Technical Guide

QuizTech

By:
G Sai Karthikeya
Sravan Nallala

1. Introduction

1.1. Project Background

This project is a console application for managing digital quiz - preparation and execution. This document is prepared by G Sai Karthikeya and Sravan Nallala for 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, user can login as an admin to add, update, delete and search questions, take a test and get score at the end of the quiz.

1.3. Project Scope

In Scope :

1. Automatically displays quiz using multiple-choice questions
2. Auto-grading for multiple choice questions
3. Creating data access object with CRUD methods

Out of Scope :

4. Incorporating open questions, Associative questions into the GUI
5. Exporting quizzes to PDF
6. Using an algorithm to arrange quizzes
7. GUI
8. Creating different user accounts other than Student
9. Creating user accounts with logging in
10. Providing real questions for the Quiz Manager application

1.4. Project Dependencies

In order to run the program, the following steps are required:

- Install h2 JDBC
- Install Java 8
- Create tables in database
- Add questions

1.5. Acronyms

Acronyms	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 Modeling 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
OQ	(Open question) Question with a free-text field response and no predefined structured answer
JDBC	(Java Database Connectivity) application programming interface (API) for the programming language Java

2. Requirements

BR#	Requirement Description
1	Use CRUD operations on a question (create, read, update, or delete a question)
2	Search questions by topic or difficulty
3	Assemble automatically a quiz (a quiz is a set of questions) that gathers all the questions covering a given list of topics.
4	Export this quiz under a plain text format
5	Run the evaluation and provide the automatic mark in the end of this execution
6	Do the same using a desktop GUI (JavaFX is preferred)
7	Export a quiz as PDF

8	Write an algorithm (or use an existing one) that will allow to get quiz based on a complexity rate. This overall complexity required by the user can be calculated on the difficulty property.

3. Classes

3.1 Answer

Answer is a helper container that holds an answer for a specific question. It is an abstract class to derive question type specific answers out of it, since the correct answer containers for questions are different for different type of questions.

3.2 Question

Question class is one of the main classes of the program. It is an abstract class used to derive other specific type questions. But it holds the most information (common attributes of questions) and most of the processes are done through this class. The common attributes include the question text (body), question owner, topics, question type, publicity, difficulty, correct answer count.

- **Multiple Choice Question**

This is the container for a question where there are multiple choices and one is correct. The user needs to pick one out of 4 choice. Correct choice hold as a single string while other false choices are hold as string array. This way correct and false choices are separated from each other.

3.3 Student

Student is one of the other classes of the program. It is required to login to the system and use the functionalities. It holds the id, username and authority status. Instead of using the User object itself most of the time the id is used by other classes

3.4 Quiz

Quiz is the last main class of the program. It is constructed from a list of questions and a user (to indicate owner of quiz).

3.5 Topics

Topics is one of the classes of the program. It holds the topics of the Quiz where the user can select and answer the questions regarding the topic.

4. Functionalities

SR#	Functionality

1	**Students to take Quiz online**
	
	private static void takeQuizOnline() {
	input = new Scanner(System.in);
	System.out.println("Welcome to Student Portal ");
	System.out.print("Please give Topic : ");
	String topic = input.next();
	System.out.println("Enter Difficulty");
	int diffclty = input.nextInt();
	
	List<Answer> ansList = questionJDBCDao.studntQuiz(topic, diffclty);
	int score = 0;
	for (int i = 0; i < ansList.size(); i++) {
	System.out.println(ansList.get(i).getQues().getQuestion());
	System.out.println("A : " + ansList.get(i).getA());
	System.out.println("B : " + ansList.get(i).getB());
	System.out.println("C : " + ansList.get(i).getC());
	System.out.println("D : " + ansList.get(i).getD());
	
	System.out.println("Please Enter your Answer ::::: ");
	String chc = input.next();
	System.out.println("chc::"+chc);
	if (chc.equalsIgnoreCase(ansList.get(i).getText())) {
	score++;
	} else {
	System.out.println("Wassup");
	}
	
	}
	
	System.out.println("Congratulations, Quiz has been ended, and your total Score is :::" + score);
	}

2	**To create the questions for the Quiz**
	
	public static void insertQ() {
	input = new Scanner(System.in);
	System.out.print("Enter Question: ");
	String questn = input.nextLine();
	System.out.println("Enter Topic: ");
	String topic = input.nextLine();
	System.out.println("Enter Difficulty");
	String diffclty = input.nextLine();
	System.out.print("Enter A: ");
	String a = input.nextLine();
	System.out.print("Enter B: ");
	String b = input.nextLine();
	System.out.print("Enter C: ");
	String c = input.nextLine();
	System.out.print("Enter D: ");
	String d = input.nextLine();
	System.out.print("Enter Actual Answer: ");
	String ans = input.nextLine();
	
	questionJDBCDao.insertQ(questn, Integer.*parseInt*(diffclty), a, b, c, d, ans);
	
	System.***out***.println("Question has been created.");
	}

3	**To update questions for the Quiz**
	
	public static void updateQ() {
	
	input = **new** Scanner(System.***in***);
	System.***out***.print("Enter Question to Update: ");
	String questn = *input*.nextLine();
	
	System.***out***.println("Enter new Difficulty");
	int diffclty = *input*.nextInt();
	
	System.***out***.print("Enter New Question ");
	String newQuestn = *input*.nextLine();
	
	System.***out***.println("Enter New topic");
	@SuppressWarnings("unused")
	String topicName = *input*.nextLine();
	questionJDBCDao.updateQ(questn, newQuestn, diffclty);

	System.***out***.println("Question has been updated."); }

4	**To Delete question from the quiz**
	
	public static void deleteQ() {
	input = **new** Scanner(System.***in***);
	
	System.***out***.print("Enter Question to Delete: ");
	String questn = *input*.next();
	questionJDBCDao.deleteQ(questn);
	System.***out***.println("Question has been deleted.");
	
	}

5	**To search for a question**
	
	public static void searchQ() {
	
	input = **new** Scanner(System.***in***);
	
	System.***out***.print("Enter Question to Search: ");
	String questn = *input*.next();
	
	questionJDBCDao.searchQ(questn);
	System.***out***.println("Search Completed.");
	
	}

6	**Users can search questions by difficulty and topics.**
	
	public static void difficultyonQ() **throws** FileNotFoundException, UnsupportedEncodingException {
	Scanner scanner = **new** Scanner(System.***in***);
	System.***out***.println("Please select Difficulty level");
	System.***out***.println("1.Easy");
	System.***out***.println("2.Medium");
	System.***out***.println("3.Hard");
	
	String diff;
	
	diff = scanner.nextLine();
	
	System.***out***.println("the difficulty you have selected is:" + diff);
	
	questionJDBCDao.difficultyonQ(diff);
	
	}

4.1. Non-functional Requirements

3.1.1. The program shall be able to run on console.

3.1.2 The data is stored in h2 database.

5. Hardware Requirements

#	Hardware	Requirement
1	Operating System	Compatible with Windows, Mac OS X, Linux
2	RAM	Minimum required 124MB
3	Disk Space	Minimum required 124MB
4	Processor	64-bit, four-core, 2.5 GHz minimum per core

6. Appendix

UML Class Diagram