**CIT 3350: MOBILE APPLICATION DEVELOPMENT**

**NAME: MUHUHU DAVID NDUNG’U**

**REG. NO.: CT203/0013/18**

**COURSE: BIT Y3S2**

1. **Discuss 4 main components of android application [4 marks]**

* Activities - They dictate the UI and handle the user interaction to the smart phone screen.
* Services - They handle background processing associated with an application.
* Broadcast Receivers - They handle communication between Android OS and applications.
* Content Providers - They handle data and database management issues.

1. **Describe the six steps of the mobile application development process. [6 marks]**

* Strategy and market research – This is where you outline the strategy for developing your idea into a successful app.
* Analysis and planning – Here the idea starts taking shape and turns into an actual project. Starts with defining use cases and capturing complete functional requirements. After you are well known with the app requirements, prepare a product roadmap that includes prioritizing the mobile app requirements and combining them into delivery milestones.
* UI/UX Design – The objective of this is to build excellent user experiences by making an app interactive, spontaneous and user friendly.
* Mobile app development – This is made up of 3 basic parts: back end/server technology, API, and the mobile app front-end.
* Testing and Quality Assurance (QA) – Executing systematic QA testing during the mobile app development process makes application stable, functional and secure.
* Deployment and support – Releasing a native mobile app involves submitting your app to the google play store for android apps and app store for IOS apps.

1. **Outline the code of manifest file as used in android applications and describe the key elements in the file. [4 marks]**

A default manifest file code is as follows:

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.example.tutorialspoint7.myapplication">

<application

android:allowBackup="true"

android:icon="@mipmap/ic\_launcher"

android:label="@string/app\_name"

android:supportsRtl="true"

android:theme="@style/AppTheme">

<activity android:name=".MainActivity">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

</application>

</manifest>

The key elements are as follows:

<manifest> - it has package attribute that describes the package of the activity class.

<application> - is the sub element of the manifest. It includes the namespace declaration. Common attributes are; icon, label, theme etc.

<activity> - is the sub element of application and represents an activity that must be defined in the AndroidManifest.xml file.

<action> - it an action for the intent-filter. The intent-filter must have at least one action element.

<category> - it adds a category name to an intent-filter.

1. **Briefly describe methods in the fragment life cycle in android application development. [4marks]**

**onAttach**() - you get in this method a reference to the activity which uses the fragment for further initialization work.

**onCreate**() - The system calls this method when creating the fragment.

**onCreateView**() - The system calls this callback when it's time for the fragment to draw its user interface for the first time. You can return null if the fragment does not provide a UI.

**onActivityCreated**() - is called after the onCreateView() method when the host activity is created. Activity and fragment instance have been created as well as the view hierarchy of the activity. At this point, view can be accessed with the findViewById() method.

**onStart**() - is called once the fragment gets visible.

**onResume**() - Fragment becomes active.

**onPause**() - The system calls this method as the first indication that the user is leaving the fragment. This is usually where you should commit any changes that should be persisted beyond the current user session.

**onStop**() - Fragment going to be stopped.

**onDestroyView**() - Fragment view will destroy after call this method.

**onDestroy**() = does final cleanup of the fragment's state but Not guaranteed to be called by the Android platform.

1. **Discuss parts of a GSM network. [4 marks]**

The GSM network contains of the following main subsystems:

* + 1. Mobile Station (MS) - Communicates across an air interface with a BTS in the same cell in which the mobile unit is located. The mobile equipment includes: Radio receiver; Digital signal processors; and Subscriber Identify Module (SIM)
    2. Base Station Subsystem (BSS) - This unit consists of a base station controller (BSC) and one or more base transceiver stations (BTS). The BSC reserves radio frequencies, manages the handoff of a mobile unit from one cell to another within the BSS, etc.
    3. Network and switching Subsystem (NSS) – NSS provides: The link between the cellular network and the PSTNs, It controls handoff between cells in different BSSs, Authenticates users and validates their accounts, Includes functions for worldwide roaming of mobile users. The central element of the NS is the mobile services switching Centre (MSC), but several databases are also part of the NSS: Home Location Register (HLR), Visitor Location Register (VLR), Equipment Identity Register (EIR), Authentication Centre etc.
    4. Operation and Support Subsystem (OSS) – is an element within the overall GSM mobile communications network architecture that is connected to components of the NSS and the BSC. It is used to control and monitor the overall GSM network and it is also used to control the traffic load of the BSS.

1. **Discuss all android layout types, using code examples [4 marks]**

**Linear Layout** – Linear layout is further divided into horizontal and vertical layout. It means it can arrange views in a single column or in a single row. Here is the code of linear layout:

<?xml version=”1.0” encoding=”utf-8”?>

<LinearLayout xmlns:android=”http://schemas.android.com/apk/res/android”

android:layout\_width=”fill\_parent”

android:layout\_height=”fill\_parent”

android:orientation=”vertical” >

<TextView

android:layout\_width=”fill\_parent”

android:layout\_height=”wrap\_content”

android:text=”@string/hello” />

</LinearLayout>

**Absolute Layout -** The Absolute Layout enables you to specify the exact location of its children. It can be declared like this:

<AbsoluteLayout

android:layout\_width=”fill\_parent”

android:layout\_height=”fill\_parent”

xmlns:android=”http://schemas.android.com/apk/res/android” >

<Button

android:layout\_width=”188dp”

android:layout\_height=”wrap\_content”

android:text=”Button”

android:layout\_x=”126px”

android:layout\_y=”361px” />

</AbsoluteLayout>

**Table Layout -** The Table Layout groups views into rows and columns. It can be declared like this:

<TableLayout

xmlns:android=”http://schemas.android.com/apk/res/android”

android:layout\_height=”fill\_parent”

android:layout\_width=”fill\_parent” >

<TableRow>

<TextView

android:text=”User Name:”

android:width =”120dp”

/>

<EditText

android:id=”@+id/txtUserName”

android:width=”200dp” />

</TableRow>

</TableLayout>

**Relative Layout -** The Relative Layout enables you to specify how child views are positioned relative to each other. It can be declared like this:

<RelativeLayout

android:id=”@+id/RLayout”

android:layout\_width=”fill\_parent”

android:layout\_height=”fill\_parent”

xmlns:android=”http://schemas.android.com/apk/res/android” >

</RelativeLayout>

**Frame Layout -** The Frame Layout is a placeholder on screen that you can use to display a single view. It can be declared like this:

<?xml version=”1.0” encoding=”utf-8”?>

<FrameLayout

android:layout\_width=”wrap\_content”

android:layout\_height=”wrap\_content”

android:layout\_alignLeft=”@+id/lblComments”

android:layout\_below=”@+id/lblComments”

android:layout\_centerHorizontal=”true” >

<ImageView

android:src = “@drawable/droid”

android:layout\_width=”wrap\_content”

android:layout\_height=”wrap\_content” />

</FrameLayout>

1. **Discuss all the layers of android software architecture [4 marks]**

**Linux Kernel -** This provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

**Libraries -** On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

**Android Runtime -** This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called Dalvik Virtual Machine which is a kind of Java Virtual Machine specially designed and optimized for Android.

**Application Framework** - layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

**Applications -** You will find all the Android application at the top layer. You will write your application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, Games etc.