Customer specification for a Trivia application

The Open Trivia Database provides a simple API from which one can fetch trivia questions and answers in JSON format. For example:

https://opentdb.com/api.php?amount=5

```
{
  "response_code":0,
  "results":[
    {
      "category":"Sports",
      "type":"multiple",
      "difficulty":"hard",
      "question":"What tool lends it&#039;s name to a last-stone advantage in an end in Curling?",
      "correct_answer":"Hammer",
      "incorrect_answers":[
        "Wrench",
        "Drill",
        "Screwdriver"
      ]
    },
    {
      "category":"Entertainment: Video Games",
      "type":"multiple",
      "difficulty":"easy",
      "question":"Which game did NOT get financed via Crowdfunding?",
      "correct_answer":"Enter the Gungeon",
      "incorrect_answers":[
        "Town of Salem",
        "Undertale",
        "Tower Unite"
```

Background

The customer wants an application that offers a set of 5 multiple-choice questions to the user and the possible answers for each. The user selects the answers and is shown whether he/she is correct or not. The customer expects a Java backend and a user interface (frontend):
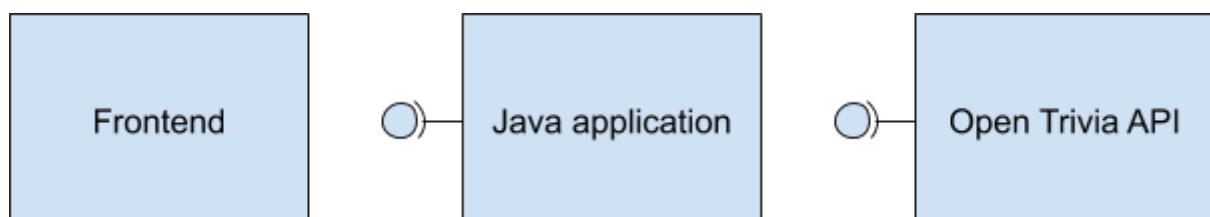
Task for backend developer

The Open Trivia API is not ideal because a clever user can inspect the JSON and discover the correct answer. Therefore, the *Java application* should prevent this by providing two APIs to the *frontend*: one that presents 5 questions and answer options and another to check if the selected answer is correct.

Objectives:
- A Java application with two REST (JSON) endpoints:
  - GET /questions
  - POST /checkanswer
- Post the link to Github where the source code can be found.
- Provide a README in the root directory containing instructions for how to build and run the application.

Suggestions

- Spring Boot is the recommended framework as it provides [web features](#) out of the box.
- Consider (unit) testing.
- There is no requirement to use a database.