Comp 319 Programming Assignment #3
GameOS+Firebase
Instructor: Seyhan Ucar

Deadline: 3 May, 23:59

1 Definition

In the first and second assignment, you developed a Quiz and Flag Game Android applications. In these assignments, you used the *res* folder or *java* classes to keep the quiz questions and flags. In this assignment, you are required to change this into a Firebase integrated version. Firebase gives you the tools and infrastructure you need to build better apps and grow successful businesses. In this assignment, you need to integrate the GameOS Android application with Firebase. The specification of GameOS+Firebase is as follows;

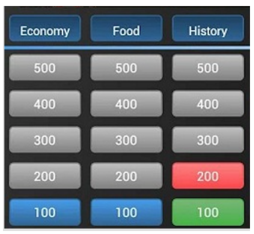

Figure 1: QuickQuiz Game

Figure 1 represents the implemented QuickQuiz (portrait mode) game where the user presses a question and a question with four possible answers is shown on the screen. In this assignment, you need to put these questions and answer into Firebase Realtime database and when a user presses a question, question and four possible answers are retrieved from Firebase. To achieve this, you need to use the Firebase Android library. When you create a request for Firebase, Android initiates an asynchronous call where the request needs time to complete. In this request and response time, you need to show a loading animation not to irritate the users. You can use the loader library in AndroidLoader

Figure 2, on the other hand, demonstrates the MemoGame (portrait mode) where a user can train his/her short-term memory and concentration. MemoGame has flag pool where you upload flags into Android *res* folder. In this assignment, you need to upload these flags into Firebase Storage and use them from there. As in QuickQuiz, not to irritate the user, you need to show a loading animation while the games retrieving the flags from Firebase.

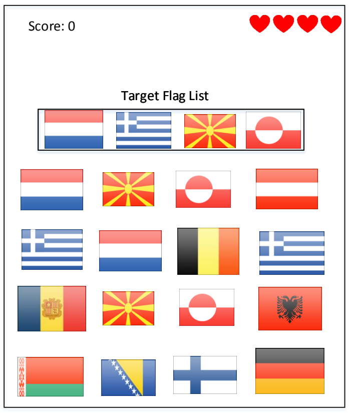

Figure 2: Memo Game

Moreover, GameOS needs to hold a configuration related to the flags in the pool. You can think this configuration as metadata related to available flags and you need to use this configuration while initiating board in MemoGame. For example, in the square board 4x4, there exist 16 boxes where 8 of them contains 4 different flag pairs. Other boxes contain random flags that are different than the target flag list. Flags that need to placed in boxes must be determined based on the configuration without downloading all flags from Firebase Storage. **In the third assignment, the MemoGame logic has some changes that are different than the second assignment**. When a user selects MemoGame, the board size will be selected by user apart from the alternatives 4x4, 5x5 and 6x6. After board selection, the game initiates itself and when the user finds all flags in target flag list, the game is completed and collected points is shown to the user. While the user is playing the MemoGame, another fragment that is placed inside the parent fragments shows the time that counts up from the beginning. After completion, MemoGame returns back to board size selection phase.

2 GameOS+Firebase Challenging to Friend

GameOS will start with a login/signup page where the user can either login into the system (already have an account) or create an account by signup page. In signup page, the user creates the account by providing username, password, name, surname and city information. User information is stored in Firebase. (You need to use Email/Password authentication method from Firebase). When a user logs in/signs up, the GameOS is represented. The user can play either QuickQuiz or MemoGame individually or challenge to a friend. GameOS provides a page where the user searches other users by using the information that is provided in signup phase. When a user finds a friend, a friend request is sent as Push Notification to other user. When the user accepts this friend request, users become a friend and they can challenge to each other both in QuickQuiz and MemoGame.

GameOS provides challenging functionality between friends. A user, the challenger, can challenge to friends in completing the QuickQuiz and MemoGame. The challenging starts with selecting a friend, the challenged user, from friend list and the game that challenger challenges. If QuickQuiz game is selected then the random category is chosen and questions start with 100 points to 500 points are asked to the challenger in order. The given answers are recorded. The randomly selected category and correlated questions are asked to challenged user in the same manner and given answers are also recorded. The winner is determined based on given answers. Challenged user can see the winner as long as he/she completed the game. The challenger, on the other hand, needs to get a push notification about the results. If the MemoGame is selected then challenger selects the board size and the MemoGame is initialized with random flags as you did in the second assignment. After challenger completes the MemoGame, the challenged user gets a push notification. When the challenged user pushes the received push notification, the MemoGame is initialized with the same configuration as the challenger completes. Challenged user completes the MemoGame and the winner is determined based on collected points and time duration that passed while completing the game.

The GameOS+Firebase has the following properties;

- The QuickQuiz needs to obey the specifications that are pointed in the first assignment. If you fail to complete any features, correct this features before starting to the third assignment.
- The MemoGame needs to obey the specifications that are pointed in the second assignment. Moreover, you need to modify the logic as pointed out in the third assignment. Apart from this, if you fail to implement any features of MemoGame in the second assignment (for example: crash in tablet version), pay attention to correct before starting the third assignment.
- In QuickQuiz, the categories, questions and answers need to be stored in Firebase. When a user selects the game QuickQuiz, you need to retrieve this information from Firebase. Pay attention, Firebase works asynchronously, where you need to put a loader on screen.
- The flags that are available on MemoGame need to be uploaded Firebase Storage. You need to hold a configuration on Firebase and initiates game via the usage of this configuration instead of downloading all images.
- Use SQLite to keep the Firebase retrieved information on Android. So, even the internet is missing, users can still play the QuickQuiz or MemoGame. If you know libraries to handle the SQLite workload such as Android SimpleNoSQL, you are allowed to use them.
- You need to use the Firebase push notification mechanism to implement the challenging part of the third assignment. Despite you can send push notification from Android device to another Android device, it is not recommended due to security reasons. To implement the challenging part, you need to write a Firebase Cloud Functions. Cloud Functions for Firebase lets you run mobile backend code that automatically responds to events triggered by Firebase features and HTTPS requests.
- GameOS+Firebase uses the Navigation Drawer and provides pages in navigation as follows;

1. QuickQuiz where the quiz game is initiated and represented to the user.
2. MemoGame where the memory game board selection phase is presented.
3. Profile where the user profile information is demonstrated with edit capabilities.
4. Friend list where the user sees the friend list and searches other users. Search needs to be in action bar.
5. Top ranking where the user can see the top rankings of MemoGame and QuickQuiz from the GameOS users.

- GameOS+Firebase needs to work on both phone (in portrait mode) and on a tablet (in landscape mode).

3 Bonus (20 points)

The challenging in GameOS+Firebase can be seen as offline where even challenged user is not online, the challenger can still play the round. Challenger's answers are recorded and challenged user will be able to complete the challenge later (pushing the notification). Challenger will be notified when challenged user finishes the challenge and he/she can then view the game results. In bonus part, you need to change this offline challenging into online where if both challenger and challenged user are online, they can play against each other in real time. The bonus requirements are not defined and the design structure depends on you. You are allowed to design the bonus freely as long as the structure is reasonable. (For example, to detect the online users, polling (periodically querying the Firebase) is naive solution but it is not a good design)

4 Requirements

The application needs to be designed in object oriented manner. **Students are expected to be aware of the underlying class model and the relation of classes**. The application has to satisfy the defined objectives and properties. Each of the objectives and properties should be tested before submission. The design guidelines are as follows;

1. You need to integrate the Firebase into GameOS both in QuickQuiz and MemoGame. The information used in both QuickQuiz and MemoGame needs to be retrieved from Firebase. Without this features, the third assignment is not counted as a valid assignment.

2. Firebase uses an asynchronous callback to complete network requests. Not to irritate users, use loading animation in GameOS. Moreover, you need to check the internet connection before performing any Firebase request.

3. GameOS needs to use the SQLite to keep the Firebase retrieved information. Even the internet is missing, your applications can be playable via the usage of SQLite data. You need to handle the SQLite related workload on your application.

4. GameOS has some additional pages and some modifications that are different than the first and second assignment. Pay attention to these changes and additional pages and ensure that all are checked before submission.

5. GameOS provides the functionality of challenging. Use Firebase Cloud functions to implement this feature. Check the challenging functionality that should work properly before submission.

6. Use Navigation Drawers for navigation in GameOS. GameOS needs to work on both phone and tablet with the usage of fragments.

5 Submission

Students need to submit their source code and screenshot of the working demo to Blackboard website. The missing or corrupted files, nonworking demos are not accepted. Students are expected to explain the code details or show working demo when it is requested. The challenging requires a real device to test the Firebase Push Notification. Test your application on real Android devices before submission.

6 Demo and Grading

In grading, each group or student needs to demonstrate the working copy of the code with the submitted files. Each defined objectives and properties is evaluated separately. **To test the challenging functionality, at least two real Android devices are required. Students are responsible for bringing two Android devices into demo.** If they cannot, please inform the TAs beforehand. Students will be asked a different kind of questions related to the application, class model and application structure. Group members can take different points based on the evaluation. The doodle poll for demo session will be announced by TAs.