AN ANDROID APP FOR BUILDING STUDENT PROFILES

Mallepally Bhanusree Reddy[1]

[1]*Student, Department of CSE, Sreyas Institute of Engineering and Technology, Telangana, India*

Abstract - *This paper aims to demonstrate the creation and evaluation of a mobile application for creating student profiles. Mobile technology is a rapidly expanding industry that is closely related to our daily lives and jobs. The plan is to develop an app for college students that allows them to display their profiles, which include interests, contact information, personal data, skills, and skill sets, to faculty and industry recruiters. Students can explain themselves much more quickly this way. With the help of this application, students can discuss other important topics with the faculty members by leaving their introduction information in the app. By entering their introduction details into the app, students can use this application to talk with faculty members about other crucial subjects.*

Key Words: Android Studio, Multiple screens, User-friendly Application, Personal Information.

1. INTRODUCTION

The majority of mobile platforms used around the world run the Android operating system. Android is the operating system for hundreds of millions of mobile devices in more than 190 different countries. It will hold roughly 75% of the global market share by the end of 2020, and this trend is only accelerating. Android, which was initially developed by the business known as the Open Handset Alliance, was created by modifying the Linux kernel and other open-source software. Before buying the entire company in 2005, Google provided funding for the project. In September 2008, the first device with the Android operating system was made available for purchase. Because of the extensive feature set it offers, Android has dominated the mobile OS market. user-friendly, has a sizable community, allows for greater customization, and a lot of manufacturers produce Android-compatible smartphones. user-friendly, has strong community support, allows for greater customization, and many businesses produce Android-compatible smartphones.

1.1 Classifications

One or more of the following four types of applications can be found on Android devices:

- **Activities:** An activity is used to implement an application with a visible user interface. An activity begins when you choose an application from the home screen or application launcher.

- **Services:** Any application that must run continuously for a long period of time, like a network monitor or an update-checking program, can use a service.

- **Content providers:** Thinking of content providers as database servers is the simplest way to conceptualize them. Controlling access to persistent data, such as phone contacts, is the responsibility of a content provider. One might not need to create a content provider if the application is very straightforward. The recommended method of accessing the data is through a content provider, though, developing a larger application or one that makes data available to multiple activities and/or applications.

- **Receivers of broadcasts:** One can run an Android application to process a specific piece of data or react to an occasion, like receiving a text message.

2. INSTALLATION PROCESS

The IDE for building native Android apps is called Android Studio. It also contains the Android SDK, which must be set up before being used on the command line. The Android emulator requires the creation of Android virtual devices, which can be done using Android Studio. Ionic apps can also be launched on a device.

2.1 Installing the Android Studio

The Android Studio website offers a download for Android Studio.

Download Android Studio and SDK tools | Android Developers

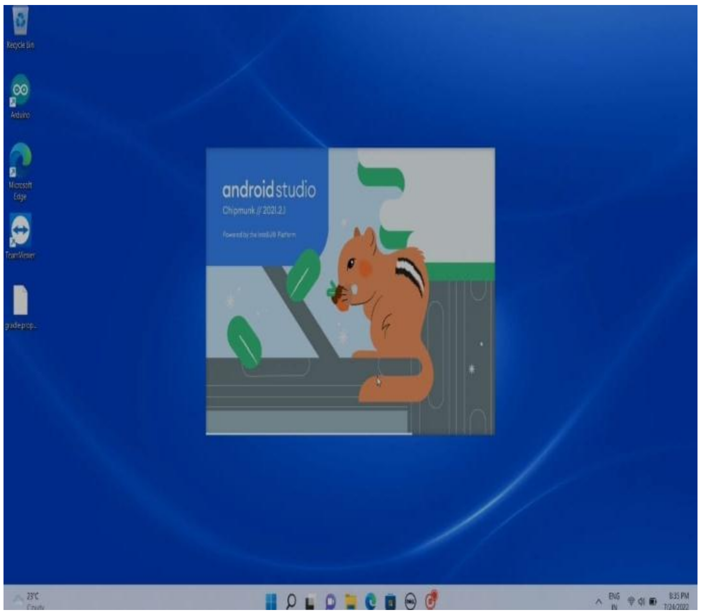
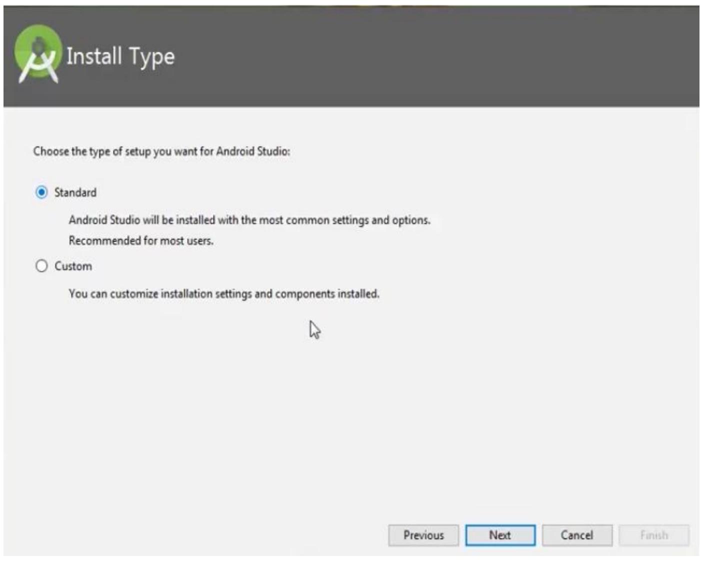


Fig -1: Windows setup



Install Type

Choose the type of setup you want for Android Studio:

● Standard

Android Studio will be installed with the most common settings and options.
Recommended for most users.

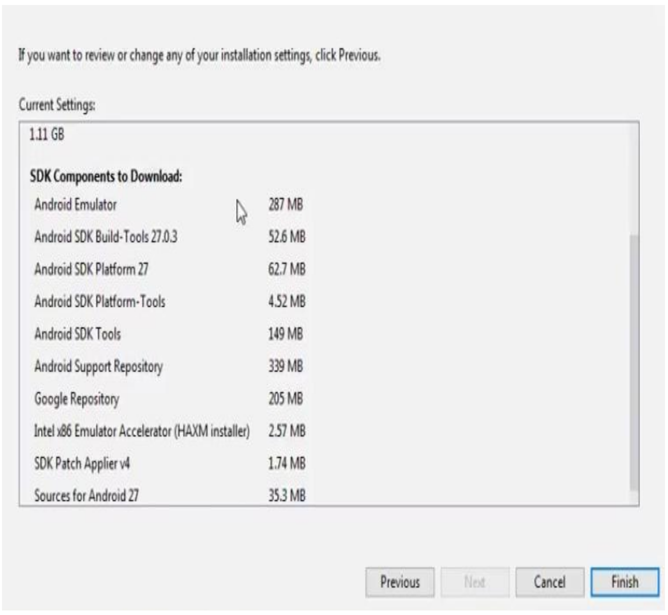
○ Custom

You can customize installation settings and components installed.

Fig -2: Installation on windows

2.2 Installing the Android SDK

Once installed, open Android Studio. Following installation, launch Android Studio. The Android SDK installation requirement should be detected by the IDE. Completely install the SDK using the SDK Components Setup screen. Make a note of where the Android SDK is located.



If you want to review or change any of your installation settings, click Previous.

Current Settings:

1.11 GB

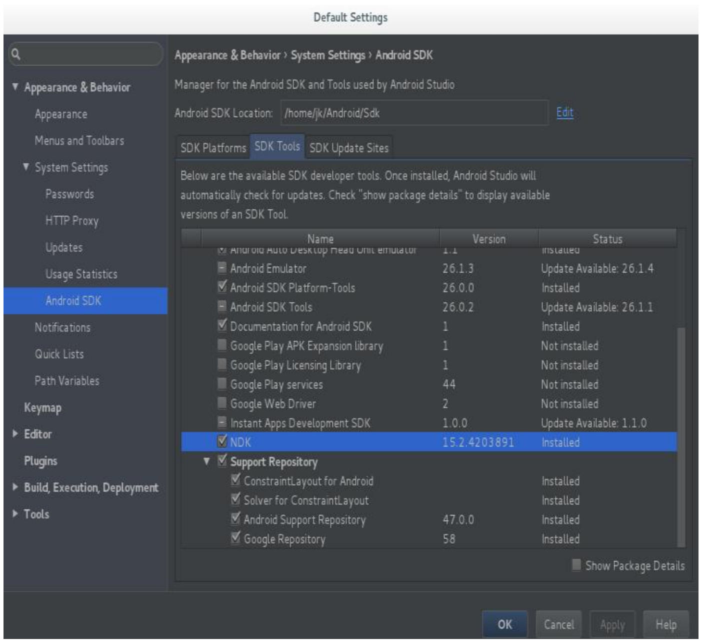
SDK Components to Download:

Android Emulator	287 MB
Android SDK Build-Tools 27.0.3	52.6 MB
Android SDK Platform 27	62.7 MB
Android SDK Platform-Tools	4.52 MB
Android SDK Tools	149 MB
Android Support Repository	339 MB
Google Repository	205 MB
Intel x86 Emulator Accelerator (HAXM installer)	2.57 MB
SDK Patch Applier v4	1.74 MB
Sources for Android 27	35.3 MB

Fig -3: SDK Installation

By default, the latest stable SDK Platform is installed, which includes a collection of packages required to target that version of Android. Installing the most recent stable SDK Platform, which includes several packages required to target that version of Android, is automatic. The Show Package Details checkbox at the bottom of the SDK Manager may need to be selected in order to install system images and other minor SDK platform packages.

SDK (Software Development Kit or "devkit") is typically a set of software development tools that allow for the creation of applications for a certain software package, software framework, or hardware platform. It may be something as simple as an application programming interface (API) in the form of some files to interface with a particular programming language or include sophisticated hardware. Debugging aids and other utilities are frequent tools that are frequently displayed in an integrated development environment (IDE). It could be something as simple as an application programming interface (API) in the form of a few files to communicate with a particular embedded system or include complex hardware to interface with a particular programming language. Common tools include utilities and debugging aids, which are frequently displayed in an integrated development environment (IDE). As soon as you unzip and load the IDE in the most recent version of ADT, the Android SDK is automatically added. The SDK Manager enables us to download Google APIs and use them in our code. Google Calendar, Google Code.



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: /home/jk/Android/Sdk Edit

Below are the available SDK developer tools. Once installed, Android Studio will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.

Name	Version	Status
✓ Android Auto Desktop Head Unit emulator	1.1	Installed
− Android Emulator	26.1.3	Update Available: 26.1.4
✓ Android SDK Platform-Tools	26.0.0	Installed
− Android SDK Tools	26.0.2	Update Available: 26.1.1
✓ Documentation for Android SDK	1	Installed
Google Play APK Expansion library	1	Not installed
Google Play Licensing Library	1	Not installed
Google Play services	44	Not installed
Google Web Driver	2	Not installed
− Instant Apps Development SDK	1.0.0	Update Available: 1.1.0
✓ NDK	15.2.4203891	Installed
✓ Support Repository		
✓ ConstraintLayout for Android		Installed
✓ Solver for ConstraintLayout		Installed
✓ Android Support Repository	47.0.0	Installed
✓ Google Repository	58	Installed

Fig -4: Android SDK manager

2.3 Android Virtual Device (AVD)

One can launch virtual Android devices or emulators on our PC and run the app inside of them with the help of the Android Virtual Device (AVD) manager, while also being able to monitor and troubleshoot each app's activity from the Logcat in our IDE.

Fig -5: AVD Launching

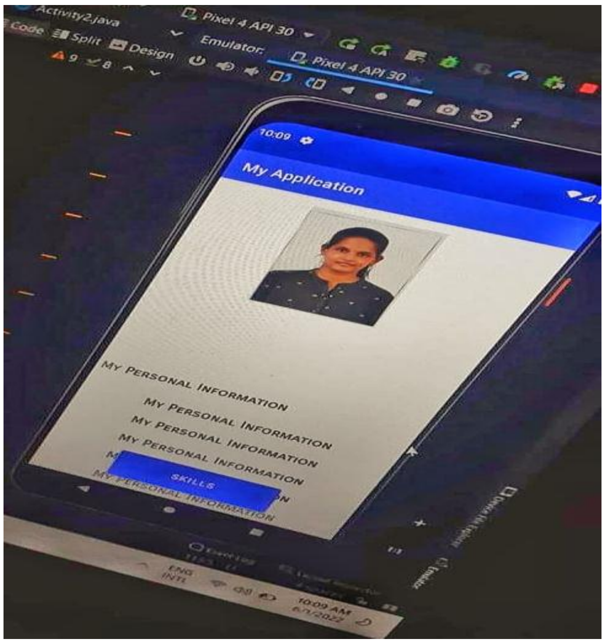


Fig -6: Pixel 4 API 30 Emulator

2.4 Android Software Development

The process by which new applications are developed for the Android operating system is known as Android software development. The Android Software Development Kit is typically used to create applications in the Java programming language. The software used to create Android apps is called ADT (Android Development Tools). Basically, it houses the Eclipse IDE, a multi-language integrated development environment (IDE) with a base workspace and an extensible plug-in system for environment customization. The latest version comes with the ADT plugin preinstalled and bundled with the IDE.

3 SEQUENCE

1. Save each icon in its own folder.
2. Keep "my information" as the name of the folder.
3. Add them to the drawable, copy the desired portion from the location, and then paste it there in the drawable file.
4. Create activity 1 now, fill in the requirements with code in the java file, and create the layout in the xml file.
5. Create activities 2, 3, and 4.
6. Check for any errors.

7. Create a virtual device with the name "Pixel 4 Mobile API 30" by opening the virtual device tab.

8. Run the program and examine the application to see if everything is in order.

4 CODE FOR APP DEVELOPMENT

main.java (Activity 1)

```
package com.example.myinformation;

import android.content.Intent;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {
  private Button button;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    button = (Button) findViewById(R.id.button);
    button.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        openActivity2();
      }
    });
  }

  public void openActivity2() {
    Intent intent = new Intent(this, Activity2.class);
    startActivity(intent);
  }
}
```

main.xml (Activity 1)

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/andr
oid"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context="com.example.myapplication.MainActivity">

  <TextView
    android:id="@+id/textView"
    android:layout_width="249dp"
    android:layout_height="wrap_content"
    android:layout_alignTop="@+id/button"
    android:layout_alignParentEnd="true"
    android:layout_marginTop="-235dp"
    android:layout_marginEnd="78dp"
    android:layout_marginBottom="30dp"
    android:text="Hello! I'm  Bhanu sree. Welcome to my
App"
    android:textColor="#30A335"
    android:textSize="25sp"
    android:textStyle="bold|italic" />

  <Button
    android:id="@+id/button"
    android:layout_width="124dp"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="71dp"
    android:layout_marginBottom="88dp"
    android:text="Enter"
```

```
        android:textSize="20sp" />


    <ImageView
        android:id="@+id/imageView7"
        android:layout_width="388dp"
        android:layout_height="349dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="3dp"
        android:layout_marginBottom="379dp"
        app:srcCompat="@drawable/hello" />


</RelativeLayout>
```

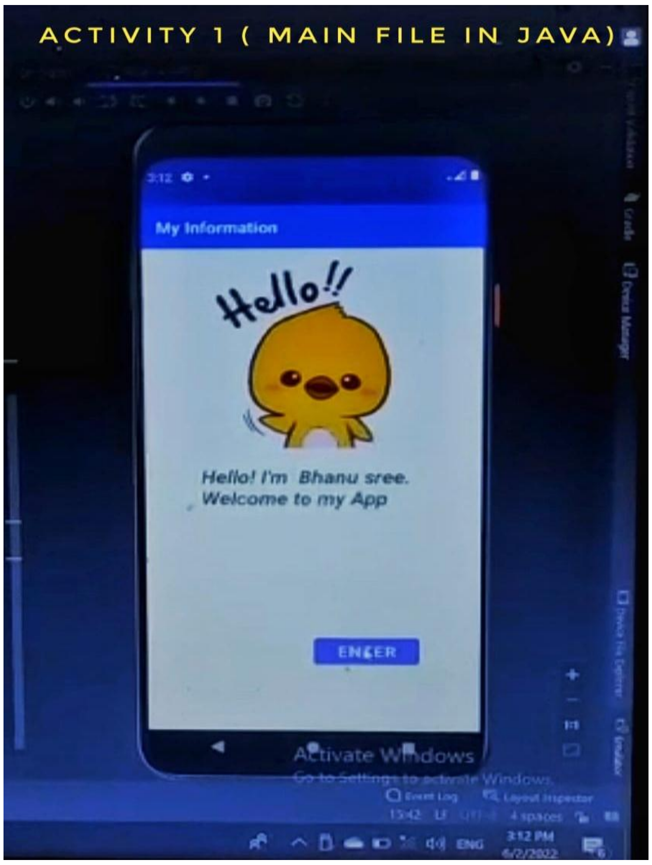
Only 1 activity code is written here for both the java and xml. Java code is to perform the logic and while xml code is used to develop the page structure. When button is pressed in 1st screen, it will activate the command and changes the screen to another activity. It follows activity 2, 3 and 4 respectively. The screens are shown in the result/output.

5. BENEFITS

There must be some important benefits if there are so many designers using multiple screens. Everyone operates differently, so the trick is to find the configuration that suits the best requirements.

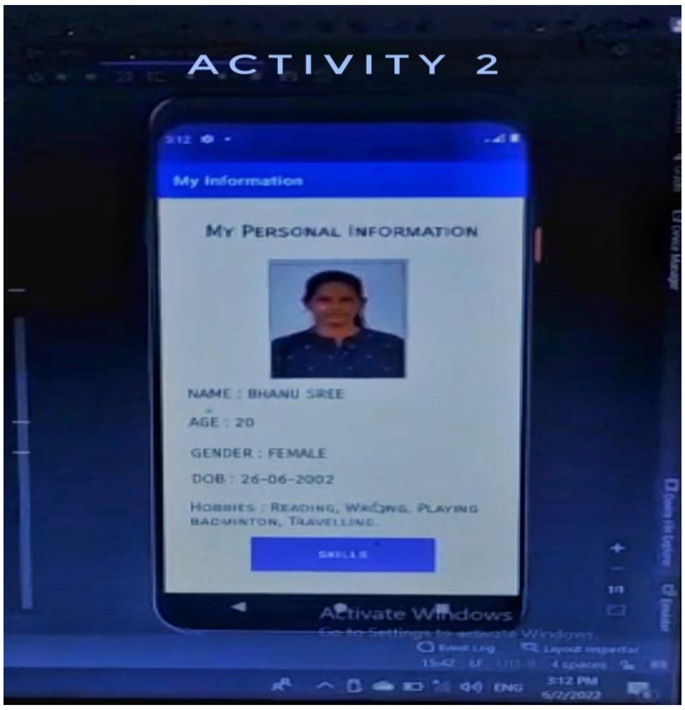
1. Rising productivity
2. Designers use a lot of programs all at once.
3. Data Sharing Among Applications Can Be Easier
4. To enable easier comparison

6 OUTPUT



Hello! I'm Bhanu sree.
Welcome to my App

Fig -7: Home page (First screen)



MY PERSONAL INFORMATION

NAME : BHANU SREE

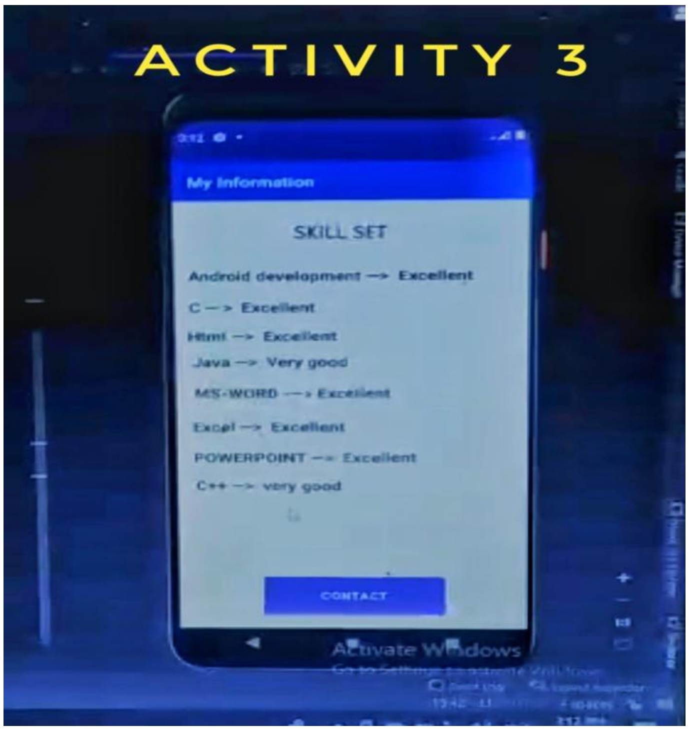
AGE : 20

GENDER : FEMALE

DOB : 26-06-2002

HOBBIES : READING, WRITING, PLAYING BADMINTON, TRAVELLING

Fig -8: Personal information (Second screen)


ACTIVITY 3

My Information

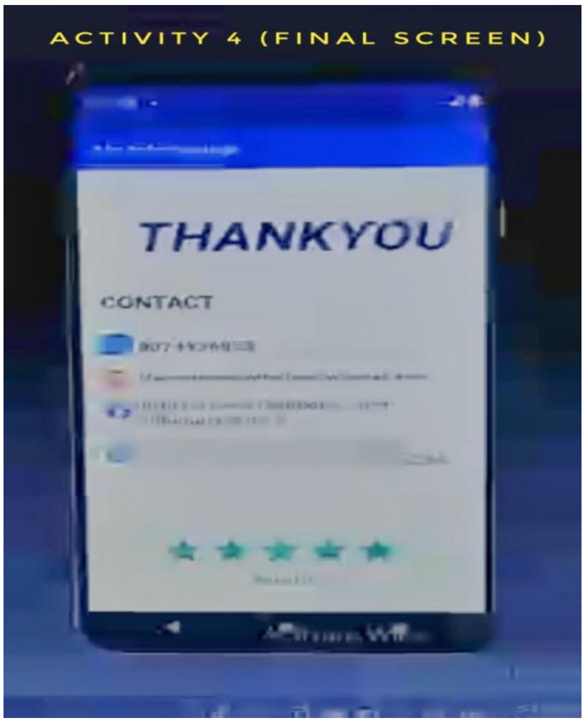
SKILL SET

Android development —> Excellent
C —> Excellent
Html —> Excellent
Java —> Very good
MS-WORD —> Excellent
Excel —> Excellent
POWERPOINT —> Excellent
C++ —> very good

CONTACT

Activate Windows

Fig -9: Skillset (Third screen)


ACTIVITY 4 (FINAL SCREEN)

THANKYOU

CONTACT

Fig -10: Contact information (Fourth screen)

The button "ENTER" on the first screen, when pressed, opens the screen two. The "SKILLS" button on the second screen, when pressed, opens screen three. The "CONTACT" button on the third screen, when pressed, opens screen four. The screen 4 is the last activity and will have information about the student's contacts, such as mobile number, LinkedIn profile, Instagram profile, college roll number, and rank of the student along with their behavioral rating.

6. CONCLUSION

This is the procedure to create multiple screens, and to access each screen, there must be a logic written in Java for each activity that leads to the next activity. Having a student profile application is useful, which can save time and allow students to showcase professional and personal details to people.

REFERENCES

[1] How to Install and Set up Android Studio on Windows? - GeeksforGeeks

[2] Introduction to Android Development - GeeksforGeeks

[3] Android development – NAAS Solutions Ltd. (naasbd.com)

[4] https://gist.github.com/codinginflow