



Fundamental Java Project : Quiz Manager

Technical Specification Document

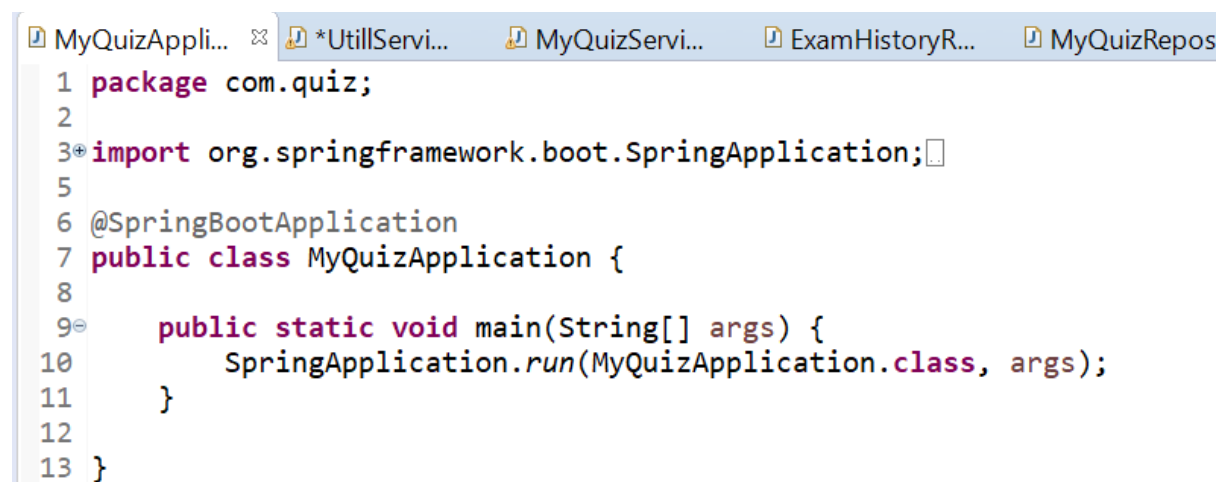
-Indhu Arivalagan

Technical Description:

The aim of this project is to develop an application to manage the preparation and execution of quiz (API, web-based, oriented).

Introduction:

This project is a console application and helps to carry out CRUD operations on open questions and MCQ – preparation and execution. This document is prepared for the Fundamental Java Project by Indhu Arivalagan for her computer science master's program at L'École Pour l'Informatique et les Techniques Avancées (EPITA).



```
1 package com.quiz;
2
3 import org.springframework.boot.SpringApplication;
4
5
6 @SpringBootApplication
7 public class MyQuizApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(MyQuizApplication.class, args);
11     }
12
13 }
```

Project Overview:

In this project, the user can add questions, search for topic-based questions, attend a quiz, evaluate the quiz and export the file to a plain text format.

Project Scope:

- Assemble quizzes automatically using open questions, Boolean and multiple choices.
- Auto-grading for questions of multiple choice.
- Data access object creation using CRUD Methods.
- Create the application's configurable properties file.
- Export quizzes in plain text format.

Software Requirements:

Technologies Used:

BACKEND/WEBSERVER	Java 8, Spring-boot, JSP , Tomcat-Embed-Jasper
FRONTEND	HTML, CSS, JavaScript
DATABASE	H2
VERSION CONTROL	GIT
TESTING	Unit Testing

Project Dependencies:

- Install Java 8
- Install H2 JDBC
- Install Spring-boot
- Html and Springframework
- Create table in database

Database:

This project uses the H2 database and the table information are provided in the database.

Acronyms and Abbreviations:

ACRONYMS	ABBREVIATIONS
CRUD	(Create, Read, Update, Delete) → Relational database application functions.
DAO	(Data Access Object) → Service class to communicate with the H2 database.
UML	(Unified Modelling Language) → a standard way of displaying the system design.
MCQ	(Multiple Choice Question) → Questions with possible responses listed in our implementation using radial buttons.
JDBC	(Java Database Connectivity) → Application Programming Interface(API) for Java Programming Language.

Project Requirements:

1. Create a quiz based on the requirements (topics) of the user.
2. Use questions stored in database (using h2 database).
3. Allows the user to take the quiz.
4. At the end of the assessment, correct open & MCQ questions and display results.
5. Export the quiz in a plain text format.
6. Search for topic based questions.
7. Use CRUD operations on a question

System Specification:

Functional Requirements:

- When the quiz is instantiated, users can customize their quiz, which takes a list of topics as one of the parameters.

```
1 package com.quiz.service;
2
3 import java.util.ArrayList;
13
14 @Service
15 public class UtitlService {
16
17     @Autowired
18     private MyQuizRepositry myQuiz;
19
20     @Autowired
21     private ExamHistoryRepositry examHistoryRepositry;
22
23     public void save() {
24
25         MyExam exam = new MyExam();
26         exam.setExamDescription("Java Language Basics Quiz");
27         exam.setExamQuestions(preparingQuestions());
28         myQuiz.save(exam);
29
30         MyExam exam1 = new MyExam();
31         exam1.setExamDescription("General Knowledge Quiz");
32         exam1.setExamQuestions(preparingQuestions2());
33         myQuiz.save(exam1);
34
35         MyExam exam2 = new MyExam();
36         exam2.setExamDescription("Job Interview Quiz");
37         exam2.setExamQuestions(preparingQuestions3());
38         myQuiz.save(exam2);
39
40         MyExam exam3 = new MyExam();
41         exam3.setExamDescription("English to french Quiz");
42         exam3.setExamQuestions(preparingQuestions4());
43         myQuiz.save(exam3);
44     }
45 }
```

- Can be able to add Questions connecting to the database(h2), i.e., The questions already used and stored in the database can be used by users.

```
@GetMapping("/addQuestions")
public String addQuestions() {
    utillService.save();
    return "Succesfully Added Questions";
}
```

- Preparing Questions Based on topics

```
public void save() {

    MyExam exam = new MyExam();
    exam.setExamDescription("Java Language Basics Quiz");
    exam.setExamQuestions(preparingQuestions());
    myQuiz.save(exam);
}
```

```
public List<ExamReleatedQuestions> preparingQuestions() {

    List<ExamReleatedQuestions> examquestion = new ArrayList<ExamReleatedQuestions>();

    ExamReleatedQuestions examReleatedQuestions = new ExamReleatedQuestions();
    examReleatedQuestions.setQuestionText("What is JAVA?");
    examReleatedQuestions.setOptionA("Java is a high-level programming language and is platform independent.");
    examReleatedQuestions.setOptionB("Java is a low-level programming language and is platform independent.");
    examReleatedQuestions.setOptionC("Java is a high-level programming language and is platform dependent.");
    examReleatedQuestions.setOptionD("None of the above");
    examReleatedQuestions.setTypeOfQuestion("MCQ");
    examReleatedQuestions.setCorrectAnswer("A");
    examquestion.add(examReleatedQuestions);
}
```

- Users can search for topic-based questions.

```
<script type="text/javascript">
$(document).ready(function() {
    $("#searchbtn").click(function() {
        var serachward=$("#searchText").val();
        if(serachward==''){
            alert("Text blank");
        }else{
            debugger;
            $.ajax({
                type : "POST",
                url : "/getExamByserachward",
                dataType : 'json',
                data : JSON.stringify(serachward),
                contentType : "application/json;charset=UTF-8",
                xhrFields : {
                    withCredentials : true
                },
                success : function(data) {
                    if(data.length>0)
                    {
                        alert("Exam Found ..! Please Check Below ");
                    }
                    else
                    {
                        alert("No result Found ");
                    }
                },
                error : function(xhr, status, error) {

                },
            });
        }
    });
});
```

MyQuizController.java

```
@PostMapping("/getExamByserachward")
public List<MyExam> getExamByserachward(@RequestBody String searchward) {

    List<MyExam> list = myQuizService.getExamList();

    List<MyExam> newlist = new ArrayList<MyExam>();
    searchward = searchward.toUpperCase().trim();
    char c = searchward.charAt(1);
    for (MyExam myExam : list) {

        int i = myExam.getExamDescription().indexOf(c);
        if (i >= 0) {
            newlist.add(myExam);
        }
    }
    return newlist;
}
```

- Users can evaluate their quiz and display the results after submission.

```
@GetMapping("/submitExam")
public ModelAndView submitExam() {
    String result = myQuizService.submitExam(globalExamId);

    ModelAndView andView = new ModelAndView("result");

    andView.addObject("result", result);

    myQuizService.deleteSubmitExam(globalExamId);
    return andView;
}
```

```
public String submitExam(Integer examId) {
    List<UserExamHistroy> list = examHistoryRepository.findByExamID(examId);
    int count = 0;
    int totalCount = list.size();
    for (UserExamHistroy userExamHistroy : list) {
        if (userExamHistroy.getCorrectAns().equalsIgnoreCase(userExamHistroy.getUserSelectdAns())) {
            count++;
        }
    }

    String result = "Dear User , You Got " + count + " Out of " + totalCount;
    return result;
}

public void deleteSubmitExam(Integer examId) {
    List<UserExamHistroy> list = examHistoryRepository.findByExamID(examId);
    for (UserExamHistroy userExamHistroy : list) {
        examHistoryRepository.delete(userExamHistroy);
    }
}
```

- Users can export the quiz as a plain text format.

```
@PostMapping("/exportFile")
public String exportFile(@RequestBody Integer id) {
    String path = "";
    List<MyExam> myexam = myQuizService.getExamList();

    for (MyExam myExam2 : myexam) {

        if (myExam2.getExamId().equals(id)) {
            path = prepareFile(myExam2);
        }

    }

    return path;
}
```

```
public String prepareFile(MyExam myExam2) {
    String str = "\n";
    String path="";
    FileOutputStream fout =null;
    try {
        path = "D:\\testout" + myExam2.getExamId() + " " + ".txt";
        fout = new FileOutputStream(path);
        str = myExam2.getExamDescription() + "\n ";
        List<ExamReleatedQuestions> e = myExam2.getExamQuestions();

        for (ExamReleatedQuestions examReleatedQuestions : e) {
            str = str + examReleatedQuestions.getQuestionText() + " \n " + "\n "
                + examReleatedQuestions.getOptionA() + " " + examReleatedQuestions.getOptionB() + " \n"
                + examReleatedQuestions.getOptionC() + " " + examReleatedQuestions.getOptionD() + " \n"
                + examReleatedQuestions.getCorrectAnswer() + " /n " + examReleatedQuestions.getTypeOfQuestion();
        }

        byte[] b = str.getBytes();// converting string into byte array
        fout.write(b);
        fout.close();
    } catch (Exception e) {
        throw new RuntimeException(null, " "+e);
    } finally {
        try {
            fout.close();
        } catch (IOException e) {
        }
    }
    return path;
}
```


- Quiz can have three different types of questions; open, MCQ and Boolean.

MCQ

```
public List<ExamReleatedQuestions> preparingQuestions() {  
  
    List<ExamReleatedQuestions> examquestion = new ArrayList<ExamReleatedQuestions>();  
  
    ExamReleatedQuestions examReleatedQuestions = new ExamReleatedQuestions();  
    examReleatedQuestions.setQuestionText("What is JAVA?");  
    examReleatedQuestions.setOptionA("Java is a high-level programming language and is platform independent.");  
    examReleatedQuestions.setOptionB("Java is a low-level programming language and is platform independent.");  
    examReleatedQuestions.setOptionC("Java is a high-level programming language and is platform dependent.");  
    examReleatedQuestions.setOptionD("None of the above");  
    examReleatedQuestions.setTypeOfQuestion("MCQ");  
    examReleatedQuestions.setCorrectAnswer("A");  
    examquestion.add(examReleatedQuestions);  
}
```

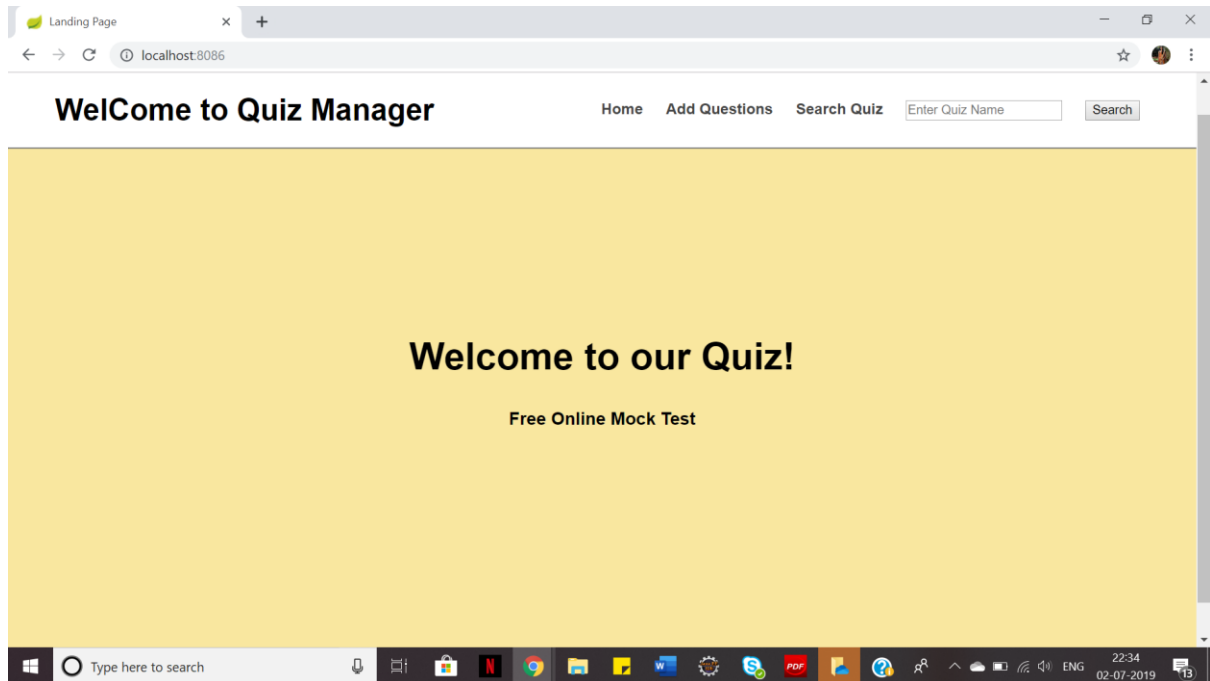
TEXT

```
ExamReleatedQuestions examReleatedQuestions7 = new ExamReleatedQuestions();  
examReleatedQuestions7.setQuestionText("Baby of a dog is called_");  
examReleatedQuestions7.setOptionA("See");  
examReleatedQuestions7.setOptionB("public final double methoda();");  
examReleatedQuestions7.setOptionC(" static void methoda(double d1);");  
examReleatedQuestions7.setOptionD("None of the above");  
examReleatedQuestions7.setTypeOfQuestion("TEXT");  
examReleatedQuestions7.setCorrectAnswer("Puppy");  
examquestion.add(examReleatedQuestions7);
```

BOOLEAN

```
ExamReleatedQuestions examReleatedQuestions9 = new ExamReleatedQuestions();  
examReleatedQuestions9.setQuestionText(" Sharks are mammals.");  
examReleatedQuestions9.setOptionA("boolean b1 = 0;");  
examReleatedQuestions9.setOptionB("boolean b2 = 'false';");  
examReleatedQuestions9.setOptionC("boolean b3 = false;");  
examReleatedQuestions9.setOptionD("None of the above");  
examReleatedQuestions9.setTypeOfQuestion("BOOLEAN");  
examReleatedQuestions9.setCorrectAnswer("TRUE");  
examquestion.add(examReleatedQuestions9);
```

- The welcome page appears as follows.



User Interface Design:

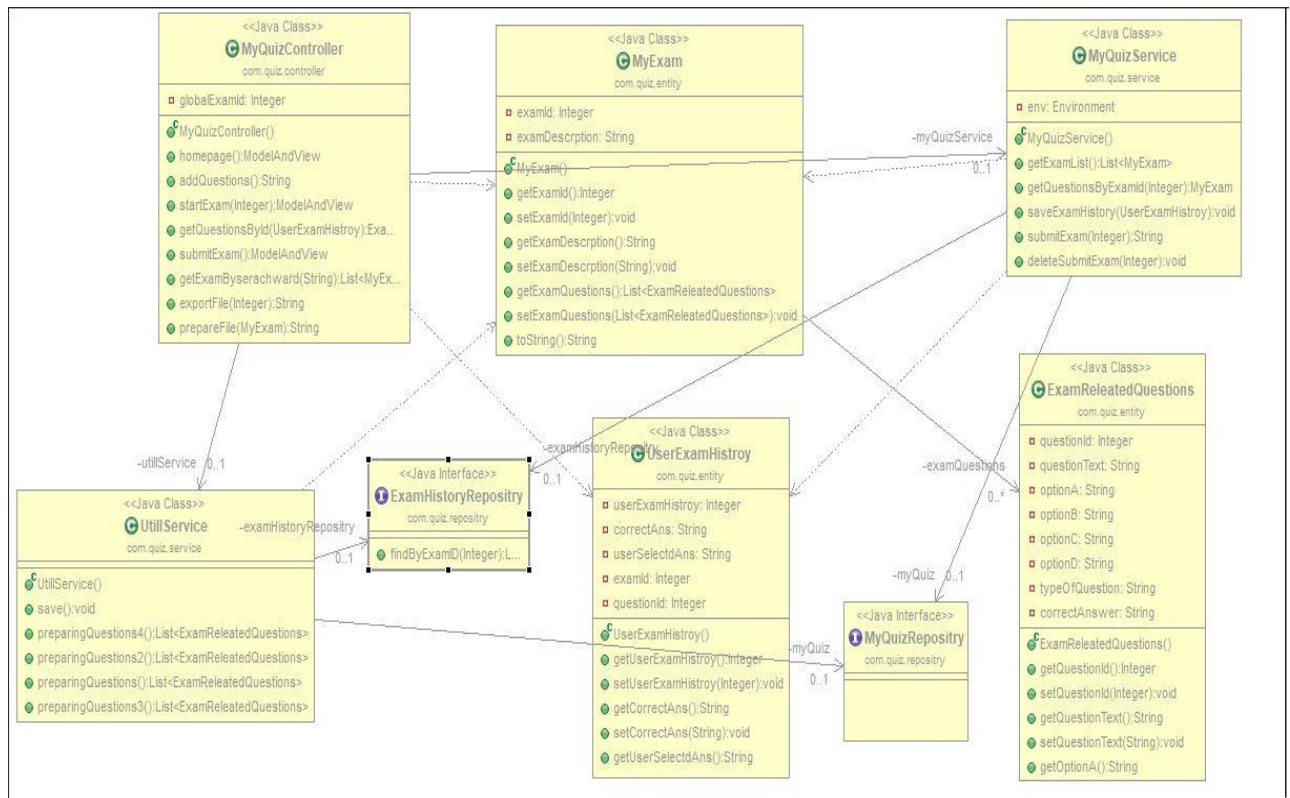
Refer to the user guide.

Hardware Interfaces

- ✓ Operating System : Windows xp or more, MAC or UNIX
- ✓ Processor : Pentium 3.0 GHz or higher
- ✓ RAM : 256 MB or more
- ✓ Hard Disk : 10 GB or more

Appendix:

UML Class Diagram



Bibliography:

<https://thomas-broussard.fr/work/java/courses/project/fundamental.xhtml>