Visvesvaraya Technological University

"JnanaSangama", Belagavi – 590018, Karnataka

A

A Mini Project Report

on

"Simple Calendar"

Submitted in partial fulfillment of the requirement for the Computer Graphics Laboratory with Mini Project (18CSL68) of VI semester, Bachelor of Engineering in Computer Science and Engineering

Submitted By

Nayan V Bhandari (1GA19CS095)

Under the guidance of

Dr. Roshan Joy Martis
Associate Professor,
Dept of CSE,
GAT

Dr. Swetha P
Assistant Professor,
Dept of CSE,
GAT

GLOBAL ACADEMY OF TECHNOLOGY

Department of Computer Science and Engineering

(Accredited by NBA 2019-2022)

Rajarajeshwari Nagar, Bengaluru – 560 098

2021-2022

GLOBAL ACADEMY OF TECHNOLOGY

Department of Computer Science and Engineering

(Accredited by NBA 2019-2022)

Rajarajeshwari Nagar, Bengaluru – 560 098

CERTIFICATE

This is to certify that VI Semester Mini Project entitled "**Simple Calendar"** is a bonafide work carried out by **Nayan V Bhandari (1GA19CS095)** as a partial fulfillment for the award of Bachelors Degree in Computer Science and Engineering for Computer Graphics Laboratory with Mini Project [18CSL68] as prescribed by **Visvesvaraya Technological University**, **Belagavi** during the year 2021-2022.

- - - - - - - - - - - - - -	- - - - - - - - - - - - - -
Dr. Roshan Joy Martis Associate Professor, Dept of CSE, GAT, Bengaluru.	Dr. Swetha P Assistant Professor, Dept of CSE, GAT, Bengaluru.

External Exam

Name of the Examiner	Signature with date
1.________________________	____________________
2.________________________	____________________

GLOBAL ACADEMY OF TECHNOLOGY

Department of Computer Science and Engineering

(Accredited by NBA 2019-2022)

Rajarajeshwari Nagar, Bengaluru – 560 098

DECLARATION

I , Nayan V Bhandari, bearing USN 1GA19CS095, student of Sixth Semester B.E, Department of Computer Science and Engineering, Global Academy of Technology, Rajarajeshwarinagar Bengaluru, declare that the Mini Project entitled "**Simple Calendar**" has been carried out by me and submitted in partial fulfillment of the course requirements for the award of degree in Bachelor of Engineering in Computer Science and Engineering from Visvesvaraya Technological University, Belagavi during the academic year 2021-2022.

Nayan V Bhandari (1GA19CS095)

Place: Bengaluru

Date: 12/07/2022

ABSTRACT

This android application "**Simple Calendar**" is meant to replace the paper copies of calendars. The purpose is to help the users to schedule their appointments by having an idea of the dates are placed in the respective months of interest. It is simple, elegant and efficient like the hardcopies but much handier.

A calendar is a system of organizing days. This is done by giving names to periods of time typically days, weeks, months and years A date is the designation of a single, specific day within such a system. A calendar is also a physical record (often paper) of such a system. A calendar can also mean a list of planned events, such as a court calendar or a partly or fully chronological list of documents, such as a calendar of wills.

Periods in a calendar (such as years and months) are usually, though not necessarily, synchronized with the cycle of the sun or the moon. The most common type of pre-modern calendar was the lunisolar calendar, a lunar calendar that occasionally adds one intercalary month to remain synchronized with the solar year over the long term.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance crowned our efforts with success.
I consider myself proud, to be part of **Global Academy of Technology** family, the institution which stood by our way in endeavors.

I express my deep and sincere thanks to our Principal **Dr. N. Ranapratap Reddy** for his support.

I am grateful to **Dr. Bhagyashri R Hanji**, Professor and HOD, Dept of CSE who is source of inspiration and of invaluable help in channelizing my efforts in right direction.

I wish to thank our internal guide **Dr. Roshan Joy Martis,** Associate Professor, Dept of CSE and **Dr. Swetha P,** Assistant Professor, Dept of CSE for guiding and correcting various documents with attention and care. She has taken lot of pain to go through the document and make necessary corrections as and when needed.

I would like to thank the faculty members and supporting staff of the Department of CSE, GAT for providing all the support for completing the Project work.

Finally, I am grateful to our parents and friends for their unconditional support and help during the course of our Project work.

Nayan V Bhandari (1GA19CS095)

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION TO MOBILE APPLICATION DEVELOPMENT

Every day the new devices are incoming to the market with innovative options thanks to growing technology. The evolution of Mobile Application Development technology with new devices made our lives easier. Being obtainable on an internet-enabled device is needed for each and every business which has given the kicking start to mobile application development.

Android is the dominant player in mobile development platforms space, it was a bit later participant to the game, first being replaced in Sept 2008, virtually a year later than iOS but it has managed to achieve a reasonably massive share of the mobile market.

Technically, Android is the mobile OS with largest dominant share of the market with around 80% share compared to iOS's 18% share. Those numbers are a bit deceiving since android may be a fragmented market consisting of many different devices created by different manufacturers, running completely different versions of the Android OS.

Types of Application

1. Native Applications: These are applications developed to be used on a particular platform or operating system such as Android, iOS etc. Native apps are usually written in languages that the platform accepts. The principal advantage of native apps is that they optimize the user experience. By being designed and developed specifically for that platform, they look and perform better.
2. Hybrid Applications: These are developed to be used across the platform i.e., can be deployed on both iOS and Android platforms. Hybrid mobile applications are built in a similar manner as websites. Today, most hybrid mobile applications

leverage Apache Cordova, a platform that provides a consistent set of JavaScript APIs to access device capabilities through plug-ins, which are built with native code.

3. Progressive Applications: It is a web app that uses modern web capabilities to deliver an app-like experience to users without requiring them to install an app from Play Store. They are usually accessible by a web URL which can always be pinned or saved on phone's home screen.

1.2 ANDROID ARCHITECTURE

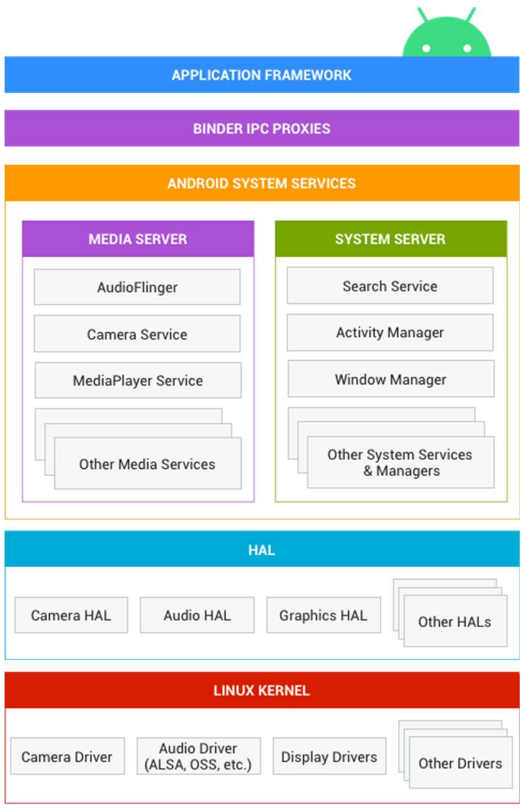


Fig 1.1: Android Architecture

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

- **Application framework:** The application framework is used most often by application developers. As a hardware developer, you should be aware of developer APIs as many maps directly to the underlying HAL interfaces and can provide helpful information about implementing drivers.

- **Binder IPC:** The Binder Inter-Process Communication mechanism allows the application framework to cross process boundaries and call into the Android system services code. This enables high level framework APIs to interact with Android system services code. This enables high level framework APIs to interact with Android system services. At the application framework level, this communication is hidden from the developer and things appear to "just work".
- **System services:** System services are modular, focused components such as Window Manager, Search Service, or Notification Manager. Functionally exposed by application framework APIs communicates with system services to access the underlying hardware. Android includes two groups of services: system and media.
- **Hardware abstraction layer (HAL):** A HAL defines a standard for interface for hardware vendors to implement, which enables Android to be agnostic about lower-level driver implementations. Using a HAL allows you to implement functionality without affecting or modifying the higher-level system. HAL implementations are packaged into modules and loaded by the Android system at the appropriate time.
- **Linux Kernel:** Developing your device drivers is similar to developing a typical Linux device driver. Android uses a version of the Linux kernel with a few special additions such as Low Memory Killer, wake locks, the Binder IPC driver, and other features important for a mobile embedded platform. These additions are primarily for system functionality and do not affect driver development.

+

CHAPTER 2

REQUIREMENTS SPECIFICATION

Requirements analysis is critical to complete a project. Requirements must be documented, actionable, measurable, testable and defined to a level of detail sufficient for system design. Requirements can be architectural, structural, behavioral, functional and non-functional. The requirements can be broken down into 2 major categories namely hardware and software requirements. The former specifies the minimal hardware facilities expected in a system in which the project has to be run. The latter specifies the essential software needed to build and run the project.

2.1 SOFTWAREREQUIREMENTS

2.1.1: Android Studio:

Android Studio provides a unified environment where you can build apps for Android phones, tablets. 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. The following features are provided in the current version:

- Gradle-based build support.
- Android-specific refactoring and quick fixes.
- Lint tools to catch performance, usability, version compatibility.
- Template-based wizards to create common Android designs and components A layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations.
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.

2.1.2: XML:

XML defines the data and is used to store and organize data. It is easily scalable and simple to develop. In Android, the XML is used to implement UI- related data, and it is a lightweight markup language that does not make layout heavy. XML only contains tags, while implementing they need to be just invoked.

2.1.3: Java:

Java is an official language Of Android development and is supported by Android Studio. It has been an official language than Kotlin, and it is popular outside of Kotlin development for many other purposes. The android platform allows developers to write managed code using Java which in turn manages and controls Android device. Android applications can be developed by using the Java programming language and the Android SDK.

2.2 HARDWARE REQUIREMENTS

- Processor –Intel i5 (7th Gen or higher)
- RAM –8GB or higher
- Memory –Minimum of 10GB
- Display –Any Display of viewable size

CHAPTER 3

SYSTEM DEFINITION

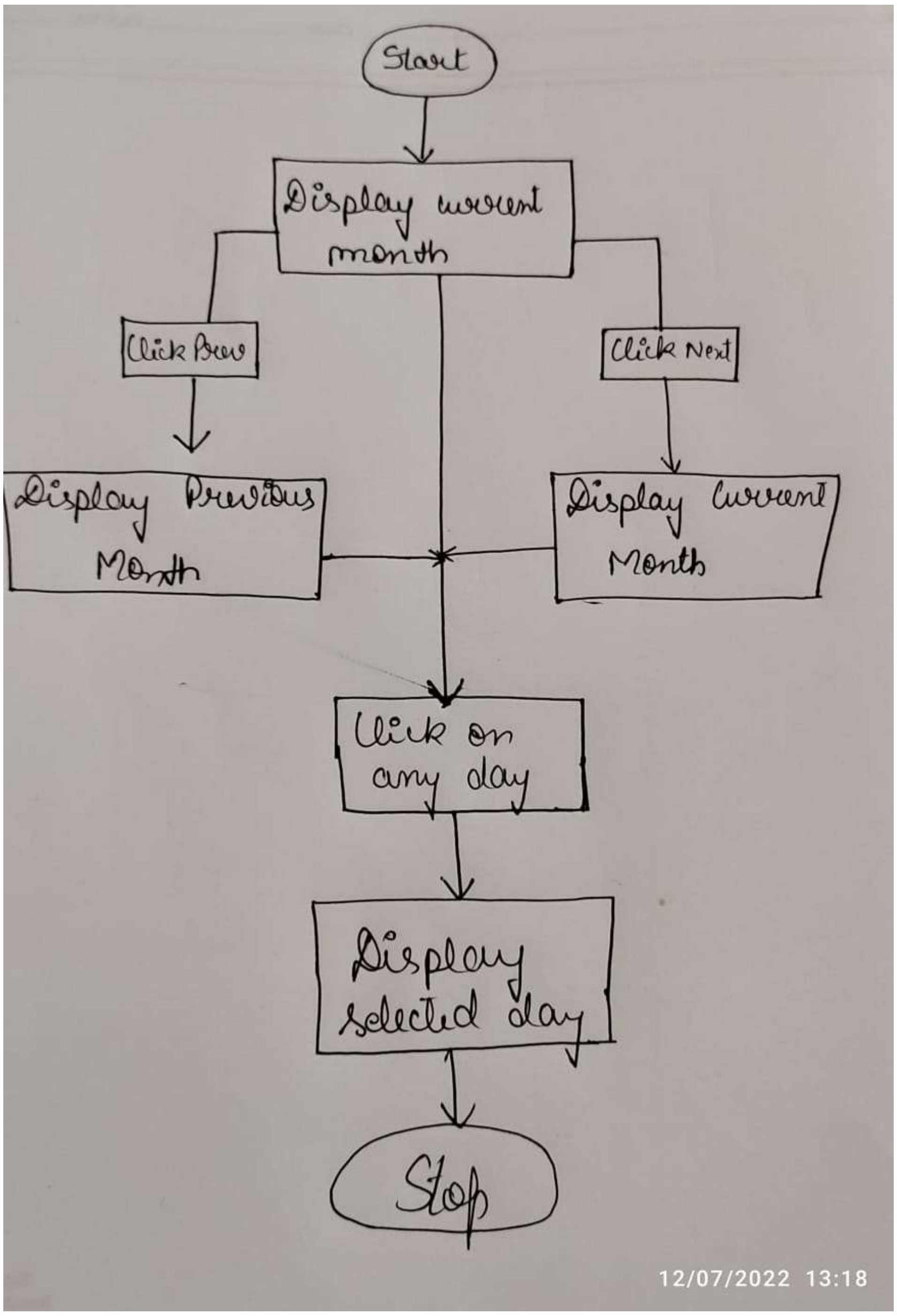


Fig 3.1: System Definition

CHAPTER 4

IMPLEMENTATION

4.1 SOURCE CODE

4.1.1: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
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:orientation="vertical"
tools:context=".MainActivity">

<LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_marginBottom="20dp"
        android:layout_marginTop="20dp">
   <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Prev"
        android:background="@null"
        android:textStyle="bold"
        android:onClick="previousMonthAction"
        android:textSize="20sp"/>

   <TextView
        android:id="@+id/monthYearTV"
        android:layout_width="0dp"
        android:layout_weight="2"
        android:layout_height="wrap_content"
        android:text="Feb 2021"
        android:textSize="20sp"
        android:textAlignment="center"
        android:textColor="@color/black"/>

  <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Next"
```

```
            android:background="@null"
            android:textStyle="bold"
            android:onClick="nextMonthAction"
            android:textSize="20sp"/>

    </LinearLayout>

    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <TextView
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="SUN"
                    android:textAlignment="center"
                    android:textSize="16sp"/>

            <TextView
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="MON"
                    android:textAlignment="center"
                    android:textSize="16sp"/>

            <TextView
               android:layout_width="0dp"
               android:layout_height="wrap_content"
               android:layout_weight="1"
               android:text="TUE"
               android:textAlignment="center"
               android:textSize="16sp"/>

          <TextView
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="WED"
                    android:textAlignment="center"
                    android:textSize="16sp"/>

          <TextView
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="THUR"
                    android:textAlignment="center"
                    android:textSize="16sp"/>
```

```
        <TextView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="FRI"
                android:textAlignment="center"
                android:textSize="16sp"/>

      <TextView
              android:layout_width="0dp"
              android:layout_height="wrap_content"
              android:layout_weight="1"
              android:text="SAT"
              android:textAlignment="center"
              android:textSize="16sp"/>

  </LinearLayout>

  <androidx.recyclerview.widget.RecyclerView
  android:id="@+id/calendarRecyclerView"
  android:layout_width="match_parent"
  android:layout_height="match_parent"/>

  </LinearLayout>
```

4.1.2: calendar_cell.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
xmlns:app="http://schemas.android.com/apk/res-auto">

<TextView
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:id="@+id/cellDayText"
   android:text="1"
   android:textSize="20sp"
   android:textColor="@color/black"
   app:layout_constraintBottom_toBottomOf="parent"
   app:layout_constraintTop_toTopOf="parent"
   app:layout_constraintRight_toRightOf="parent"
   app:layout_constraintLeft_toLeftOf="parent"
   app:layout_constraintVertical_bias="0.25"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```

4.1.3: MainActivity.java

```
package com.example.simplecalendar;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;

import java.time.LocalDate;
import java.time.YearMonth;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity implements CalendarAdapter.OnItemListener
{
        private TextView monthYearText;
        private RecyclerView calendarRecyclerView;
        private LocalDate selectedDate;

@Override
protected void onCreate(Bundle savedInstanceState)
{
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  initWidgets();
  selectedDate = LocalDate.now();
  setMonthView();
}

private void initWidgets()
{
        calendarRecyclerView = findViewById(R.id.calendarRecyclerView);
        monthYearText = findViewById(R.id.monthYearTV);
    }

    private void setMonthView()
    {
        monthYearText.setText(monthYearFromDate(selectedDate));
        ArrayList<String> daysInMonth = daysInMonthArray(selectedDate);

        CalendarAdapter calendarAdapter = new CalendarAdapter(daysInMonth, this);
        RecyclerView.LayoutManager layoutManager = new GridLayoutManager(getApplicationContext(), 7);
        calendarRecyclerView.setLayoutManager(layoutManager);
        calendarRecyclerView.setAdapter(calendarAdapter);
    }

    private ArrayList<String> daysInMonthArray(LocalDate date)
    {
        ArrayList<String> daysInMonthArray = new ArrayList<>();
        YearMonth yearMonth = YearMonth.from(date);

        int daysInMonth = yearMonth.lengthOfMonth();
        LocalDate firstOfMonth = selectedDate.withDayOfMonth(1);
        int dayOfWeek = firstOfMonth.getDayOfWeek().getValue();

       for(int i = 1; i <= 42; i++)
    {
            if(i <= dayOfWeek || i > daysInMonth + dayOfWeek)
            {
                daysInMonthArray.add("");
            }
            else
            {
                daysInMonthArray.add(String.valueOf(i - dayOfWeek));
            }
    }
    return  daysInMonthArray;
    }

    private String monthYearFromDate(LocalDate date)
    {
      DateTimeFormatter formatter = DateTimeFormatter.ofPattern("MMMM yyyy");
      return date.format(formatter);
    }

    public void previousMonthAction(View view)
    {
      selectedDate = selectedDate.minusMonths(1);
      setMonthView();
    }

    public void nextMonthAction(View view)
    {
      selectedDate = selectedDate.plusMonths(1);
      setMonthView();
    }

    @Override
    public void onItemClick(int position, String dayText)
    {
      if(!dayText.equals(""))
      {
          String message = "Selected Date " + dayText + " " + monthYearFromDate(selectedDate);
          Toast.makeText(this, message, Toast.LENGTH_LONG).show();
      }
    }
}
```

4.1.4:CalendarAdapter.java

```
package com.example.simplecalendar;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
import java.util.ArrayList;
class CalendarAdapter extends RecyclerView.Adapter<CalendarViewHolder>
{
  private final ArrayList<String> daysOfMonth;
  private final OnItemListener onItemListener;

  public CalendarAdapter(ArrayList<String> daysOfMonth, OnItemListener onItemListener)
  {
    this.daysOfMonth = daysOfMonth;
    this.onItemListener = onItemListener;
  }

  @NonNull
  @Override
  public CalendarViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)
  {
    LayoutInflater inflater = LayoutInflater.from(parent.getContext());
    View view = inflater.inflate(R.layout.calendar_cell, parent, false);
    ViewGroup.LayoutParams layoutParams = view.getLayoutParams();
    layoutParams.height = (int) (parent.getHeight() * 0.166666666);
    return new CalendarViewHolder(view, onItemListener);
  }

  @Override
  public void onBindViewHolder(@NonNull CalendarViewHolder holder, int position)
  {
    holder.dayOfMonth.setText(daysOfMonth.get(position));
  }

  @Override
  public int getItemCount()
  {
    return daysOfMonth.size();
  }

  public interface  OnItemListener
  {
    void onItemClick(int position, String dayText);
  }
}
```

4.1.5:CalendarViewHolder.java

```
package com.example.simplecalendar;
import android.view.View;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
public class CalendarViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener
{
    public final TextView dayOfMonth;
    private final CalendarAdapter.OnItemListener onItemListener;
    public CalendarViewHolder(@NonNull View itemView, CalendarAdapter.OnItemListener
onItemListener)
    {
        super(itemView);
        dayOfMonth = itemView.findViewById(R.id.cellDayText);
        this.onItemListener = onItemListener;
        itemView.setOnClickListener(this);
    }

    @Override
```

```
        public void onClick(View view)
        {
           onItemListener.onItemClick(getAdapterPosition(), (String) dayOfMonth.getText());
        }
    }
```

4.2: Layout Design

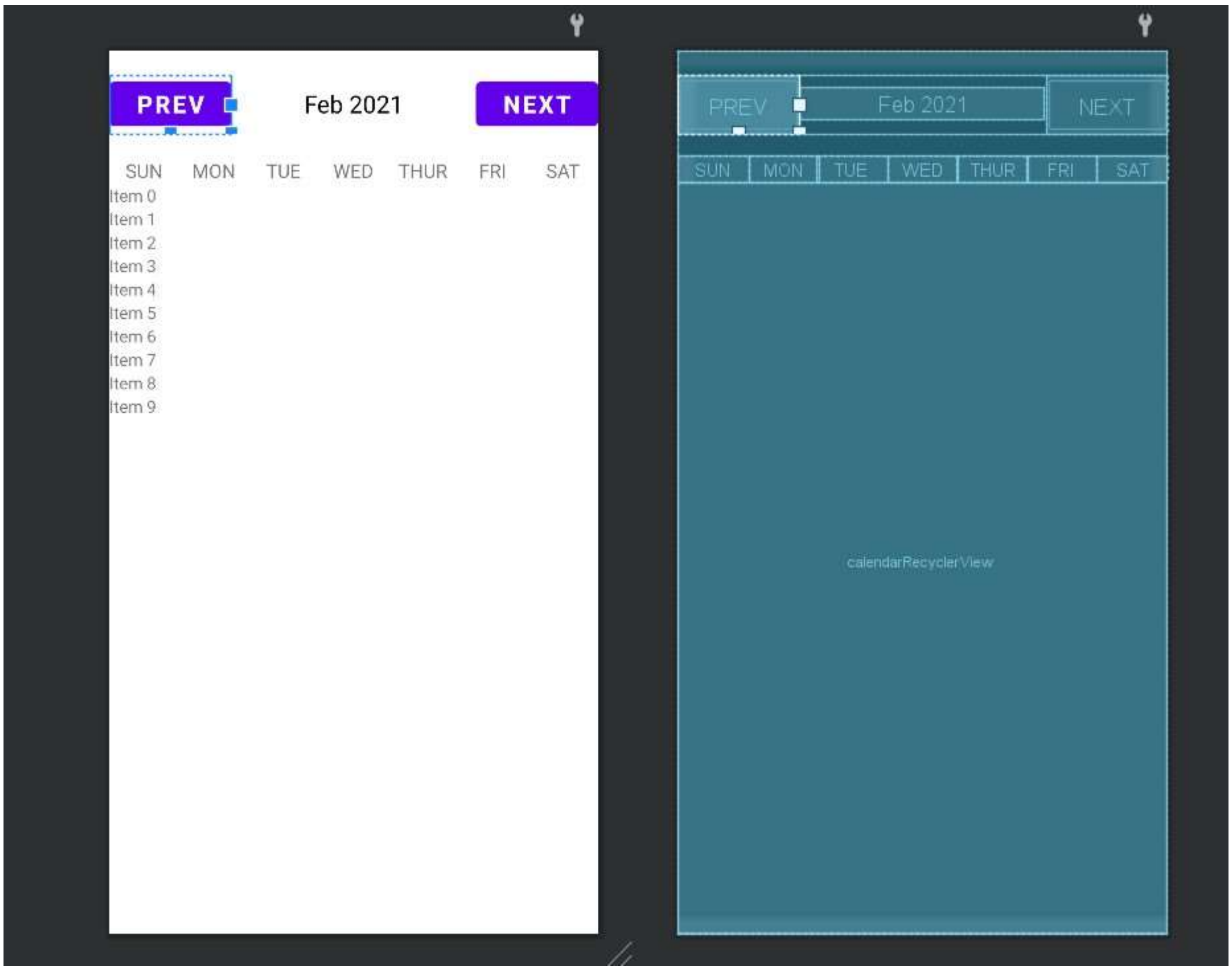


Fig 4.2.1: Layout Design

CHAPTER 5

TESTING AND RESULTS

5.1 DIFFERENT TYPES OF TESTING

1. Unit Testing

Individual components are tested to ensure that they operate correctly. Each component is tested independently, without other system components.

2. Module Testing

A module is a collection of dependent components such as a object class, an abstract Data type or some looser collection of procedures and functions. A module related Components, so can be tested without other system modules.

3. System Testing

This is concerned with finding errors that result from unanticipated interaction between Sub-system interface problems.

4. Acceptance Testing

The system is tested with data supplied by the system customer rather than simulated test data.

5.2 TEST CASES

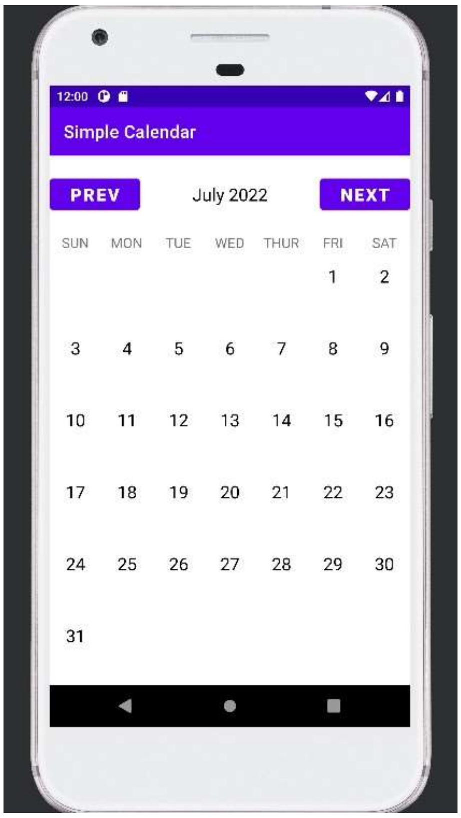
The test cases provided here test the most important features of the project.

Table 5.2.1: Test Case

SL No	Test Input	Expected Results	Observed Results	Remarks
1	Open the app	Shows the calendar	Calendar is seen	Pass
2	Click on button "PREV"	Shows previous month calendar	Shows previous month calendar	Pass
3	Click on button "NEXT"	Shows next month calendar	Shows next month calendar	Pass
4	Click on a date	Shows the selected data	Shows the selected data	Pass

CHAPTER 6
SNAPSHOTS

Snapshots of Simple calendar:

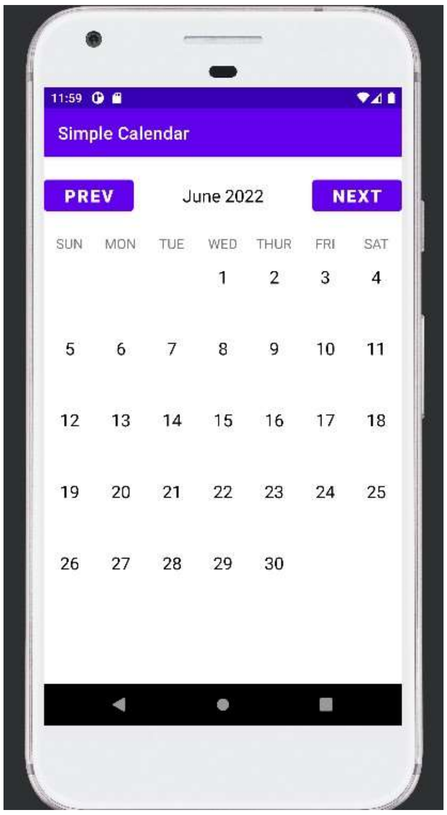

12:00

Simple Calendar

PREV July 2022 NEXT

SUN	MON	TUE	WED	THUR	FRI	SAT
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

Fig 6.1: On opening the application

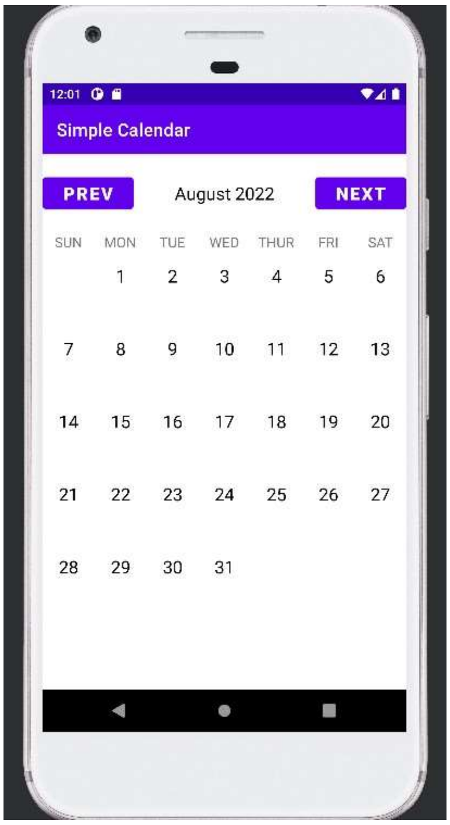

11:59

Simple Calendar

PREV June 2022 NEXT

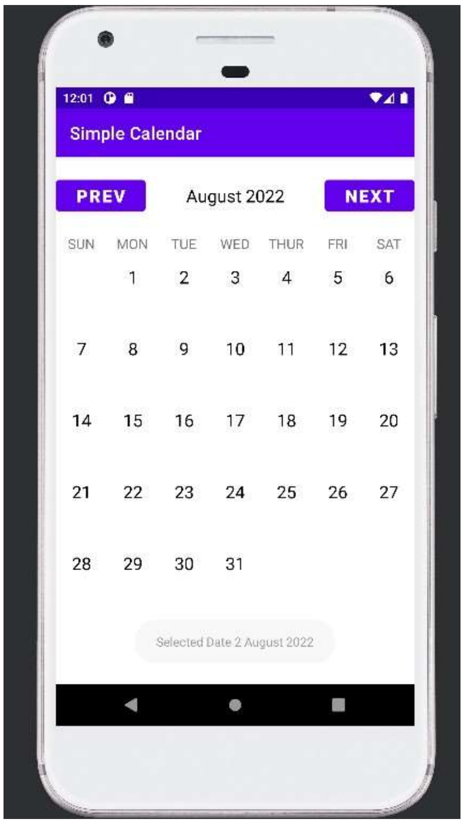
SUN	MON	TUE	WED	THUR	FRI	SAT
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Fig 6.2: On clicking the prev button



SUN	MON	TUE	WED	THUR	FRI	SAT
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

Fig 6.3: On clicking the next button



SUN	MON	TUE	WED	THUR	FRI	SAT
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

Selected Date 2 August 2022

Fig 6.4: On clicking on a date

CONCLUSION

Thus, in this project we have acquired a lot of knowledge about various techniques in XML and Java programming. We have explored many new concepts. The outcome of the experiments shows that technology is quite reliable and can be implemented in different applications on different platforms. The application created during research can be used for both scientific and entertainment purposes. Thus we thank the Principal, the HOD and the faculty for their continuous support and belief in us.

BIBLIOGRAPHY

1. Android developer fundamentals course-concepts:
2. Erik Hellman. "ANDROID PROGRAMMING-PUSHING THE LIMITS".1" edition. India Pvt ltd,2014.
3. Dawn Griffiths and David Griffiths, "HEAD FIRST ANDROID DEVELOPER", 1" edition, O'Reilly SPD publishers 2015.
4. W.-Y. Chen, P.-J. Lin, and D.-Y. Kuo, "Dice image recognition scheme using pattern comparison technique," in in Proc. Int. Symp. Computer Consumer Control, Taichung, Taiwan, June 4-6 2012, pp.
5. http://www.developers.android.com/