QUAD

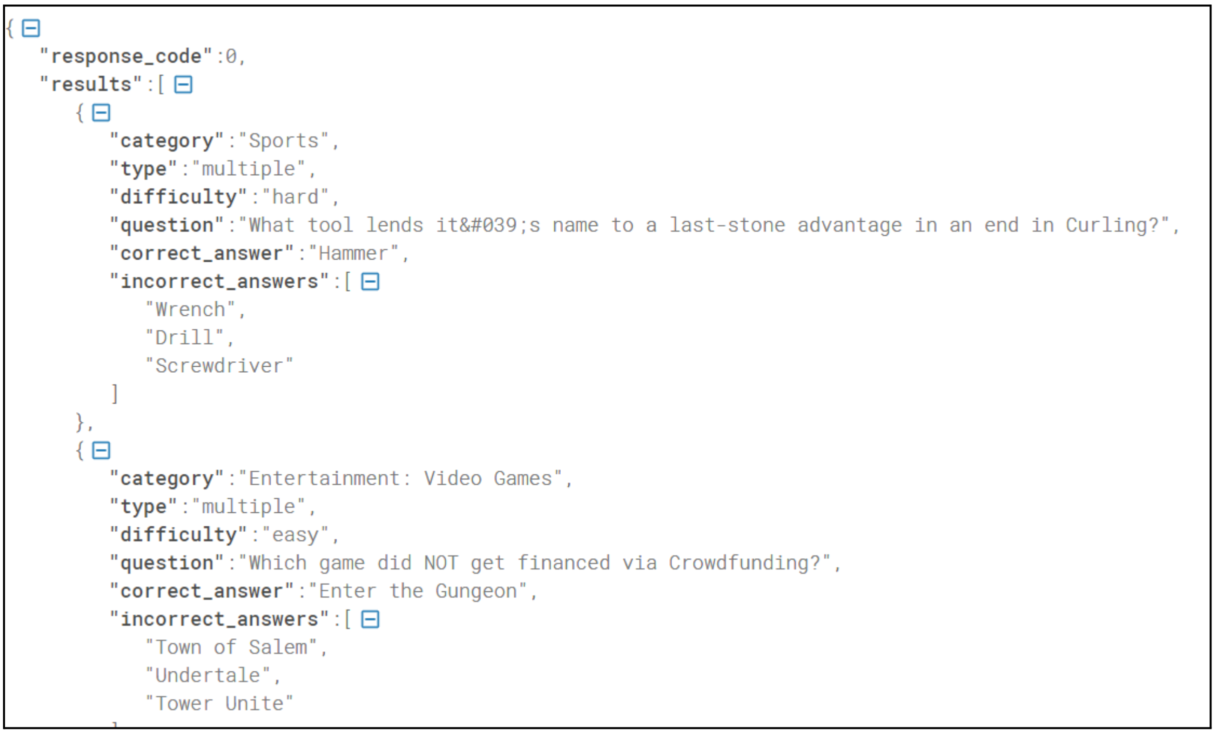
Technical Assignment

Requirements

The customer wants a web application that offers a set of 5 multiple-choice questions. The user selects the answers and for each one is shown whether they have the correct answer or not.

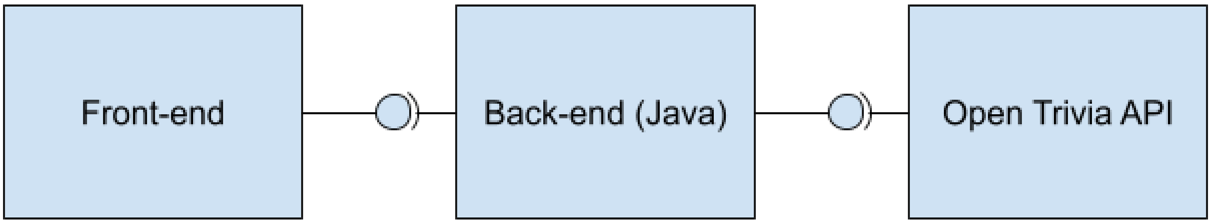
The Open Trivia Database provides a simple API to access trivia questions and answers in JSON format. For example:

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"
```

The customer expects a Java backend and a user interface (front-end) of some kind.



QUAD

Assignment specification

The Open Trivia API is not ideal because a clever user can inspect the JSON and discover the correct answer. Therefore, the *back-end* should prevent this by providing two APIs to the *front-end*; one that presents 5 questions and possible answers and another to check if the selected answer is correct.

Objectives

- Build a back-end that exposes two REST (JSON) endpoints:
  - `GET /questions`
  - `POST /checkanswers`
- Build a simple UI for the user to interact with

Deliverables

- A link to Github where the source code can be found.
- A README in the root directory with instructions for how to build and run the application
- A link to the running application on a cloud platform (optional)

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.
- Front-end technology is not important, choose what you want.