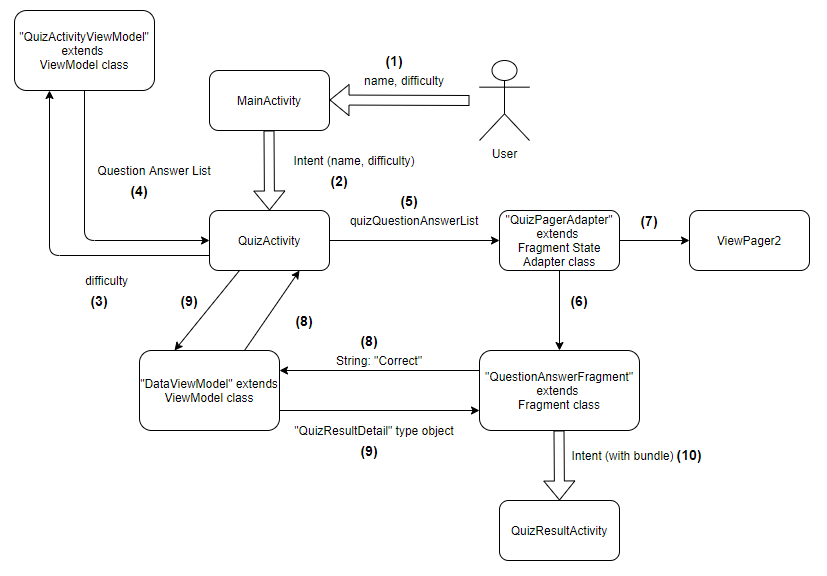
# **Block Diagram**



# **Process**

1. User enters the detail on the first screen and press “Start Quiz” button.
2. MainActivity sends the detail to the QuizActivity through intent and bundle. QuizActivity extracts the difficulty from intent bundle.
3. QuizActivity sends the difficulty extracted from bundle to the QuizActivityViewModel.
4. QuizActivityViewModel calls the API using difficulty. It returns the list back to the QuizActivity. The ViewModel does not get destroyed when the activity gets destroyed temporarily due to screen rotation and it keeps the data as it was before the activity being destroyed.
5. QuizActivity sends the received list to the QuizPagerAdapter class.
6. QuizPagerAdapter class extends FragmentStateAdapter class which creates the fragment and returns it back to ViewPager2. The FragmentStateAdapter destroys the fragments when the user goes far from the particular fragments. If the user wants to see the destroyed fragments, then FragmentStateAdapter recreates them for user to view. FragmentStateAdapter uses the bundle to retain the state of the user page. State in our game means the user answered questions.
7. QuizActivity has the QuizPagerAdapter set which sends the fragment pages to ViewPager2. ViewPager2 hosts the pages from QuizPagerAdapter with the condition of hosting only one page at a time. User can swipe right to left to see new fragment page or left to right to see the old fragment pages.
8. When the user presses the “Submit” button to lock its answer and if the answer is correct, then a string of “Correct” is send from the QuestionAnswerFragment class to the QuizActivity using DataViewModel. The QuizActivity increases the score and set it to the TextView in the QuizActivity.
9. On last page, the QuizActivity using the ViewPager2 property sends the user details along with the score to the last fragment
10. When the user presses the “Finish Quiz” button on the last page, the fragment sends the user details in a bundle to the QuizResultActivity. The QuizResultActivity displays the result on screen.

# **Advantages of ViewPager2**

1. ViewPager2 provides vertical orientation support and Right-to-left orientation support.
2. When a fragment is modified, the notifyDatasetChanged() call updates the UI. ViewPager2 supports it by displaying the modified fragments.
3. ViewPager2 offers many methods to access the fragment when using the FragmentStateAdapter for example onPageChangeCallback() method.
4. ViewPager2 is built on RecyclerView. The difference here is that ViewPager2 can show one page (item) at a time whereas a Recycler view shows more than one page (item) at a time. A RecyclerView can be made to show one page/item at a time but with few more lines of code as compared to ViewPager2.
5. ViewPager2 uses FragmentStateAdapter. FragmentStateAdapter creates and supplies instances of new fragment each time it is used.

# **Problems faced during ViewPager2 Implementation**

1. **Screen rotation**: when rotating the screen, the activity (with fragments in my game) temporarily gets destroyed and recreated due to configurations change. When the activity is recreated, if the API call is in the activity, then it would call the API again which will give a new list of question answers. I was able to overcome this issue with the use of ViewModel whose lifecycle remains unaffected with activity being destroyed and recreated. I called the API in the ViewModel which provides the same data back to the activity once it is recreated.

**Maintaining state**: the fragment state adapter destroys the fragment when the user goes far from the fragment and it recreates the fragment when the user comes back to see it. The challenge for me was to maintain the state which in my game is user selected option (correct/incorrect) and the correct answer option.

# **References**

Android Developers. (n.d.). *Slide between fragments using ViewPager2*. https://developer.android.com/training/animation/screen-slide-2

Android Developers. (n.d.). *Migrate from ViewPager to ViewPager2*. https://developer.android.com/training/animation/vp2-migration

Android Developers. (n.d.). *Create swipe views with tabs using ViewPager2*. https://developer.android.com/guide/navigation/navigation-swipe-view-2

Android Developers. (n.d.). *Handle configuration changes*. https://developer.android.com/guide/topics/resources/runtime-changes

Android Developers. (n.d.). *ViewModel Overview*. https://developer.android.com/topic/libraries/architecture/viewmodel

Android Developers. (n.d.). *Communicating with fragments*. https://developer.android.com/guide/fragments/communicate

Android Developers. (n.d.). *Android KTX*. https://developer.android.com/kotlin/ktx

Android Developers. (n.d.). *Saving state with fragments*. https://developer.android.com/guide/fragments/saving-state

Elye. (2020, June 19). *Is ViewPager2 just a RecyclerView?* https://medium.com/mobile-app-development-publication/is-viewpager2-just-a-recyclerview-5bd90559d4df

Jones, B. [Brandan Jones]. (2018, Jul 22). *Use Retrofit to parse JSON in Kotlin Android app* [Video]. YouTube. https://www.youtube.com/watch?v=FW7sY7M\_E8k

Pandey R. [Rahul Pandey]. (2019, Nov 03). *Using ViewModel to Fix Screen Rotation Issues - Android Kotlin example* [Video]. https://www.youtube.com/watch?v=foC438dehJY