CSIA
Development

Hakobyan Aram

March 13, 2018

1 Libraries used

1.1 Dagger 2

Dagger 2 is a fast dependency-injection framework for java, maintained by Google. Dependency injection allows to create reusable and interchangeable code.

1.2 AWS DynamoDB SDK

Amazon DynamoDB is a NoSQL database service, which was chosen to store quiz questions and answers for the application.

1.3 JavaFX

JavaFX is java framework for creating GUI applications. It was chosen to create the application for its simplicity, good documentation and rich support.

2 Classes

2.1 Overview

Below is the UML diagram of the main classes in the project.

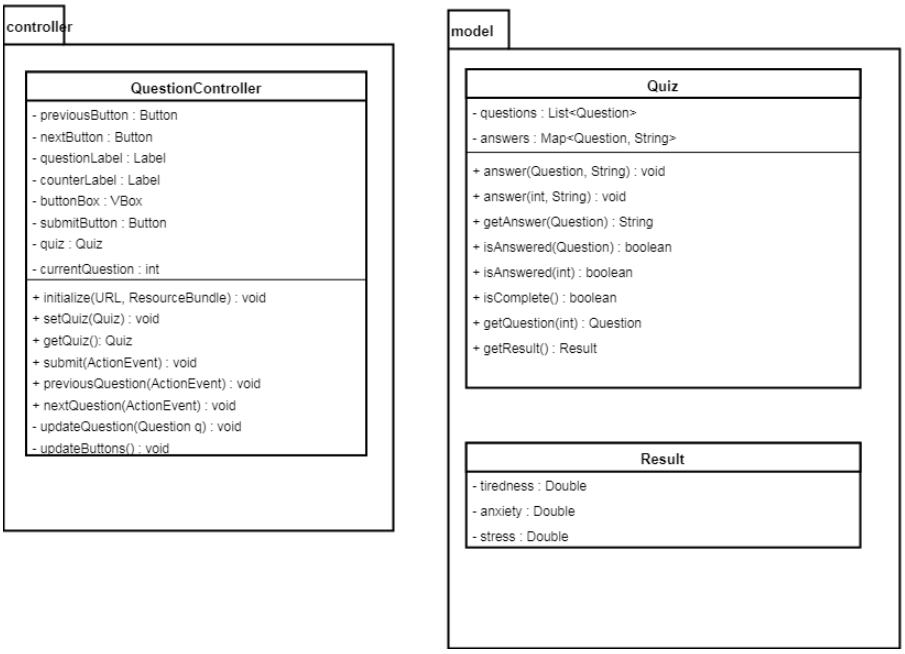

Figure 1: Class diagram

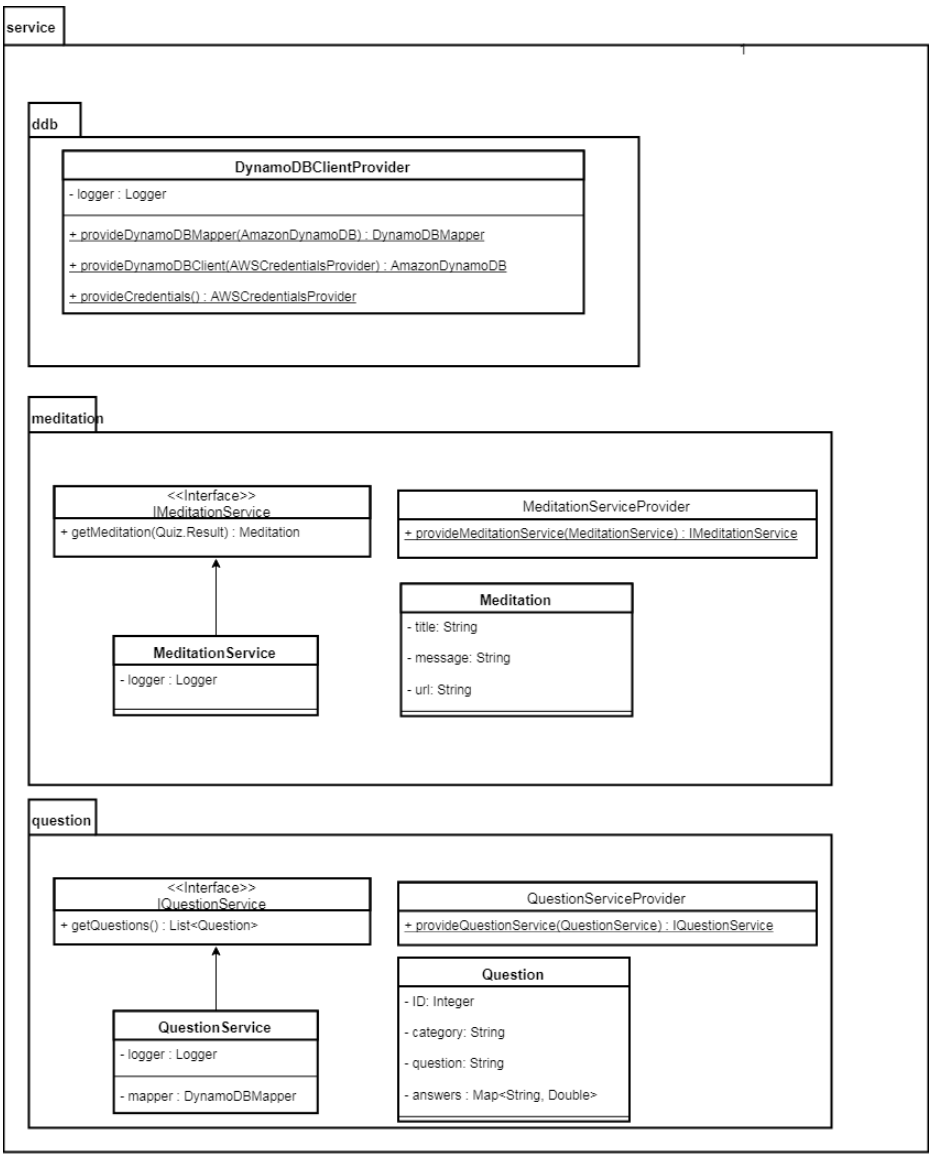

Figure 2: Class diagram

2.2 QuizController

QuizController is the controller class responsible for displaying the quiz questions and accepting user input.

```
q.getAnswers().entrySet()
        .stream()
        .sorted(Map.Entry.comparingByValue(Collections.reverseOrder()))
        .forEach(entry -> {
            RadioButton radio = new RadioButton(entry.getKey());
            radio.setToggleGroup(toggleGroup);
            if(quiz.isAnswered(q) && quiz.getAnswer(q).equals(entry.getKey())){
                radio.setSelected(true);
            }
            radio.setOnAction(event -> {
                quiz.answer(q, ((RadioButton)event.getSource()).getText());
                this.updateButtons();
            });
            this.buttonBox.getChildren().add(radio);
        });
```

Figure 3: QuizController

The code above iterates over the answers of a particular question assigns each answer to a radio button. Also, it makes sure that the answer to the question is updated when the user checks a radio button.

2.3 MeditationService

The MeditationService class calculates the most appropriate meditation session based on the quiz results.

```
if(result.getTiredness() > result.getAnxiety() &&
        result.getTiredness() > result.getStress() &&
        result.getStress() < 5 &&
        result.getAnxiety() < 5){
    return Meditation.builder().message("Enjoy your meditation").url("https://www.youtube.com/watch?v=
```

Figure 4: MeditationService

The meditation is calculated using an algorithm that takes into account the level of anxiety, tiredness and stress of the user, and the relationships between the three.

2.4 QuestionService

The QuestionService class is used to fetch quiz questions from remote DynamoDB database.

```java
@Override
public List<Question> getQuestions() {
    List<Question> questions = new ArrayList<>();
    PaginatedScanList<Question> results = mapper.scan(Question.class, new DynamoDBScanExpression());
    questions.addAll(results);
    return questions;
}
```

Figure 5: QuestionService

The class uses an instance of DynamoDBMapper class from the AWS DynamoDB SDK that is used to communicate with the database.

2.5 Quiz

The Quiz class represents the model of the quiz in the application.

```java
public Result getResult(){
    Double tiredness = 0.0;
    Double anxiety = 0.0;
    Double stress = 0.0;
    for (Map.Entry<Question, String> entry : answers.entrySet()) {
        Question q = entry.getKey();
        Double answer = q.getAnswers().get(entry.getValue());
        switch(entry.getKey().getCategory()){
            case "Stress":
                stress += answer; break;
            case "Tiredness":
                tiredness += answer; break;
            default:
                anxiety += answer; break;
        }
    }
    return Result.builder().anxiety(anxiety).stress(stress).tiredness(tiredness).build();
}
```

Figure 6: Quiz

The method getResult creates a Result object that stores the stress, anxiety and tiredness levels that are calculated based on the user's answers to the quiz questions.

2.6 MeditationApp

MeditationApp is a class extending from JavaFX Application, it handles the initialization of the application and changes of views.

```
public void submitResult(Quiz.Result result){
    logger.trace(result);
    Meditation meditation = meditationService.getMeditation(result);
    AnchorPane pane = new AnchorPane();
    VBox vBox = new VBox();
    vBox.setAlignment(Pos.CENTER);
    vBox.setSpacing(10.0);
    Hyperlink link = new Hyperlink(meditation.getUrl());
    link.setOnAction(t -> getHostServices().showDocument(link.getText()));
    Label label = new Label(meditation.getMessage());
    label.setTextAlignment(TextAlignment.CENTER);
    Button resetButton = new Button();
    resetButton.setText("Start over");
    resetButton.setOnAction(e-> this.displayQuestions());
    vBox.getChildren().addAll(label, link, resetButton);
    pane.getChildren().add(vBox);
    Scene scene = new Scene(pane,  width: 300,  height: 200);
    primaryStage.setTitle("Meditation");
    primaryStage.setScene(scene);
    primaryStage.show();
}
```

Figure 7: MeditationApp

The method submitResult submits the result of the quiz to the MeditationService, receives the Meditation objects, then proceeds to create a view to display the meditation to the user.

```
public void displayQuestions()  {
    try {
        List<Question> questions = questionService.getQuestions();
        Quiz quiz = Quiz.builder().questions(questions).answers(new HashMap<>()).build();
        QuestionController controller = updateStage( fxml: "/fxml/QuestionController.fxml",  title: "",  width: 400,  height: 250);
        controller.setQuiz(quiz);
        this.primaryStage.show();
    } catch (Exception e){
        logger.error(e);
        AlertHelper.generateErrorAlert( title: "Error",  header: "There seems to be a problem",  content: "Check your internet connection").s
    }
}
```

Figure 8: MeditationApp

The method displayQuestions fetches the questions from the QuestionService, creates a Quiz object. Then it proceeds to create a QuizController instance, populates it with the Quiz and displays it. It also makes sure that if the application is unable to fetch the questions, and error message is displayed to the user.

3 Reference

- Docs.aws.amazon.com. (2018). Amazon DynamoDB. [online] Available at: https://docs.aws.amazon.com/amazondynamodb/latest/developerguide [Accessed 10 Mar. 2018].
- Google.github.io. (2018). User's Guide. [online] Available at: https://google.github.io/dagger/users-guide [Accessed 5 Mar. 2018].
- Docs.oracle.com. (2018). Getting Started with JavaFX: About This Tutorial — JavaFX 2 Tutorials and Documentation. [online] Available at: https://docs.oracle.com/javafx/2/get_started/jfxpub-get_started.htm [Accessed 2 Mar. 2018].