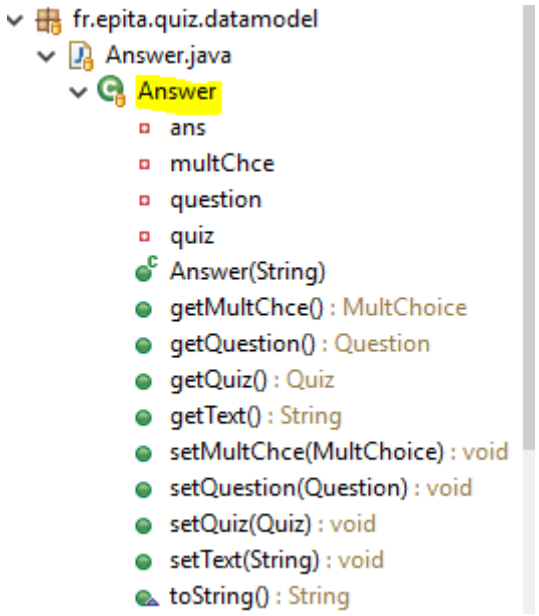
8.2. Project Structure

Project Structure with Folders, Data models and other Services are given below.

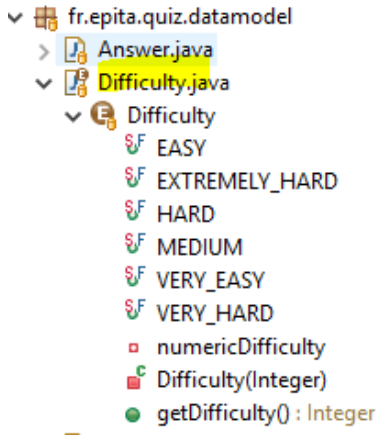


- The datamodels are the replica of tables involved in our application

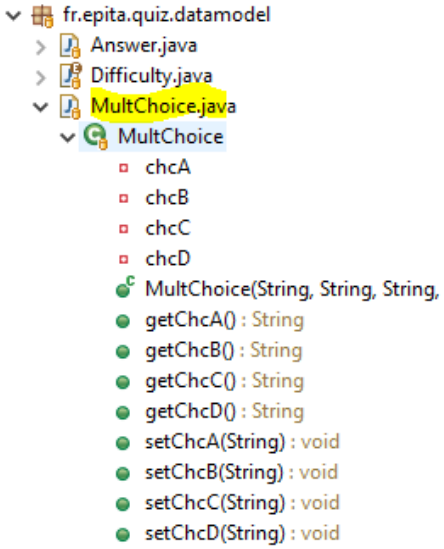
Answer



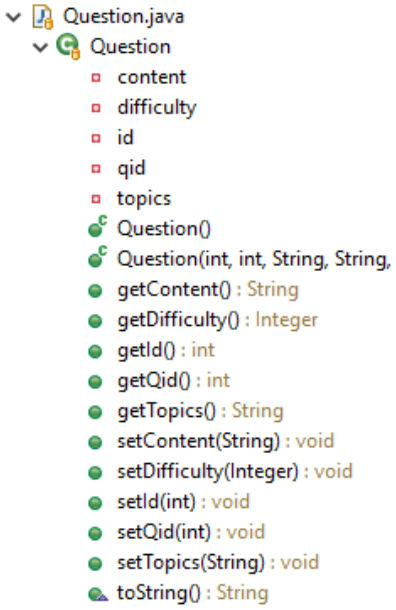
Difficulty Class



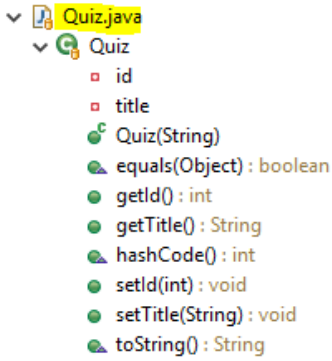
Multiple Choice Class



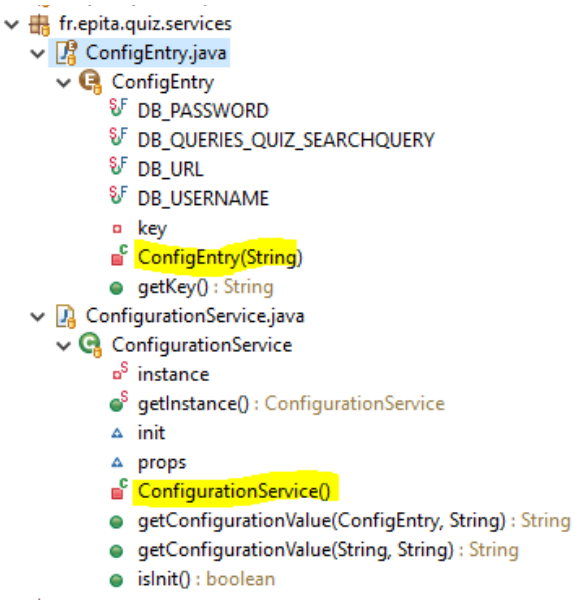
Question Class



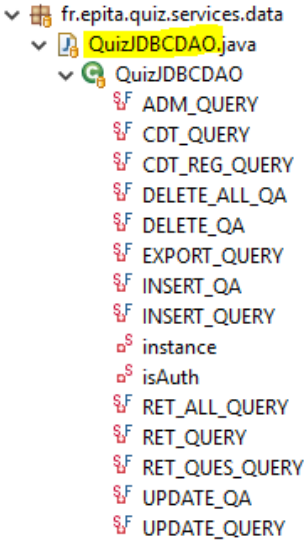
Quiz Data Model Class

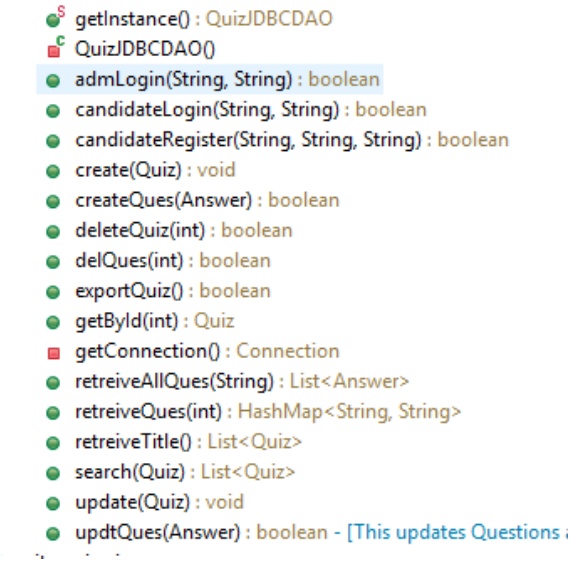


- **Services contain the database configuration and database access layers which has the core CRUD operation code and business logic**

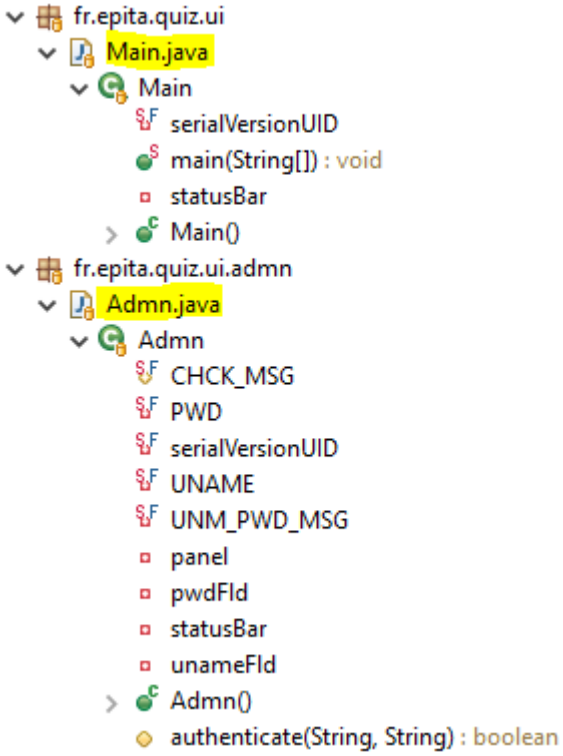


QuizMgrJDBCDAO Class : CRUD operation performed for the admin usage , on both questions and answers





- UI folder as the name says contains the class which is to be run to invoke the application. (It has the main method)



1.1. UML Diagram