|  |  |  |
| --- | --- | --- |
| **Mod-3: Activity(1-11)** | **Start: 1:30 pm** | **Stop: 4:00 pm** |
| * An intent allows you to start an activity in another app by describing a simple action you'd like to perform in an Intent object. * With the Google Maps and Places APIs for Android, your app can be location-aware, include data-rich maps, find relevant places nearby, and more * Creating a method called showmap with a uri as the single parameter. * Replacing the toast message with a call to showmap. * As an application developer, you can use any media codec that is available on any Android-powered device, including those provided by the Android platform and those that are device-specific. * Creating a void method called share text that accepts a string as a parameter. * Setting the media type and title of the chooser. * Adding new empty activity to sunshine called detail activity. * Adding textview and sending the string. * Also adding a share weather details when clicked on the menu item. | | |

|  |  |  |
| --- | --- | --- |
| **Activity(12-16)** | **Start: 4:25 pm** | **Stop: 6:00 pm** |
| * As a user navigates through, out of, and back to your app, the Activity instances in your app transition through different states in their lifecycle. * The Activity class provides a number of callbacks that allow the activity to know that a state has changed: that the system is creating, stopping, or resuming an activity, or destroying the process in which the activity resides Within the lifecycle callback methods, you can declare how your activity behaves when the user leaves and re-enters the activity. * .Crashing if the user receives a phone call or switches to another app while using your app. * To navigate transitions between stages of the activity lifecycle, the Activity class provides a core set of six callbacks: onCreate(), onStart(), onResume(), onPause(), onStop(), and onDestroy(). * The system invokes each of these callbacks as an activity enters a new state. | | |

|  |  |  |
| --- | --- | --- |
| **Activity(17-25)** | **Start: 7:00 pm** | **Stop: 9:00 pm** |
| * Preserving and restoring an activity’s UI state in a timely fashion across system-initiated activity or application destruction is a crucial part of the user experience. * In these cases the user expects the UI state to remain the same, but the system destroys the activity and any state stored in it. * Android provides several options for you to save your app data. * The solution you choose depends on your specific needs, such as how much space your data requires, what kind of data you need to store, and whether the data should be private to your app or accessible to other apps and the user. * A cached process is one that is not currently needed, so the system is free to kill it as desired when memory is needed elsewhere. * In a normally behaving system, these are the only processes involved in memory management: a well running system will have multiple cached processes always available and regularly kill the oldest ones as needed. * Only in very critical situations will the system get to a point where all cached processes are killed and it must start killing service processes. | | |

|  |
| --- |
| **Deliverable Status** |

**Module-3:**

<https://github.com/manojbandari/Android/tree/master/Module%203>