95-702 Distributed System Management

Name: Seon Jhang
Andrew ID: sjhang

Project 4-2 – Trivia Quiz App

Description

My application is a native Android trivia quiz app that fetches multiple-choice questions from a cloud-deployed servlet-based web service and displays them to the user. Users can enter their names, answer multiple-choice questions, receive instant feedback, and repeat the game. The server communicates with a third-party API and returns data in JSON format.

1. Implement a Native Android Application

TriviaQuizAndroidApp

a. Layout with Multiple Views

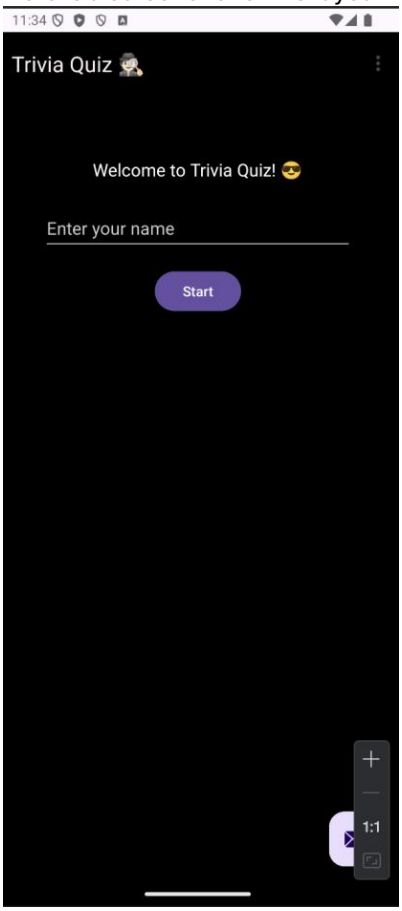
The application includes at least three different Android Views in the layout:

- TextView – displays the question text and score summary
- EditText – allows the user to input their name
- Button – used for submitting answers and navigating questions

The layout includes a TextView for questions and feedback, an EditText for name input, and Buttons for selecting answers and moving to the next question – covering user interaction, display, and input.

These are implemented across FirstFragment and SecondFragment with ViewBinding.

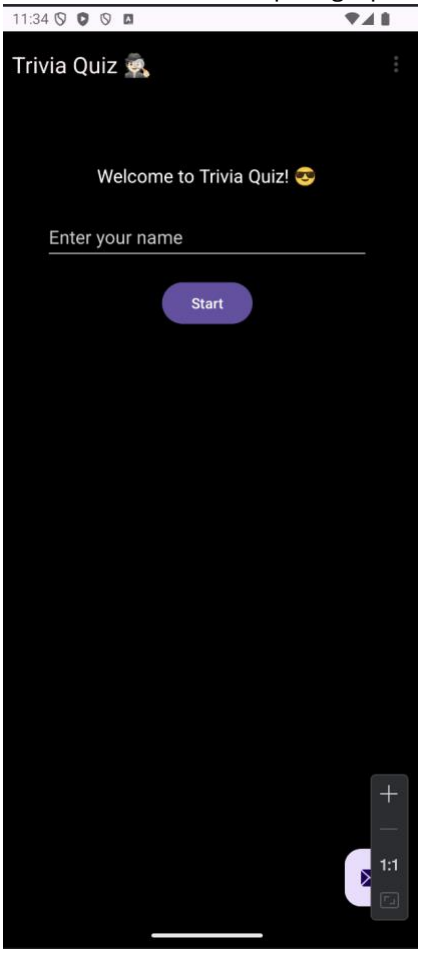
Here is a screenshot of the layout when first entered the app.



b. Requires Input from the User

In FirstFragment.java, users input their name via EditText.
In SecondFragment.java, they interact by selecting an answer from four buttons.

Here is a screenshot requiring input from the user – their name/nickname



c. Make an HTTP request from a background thread

The Android app performs a network request to the servlet-based web service using a background thread, ensuring that the main UI thread remains responsive.

This functionality is implemented in the GetTrivia class:

```
1 usage
public void fetch() {
    new Thread(() -> {
        List<TriviaQuestion> questions = fetchTrivia();
        activity.runOnUiThread(() -> {
            if (questions == null) {
                callback.onTriviaError( errorMessage: "Network error. Please try again.");
            } else {
                callback.onTriviaReady(questions);
            }
        });
    }).start();
}
```

A new Thread is spawned to handle the HTTP GET request to the servlet at:

```
1 usage
private List<TriviaQuestion> fetchTrivia() {
    try {
        URL url = new URL( spec: "https://upgraded-engine-6659jqvq9j3r7q7-8080.app.github.dev/trivia");
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setRequestMethod("GET");

```

```
https://upgraded-engine-6659jqvq9j3r7q7-8080.app.github.dev/trivia
```

```
no usages
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {
    try {
        List<TriviaQuestion> questions = triviaService.fetchQuestionsAndLog(
                req.getRemoteAddr(),
                req.getHeader( s: "User-Agent")
        );
```

Once the response is received and parsed, the UI is updated via activity.runOnUiThread()

d. Receives and parses JSON response

```
no usages
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {
    try {
        List<TriviaQuestion> questions = triviaService.fetchQuestionsAndLog(
                req.getRemoteAddr(),
                req.getHeader( s: "User-Agent")
        );

        JSONArray jsonArray = new JSONArray();
        for (TriviaQuestion q : questions) {
            JSONObject obj = new JSONObject();
            obj.put("question", q.getQuestion());
            obj.put("options", q.getOptions());
            obj.put("correct_index", q.getCorrectIndex());
            jsonArray.put(obj);
        }

        res.setContentType("application/json");
        res.setCharacterEncoding("UTF-8");
        res.getWriter().print(jsonArray.toString());

```

The response is a JSON array of trivia questions like:

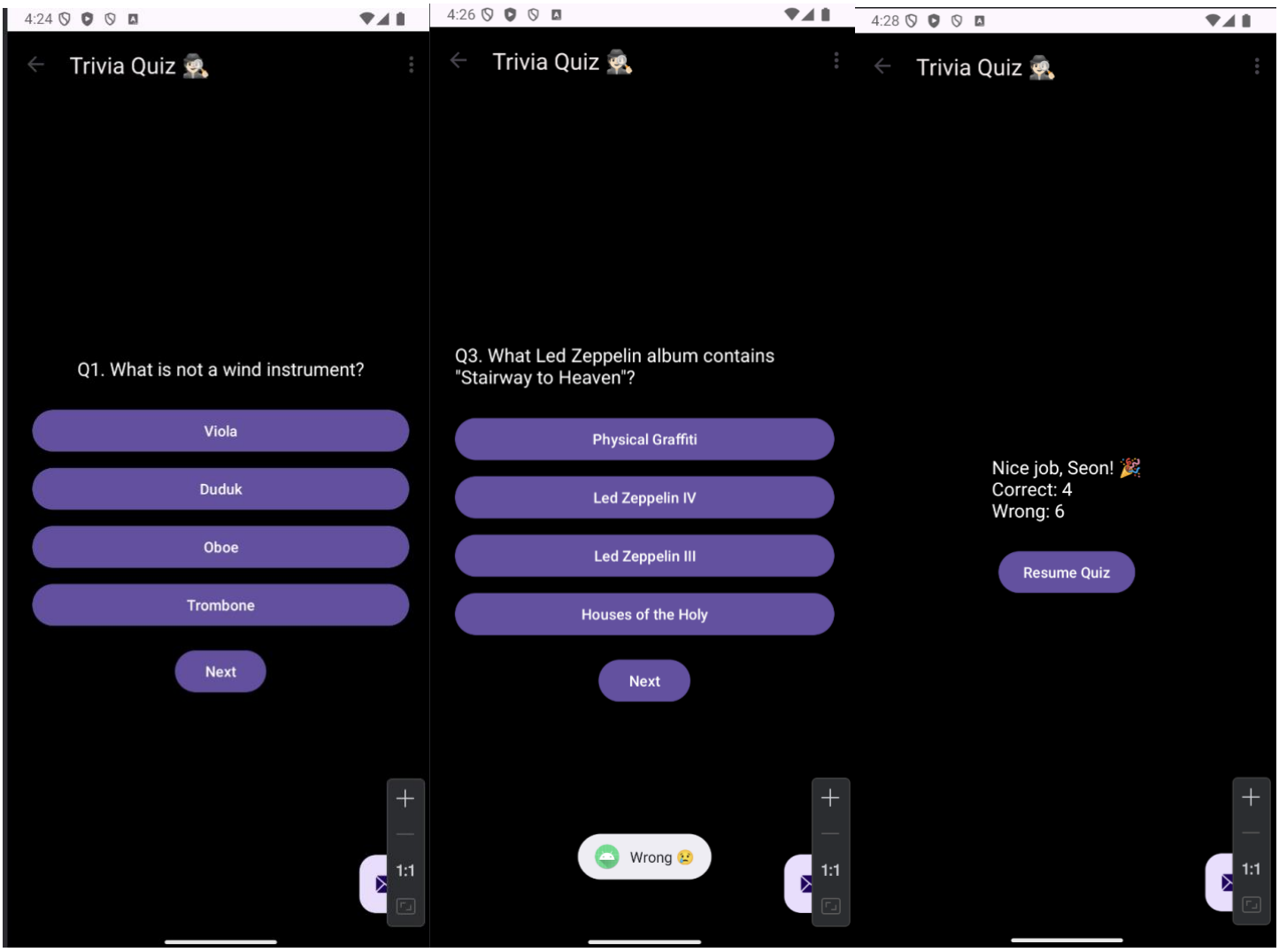

```
[
  {
    "correct_index": 2,
    "question": "Which of these songs by Skrillex features Fatman Scoop as a side artist?",
    "options": [
      "All is Fair in Love and Brostep",
      "Rock N Roll (Will Take You to the Mountain)",
      "Recess",
      "Scary Monsters and Nice Sprites"
    ]
  },
  {
    "correct_index": 3,
    "question": "Which year was the album \"Year of the Snitch\" by Death Grips released?",
    "options": [
      "2013",
      "2017",
      "2011",
      "2018"
    ]
  },
```

This is parsed in GetTrivia.java using org.json.

e. Display new information to the user

Once questions are received, SecondFragment dynamically updates the UI with:

- Current questions and options
- Toast notifications for correct/wrong answers
- Final score screen with a “Resume Quiz” option



f. Is repeatable (i.e. the user can repeatedly reuse the application without restarting it)

Users can repeat the quiz infinitely. The SecondFragment includes a “Resume Quiz” button, which resets the quiz state and re-fetches a new set of questions, enabling endless repeat usage.

2. Implement a web service

Trivia Servlet

a. Implement a simple (can be a single path) API

The web service implements a simple single-path API endpoint: /trivia
The endpoint is defined in the TriviaServlet.java class, using HttpServlet. It is responsible for:

- Handling incoming HTTP GET requests from the Android client
- Fetches 10 trivia questions from a third-party API: Open Trivia Database (https://opentdb.com/)
- Parsing and processing the response
- Logging the transaction to MongoDB
- Sending back a custom-formatted JSON response to the Android app

b. Receives an HTTP request from the native Android application

The Android application sends an HTTP GET request to this endpoint via a background thread (using Thread and HttpURLConnection).
On the server side, the doGet() method of TriviaServlet receives the request and handles it synchronously.
Additionally, the servlet extracts basic request metadata like Client IP address and User-Agent header.

c. Executes business logic appropriate to your application. This includes fetching XML or JSON information from some 3rd party API and processing the response.

Upon receiving the request, the servlet delegates processing to a service class that:

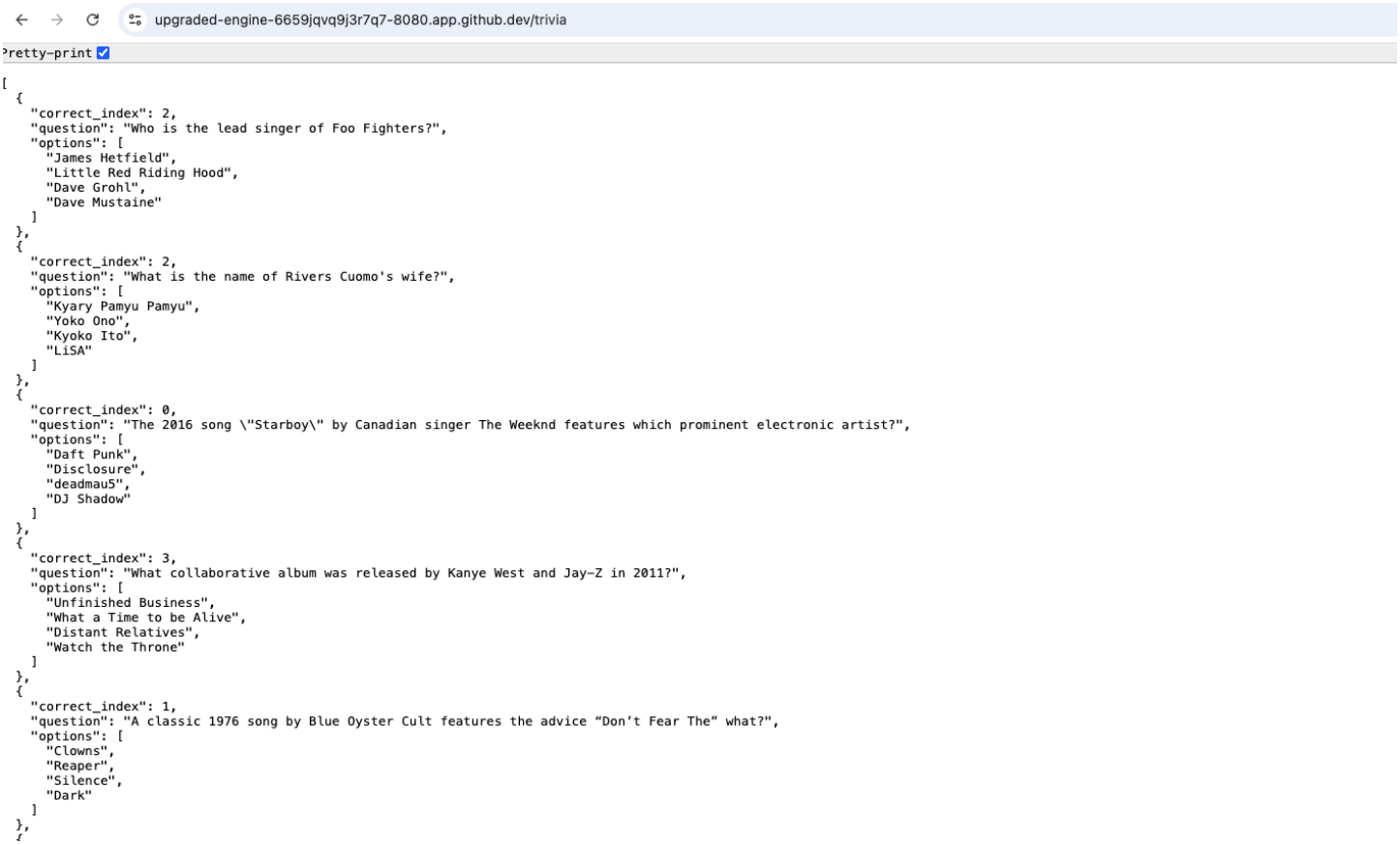
- Fetches JSON data from the published and documented third-party API

```
3 usages
public class TriviaService {
    2 usages
    private static final String API_URL = "https://opentdb.com/api.php?amount=10&category=12&difficulty=easy&type=multiple";

    1 usage
    public List<TriviaQuestion> fetchQuestionsAndLog(String clientIp, String userAgent) throws Exception {
        long startTime = System.currentTimeMillis();

        HttpURLConnection conn = (HttpURLConnection) new URL(API_URL).openConnection();
        conn.setRequestMethod("GET");
```

- Parses the JSON response to extract: Question text, Correct answer, Incorrect answers
- Shuffles all answers answers and tracks the index of the correct one
- Logs the request to a MongoDB Atlas database

d. Replies to the Android application with an XML or JSON formatted response. The schema of the response can be your own design.

The servlet returns a JSON array of trivia questions, structured using a custom schema.



Web Service Logging and Analysis Dashboard

Trivia Quiz App

Overview

As part of the distributed Trivia Quiz App, a web-based dashboard has been implemented to monitor, analyze, and visualize usage data of the trivia Api accessed by the Android client. The dashboard is accessible from a browser and is built using JSP and Servlets, with all logs stored in a cloud-hosted MongoDB Atlas database.

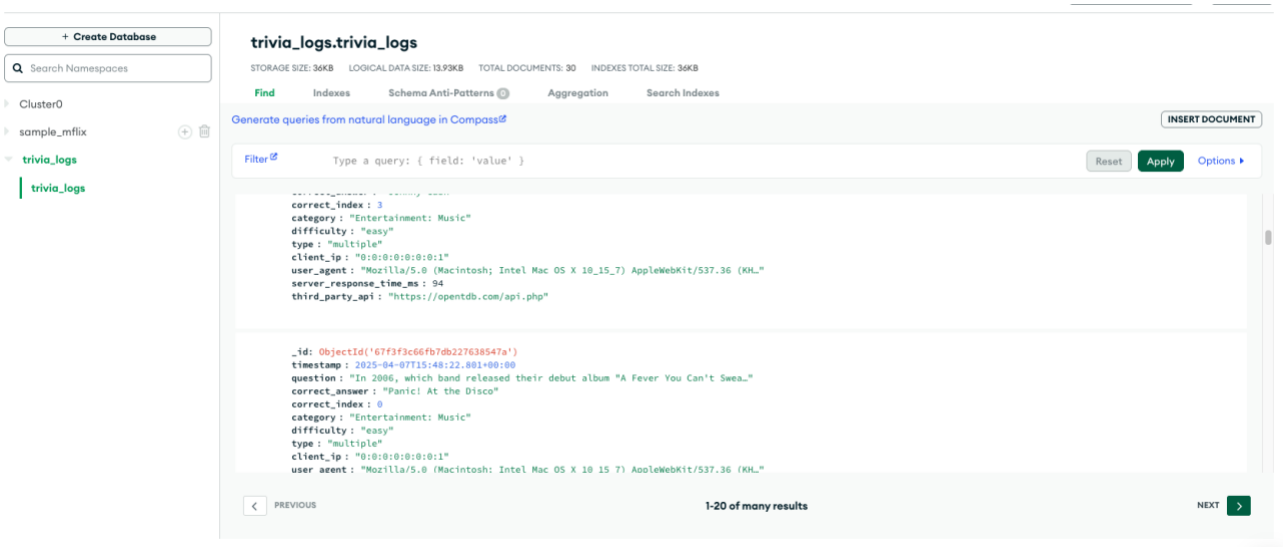
4. Log useful information

My web service logs at least six distinct and meaningful pieces of information for each trivia request made by the Android app.

```
Document logDoc = new Document()
        .append("timestamp", Instant.now().toString())
        .append("question", question)
        .append("correct_answer", correct)
        .append("correct_index", correctIndex)
        .append("category", trivia.optString( key: "category"))
        .append("difficulty", trivia.optString( key: "difficulty"))
        .append("type", trivia.optString( key: "type"))
        .append("client_ip", clientIp)
        .append("user_agent", userAgent)
        .append("server_response_time_ms", duration)
        .append("third_party_api", API_URL);
```

5. Store the log information in a database

All log entries are stored persistently using a MongoDB Atlas cloud database. This is achieved via the MongoLogger.java class.

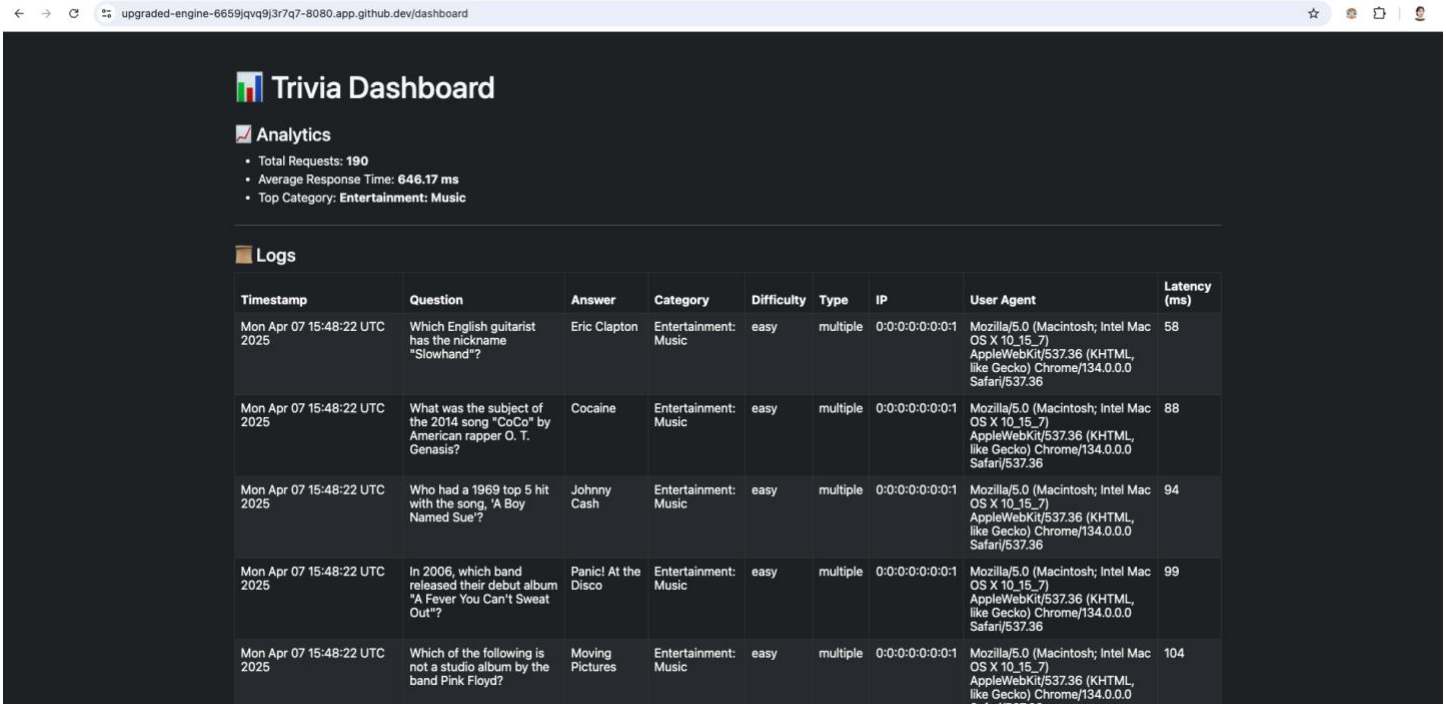


6. Display operations analytics and full logs on a web-based dashboard

a. A unique URL Addresses the Web interface

The dashboard is accessible at:
https://upgraded-engine-6659jqvq9j3r7q7-8080.app.github.dev/dashboard

This URL points to the DashboardServlet, which renders a JSP view (index.jsp) containing analytics and full logs.



b. The Dashboard displays at least 3 interesting analytics

In DashboardServlet.java, the following analytics are computed from the MongoDB lgos and displayed:

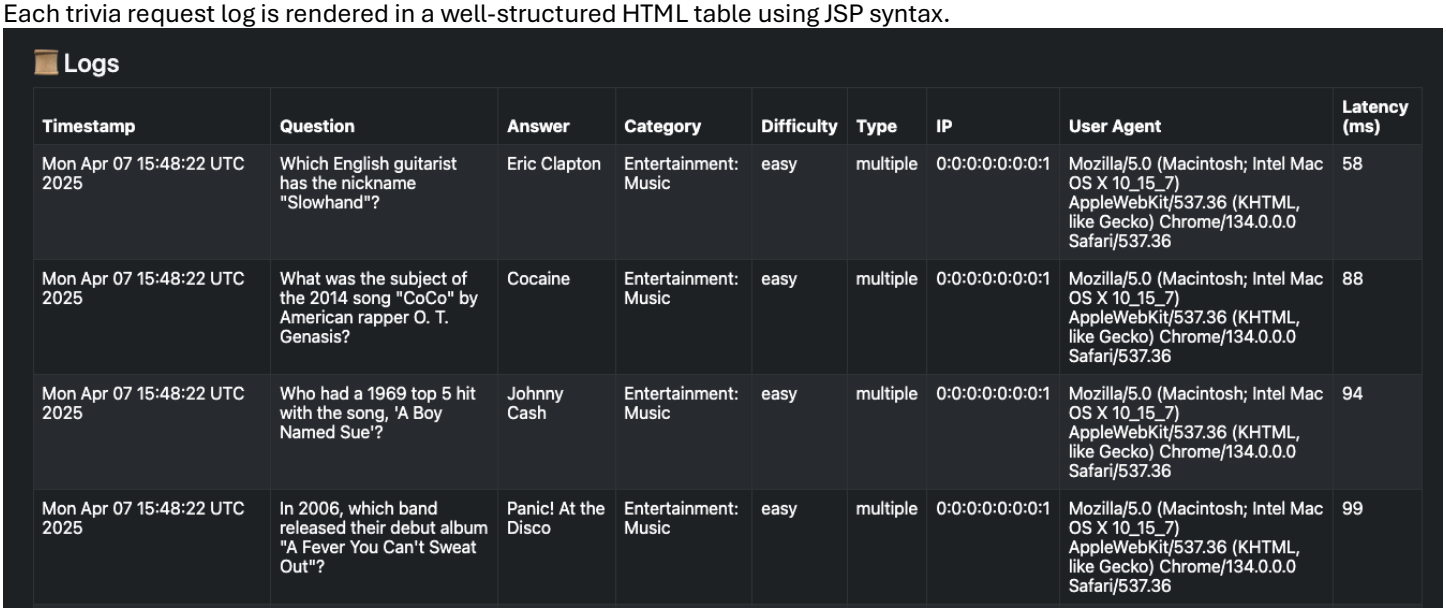
1. Total number of requests
2. Average server response time

3. Top requested trivia category

These are displayed at the top of the dashboard page.

c. The Dashboard displays formatted full logs

Each trivia request log is rendered in a well-structured HTML table using JSP syntax.



Logs

Timestamp	Question	Answer	Category	Difficulty	Type	IP	User Agent	Latency (ms)
Mon Apr 07 15:48:22 UTC 2025	Which English guitarist has the nickname "Slowhand"?	Eric Clapton	Entertainment: Music	easy	multiple	0:0:0:0:0:0:0:1	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36	58
Mon Apr 07 15:48:22 UTC 2025	What was the subject of the 2014 song "CoCo" by American rapper O. T. Genasis?	Cocaine	Entertainment: Music	easy	multiple	0:0:0:0:0:0:0:1	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36	88
Mon Apr 07 15:48:22 UTC 2025	Who had a 1969 top 5 hit with the song, 'A Boy Named Sue'?	Johnny Cash	Entertainment: Music	easy	multiple	0:0:0:0:0:0:0:1	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36	94
Mon Apr 07 15:48:22 UTC 2025	In 2006, which band released their debut album "A Fever You Can't Sweat Out"?	Panic! At the Disco	Entertainment: Music	easy	multiple	0:0:0:0:0:0:0:1	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36	99

7. Deploy the web service to GitHub Codespaces

I successfully deployed my servlet-based Trivia web service to the cloud using GitHub Codespaces. Below are the steps I completed to meet the deployment requirements.

Deployment URL: https://upgraded-engine-6659jqvq9j3r7q7-8080.app.github.dev

Endpoints:

- /trivia → for mobile clients (JSON API)
- /dashboard → for desktop browsers (JSP-based analytics)