

Code Busters
Project 02: Trivia Game
Dr. C



Trivia Game is a game where users will get asked random questions, users will have to select the correct answer to obtain points. The points will be collected and each player that logs in will have tracking history to keep track of one's high scores. The administrator will be able to edit the questions.

Github: <https://github.com/FlavioCervantes/Dev-p2-android-application>

YouTube:  Project2 Code Busters

Table of contents

Completed Rubric Items	2
Initial Layout	3
Entity Relationship Diagram	4
Team members	5
Flavio Cervantes	5
Raquel Zuelke	5
Monica Um	5
Jeremy Jun	5
User Stories	6

Completed Rubric Items

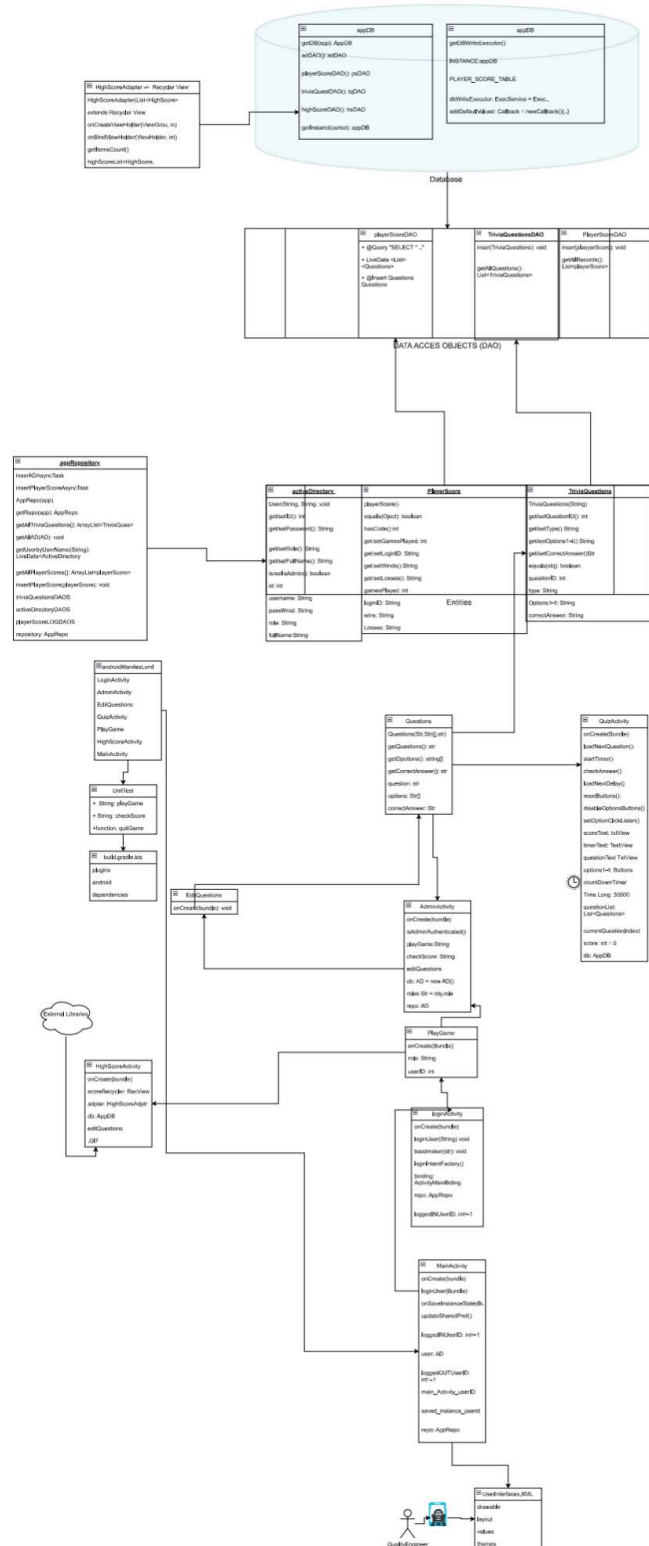
Please **Highlight** the items you have completed

Item	The following are Required	Score
Persistence	Minimum 3 tables implemented using the Room database wrapper If you do not use the Room Database wrapper the project will be marked as 0.	30
Github issues	There must be at least 5 GitHub issues PER TEAMMATE	10
Presentation	Each application must have a video showing its functionality. The video must be between 6 – 8 minutes long. Each person must present their portion for 1-2 minutes	35
Highlighted Rubric	This rubric must be included in the video AND the submission. All completed tasks must be highlighted	5
Activities	Each team member must create 2-3 activities, one of which must interact with the database.	12
Activity: Login Page	An activity where a username and password are entered. The username and password must be stored in the local room database.	4
Activity: Landing Page	After successful login an activity should start that displays the username and some other information retrieved from the local database.	4
Admin landing page	If the user is an administrator the landing page must indicate this with additional options. The same Activity may be used for both admin and non-admin users.	4
Github	Code must be pushed to Github in a public repository. Each teammate must have at least 3 branches, with 5 commits EACH; each branch must be merged into main using a pull request.	10
Unit tests	Each teammate must contribute at LEAST 2 unit tests. The tests must pass.	20

Optional Item	The following are Optional	Score
Intent Factories	All Intents to start an activity must use the factory pattern Either each activity must have an Intent factory method OR use a Factory class.	10
Intent Tests	At least 1 unit test per activity transition. That is each intent must have a unit test.	20
Database Tests	Each database table must have at least <u>three tests</u> (insert, update, delete)	20
Recycler View	At least 1 activity has a <u>recycler view</u> for displaying information	10
Livedata	Incorporate <u>liveData</u> in your project. This dovetails nicely with adding a recycler view	10
API Integration	Use <u>Retrofit</u> to consume an external API	20
External Library	Use an external Library. <u>Here</u> is a list of examples. <u>Retrofit</u> is included and would count for this.	16
Fragments	Incorporate <u>fragments</u> into your application	8
OAuth	Implement login with OAuth 2. This can replace the login procedure described above	16

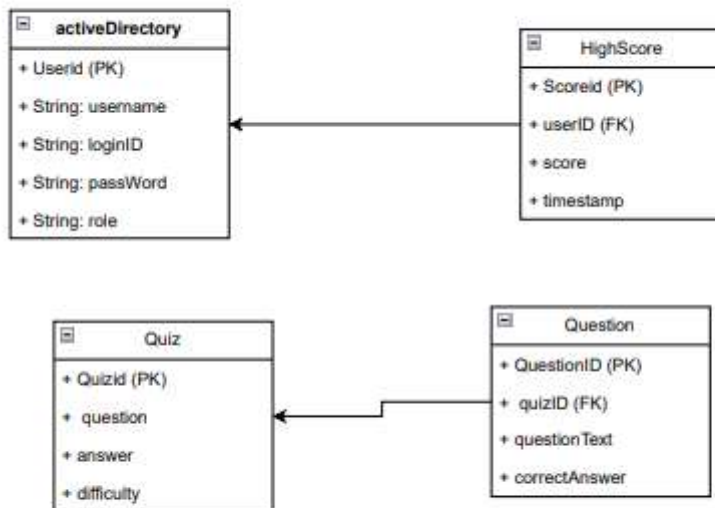
Layout v2 - Database Implementations

<https://github.com/FlavioCervantes/Dev-p2-android-application/wiki>



Trivia Game - ERD

<https://github.com/FlavioCervantes/Dev-p2-android-application/wiki>



Team members

Monica Um - Full stack Developer

Raquel Z - Full stack Developer

Jeremy Jun - Frontend Developer

Flavio Cervantes - Project Mgr./ Full stack Developer/QA Engineer

Flavio Cervantes

1. Project Manager / Backend Developer:

Responsibilities:

- ☒ create a **public** github repository and invite the rest of the team to contribute
- ☒ Setting up the project structure in Android Studio.
- ☒ Managing the project timeline and coordinating between team members.
- ☒ Developing the backend server (appDatabase)
- ☒ Diagramming the basic logic for handling quizzes, questions, answers, etc.
- ☒ Setting up the database for storing quiz data and user information.
- ☒ at least one branch merged into main (MainActivity-Debugging-02)
- ☒ The Database should be functional
- ☒ Unit tests - must contribute at LEAST 2 unit tests. The tests must pass.
- ☒ Database tests - contributed 3 tests that passed.
- ☒ Created XML UI files - activity_main, admin_ui, playe_game, timer_scoreboard, user_ui
- ☒ Created Trivia Game logo
- ☐

Jeremy Jun

- ☐ **2. Frontend Developer (Admin Interface):**
- ☐ Responsibilities:
- ☐ Designing and developing the Admin Interface for publishing questions.
- ☐ Ensuring the **Admin Interface communicates effectively with the backend database.**
- ☐ Implementing features for question creation, management, and real-time publishing.
- ☐ Handling admin user authentication and authorization.
- ☐ at least one branch merged into main
- ☐ Unit tests - must contribute at LEAST 2 unit tests. The tests must pass.

Monica Um

☐ 3. Full stack Developer:

☐ Responsibilities:

- ☒ Designing and developing the Player Interface (Android app) for players to participate in quizzes.
- ☒ Implementing the UI/UX for the player experience, including question display and answer submission.
- ☒ Integrating the player app with the backend dtb.
- ☒ At least one branch merged into main
- ☒ Unit tests – must contribute at LEAST 2 unit tests. The tests must pass.

Raquel Zuelke

☐ 4. Full stack Developer / Deployment Specialist:

☐ Responsibilities:

- ☒ Testing the app for bugs and ensuring all functionalities work as expected.
- ☒ At least one branch merged into main
- ☒ Performing unit tests, integration tests, and user acceptance tests.
- ☒ Video editing, will compile all videos together and ensure seamless integration.
- ☒ Handling the submission process and ensuring compliance with guidelines.
- ☒ Addressing any issues that arise.
- ☒ The above mentioned video
 - ☒ Youtube links are preferred but you may also submit a video file saved to a Google drive
- ☐ A .pdf with all of the required items
- ☒ Unit tests – must contribute at LEAST 2 unit tests. The tests must pass.

User Stories

[A User story tells us what the user wants to accomplish. It is a description of what the user will do, not necessarily how they will do it.]

Some example user stories:

- As a triviaGameUser I want to be able to log in to the application and play the game and view my highscores.
- As a triviaGameUser I want to be able to logout
- As a triviaGame Admin I want to be able to play the game, view the scores, and edit questions
- As a triviaGame Admin I want to be able to logout
- As a triviaGameUser and as a triviaGameAdmin I want to be able to quit the game if needed.
- As a triviaGameUser and as a triviaGame Admin I want to be able to see my score when playing the game
- As a triviaGameUser and a triviaGame Admin I want to be able to see the time I have left to answer my question when playing the game
- As a triviaGameUser and as a triviaGame Admin I want to be able to see if I got the answer right by showing what I clicked as green or red. Green meaning I got the question right and red meaning I got the question wrong.

From these use cases we generate the following issues:

- Implement a Room database
 - Add the Gradle requirement
 - Setup the Asynchronous method to call the database
- Create a Repository class to access the database
- Create a TriviaGame entity object
 - It should store: Name of the game, question, correct answer, and the score of the player once they quit or the timer ends.
 - Create a Room DAO for the TriviaGame entity
 - Create methods in the DataRepository to access the TriviaGame records
- Create a User Entity object
 - Required fields: username, password, isAdmin
 - Create a Room DAO for the User entity
 - Create methods in the DataRepository to access the User score, Trivia Questions, Trivia Answers.