VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI -590 018, KARNATAKA

A Mini Project Report on

"STUDENT OFFICE"

Submitted in the partial fulfillment for the requirements for the Mobile Application Development Lab with Mini Project (18CSMP68)

INFORMATION SCIENCE AND ENGINEERING

By

Mr. Vaibhav Jain USN:1BY20IS187
Mr.Mahadeva Kumar N USN:1BY20IS408

Under the guidance of

Dr .Geeta Amol patil.
Associate Professor
Department of ISE, BMSIT&M.

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

BMS INSTITUTE OF TECHNOLOGY & MANAGEMNT

YELAHANKA, BENGALURU-560064

2022-2023

VISVESVARAYA TECHNOLOGICAL UNIVERSITYBELAGAVI – 590 018, KARNATAKA

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

YELAHANKA, BENGALURU-560064

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Project work entitled **"Student office"** is a bonafide work carried out by **Mr. Vaibhav Jain(1BY20IS187) and Mr.Mahadeva Kumar N(1BY20IS408)** in partial fulfillment of Mobile Application Development Laboratory with Mini Project (18CSMP68) for the award of **Bachelor of Engineering Degree in Information Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the year 2022-23. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this report. The project report has been approved as it satisfies the academic requirements in respect of Mini Project work for the B.E Degree.

Signature of the Guide	**Signature of the HOD**
Dr.Geeta Amol Patil	Dr. Pushpa S.K.
Associate Professor	Professor and Head
Department of ISE	Department of ISE

EXTERNAL EXAMINERS

Name of the Examiners

Signature with Date

1.

2.

ACKNOWLEDGEMENT

We are happy to present this mini project after completing it successfully. This mini project would not have been possible without the guidance, assistance and suggestions of many individuals. We would like to express our deep sense of gratitude and indebtedness to each and every one who has helped us make this mini project a success.

We heartly thank our **Principal, Dr. Mohan Babu G.N, B M S Institute of Technology & Management** for his constant encouragement and inspiration in taking up this mini project.

We heartly thank our **Head of Department, Dr. Pushpa S.K, Dept. of Information Science and Engineering, B M S Institute of Technology& Management** for her constant encouragement and inspiration in taking up this mini project.

We gracefully thank our Project guide**, Dr. Geeta Amol Patil ,Associate Professor, Dept. of Information Science and Engineering,** for her encouragement and advice throughout the course of the mini project work.

Special thanks to all the staff members of Information Science Department for their help and kind co-operation.

We also thank our parents and friends for their unconditional love and encouragement and support given to us in order to finish this precious work.

Last but not the least we would like to thank God for giving us the strength and motivation through the course of this Project.

BY,
Mahadeva Kumar N
Vaibhav Jain

ABSTRACT

At present day, we want everything in one space to save our time and maximize the utility. If we install different applications for different purposes then it will occupy more device memory and reduce device's efficiency. Therefore, we need to develop such kind of application which can be used for many purposes. To fulfill user-need, we have developed Multi-Utility Application ”Student Office” where user will get four kind of functionality in a single App. Our proposed application can be installed in any android device and this is easy to use. This Multi-Utility App provides Scheduler, Scientific Calculator, CGPA calculator and Results web-extension.

Scheduler Application provides an Android-based schedule planner for students at institutions. Scheduler App automates the human pen paper work and presents the student with every possible schedule option.

This Application is efficient and user-friendly. Furthermore, this App is suitable for low memory and less-advanced devices. We have developed simple user interface of this application to reduce power consumption power consumption. Its light dark background saves power.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Overview

Mobile application development is the process to making software for smartphones and digital assistants, most commonly for Android and iOS. The software can be preinstalled on the device, downloaded from a mobile app store or accessed through a mobile web browser. The programming and markup languages used for this kind of software development include Java, Swift, C# and HTML5.

Mobile app development is rapidly growing. From retail, telecommunications and e-commerce to insurance, healthcare and government, organizations across industries must meet user expectations for real-time, convenient ways to conduct transactions and access information. Today, mobile devices and the mobile applications that unlock their value are the most popular way for people and businesses to connect to the internet. To stay relevant, responsive and successful, organizations need to develop the mobile applications that their customers, partners and employees demand.

Yet mobile application development might seem daunting. Once you've selected the OS platform or platforms, you need to overcome the limitations of mobile devices and usher your app all the way past the potential hurdles of distribution. Fortunately, by following a few basic guidelines and best practices, you can streamline your application development journey. We can start explaining mobile development, which is not about building phone apps, though it is a huge part of it. Actually, It's doing any reasonably development for any kind of mobile devices such as developing apps for phones, tablets, smart watches, and every form of wearable devices that run any kind of mobile operating system.

Mobile development presents a reasonably distinctive chance for a one-person development team to build an actual, usable, significant app end-to-end during a period.

What is Mobile App?

1.2

In the context of Android development, a mobile app refers to an application that is specifically developed to run on Android devices. Android Studio is the official integrated development environment (IDE) for creating Android apps. It provides tools, libraries, and a code editor that assist developers in designing, coding, testing, and deploying Android applications

Android Studio provides a range of features and resources to streamline development including

- Code Editor: Android Studio offers a powerful code editor with features like syntax highlighting, auto-completion, code navigation, and refactoring capabilities to improve productivity.
- Layout Editor: The Layout Editor enables you to visually design the user interface (UI) of your app by dragging and dropping UI components, adjusting properties, and arranging layouts.
- Emulator: Android Studio provides an emulator that allows you to test your app on various virtual Android devices, simulating different screen sizes, resolutions, and versions of Android.
- Build System: Android Studio utilizes Gradle as the build system, which automates tasks such as compiling code, managing dependencies, and generating the APK (Android Package) file for distribution.
- Debugging Tools: Android Studio includes tools for debugging your app, such as breakpoints, a debugger, and logcat, which displays logs and error messages during runtime.
- Testing Frameworks: Android Studio supports various testing frameworks to facilitate unit testing, integration testing, and UI testing of your app.

1.2 What is Mobile OS?

- A mobile operating system (OS) is the software platform that manages the hardware and software resources of a mobile device, such as smartphones or tablets. It provides a set of essential services and interfaces that enable mobile apps to run and interact with the device's hardware components. In the case of Android app development using Android Studio, the mobile OS refers to the Android operating system.
- Android is an open-source mobile OS developed by Google and based on the Linux kernel. It is designed to be flexible, customizable, and compatible with a wide range of devices. Android provides developers with a rich set of APIs (Application Programming Interfaces) and frameworks that allow them to create powerful and feature-rich mobile applications.
- When developing an Android app using Android Studio, you leverage the Android OS by utilizing its features and services. Some key aspects of the Android OS include:
- Application Framework: Android provides a comprehensive framework that includes various components and APIs for building apps, such as activity management, UI design, data storage, networking, multimedia, location services, and more.
- User Interface: Android offers a user interface framework that allows developers to create visually appealing and interactive interfaces using XML layouts and resources. This includes support for different screen sizes, resolutions, and device orientations.
- Hardware Compatibility: Android OS is designed to work with a wide range of hardware configurations. Android Studio provides tools and APIs to access device features like the camera, GPS, accelerometer, and sensors, allowing developers to create apps that leverage these capabilities.
- Security and Permissions: Android incorporates a robust security model that includes app sandboxing, permission-based access to device resources, and built-in security features to protect user data and ensure app integrity.

1.3 Android Studio

In recent times, Android became the world's most popular operating system for various reasons. As an Android programmer, I want to share what the Android Studio is? Android Studio is an IDE for Google Android Development launched on 16th May 2013, during Google's I/O 2013 event. Android Studio contains all the Android tools to design, test, debug, and profile your application. The Android Studio uses Gradle to manage your project, a Build Automation Tool.

For developing your first app, you need to download Android Studio for your preferred platform (Windows®, Mac OS X, or Linux) from the Android developers site. Android Studio can develop and test your application on either a real device or an emulator.

Android Studio can be installed on Windows operating systems, OSX and Linux and is recommended by Google itself that the hardware must have at least 4 GB of memory and 1GB of free hard disk space, but we recommend that you have more memory because it was noted that Android Studio is still a little slow. You must have Java installed on the machine via the JDK (Java Development Kit), not the JRE, as it is usually installed, once to develop on Android is necessary for all Java development classes to be present on the machine.

Android Studio has many exciting features that can help you to develop your Android application like:

- Powerful code editor with smart editing and code re-factoring.
- Emulator to show your code output in various resolutions, including Nexus 4, Nexus 7, Nexus 10, and many other android phones.
- Gradle based build support.
- Maven Support.
- Template-based wizards.
- Dracula Theme Environment to enjoy your coding experience.

You can experience all the awesome features by using Android Studio in-hand.

The user interface

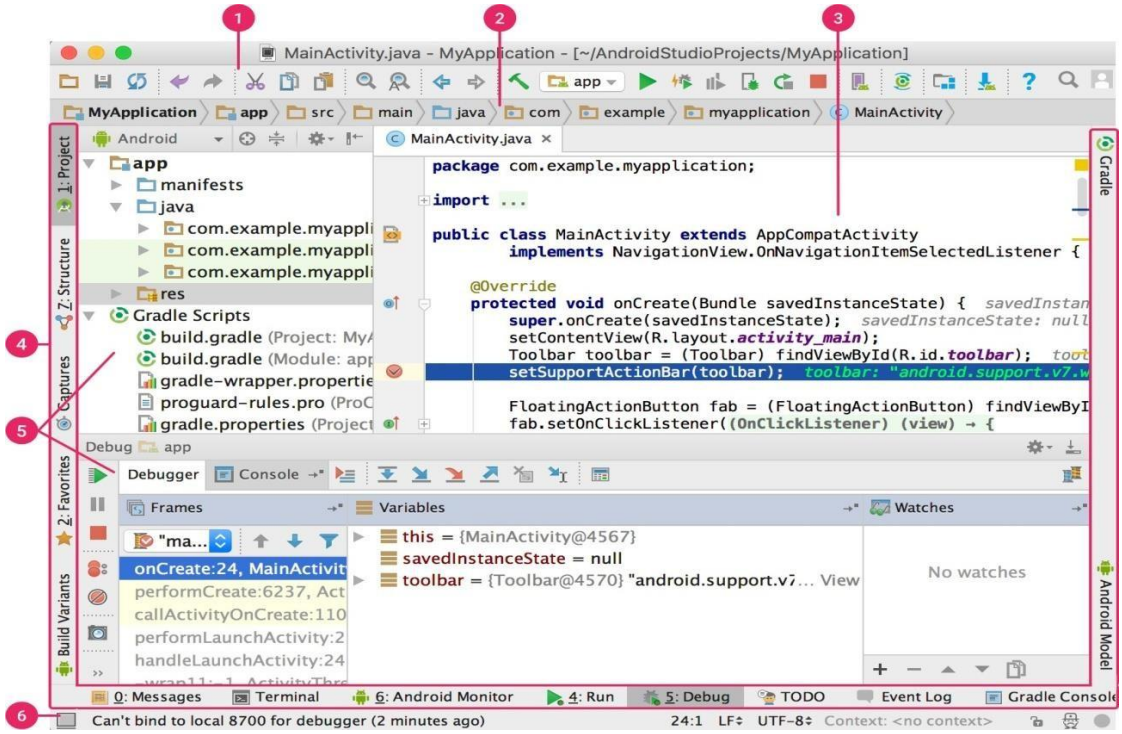


Figure 1.4.1 The Android Studio main window.

- **The toolbar** lets you carry out a wide range of actions, including running your app and launching Android tools.
- **The navigation bar** helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the Project window.
- **The editor window** is where you create and modify code. Depending on the current file type, the editor can change.
- **The tool window bar** runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
- **The tool windows** give you access to specific tasks like project management, search, version control, and more.
- **The status bar** displays the status of your project and the IDE itself, as well as any warnings or messages

1.4 Android Architecture.

The Android architecture is the structure and organization of the Android operating system, which provides the foundation for developing and running Android applications. It can be simplified into four main components:

- Linux Kernel: The Linux kernel is the core of the Android operating system. It manages the device's hardware resources, such as memory, processors, and drivers, and provides the basic functionalities for the system to operate.
- Libraries: Android includes a set of libraries written in C and C++ that provide essential functions and services to the operating system and applications. These libraries handle tasks like graphics rendering, multimedia playback, database access, and more.
- Android Runtime: The Android Runtime (ART) is the virtual machine responsible for running Android applications. It executes the code written in Java or Kotlin languages and converts it into machine code for the device's processor. This process improves performance and efficiency.
- Application Framework: The Application Framework consists of Java classes and APIs that developers use to build Android applications. It provides a wide range of pre-built components and services, such as user interface controls, content providers, data storage, networking, and location services. The framework simplifies app development by providing high-level abstractions.
- On top of these four components, Android applications themselves are developed using Java or Kotlin and run within their own sandboxed environment. Each app runs in its own process, isolated from other apps, ensuring security and stability.
- Overall, the Android architecture follows a layered approach, with the Linux kernel at the core, libraries providing essential functionalities, the Android runtime executing application code, and the application framework offering a rich set of tools and services for app development. This modular structure allows for flexibility, customization, and the ability to run Android applications on a variety of devices with different hardware configurations.

1.5 Problem Statement

At present day, we want everything in one space to save our time and maximize the utility. If we install different applications for different purposes then it will occupy more device memory and reduce device's efficiency.Application is efficient and user- friendly. Furthermore, this App is suitable for low memory and less-advanced devices. We have developed simple user interface of this application to reduce power consumption power consumption. Its light dark background saves power.

Therefore, we need to develop such kind of application which can be used for many purposes. To fulfill user-need, we have to develop Multi-Utility Application where user will get multiple kind of functionality in a single App

1.6 Motivation and Objectives

The motivation behind creating a student office application using Android Studio can be to provide students with a convenient and efficient tool for managing their academic activities and administrative tasks. The objectives of such an application can include:

- Organization and Accessibility: The application aims to help students keep track of their academic schedules, assignments, exams, and other important events. It provides a centralized platform where students can easily access and manage their study-related information.
- Task Management: The application can offer features for creating to-do lists, setting reminders, and managing deadlines. Students can prioritize tasks, receive notifications, and ensure they stay on top of their assignments and projects.
- Communication and Collaboration: The student office application can include communication tools to facilitate interaction among students, teachers, and administrative staff. Features like chat, discussion forums, and announcements can help students stay connected and engaged with their educational community.
- Course Enrollment and Registration: The application can streamline the course enrollment process by providing an intuitive interface for browsing available courses, checking course

availability, and registering for classes. It can also handle prerequisites and provide information on class schedules and locations.

- Grades and Progress Tracking: The student office app can allow students to view their grades, track their academic progress, and monitor their performance in different courses. It can provide visual representations of progress, such as charts or graphs, to help students assess their achievements.
- Resources and Support: The application can serve as a platform to access educational resources, including lecture notes, study materials, and online libraries. It can also offer links to support services, such as academic advising, counseling, and campus facilities.
- Personalization and Customization: The student office app can provide options for personalizing the user interface, such as choosing themes, layouts, and preferences. This allows students to tailor the app to their individual preferences and needs.

1.7 Project Applications

- Class Schedule Management: The app can allow students to view and manage their class schedules, including details such as course names, instructors, locations, and times. Students can receive reminders for upcoming classes and be notified of any schedule changes.
- Assignment Tracker: The app can provide a feature for students to track and manage their assignments. Students can add assignment details, set due dates, receive reminders, and mark completed tasks. They can also attach files or notes related to assignments.
- Exam and Test Planner: The app can include a feature to help students plan and track their exams and tests. Students can create a study plan, set reminders for important dates, and receive notifications for upcoming exams. They can also record their exam results and track their performance over time.
- Grade Viewer: The app can allow students to view their grades for individual courses or overall academic performance. Students can access their grade reports, see grade distributions, and track their progress throughout the semester or academic year.
- Course Registration: The app can provide a streamlined process for course registration. Students can browse available courses, view course descriptions and prerequisites, check seat availability, and register for classes directly from the app.

- Campus News and Announcements: The app can include a section for important campus news and announcements. Students can stay updated on university events, academic deadlines, campus closures, and other relevant information.
- Student Resources: The app can serve as a hub for accessing student resources and support services. It can provide links to the library catalog, online databases, academic advising, career services, counseling, and other campus facilities.
- Social Interaction and Collaboration: The app can include features to facilitate social interaction and collaboration among students. This can include chat functionality, discussion forums, study groups, and the ability to connect with classmates and exchange information or study materials.
- Campus Maps and Navigation: The app can integrate campus maps and navigation features to help students find their way around the campus. It can provide directions to specific buildings, classrooms, libraries, and other facilities.

Chapter 2

SYSTEM DESIGN

The student office system designed using Android Studio is a mobile application that simplifies various tasks related to student management. It incorporates user authentication for secureaccess, ensuring that only authorized individuals can use the app. The system features a user- friendly dashboard that serves as a central hub, providing quick access to essential information and features. Students can create and manage their profiles, updating personal details as needed. They can register for courses, view their attendance records, and access their grades and transcripts. The app facilitates communication through messaging and notifications, keeping students informed about important announcements and deadlines. It also offers a repository of educational resources, aiding students in their studies. The system integrates with existing university systems to synchronize data accurately and efficiently. Privacy and security measures are implemented to protect user information. Overall, this system design aims to enhance the student office experience, making administrative tasks more accessible and convenient for students.

Scheduler:

The scheduler feature allows students to manage their schedules effectively.

Students can input their class timings, exam dates, and other important events.

The scheduler provides reminders and notifications to help students stay organized and attend their classes and exams on time.

Calculator:

The calculator feature provides a convenient tool for students to perform mathematical calculations.It includes basic arithmetic operations such as addition, subtraction, multiplication, and division.

The calculator may also include advanced functions like square root, percentage calculations, and trigonometric functions to assist students in solving complex problems.

CGPA Calculation:

The CGPA calculation feature enables students to calculate their Cumulative Grade Point Average (CGPA).

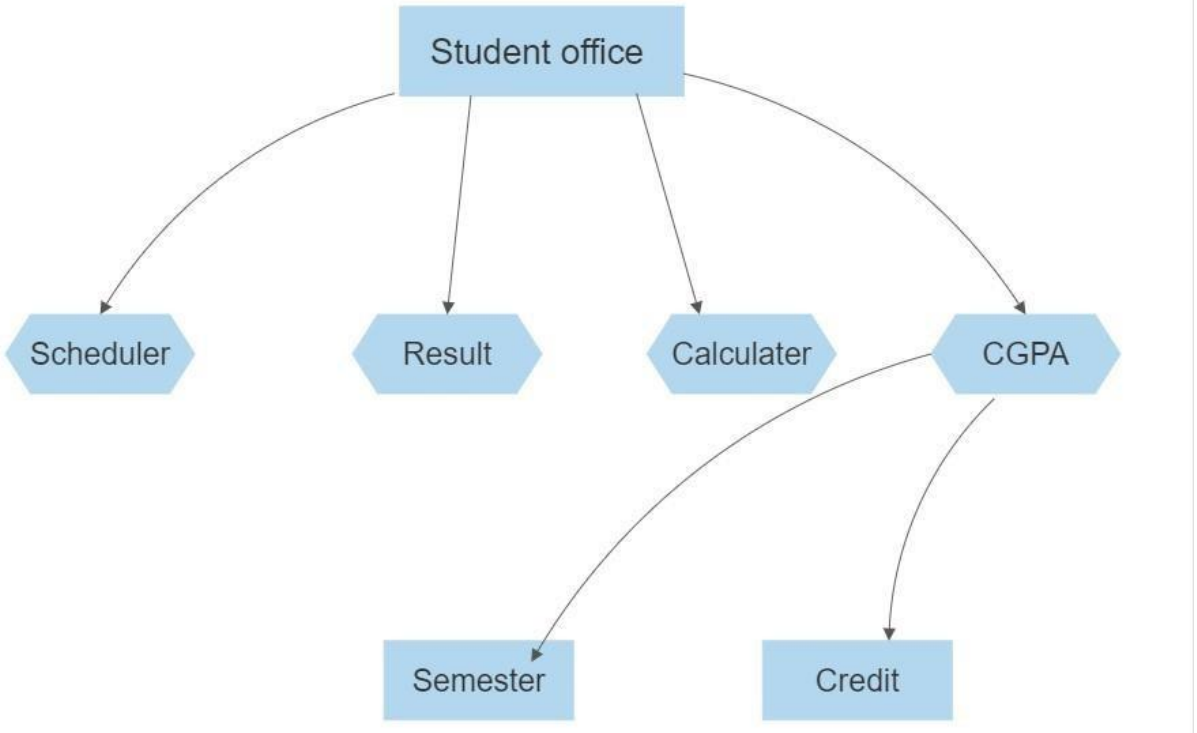
Students can input their course grades and credit hours to determine their overall CGPA.

The system automatically calculates the CGPA based on the entered grades and credit hours, providing students with an accurate representation of their academic performance.

Result Management:

The result management feature allows students to view and manage their course results. Students can access their grade sheets, which display individual course grades, credit hours, and overall GPA for each semester.

The system can generate semester-wise result summaries and cumulative transcripts for students to track their academic progress over time.



Chapter 3

IMPLEMENTATION

3.1 XML Design

The proposed system we will be building a Multi-Utility Application that can Help and reduce the work of students respectively,

The home page will be given with four option.

Scheduler : An application provides an Android-based schedule planner for students at institutions. Scheduler App automates the human pen paper work and presents the student with every possible schedule option.

It Consists of : Title - Day - Subject -Timings & CRUD options.

1. **Calculator :** An application provides an Android-based scientifc calculator app which is very simple to solve arithmetic operations and scientific notation type math functions like sin.cos,tan,log etc.

   Consists of : Numerics - Key sections - & Display options.

2. **CGPA :** An application provides an Android-based CGPA calculator for students atinstitutions that calculates your CGPA score .

   Consists of : Semester - Credits - & Calculating options.

3. **Results** : An application provides an Android-based results forecasting app whichdisplays the result data and notes for the students .

   Consists of : Resource Menu - Result forecaster - & other Utilities.

3.2 XML Code

Activity_main.xml

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

<androidx.drawerlayout.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    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"
    android:id="@+id/drawer_layout"
    tools:context=".MainActivity"
    android:background="#ffffffff">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
    android:background="@drawable/card_bg"
    android:orientation="vertical"
    android:weightSum="10">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="2">

        <ImageView
            android:id="@+id/dbtn"
            android:layout_width="35dp"
            android:layout_height="35dp"
            android:layout_margin="10dp"
            android:layout_gravity="center"
            android:clickable="true"
            android:focusable="true"
            android:src="@drawable/menu"/>

        <TextView
            android:id="@+id/textBuddy"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"
            android:text="VTU"
            android:textColor="@android:color/white"
            android:textSize="34sp" />

    </RelativeLayout>

    <GridLayout
```

```
android:id="@+id/mainGrid"
android:layout_width="match_parent"
android:layout_height="0dp"
android:layout_weight="8"
android:alignmentMode="alignMargins"
android:columnCount="2"
android:columnOrderPreserved="false"
android:padding="14dp"
android:rowCount="3">

<!-- Row 1 -->

<!-- Column 1 -->
<androidx.cardview.widget.CardView
   android:id="@+id/cardScheduler"
   android:layout_width="0dp"
   android:layout_height="0dp"
   android:layout_columnWeight="1"
   android:layout_marginBottom="16dp"
   android:layout_marginLeft="8dp"
   android:layout_marginRight="8dp"
   android:layout_rowWeight="1"
   app:cardCornerRadius="8dp"
   app:cardElevation="8dp">

   <LinearLayout
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_gravity="center_horizontal|center_vertical"
      android:layout_margin="16dp"
      android:orientation="vertical">

      <ImageView
         android:layout_width="101dp"
         android:layout_height="50dp"
         android:layout_gravity="center_horizontal"
         android:src="@drawable/schedule" />

      <TextView
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="@string/scheduler"
         android:textAlignment="center"
         android:textColor="@android:color/black"
         android:textSize="18sp"
         android:textStyle="bold" />

   </LinearLayout>

</androidx.cardview.widget.CardView>
```

```
<!-- Column 2 -->
<androidx.cardview.widget.CardView
   android:id="@+id/cardNotes"
   android:layout_width="0dp"
   android:layout_height="0dp"
   android:layout_columnWeight="1"
   android:layout_marginBottom="16dp"
   android:layout_marginLeft="8dp"
   android:layout_marginRight="8dp"
   android:layout_rowWeight="1"
   app:cardCornerRadius="8dp"
   app:cardElevation="8dp">

   <LinearLayout
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_gravity="center_horizontal|center_vertical"
      android:layout_margin="16dp"
      android:orientation="vertical">

      <ImageView
         android:layout_width="47dp"
         android:layout_height="66dp"
         android:layout_gravity="center_horizontal"
         android:src="@drawable/research" />

      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="@string/results"
         android:textAlignment="center"
         android:textColor="@android:color/black"
         android:textSize="18sp"
         android:textStyle="bold" />

   </LinearLayout>

</androidx.cardview.widget.CardView>

<!-- Row 2 -->

<!-- Column 1 -->
<androidx.cardview.widget.CardView
   android:id="@+id/cardCalculator"
   android:layout_width="0dp"
   android:layout_height="0dp"
   android:layout_columnWeight="1"
   android:layout_marginBottom="16dp"
   android:layout_marginLeft="8dp"
   android:layout_marginRight="8dp"
   android:layout_rowWeight="1"
```

```
        app:cardCornerRadius="8dp"
        app:cardElevation="8dp">

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal|center_vertical"
            android:layout_margin="16dp"
            android:orientation="vertical">

            <ImageView
                android:layout_width="89dp"
                android:layout_height="60dp"
                android:layout_gravity="center_horizontal"
                android:src="@drawable/calculator" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/calculator"
                android:textAlignment="center"
                android:textColor="@android:color/black"
                android:textSize="18sp"
                android:textStyle="bold" />

        </LinearLayout>

    </androidx.cardview.widget.CardView>

    <!-- Column 2 -->
    <androidx.cardview.widget.CardView
        android:id="@+id/cardCgpa"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_columnWeight="1"
        android:layout_marginBottom="16dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_rowWeight="1"
        app:cardCornerRadius="8dp"
        app:cardElevation="8dp">

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal|center_vertical"
            android:layout_margin="16dp"
            android:orientation="vertical">

            <ImageView
                android:layout_width="wrap_content"
```

```
                android:layout_height="wrap_content"
                android:layout_gravity="center_horizontal"
                android:src="@drawable/calculations" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="@string/cgpa"
                android:textAlignment="center"
                android:textColor="@android:color/black"
                android:textSize="18sp"
                android:textStyle="bold" />
        </LinearLayout>
    </androidx.cardview.widget.CardView>
  </GridLayout>
</LinearLayout>

  <androidx.coordinatorlayout.widget.CoordinatorLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginStart="20dp"
    android:layout_marginEnd="20dp"
    android:layout_marginBottom="5dp"
    android:id="@+id/snackbarPosition"/>

  <com.google.android.material.navigation.NavigationView
    android:id="@+id/nav_drawer"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    app:menu="@menu/menu_drawer"
    android:background="@drawable/card_bg"
    app:itemIconTint="@color/colorAccent"/>

</androidx.drawerlayout.widget.DrawerLayout>
```

activity_new_schedule.xml : contains scheduler UI.

activity_calculator.xml : contains calculator UI.

activity_cgpa.xml : contains cgpa calculator UI.

activity_notes.xml : contains results and notes UI.

3.3Description

- **WebViewClient{ }**

  WebView objects allow you to display web content as part of your activity layout, but lack some of the features of fully-developed browsers. A WebView is useful when you need increased control over the UI and advanced configuration options that will allow you to embed web pages in a specially-designed environment .

- **Void onCreate (Bundle)**

  The **entire lifetime** of an activity happens between the first call to onCreate(Bundle) and onDestroy(). An activity will do all setup of "global" state in onCreate(), and release all remaining resources in onDestroy().

- **public static interface View.OnClickListener( )**

  Interface definition for a callback to be invoked when a view is clicked.

- **public class Intent( )**

  An Intent provides a facility for performing late runtime binding between the code in different applications. Its most significant use is in the launching of activities.

- **public void onRequestPermissionsResult ( )**

  public abstract void **onRequestPermissionsResult** (int requestCode, String[] permissions, int[] grantResults) Callback for the result from requesting permissions. This method is invoked for every call on ActivityCompat.

MainActivity.java

```
package com.example.vtu;

import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;

import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;

public class MainActivity extends AppCompatActivity implements View.OnClickListener

    {private CardView cardScheduler, cardNotes, cardCalculator, cardCgpa;
    private static Context context;

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



        cardScheduler = findViewById(R.id.cardScheduler);
        cardNotes = findViewById(R.id.cardNotes);
        cardCalculator = findViewById(R.id.cardCalculator);
        cardCgpa = findViewById(R.id.cardCgpa);

        cardScheduler.setOnClickListener(this);
        cardNotes.setOnClickListener(this);
        cardCalculator.setOnClickListener(this);
        cardCgpa.setOnClickListener(this);

        MainActivity.context = getApplicationContext();
    }

    public static Context getAppContext()
        {return MainActivity.context;
    }

    public static int getPx(Context context, int dimensionDp) {
        float density = context.getResources().getDisplayMetrics().density;
        return (int) (dimensionDp * density + 0.5f);
    }

    @Override
    public void onClick(View view) {
```

```
        switch (view.getId())
            { case
            R.id.cardScheduler:
                startActivity(new Intent(this, SchedulerActivity.class));break;
            case R.id.cardNotes:
                startActivity(new Intent(this, NotesActivity.class));
                break;
            case R.id.cardCalculator:
                startActivity(new Intent(this, CalculatorActivity.class));break;
            case R.id.cardCgpa:
                startActivity(new Intent(this, CgpaActivity.class));
                break;
        }
    }
}
```

SchedulerActivity.java : contains Scheduler operations.

CalculatorActivity.java : contains Calculator operations.

NotesActivity.java : contains results and notes display operations.

CgpaActivity.java : contains CGPA calculator operations.

Chapter 4

RESULTS

- **Initial Page:** This is the Home - UI which displays four application cards and menu bar.

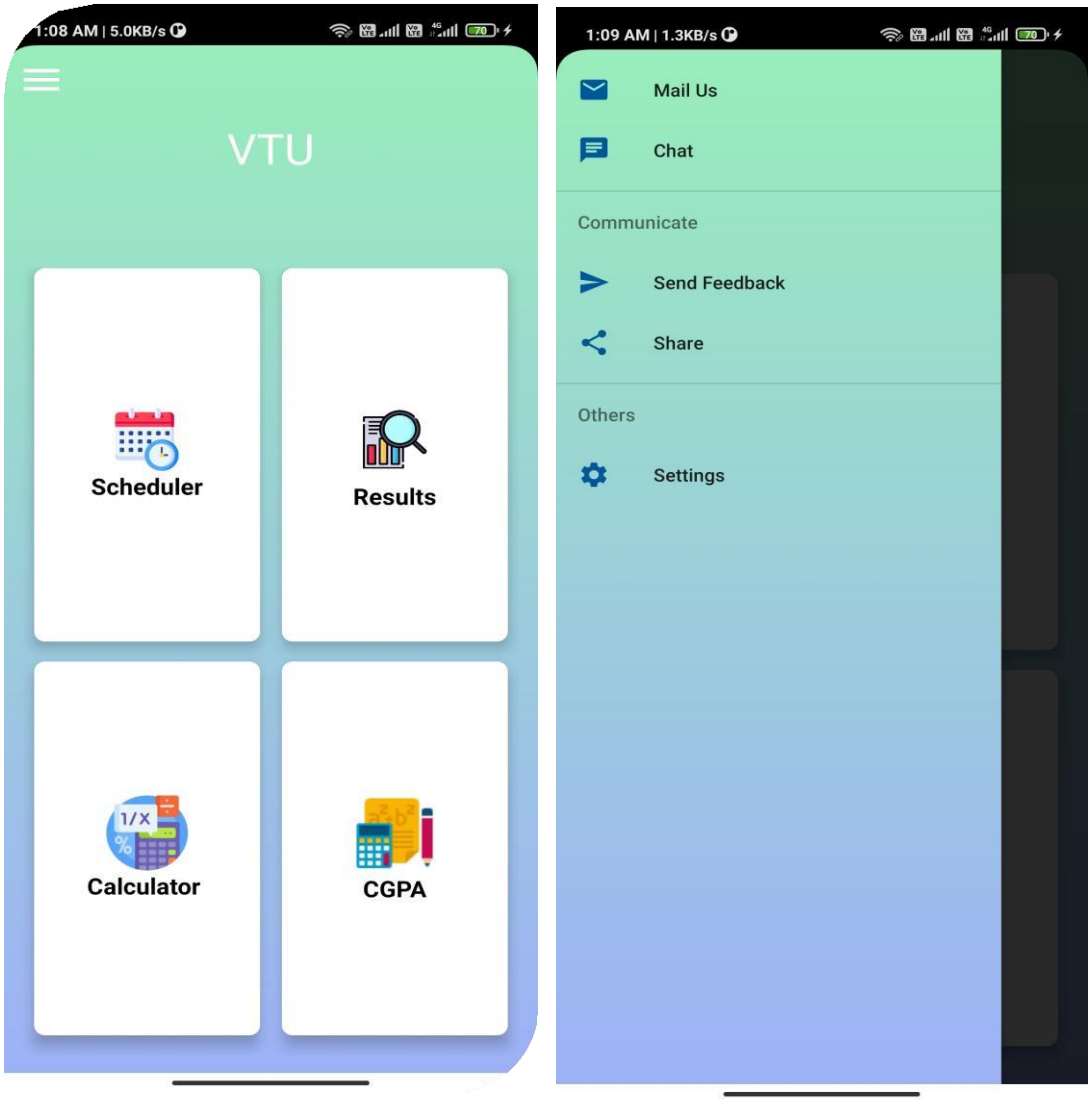


Figure 5.1 Home page of the application.

- **Scheduler:** here it allows users to enter any kind of schedule or event using the interface provided.

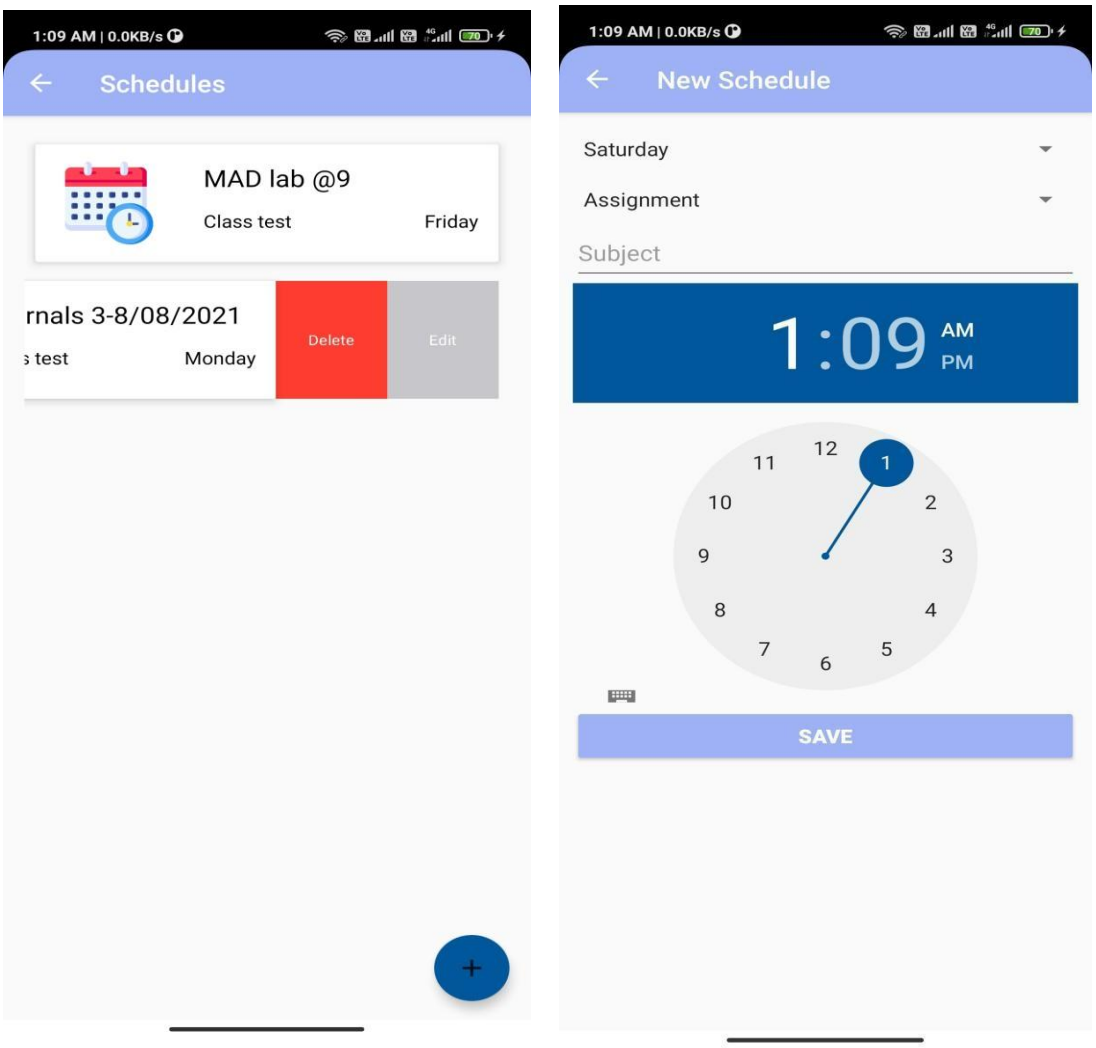


Figure 5.2 Scheduler Layout

- **Results:** here it allows user to see results browse through notes and syllabus for their respective semesters.

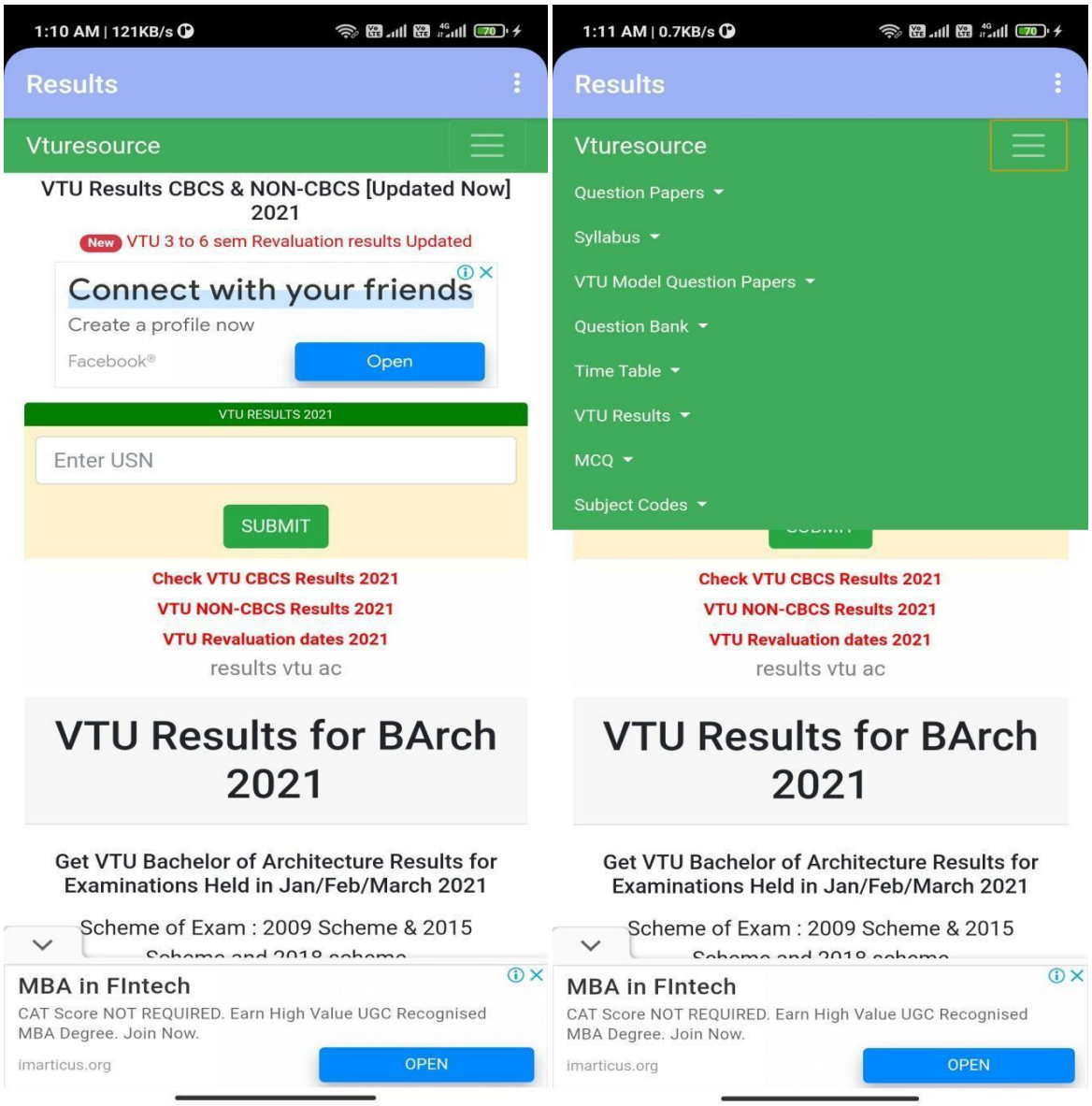


Figure 5.3 Results Layout

- **CGPA:** here it allows user to aggregatetheir cgpa's by entering credits and aquired sgpa's.

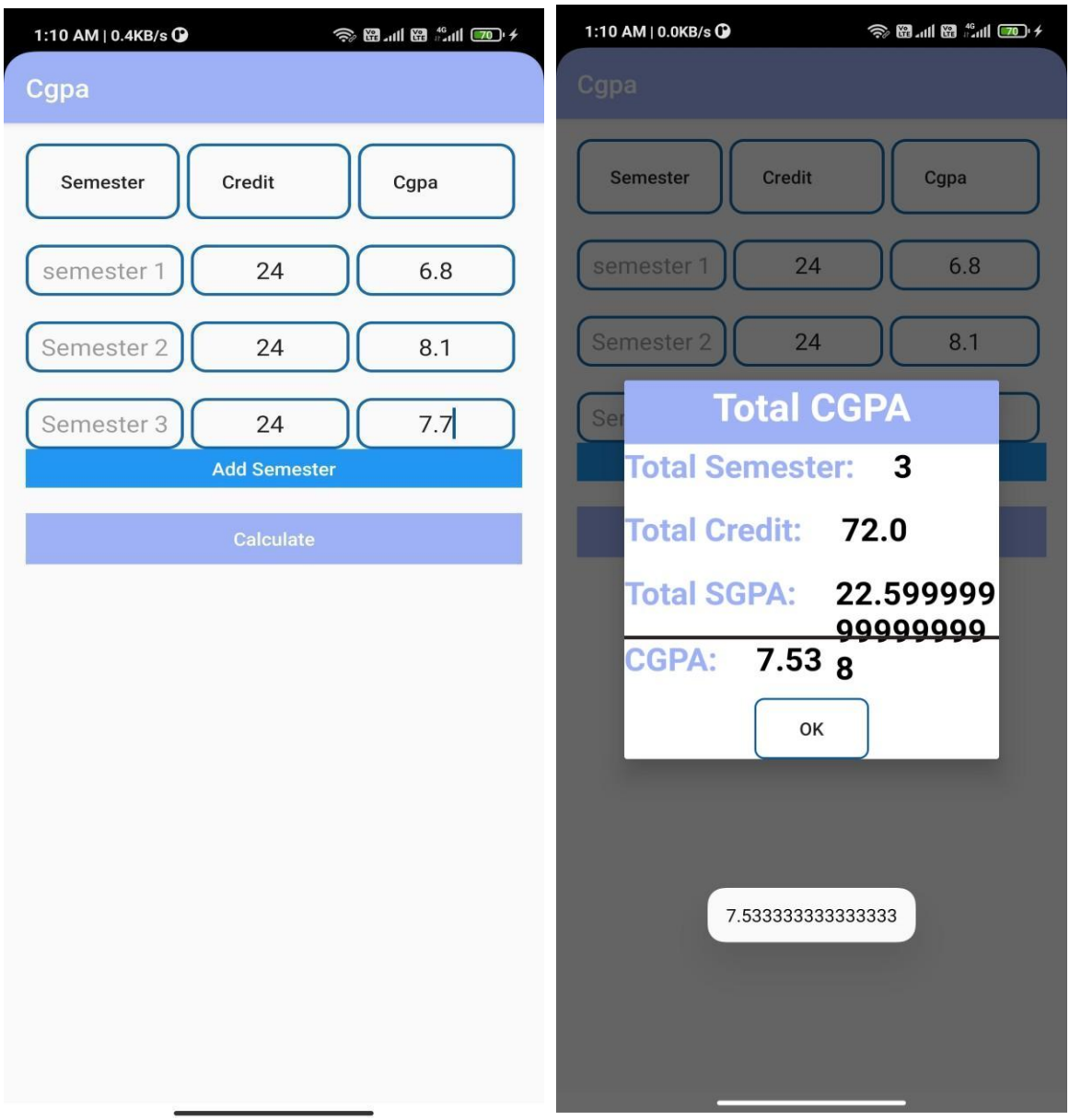


Figure 5.4 CGPA calc Layout

- **Calculator :** here it allows user to calculate which is very simple to solve arithmetic operations and scientific notation type math functions like sin, cos, tan, log etc.

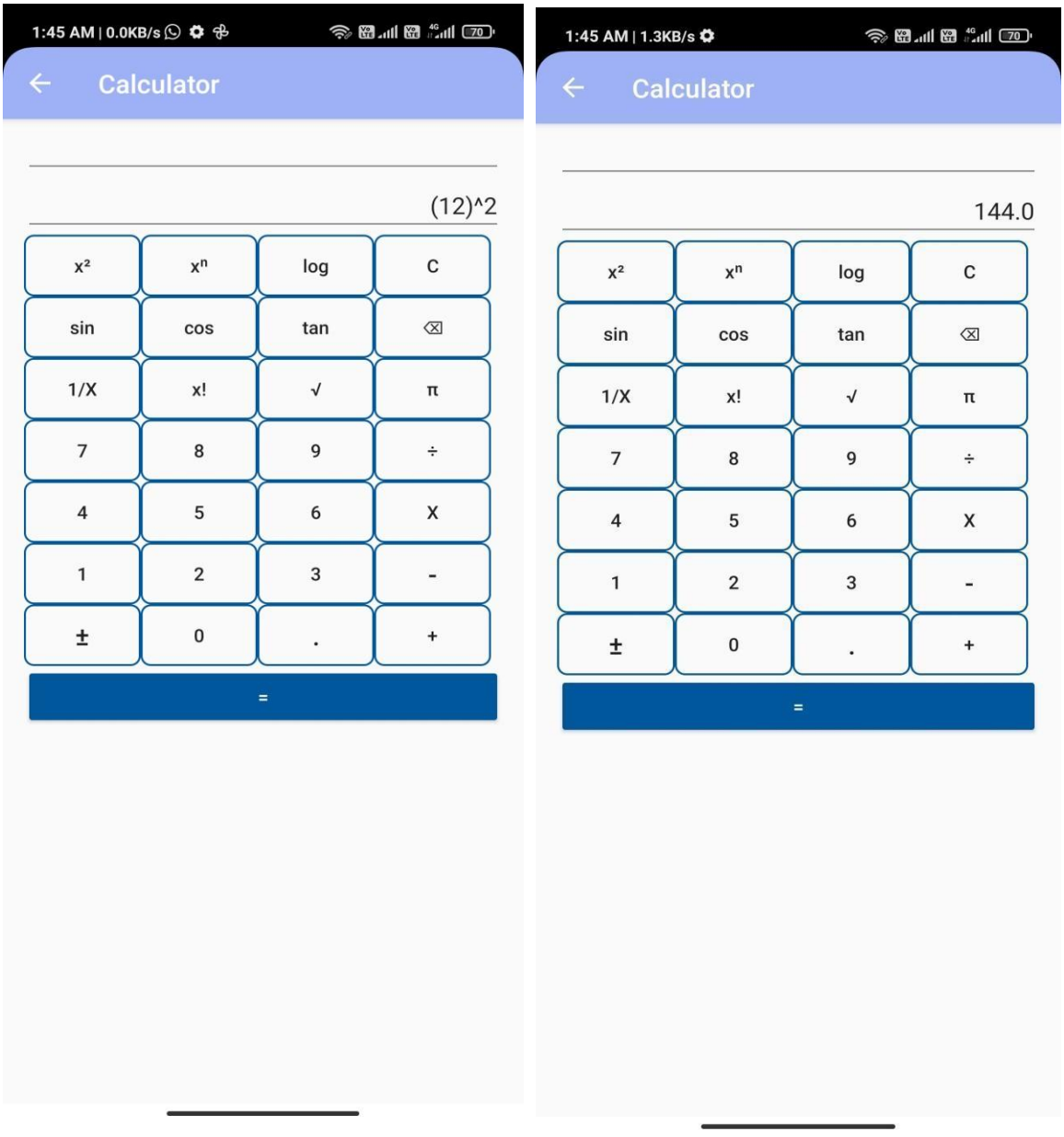


Figure 5.5 Calculator Layout

Chapter 5

Application Testing

- Application testing is an essential aspect of the development process, including the student office app developed using Android Studio. It involves evaluating the app's functionality, performance, usability, and reliability to ensure that it meets the desired quality standards.

- Unit Testing: Unit testing involves testing individual units of code, such as methods or classes, in isolation to ensure they function correctly. In the case of a student office app, unit tests can be written to verify the behavior of specific functionalities, such as scheduling events, calculating grades, or handling assignments.

- Integration Testing: Integration testing focuses on testing the interaction between different components or modules of the app to ensure they work together seamlessly. For a student office app, this can involve testing the integration between the class scheduling feature, assignment tracker, grade viewer, and other functionalities.

- User Interface (UI) Testing: UI testing involves testing the app's user interface to ensure that it functions as intended and provides a smooth user experience. In the context of a student office app, UI testing can include verifying that buttons, input fields, menus, and other UI elements are responsive and correctly trigger the expected actions.

- Performance Testing: Performance testing evaluates how the app performs under different conditions, such as varying network speeds or different device configurations. It helps identify performance bottlenecks, memory leaks, and other issues that may affect the app's responsiveness and resource usage.

- Usability Testing: Usability testing focuses on evaluating the app from a user's perspective. It involves testing how intuitive the app's interface is, how easily users can navigate through the features, and whether it meets the users' needs effectively.

- Compatibility Testing: Compatibility testing ensures that the app functions correctly across different devices, screen sizes, and Android versions. It involves testing the app on a variety of emulators and physical devices to identify any compatibility issues and ensure a consistent user experience.

- Regression Testing: Regression testing involves retesting previously tested functionalities to ensure that recent code changes or updates have not introduced new issues or caused existing features to break.

Chapter 6

CONCLUSION AND FUTURE ENHANCEMENTS

The problem of wanting everything in one space to save our time and maximize the utility. If we install different applications for different purposes then it will occupy more device memory and reduce device's efficiency. Therefore, The application is fulfilled user-needs with a Multi-Utility Application ”Student Office”

FUTURE SCOPE:

Scope of this project is very broad in terms of Hosting and Maintaining the application for student support and stay connected with the users. It is online as well as offline system. This can be a great work towards education.

Chapter 7

References

[1] Android Programming Tutorials by Mark L. Murphy

[2] Scheduler ideas :

https://developer.android.com/reference/android/app/job/JobScheduler

[3] Android CardView is a component that used to implement card layout effect. It is provided in appcompat-v7 library. Use CardView, you can add circle corner and shadow effect to the card frame. https://www.dev2qa.com/android-cardview-with-image-and-text-example/

[4] http://en.wikipedia.org/wiki/andorid

[5] Web view : https://developer.android.com/reference/android/webkit/WebViewClient