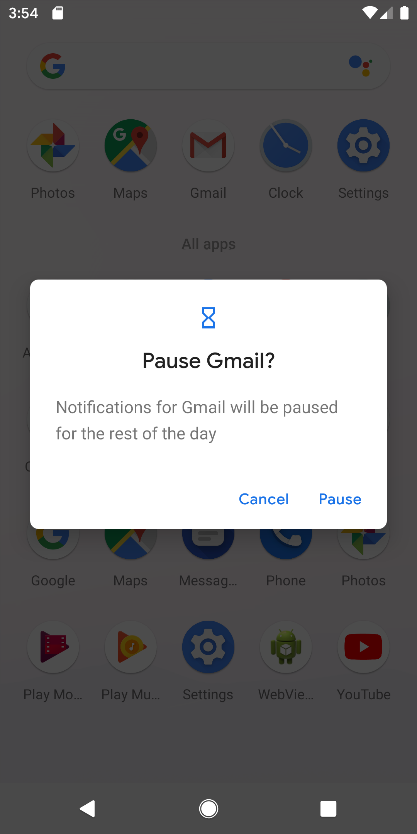
## Homework 3: State Management

**(5 Points)** The various states that an app can enter on your platform of choice

**The four states that an app on Android can enter are Active, Pause, Stop, and Destroy.**

1. Active – The active state is when the app is at the forefront and the user is actively working or using the app for their tasks or enjoyment
2. Pause – This state is when the app has stopped running due to a potentially pop up of high priority, like a call or a pop-up message. It doesn’t exit the app but puts it in a stasis until the pop up is resolved. A good example is the image below:



1. Stop – Stop is when you switch away from the app and go to another app. The best example is the app carousel in Android that shows how many apps your phone has currently open.
2. Destroy – Closing and exiting the app entirely, think like the Kill switch.

Compose uses keywords such remember and mutable state to have observable objects that will change with any given input. Remember saves the state with each change in the composable through the mutable state. There is also rememberSaveable and unlike remember, this helps save all values after any type configuration changes.

Text

Description automatically generated

Text

Description automatically generated

A good example of it is in the Google map code of the project. The position of the main user is saved but without creating a proper state and saving the location values through the use of remember, we will not be able to recompose the map to center on the main user as they move. CameraPositionState does have a remember key however it doesn’t work proper in the sense that it will only save the initial location and will not recompose with each change in location. By making the user position a remembered state, and having the cameraPosition be a saveable, this allows for recomposition where the map view will move with the user.

**(5 Points)** The various states that you must consider for your app, why you must consider it, and what must happen in each state.

The main states for the app to consider would be these three states:

1. “Tracking in progress” – During this phase the user will be just lead to the map screen if they already have a session open. The location of the user at this point is being updated in the back ground as they move.
2. “Not tracking” – If a session is not in progress then they are in the non-tracking state and will be staying there until a tracking session happens or they are waiting for consent
3. “Waiting for consent” – If the user has sent a request, they will be lead to this screen until at the point that the person they sent the request to either accepts or rejects the request. Accepting will lead them to the tracking state with the map screen and rejecting the request will send them back to the none tracking state.

Other types of state management would be:

1. The map needs to move with the user and not stay static in location
2. If the app crashes and the user is on a text field and filling out something, we want to be able to save the text field as is so that when they open the app again, the test field is the same
3. If the app crashes and the person was in a tracking session, the app should have saved the session details and gone back to the session like nothing happened. The only potential change would be the location of the user if they had moved since the app crash.
4. Changing the view of the app, either vertical or horizontal shouldn't change the test feilds or the state of the app