**Description:**

Our application takes search parameters of category, number of questions, difficulty and type of question from the user and uses it to retrieve questions from the Open Trivia Database API.

Here is how our application meets the task requirements

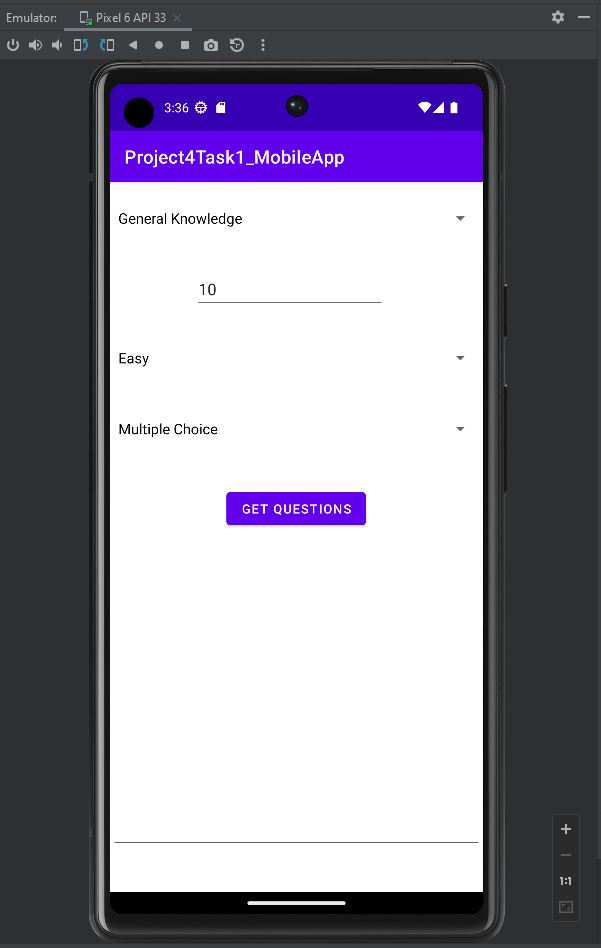
1. Implement a native Android application

The name of the native Android application project in Android Studio is: *Project4Task1\_MobileApp*

1. Has at least three kinds of views in our Layout:

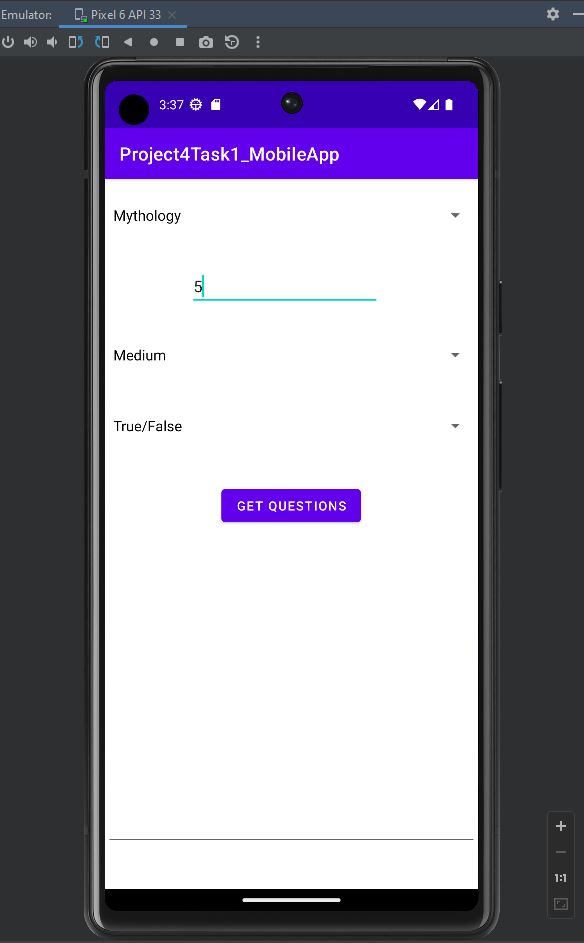
Our application uses MultilineText, EditText, Button and Spinner. See content\_main.xml for details of how they are incorporated.

Here is a screenshot of the layout (with default input values) prior to the questions being retrieved:



1. Requires input from the user

Here is a screenshot of the modified inputs from the user:



1. Makes an HTTP request (using an appropriate HTTP method) to our web service

Our application does HTTP GET requests from GetCategories.java and GetQuestions.java. The HTTP request for GetCategories.java is:

http://10.0.2.2:9090/Project4Task2-1.0-SNAPSHOT/getCategory

which requires no parameters and returns all categories tracked by the Open Trivia Database in order to populate the dropdown menu for the user to select a category (\*note: This portion of our app was modified to fill in from a manually input ArrayList since we were having issues with the GET request not completing prior to the layout loading once deployed to Heroku. We still meet the necessary HTTP request requirement while retrieving the questions). The HTTP request for GetQuestions.java is:

https://limitless-plains-67143.herokuapp.com/getQuestions?amount="+amount+"&category="+  
 category+"&difficulty="+difficulty+"&type="+type

where the parameters are the user’s requested number of questions, category, difficulty and type of question.

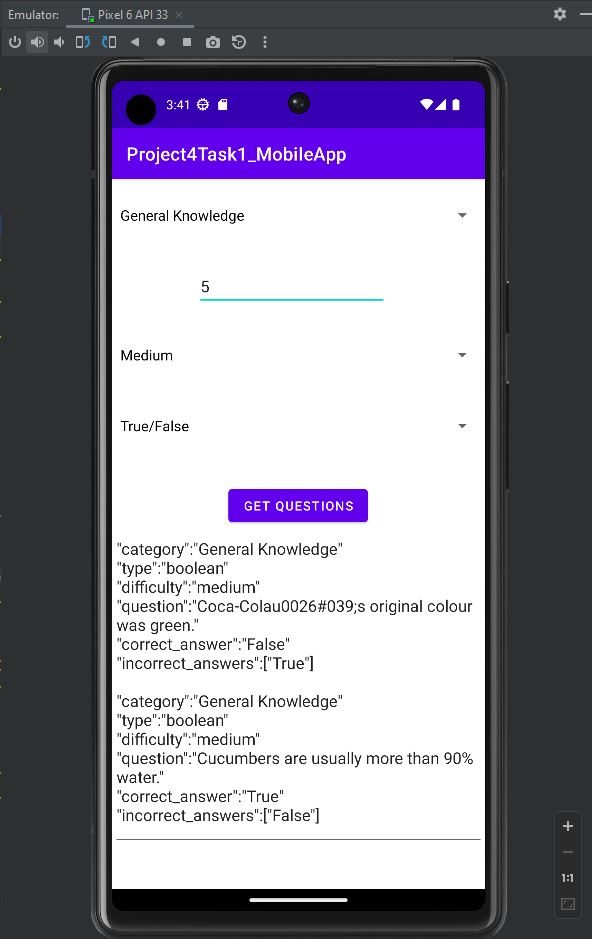
1. Receives and parses an XML or JSON formatted reply from the web service

An example of the JSON reply is:

*{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"What geometric shape is generally used for stop signs?","correct\_answer":"Octagon","incorrect\_answers":["Hexagon","Circle","Triangle"]},{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"What is the shape of the toy invented by Hungarian professor Ernu0151 Rubik?","correct\_answer":"Cube","incorrect\_answers":["Sphere","Cylinder","Pyramid"]},{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"What machine element is located in the center of fidget spinners?","correct\_answer":"Bearings","incorrect\_answers":["Axles","Gears","Belts"]},{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"How many furlongs are there in a mile?","correct\_answer":"Eight","incorrect\_answers":["Two","Four","Six"]},{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"Earth is located in which galaxy?","correct\_answer":"The Milky Way Galaxy","incorrect\_answers":["The Mars Galaxy","The Galaxy Note","The Black Hole"]}]}""*

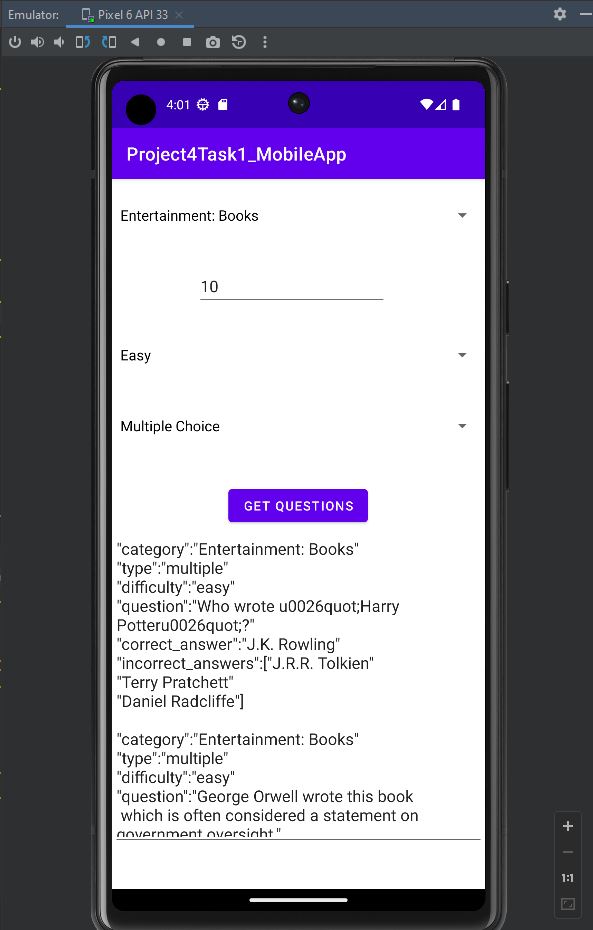
1. Displays new information to the user

Here is the screenshot after the questions have been returned to the user:



1. Is repeatable (the user can repeatedly reuse the application without restarting it)

The user can change the inputs and hit “Get Questions” to immediately get new questions. Here is an example after changing input parameters:



1. Implement a web application, deployed to Heroku

The URL of our web service deployed to Heroku is:

http://limitless-plains-67143.herokuapp.com/

The project directory name is *elliottc\_ericryu\_Project4*

1. Using an HttpServlet to implement a simple (can be single path) API

In our web app project:

Model: *TriviaModel.java*

View: *index.jsp and trivia-app-dashboard.jsp*

Controller: *CategoryServlet.java and TriviaServlet.java*

1. Receives an HTTP request from the native Android application

*CategoryServlet.java* receives the HTTP GET request for categories and *TriviaServlet.java* receives the HTTP GET request with the arguments “amount, category, difficulty and type” to retrieve the questions. It passes these parameters on to the model.

1. Executes business logic appropriate to your application

*TriviaModel.java* makes an HTTP request to:

*https://opentdb.com/api\_category.php*

and also makes an HTTP GET request to:

*https://opentdb.com/api.php?amount=****amount****&category=****category****&difficulty=****difficulty****&type=****type***

and returns the response to the mobile application in a simple JSON format:

*{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"What geometric shape is generally used for stop signs?","correct\_answer":"Octagon","incorrect\_answers":["Hexagon","Circle","Triangle"]},{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"What is the shape of the toy invented by Hungarian professor Ernu0151 Rubik?","correct\_answer":"Cube","incorrect\_answers":["Sphere","Cylinder","Pyramid"]},{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"What machine element is located in the center of fidget spinners?","correct\_answer":"Bearings","incorrect\_answers":["Axles","Gears","Belts"]},{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"How many furlongs are there in a mile?","correct\_answer":"Eight","incorrect\_answers":["Two","Four","Six"]},{"category":"General Knowledge","type":"multiple","difficulty":"easy","question":"Earth is located in which galaxy?","correct\_answer":"The Milky Way Galaxy","incorrect\_answers":["The Mars Galaxy","The Galaxy Note","The Black Hole"]}]}""*

**To document the rest of the requirements:**

1. Handle error conditions – Does not need to be documented.
2. Log useful information – Itemize what information you log and why you chose it.

Information is logged on the Trivia Web App Dashboard. The logged information displays the Category, Number of Questions Requested, Difficulty of Questions, Type of Questions, Time Taken to Reply with Questions and the App Connection Type.

1. Store the log information in a database – Give your Atlas connection string with the three shards

Connection String from Atlas:

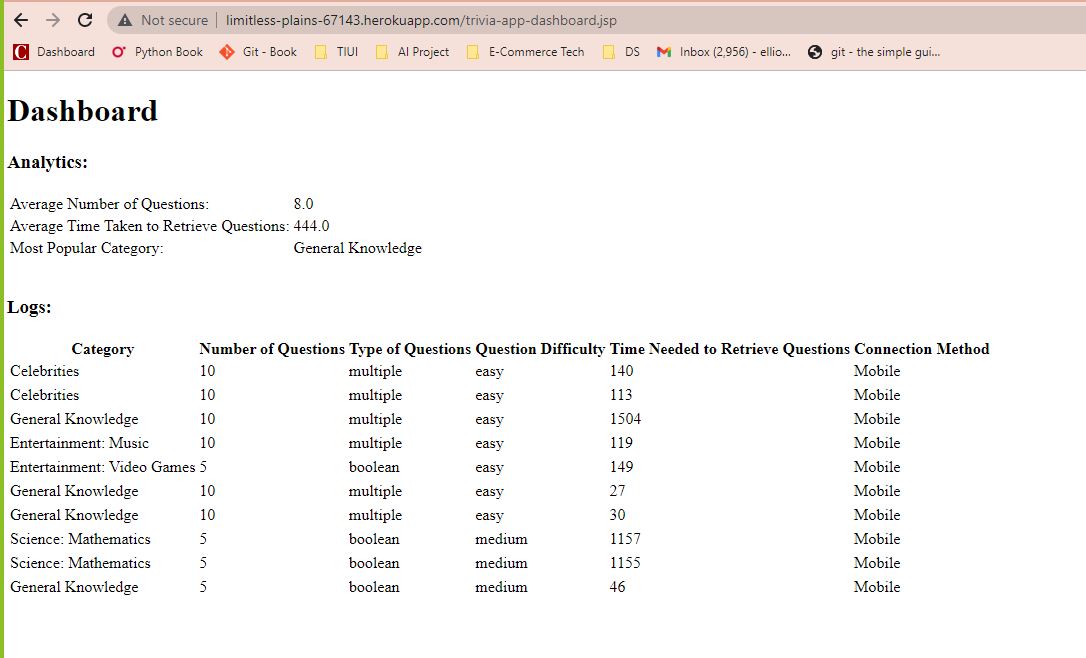
|  |
| --- |
| ConnectionString connectionString = new ConnectionString("mongodb+srv://user1234:<password>@cluster0.ywll23i.mongodb.net/?retryWrites=true&w=majority"); |
| MongoClientSettings settings = MongoClientSettings.builder() |
| .applyConnectionString(connectionString) |
| .serverApi(ServerApi.builder() |
| .version(ServerApiVersion.V1) |
| .build()) |
| .build(); |
| MongoClient mongoClient = MongoClients.create(settings); |
| MongoDatabase database = mongoClient.getDatabase("Project4"); |

Connection String from App:

ConnectionString connectionString = new ConnectionString("mongodb+srv://user1234:password1234@cluster0.ywll23i.mongodb.net/?retryWrites=true&w=majority");  
MongoClientSettings settings = MongoClientSettings.*builder*()  
 .applyConnectionString(connectionString)  
 .serverApi( ServerApi.*builder*()  
 .version( ServerApiVersion.*V1*)  
 .build())  
 .build();  
MongoClient mongoClient = MongoClients.*create*(settings);  
database = mongoClient.getDatabase("Project4");

All Shards:

ac-tjgwd0e-shard-00-00.ywll23i.mongodb.net:27017, ac-tjgwd0e-shard-00-01.ywll23i.mongodb.net:27017, ac-tjgwd0e-shard-00-02.ywll23i.mongodb.net:27017

1. Display operations analytics and full logs on a web-based dashboard – Provide a screenshot.