Hello everyone, my name is Arya Dewa Wijaya, ok I will answer these five questions:

1.What project have you worked about?

In this project I am working on a habit application, which this application can save and delete habit schedule, displays data in the form of a list based on length of time, priority and name. This application can also perform a countdown and display a notification when the countdown is complete if the user want it. In addition, this application also has a feature to change the theme to dark mode.

2.Which part is hardest?

In my opinion, the tricky part is in the notification section.

Regarding work managers, I don't understand how work managers work,

but I re-learn about work manager in Android Fundamentals. So that I can do well

3.How is the flow of showing random habit using ViewPager2?

First we need to identified the viewpager inside layout with the id, after that

There is a RandomHabitAdapter that bridges between ViewPager2 and data. The data displayed is taken from the viewmodel which is taken randomly for each priority from the database room. The data will then be displayed with the help of an adapter where the parameters entered are PageType which will be submit according to priority, low priority, medium and high. Then the data will appear with the adapter in the form of a textview for title, countdown minutes, start time, and image view for priority.

4.How does notification reminder work?

simple explanation, the notification will appear when the user clicks the SwitchPreference to on mode and the countdown has finished then the notification will shows.

for details, theres

data builder that contain the tittle and id of data habit, also NOTIFICATION\_CHANNEL\_ID as identification of the data

After that theres viewmodel observes the countdown. if the countdown is complete then a workmanager will be created.

WorkManager is configured to show a notification once with OneTimeWorkRequest containing NOTIF\_UNIQUE\_WORK as the unique work name.

If the SwitchPreference to Reminder is on then create pending intent to show notification.

Theres also notification object using NotificationCompat.Builder which contain notification channel id. In this object I add configuration like the icon is notification icon, the title is habit title , the context is time up, sound, and I also passing in the pending intent that created before.

After that check the user's phone is Android version Oreo and above, because no need to create a notification channel for older versions.

Then last create the notification using the NotifcationManagerCompat, passing in the Notification channel ID and the notification Object

and that's how notifications work

5.Why do we need a ViewModel?

We need the ViewModel because we want the code to be more organized. Using ViewModel will place the code separately to other class and away from the View and Controller. This will make the code easier to maintain and more testable. With this we can use more than one model into one object, it will encapsulate data for ui controller to let data persist in case is there any configuration change.