You are an expert Android developer. Please help me implement a complete flow for handling quizzes in my Android app (Java). My app allows both \*\*guest users\*\* (not authenticated) and \*\*logged-in users\*\* (authenticated via Firebase Authentication).

Please write the necessary code and logic for the following requirements:

1. When a guest user accesses a quiz, they should be able to:

- Fetch and view the quiz from Firestore.

- Answer the quiz questions.

- View their result at the end.

- All quiz answers and results should be stored \*\*locally only\*\* (in memory or in a local SQLite/Room database).

- No data should be sent to Firestore.

2. When a \*\*logged-in user\*\* accesses a quiz, they should:

- Fetch and view the quiz from Firestore.

- Answer the quiz questions.

- View their result at the end.

- All answers should be stored locally first.

- When the app detects that the user is online, all locally stored answers should automatically be uploaded to Firestore.

3. The app must handle offline mode gracefully:

- If the user is logged in but offline, they should still be able to answer quizzes and their data should be saved locally.

- When internet connectivity is restored, all locally saved data should be uploaded to Firestore.

4. Include a \*\*QuizManager class\*\* that handles:

- Checking if the user is authenticated.

- Deciding whether to save data locally only (for guests) or both locally and in Firestore (for logged-in users).

- A method to sync all local data to Firestore when the device is back online.

5. Use a combination of \*\*Room\*\* (for local persistence), \*\*Firebase Firestore\*\* (for cloud storage), and a \*\*ConnectivityListener\*\* to detect internet status changes.

6. Write all code in Java. Structure it to fit a standard MVVM architecture with a ViewModel, Repository, and Data Layer (Room + Firestore).

7. Ensure the logic supports \*\*automatic background sync\*\* when connectivity is restored, but only for authenticated users.

8. Add relevant comments to explain the logic.

Please generate a clear and structured example that follows modern Android development practices.