

# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

## Jnana Sangama, Belagavi – 590018

**Mobile Application Development**

**mini project report**

**on**

**“To-Do List Application”**

Submitted in partial fulfillment of the requirements for the award of the degree of

**BACHELOR OF ENGINEERING**

**in**

**INFORMATION SCIENCE & ENGINEERING**

## By

**SRAJAN S SHETTY 4MT20IS045**

## Under the Guidance of

**Mr. Manjunath H**

**Associate Professor & Head of the Department**

**DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING**

*Accrediated by NBA*

## MANGALORE INSTITUTE OF TECHNOLOGY&ENGINEERING

**Badaga Mijar, Moodabidri-574225, Karnataka 2022-2023**



## MANGALORE INSTITUTE OF TECHNOLOGY & ENGINEERING

**Badaga Mijar, Moodabidri-574225, Karnataka**

**DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING**

*Accrediated by NBA*

**CERTIFICATE**

This is to certify that **Mr. Srajan S Shetty** (4MT20IS045), has satisfactorily completed the mini project entitled “**TO-DO LIST APPLICATION**” for the Mobile Application Development (18CSMP68) lab as prescribed by the VTU for 6th semester B.E. Information Science and Engineering branch for the academic year 2022 – 2023.

………………………… ………………………

Signature of the Guide Signature of the HOD

**Prof. Manjunath H Prof. Manjunath H**

**Name of Examiners Signature of the Examiners**

1. ………………………….. …………………………..

2. ………………………….. ………………………….

# ABSTRACT

In today's fast-paced digital era, managing tasks and staying organized has become increasingly challenging. As a result, online to-do lists have emerged as a popular solution to help individuals keep track of their responsibilities and prioritize their activities effectively. These virtual task management tools provide a convenient platform for creating, organizing, and updating tasks in a centralized location accessible from various devices. By leveraging the power of the internet and cloud storage, online to-do lists enable users to access their tasks anytime, anywhere, fostering flexibility and productivity. With features such as due dates, reminders, categorization, and collaboration options, these digital tools offer enhanced functionality and convenience compared to traditional pen-and-paper lists. Additionally, some online to-do lists integrate with calendars, email clients, and other applications, streamlining workflows and ensuring seamless task management across different platforms. Furthermore, these tools often offer the ability to set priorities, assign tasks to specific individuals, and track progress, empowering users to efficiently manage both personal and professional responsibilities. Overall, online to-do lists have revolutionized task management by providing an intuitive, accessible, and feature-rich solution to help individuals stay organized, meet deadlines, and achieve their goals in today's interconnected world.

# ACKNOWLEDGEMENT

The satisfaction and the successful completion of this mini project would be incomplete without the mention of the people who made it possible, whose constant guidance encouragement crowned our efforts with success.

This Mini project is made under the guidance of **Prof.Manjunath H**, Professor, in the Department of Information Science and Engineering. I would like to express my sincere gratitude to my guide for all the helping hand and guidance in this project

I would like to thank my Project Coordinator and Head of Department **Prof.Manjunath H** Department of Information Science and Engineering for her constant encouragement and support extended throughout

I would like to thank our Principal **Dr.Prashanth CM**, for encouraging me and giving me an opportunity to accomplish the mini project.

I also thank our management who helped me directly or indirectly in the completion of this mini project.

My special thanks to faculty members and others for their constant help and support. Above all, I extend my sincere gratitude to my parents and friends for their constant encouragement with moral support.

SRAJAN S SHETTY (4MT20IS045)

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **Chapter no.** | **Chapters** | **PageNo.** |
|  | Abstract | i |
|  | Acknowledgement | ii |
|  | Table of contents | iii |
|  | List of figures | iv |
| 1 | Introduction | 1 |
| 2 | Technologies used | 2 |
| 3 | Design | 4 |
| 4 | Implementation | 5 |
| 5 | Result and Description | 22 |
| 6  7 | Conclusion  Reference | 25  26 |

# 

# 

**LIST OF FIGURES**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fig.No** | | **Fig.Name** | | **Page No.** | |
| 3.1 | | Flow Chart | | 4 | |
| 5.1 | | Welcome page | | 22 | |
| 5.2 | | Dashboard | | 22 | |
| 5.3 | | Add task | | 23 | |
| 5.4 | | Complete task | | 23 | |
| 5.5 | | Calendar | | 24 | |
| 5.6 | | Notification | | 24 | |

**Chapter 1**

# INTRODUCTION

In an increasingly digital and fast-paced world, managing our daily tasks and responsibilities can be a daunting challenge. From personal to professional commitments, it is essential to stay organized and efficient to ensure productivity and success. Thankfully, the advent of online to-do list applications has transformed the way we manage our tasks, offering a convenient and streamlined solution. An online to-do list application is a digital tool designed to help individuals and teams organize, prioritize, and track their tasks in a centralized and accessible manner. With just a few clicks, users can create, edit, and manage their to-do lists using these intuitive and user-friendly applications. One of the key advantages of an online to-do list application is its accessibility. Unlike traditional pen-and-paper lists, these digital tools can be accessed from various devices, including computers, smartphones, and tablets. This accessibility allows users to stay updated and make modifications on the go, ensuring that important tasks are never forgotten or overlooked. Additionally, some online to-do lists integrate with calendars, email clients, and other applications, streamlining workflows and ensuring seamless task management across different platforms. Furthermore, these tools often offer the ability to set priorities, assign tasks to specific individuals, and track progress, empowering users to efficiently manage both personal and professional responsibilities. Overall, online to-do lists have revolutionized task management by providing an intuitive, accessible, and feature-rich solution to help individuals stay organized, meet deadlines, and achieve their goals in today's interconnected world.

**Chapter 2**

# TECHNOLOGIES USED

## 1.1 FRONT END : XML

Extensible Markup Language (XML) allows you to describe and organize information in ways that are easily understandable by both humans and computers. You can then share that information and its description with others over the Internet, an extranet, network, or in other ways like Standard Generalized Markup Language (SGML), is a metalanguage. A metalanguage all you to define a document markup language and its structure. For example, both XML and Hypertext Markup Language (HTML) are derived from SGML. You can use XML to create your own markup language that includes a set of rules and tags that describe information suits to your needs, for example, name, title, address, and zip code. You define this markup language in a document type definition (DTD) or XML Schema file that functions as the standard way to describe your information. Using XML to share standardized information me you are no longer required to write programs to focus on proprietary software or convert and translate different data formats .

## 1.2 BACK END : JAVA

Java is a high-level and purely [object oriented programming language](https://www.interviewbit.com/java-interview-questions/). It is platform independent, robust, secure, and multithreaded programming language which makes it popular among other OOP languages. It is widely used for software, web, and mobile application development, along with this it is also used in big data analytics and server-side technology. Before moving towards features of Java, let us see how Java originated.  The first and foremost reason is the platform-independent feature of Java. Platform independence means programmers who developed their software in one platform can execute it on any platform or operating system regardless of underlying configuration, because of Java Virtual Machine. In other words, we can say that Java follows the WORA rule which says Write Once Run Anywhere. Besides that Java provides security over eavesdropping, tampering, impersonation, and virus threat*.* Java also provides a multi-threading feature.

## IDE : ANDRIOD STUDIO

 Android Studio is Google's official integrated development environment (IDE), built on JetBrains' IntelliJ IDEA software and developed exclusively for Android programming. From 2020, it was available for download on Windows, macOS, and Linux-based operating systems, as well as a subscription-based service. It is the major IDE for native Android application development, replacing the Eclipse Android Development Tools (E-ADT). Android Studio was announced during the Google I/O conference on May 16, 2013. It was in the early access preview stage beginning with version 0.1 in May 2013, then entered the beta stage beginning with version 0.8 in June 2014. Starting with version 1.0, the first stable build was released in December 2014.

,

**Chapter 3**

# DESIGN

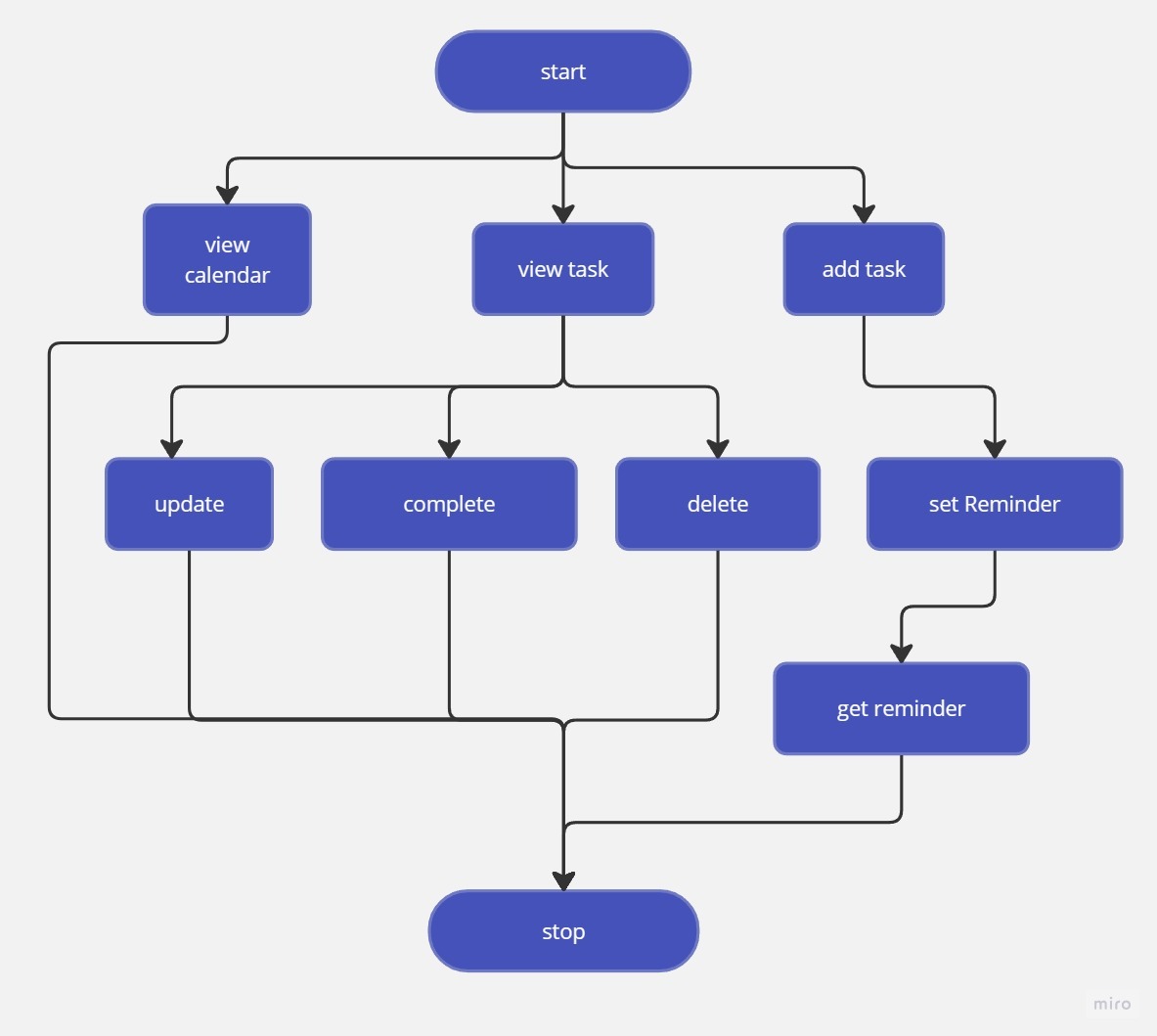
## FLOW CHART DIAGRAM

A diagram of the sequence of movements or actions of people or things involved in a

Complex system or activity.

A graphical representation of a computer program in relation to its sequence of functions

(as distinct from the data it processes).



**Figure.3.1 Flow Chart Diagram**

## CHAPTER 4

## IMPLEMENTATION

**4.1 Frontend**

**4.1.1 Main Activity:**

<?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:paddingRight="20dp"

android:orientation="vertical"

android:background="@color/colorPrimary"

android:paddingLeft="20dp"

tools:context=".activity.MainActivity">

<FrameLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Hello"

android:layout\_marginTop="20dp"

android:textColor="@color/colorAccent"

android:textSize="18sp"/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="All tasks"

fontPath="fonts/nunito\_extra\_bold.ttf"

android:textStyle="bold"

android:textColor="@color/colorAccent"

android:textSize="28sp"

tools:ignore="MissingPrefix" />

</LinearLayout>

<ImageView

android:layout\_width="30dp"

android:layout\_height="30dp"

android:layout\_gravity="right|center"

android:background="@drawable/calendar"

android:id="@+id/calendar"/>

</FrameLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginTop="20dp"

android:orientation="vertical">

<FrameLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_weight="1">

<ImageView

android:layout\_width="300dp"

android:layout\_height="300dp"

android:id="@+id/noDataImage"

android:layout\_gravity="center"/>

<androidx.recyclerview.widget.RecyclerView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/taskRecycler"/>

</FrameLayout>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="70dp"

android:text="Add Task"

android:drawablePadding="10dp"

android:paddingTop="10dp"

android:id="@+id/addTask"

android:layout\_gravity="center"

android:textColor="@color/colorAccent"

android:textSize="20sp"

android:drawableLeft="@drawable/ic\_add\_black\_24dp"

android:textAlignment="center"

android:layout\_weight="0.5"

android:layout\_marginBottom="20dp"/>

</LinearLayout>

</LinearLayout>

## Backend Implementation

* + 1. **Alarm activity:**

package com.codegama.todolistapplication.activity;

import android.media.MediaPlayer;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import com.bumptech.glide.Glide;

import com.codegama.todolistapplication.R;

import butterknife.BindView;

import butterknife.ButterKnife;

public class AlarmActivity extends BaseActivity {

private static AlarmActivity inst;

@BindView(R.id.imageView)

ImageView imageView;

@BindView(R.id.title)

TextView title;

@BindView(R.id.description)

TextView description;

@BindView(R.id.timeAndData)

TextView timeAndData;

@BindView(R.id.closeButton)

Button closeButton;

MediaPlayer mediaPlayer;

public static AlarmActivity instance() {

return inst;

}

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_alarm);

ButterKnife.bind(this);

mediaPlayer = MediaPlayer.create(getApplicationContext(), R.raw.notification);

mediaPlayer.start();

if(getIntent().getExtras() != null) {

title.setText(getIntent().getStringExtra("TITLE"));

description.setText(getIntent().getStringExtra("DESC"));

timeAndData.setText(getIntent().getStringExtra("DATE") + ", " + getIntent().getStringExtra("TIME"));

}

Glide.with(getApplicationContext()).load(R.drawable.alert).into(imageView);

closeButton.setOnClickListener(view -> finish());

}

@Override

protected void onDestroy() {

super.onDestroy();

mediaPlayer.release();

    }

}

## Base Activity

## package com.codegama.todolistapplication.activity;

## import android.content.Context;

## import android.content.res.Resources;

## import android.os.Bundle;

## import androidx.annotation.Nullable;

## import androidx.appcompat.app.AppCompatActivity;

## import java.util.Locale;

## import io.github.inflationx.viewpump.ViewPumpContextWrapper;

## public class BaseActivity extends AppCompatActivity {

## @Override

## protected void onCreate(@Nullable Bundle savedInstanceState) {

## super.onCreate(savedInstanceState);

## }

## @Override

## public void onResume() {

## super.onResume();

## }

## @Override

## protected void attachBaseContext(Context newBase) {

## super.attachBaseContext(ViewPumpContextWrapper.wrap(newBase));

## }

## }

## Main Activity

## package com.codegama.todolistapplication.activity;

## import android.content.ComponentName;

## import android.content.pm.PackageManager;

## import android.os.AsyncTask;

## import android.os.Bundle;

## import android.view.View;

## import android.view.WindowManager;

## import android.widget.ImageView;

## import android.widget.LinearLayout;

## import android.widget.TextView;

## import androidx.recyclerview.widget.LinearLayoutManager;

## import androidx.recyclerview.widget.RecyclerView;

## import com.bumptech.glide.Glide;

## import com.codegama.todolistapplication.R;

## import com.codegama.todolistapplication.adapter.TaskAdapter;

## import com.codegama.todolistapplication.bottomSheetFragment.CreateTaskBottomSheetFragment;

## import com.codegama.todolistapplication.bottomSheetFragment.ShowCalendarViewBottomSheet;

## import com.codegama.todolistapplication.broadcastReceiver.AlarmBroadcastReceiver;

## import com.codegama.todolistapplication.database.DatabaseClient;

## import com.codegama.todolistapplication.model.Task;

## import java.util.ArrayList;

## import java.util.List;

## import butterknife.BindView;

## import butterknife.ButterKnife;

## public class MainActivity extends BaseActivity implements CreateTaskBottomSheetFragment.setRefreshListener {

## @BindView(R.id.taskRecycler)

## RecyclerView taskRecycler;

## @BindView(R.id.addTask)

## TextView addTask;

## TaskAdapter taskAdapter;

## List<Task> tasks = new ArrayList<>();

## @BindView(R.id.noDataImage)

## ImageView noDataImage;

## @BindView(R.id.calendar)

## ImageView calendar;

## @Override

## protected void onCreate(Bundle savedInstanceState) {

## super.onCreate(savedInstanceState);

## setContentView(R.layout.activity\_main);

## ButterKnife.bind(this);

## setUpAdapter();

## getWindow().addFlags(WindowManager.LayoutParams.FLAG\_KEEP\_SCREEN\_ON);

## ComponentName receiver = new ComponentName(this, AlarmBroadcastReceiver.class);

## PackageManager pm = getPackageManager();

## pm.setComponentEnabledSetting(receiver, PackageManager.COMPONENT\_ENABLED\_STATE\_ENABLED, PackageManager.DONT\_KILL\_APP);

## Glide.with(getApplicationContext()).load(R.drawable.first\_note).into(noDataImage);

## addTask.setOnClickListener(view -> {

## CreateTaskBottomSheetFragment createTaskBottomSheetFragment = new CreateTaskBottomSheetFragment();

## createTaskBottomSheetFragment.setTaskId(0, false, this, MainActivity.this);

## createTaskBottomSheetFragment.show(getSupportFragmentManager(), createTaskBottomSheetFragment.getTag());

## });

## getSavedTasks();

## calendar.setOnClickListener(view -> {

## ShowCalendarViewBottomSheet showCalendarViewBottomSheet = new ShowCalendarViewBottomSheet();

## showCalendarViewBottomSheet.show(getSupportFragmentManager(), showCalendarViewBottomSheet.getTag());

## });

## }

## public void setUpAdapter() {

## taskAdapter = new TaskAdapter(this, tasks, this);

## taskRecycler.setLayoutManager(new LinearLayoutManager(getApplicationContext()));

## taskRecycler.setAdapter(taskAdapter);

## }

## private void getSavedTasks() {

## class GetSavedTasks extends AsyncTask<Void, Void, List<Task>> {

## @Override

## protected List<Task> doInBackground(Void... voids) {

## tasks = DatabaseClient

## .getInstance(getApplicationContext())

## .getAppDatabase()

## .dataBaseAction()

## .getAllTasksList();

## return tasks;

## }

## @Override

## protected void onPostExecute(List<Task> tasks) {

## super.onPostExecute(tasks);

## noDataImage.setVisibility(tasks.isEmpty() ? View.VISIBLE : View.GONE);

## setUpAdapter();

## }

## }

## GetSavedTasks savedTasks = new GetSavedTasks();

## savedTasks.execute();

## }

## @Override

## public void refresh() {

## getSavedTasks();

## }

## }

## Show calendar

package com.codegama.todolistapplication.bottomSheetFragment;

import android.annotation.SuppressLint;

import android.app.Dialog;

import android.os.AsyncTask;

import android.os.Build;

import android.view.View;

import android.widget.ImageView;

import androidx.annotation.NonNull;

import androidx.annotation.RequiresApi;

import com.applandeo.materialcalendarview.CalendarView;

import com.applandeo.materialcalendarview.EventDay;

import com.codegama.todolistapplication.R;

import com.codegama.todolistapplication.activity.MainActivity;

import com.codegama.todolistapplication.database.DatabaseClient;

import com.codegama.todolistapplication.model.Task;

import com.google.android.material.bottomsheet.BottomSheetBehavior;

import com.google.android.material.bottomsheet.BottomSheetDialogFragment;

import com.google.android.material.drawable.DrawableUtils;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.GregorianCalendar;

import java.util.List;

import butterknife.BindView;

import butterknife.ButterKnife;

import butterknife.Unbinder;

public class ShowCalendarViewBottomSheet extends BottomSheetDialogFragment {

Unbinder unbinder;

MainActivity activity;

@BindView(R.id.back)

ImageView back;

@BindView(R.id.calendarView)

CalendarView calendarView;

List<Task> tasks = new ArrayList<>();

private BottomSheetBehavior.BottomSheetCallback mBottomSheetBehaviorCallback = new BottomSheetBehavior.BottomSheetCallback() {

@Override

public void onStateChanged(@NonNull View bottomSheet, int newState) {

if (newState == BottomSheetBehavior.STATE\_HIDDEN) {

dismiss();

}

}

@Override

public void onSlide(@NonNull View bottomSheet, float slideOffset) {

}

};

@RequiresApi(api = Build.VERSION\_CODES.O)

@SuppressLint({"RestrictedApi", "ClickableViewAccessibility"})

@Override

public void setupDialog(Dialog dialog, int style) {

super.setupDialog(dialog, style);

View contentView = View.inflate(getContext(), R.layout.fragment\_calendar\_view, null);

unbinder = ButterKnife.bind(this, contentView);

dialog.setContentView(contentView);

calendarView.setHeaderColor(R.color.colorAccent);

getSavedTasks();

back.setOnClickListener(view -> dialog.dismiss());

}

@Override

public void onDestroyView() {

super.onDestroyView();

}

private void getSavedTasks() {

class GetSavedTasks extends AsyncTask<Void, Void, List<Task>> {

@Override

protected List<Task> doInBackground(Void... voids) {

tasks = DatabaseClient

.getInstance(getActivity())

.getAppDatabase()

.dataBaseAction()

.getAllTasksList();

return tasks;

}

@Override

protected void onPostExecute(List<Task> tasks) {

super.onPostExecute(tasks);

calendarView.setEvents(getHighlitedDays());

}

}

GetSavedTasks savedTasks = new GetSavedTasks();

savedTasks.execute();

}

public List<EventDay> getHighlitedDays() {

List<EventDay> events = new ArrayList<>();

for(int i = 0; i < tasks.size(); i++) {

Calendar calendar = Calendar.getInstance();

String[] items1 = tasks.get(i).getDate().split("-");

String dd = items1[0];

String month = items1[1];

String year = items1[2];

calendar.set(Calendar.DAY\_OF\_MONTH, Integer.parseInt(dd));

calendar.set(Calendar.MONTH, Integer.parseInt(month) - 1);

calendar.set(Calendar.YEAR, Integer.parseInt(year));

events.add(new EventDay(calendar, R.drawable.dot));

}

return events;

    }

}

* + 1. **Create task**

package com.codegama.todolistapplication.bottomSheetFragment;

import android.annotation.SuppressLint;

import android.app.AlarmManager;

import android.app.DatePickerDialog;

import android.app.Dialog;

import android.app.NotificationChannel;

import android.app.NotificationManager;

import android.app.PendingIntent;

import android.app.TimePickerDialog;

import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.DialogInterface;

import android.content.Intent;

import android.os.AsyncTask;

import android.os.Build;

import android.os.SystemClock;

import android.util.Log;

import android.view.MotionEvent;

import android.view.View;

import android.widget.Button;

import android.widget.DatePicker;

import android.widget.EditText;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.annotation.RequiresApi;

import androidx.core.app.NotificationCompat;

import com.codegama.todolistapplication.R;

import com.codegama.todolistapplication.activity.AlarmActivity;

import com.codegama.todolistapplication.activity.MainActivity;

import com.codegama.todolistapplication.broadcastReceiver.AlarmBroadcastReceiver;

import com.codegama.todolistapplication.database.DatabaseClient;

import com.codegama.todolistapplication.model.Task;

import com.google.android.material.bottomsheet.BottomSheetBehavior;

import com.google.android.material.bottomsheet.BottomSheetDialogFragment;

import com.zubair.alarmmanager.builder.AlarmBuilder;

import com.zubair.alarmmanager.enums.AlarmType;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

import java.util.GregorianCalendar;

import java.util.Locale;

import butterknife.BindView;

import butterknife.ButterKnife;

import butterknife.Unbinder;

import static android.content.Context.ALARM\_SERVICE;

public class CreateTaskBottomSheetFragment extends BottomSheetDialogFragment {

Unbinder unbinder;

@BindView(R.id.addTaskTitle)

EditText addTaskTitle;

@BindView(R.id.addTaskDescription)

EditText addTaskDescription;

@BindView(R.id.taskDate)

EditText taskDate;

@BindView(R.id.taskTime)

EditText taskTime;

@BindView(R.id.taskEvent)

EditText taskEvent;

@BindView(R.id.addTask)

Button addTask;

int taskId;

boolean isEdit;

Task task;

int mYear, mMonth, mDay;

int mHour, mMinute;

setRefreshListener setRefreshListener;

AlarmManager alarmManager;

TimePickerDialog timePickerDialog;

DatePickerDialog datePickerDialog;

MainActivity activity;

public static int count = 0;

private BottomSheetBehavior.BottomSheetCallback mBottomSheetBehaviorCallback = new BottomSheetBehavior.BottomSheetCallback() {

@Override

public void onStateChanged(@NonNull View bottomSheet, int newState) {

if (newState == BottomSheetBehavior.STATE\_HIDDEN) {

dismiss();

}

}

@Override

public void onSlide(@NonNull View bottomSheet, float slideOffset) {

}

};

public void setTaskId(int taskId, boolean isEdit, setRefreshListener setRefreshListener, MainActivity activity) {

this.taskId = taskId;

this.isEdit = isEdit;

this.activity = activity;

this.setRefreshListener = setRefreshListener;

}

@RequiresApi(api = Build.VERSION\_CODES.O)

@SuppressLint({"RestrictedApi", "ClickableViewAccessibility"})

@Override

public void setupDialog(Dialog dialog, int style) {

super.setupDialog(dialog, style);

View contentView = View.inflate(getContext(), R.layout.fragment\_create\_task, null);

unbinder = ButterKnife.bind(this, contentView);

dialog.setContentView(contentView);

alarmManager = (AlarmManager) getActivity().getSystemService(ALARM\_SERVICE);

addTask.setOnClickListener(view -> {

if(validateFields())

createTask();

});

if (isEdit) {

showTaskFromId();

}

taskDate.setOnTouchListener((view, motionEvent) -> {

if(motionEvent.getAction() == MotionEvent.ACTION\_UP) {

final Calendar c = Calendar.getInstance();

mYear = c.get(Calendar.YEAR);

mMonth = c.get(Calendar.MONTH);

mDay = c.get(Calendar.DAY\_OF\_MONTH);

datePickerDialog = new DatePickerDialog(getActivity(),

(view1, year, monthOfYear, dayOfMonth) -> {

taskDate.setText(dayOfMonth + "-" + (monthOfYear + 1) + "-" + year);

datePickerDialog.dismiss();

}, mYear, mMonth, mDay);

datePickerDialog.getDatePicker().setMinDate(System.currentTimeMillis() - 1000);

datePickerDialog.show();

}

return true;

});

taskTime.setOnTouchListener((view, motionEvent) -> {

if(motionEvent.getAction() == MotionEvent.ACTION\_UP) {

// Get Current Time

final Calendar c = Calendar.getInstance();

mHour = c.get(Calendar.HOUR\_OF\_DAY);

mMinute = c.get(Calendar.MINUTE);

// Launch Time Picker Dialog

timePickerDialog = new TimePickerDialog(getActivity(),

(view12, hourOfDay, minute) -> {

taskTime.setText(hourOfDay + ":" + minute);

timePickerDialog.dismiss();

}, mHour, mMinute, false);

timePickerDialog.show();

}

return true;

});

}

public boolean validateFields() {

if(addTaskTitle.getText().toString().equalsIgnoreCase("")) {

Toast.makeText(activity, "Please enter a valid title", Toast.LENGTH\_SHORT).show();

return false;

}

else if(addTaskDescription.getText().toString().equalsIgnoreCase("")) {

Toast.makeText(activity, "Please enter a valid description", Toast.LENGTH\_SHORT).show();

return false;

}

else if(taskDate.getText().toString().equalsIgnoreCase("")) {

Toast.makeText(activity, "Please enter date", Toast.LENGTH\_SHORT).show();

return false;

}

else if(taskTime.getText().toString().equalsIgnoreCase("")) {

Toast.makeText(activity, "Please enter time", Toast.LENGTH\_SHORT).show();

return false;

}

else if(taskEvent.getText().toString().equalsIgnoreCase("")) {

Toast.makeText(activity, "Please enter an event", Toast.LENGTH\_SHORT).show();

return false;

}

else {

return true;

}

}

@Override

public void onDestroyView() {

super.onDestroyView();

}

private void createTask() {

class saveTaskInBackend extends AsyncTask<Void, Void, Void> {

@SuppressLint("WrongThread")

@Override

protected Void doInBackground(Void... voids) {

Task createTask = new Task();

createTask.setTaskTitle(addTaskTitle.getText().toString());

createTask.setTaskDescrption(addTaskDescription.getText().toString());

createTask.setDate(taskDate.getText().toString());

createTask.setLastAlarm(taskTime.getText().toString());

createTask.setEvent(taskEvent.getText().toString());

if (!isEdit)

DatabaseClient.getInstance(getActivity()).getAppDatabase()

.dataBaseAction()

.insertDataIntoTaskList(createTask);

else

DatabaseClient.getInstance(getActivity()).getAppDatabase()

.dataBaseAction()

.updateAnExistingRow(taskId, addTaskTitle.getText().toString(),

addTaskDescription.getText().toString(),

taskDate.getText().toString(),

taskTime.getText().toString(),

taskEvent.getText().toString());

return null;

}

@Override

protected void onPostExecute(Void aVoid) {

super.onPostExecute(aVoid);

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {

createAnAlarm();

}

setRefreshListener.refresh();

Toast.makeText(getActivity(), "Your event is been added", Toast.LENGTH\_SHORT).show();

dismiss();

}

}

saveTaskInBackend st = new saveTaskInBackend();

st.execute();

}

@RequiresApi(api = Build.VERSION\_CODES.M)

public void createAnAlarm() {

try {

String[] items1 = taskDate.getText().toString().split("-");

String dd = items1[0];

String month = items1[1];

String year = items1[2];

String[] itemTime = taskTime.getText().toString().split(":");

String hour = itemTime[0];

String min = itemTime[1];

Calendar cur\_cal = new GregorianCalendar();

cur\_cal.setTimeInMillis(System.currentTimeMillis());

Calendar cal = new GregorianCalendar();

cal.set(Calendar.HOUR\_OF\_DAY, Integer.parseInt(hour));

cal.set(Calendar.MINUTE, Integer.parseInt(min));

cal.set(Calendar.SECOND, 0);

cal.set(Calendar.MILLISECOND, 0);

cal.set(Calendar.DATE, Integer.parseInt(dd));

Intent alarmIntent = new Intent(activity, AlarmBroadcastReceiver.class);

alarmIntent.putExtra("TITLE", addTaskTitle.getText().toString());

alarmIntent.putExtra("DESC", addTaskDescription.getText().toString());

alarmIntent.putExtra("DATE", taskDate.getText().toString());

alarmIntent.putExtra("TIME", taskTime.getText().toString());

PendingIntent pendingIntent = PendingIntent.getBroadcast(activity,count, alarmIntent, PendingIntent.FLAG\_UPDATE\_CURRENT);

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {

alarmManager.setAndAllowWhileIdle(AlarmManager.RTC\_WAKEUP, cal.getTimeInMillis(), pendingIntent);

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.KITKAT) {

alarmManager.setExact(AlarmManager.RTC\_WAKEUP, cal.getTimeInMillis(), pendingIntent);

} else {

alarmManager.set(AlarmManager.RTC\_WAKEUP, cal.getTimeInMillis(), pendingIntent);

}

count ++;

PendingIntent intent = PendingIntent.getBroadcast(activity, count, alarmIntent, 0);

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {

alarmManager.setAndAllowWhileIdle(AlarmManager.RTC\_WAKEUP, cal.getTimeInMillis() - 600000, intent);

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.KITKAT) {

alarmManager.setExact(AlarmManager.RTC\_WAKEUP, cal.getTimeInMillis() - 600000, intent);

} else {

alarmManager.set(AlarmManager.RTC\_WAKEUP, cal.getTimeInMillis() - 600000, intent);

}

}

count ++;

}

} catch (Exception e) {

e.printStackTrace();

}

}

private void showTaskFromId() {

class showTaskFromId extends AsyncTask<Void, Void, Void> {

@SuppressLint("WrongThread")

@Override

protected Void doInBackground(Void... voids) {

task = DatabaseClient.getInstance(getActivity()).getAppDatabase()

.dataBaseAction().selectDataFromAnId(taskId);

return null;

}

@Override

protected void onPostExecute(Void aVoid) {

super.onPostExecute(aVoid);

setDataInUI();

}

}

showTaskFromId st = new showTaskFromId();

st.execute();

}

private void setDataInUI() {

addTaskTitle.setText(task.getTaskTitle());

addTaskDescription.setText(task.getTaskDescrption());

taskDate.setText(task.getDate());

taskTime.setText(task.getLastAlarm());

taskEvent.setText(task.getEvent());

}

public interface setRefreshListener {

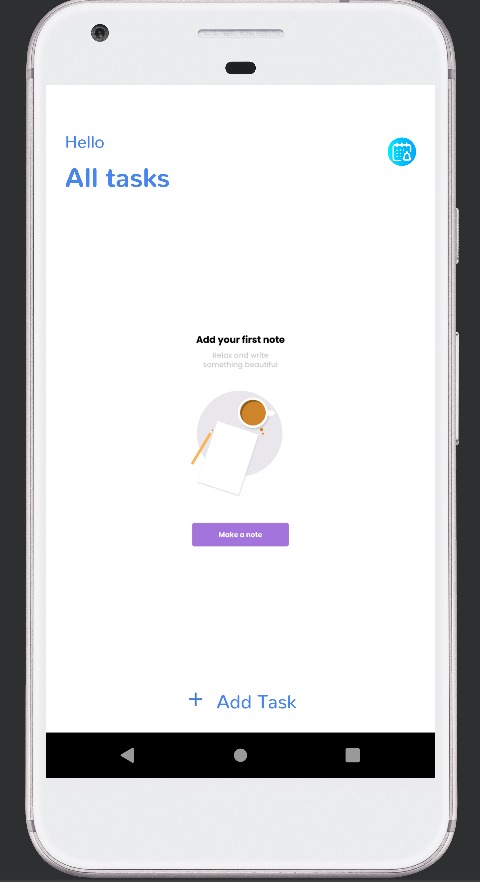
void refresh();

    }

}

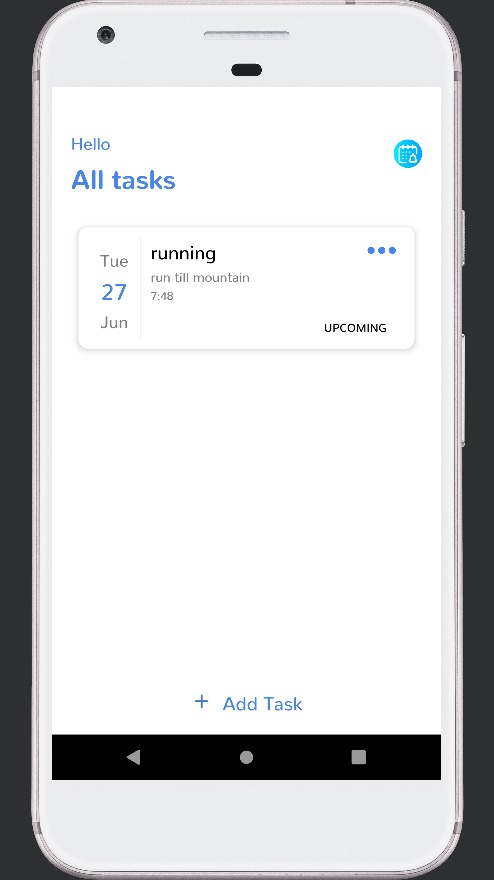
**Chapter 5**

# RESULT AND DESCRIPTION



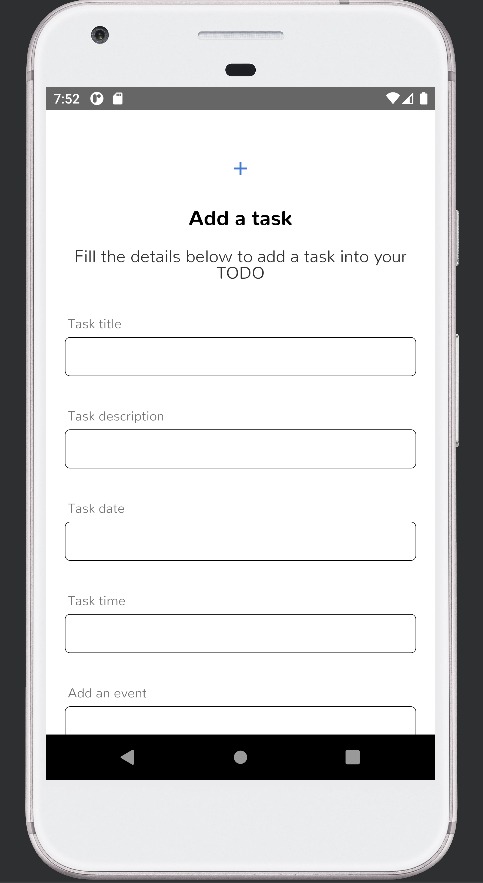
**Figure.5.1 Welcome Page**

**(This is the welome page(home page)**



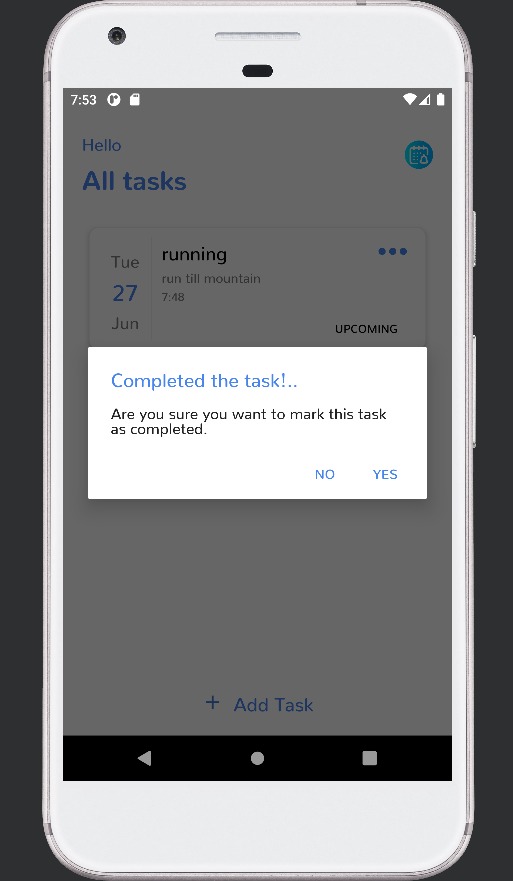
**Figure.5.2 Dashboard**

**(This is the dashboard to add or view tasks)**



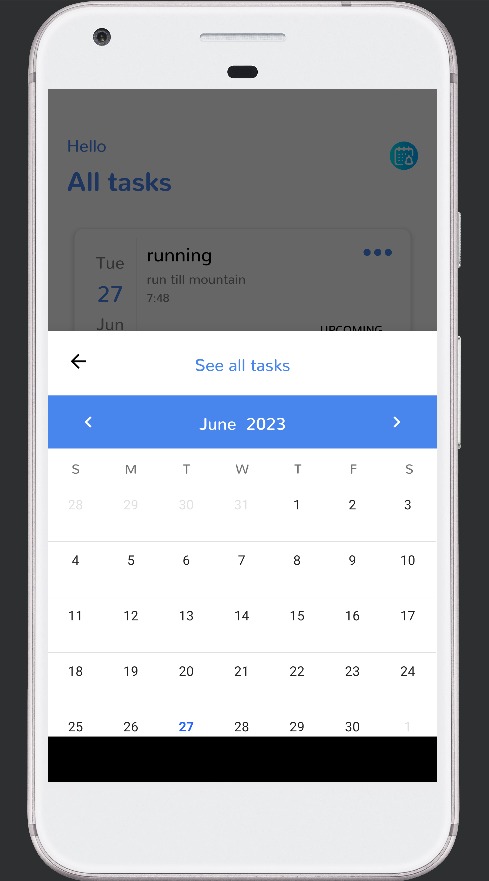
**Figure.5.3 Add task**

**(This is the section to add task with the task information)**



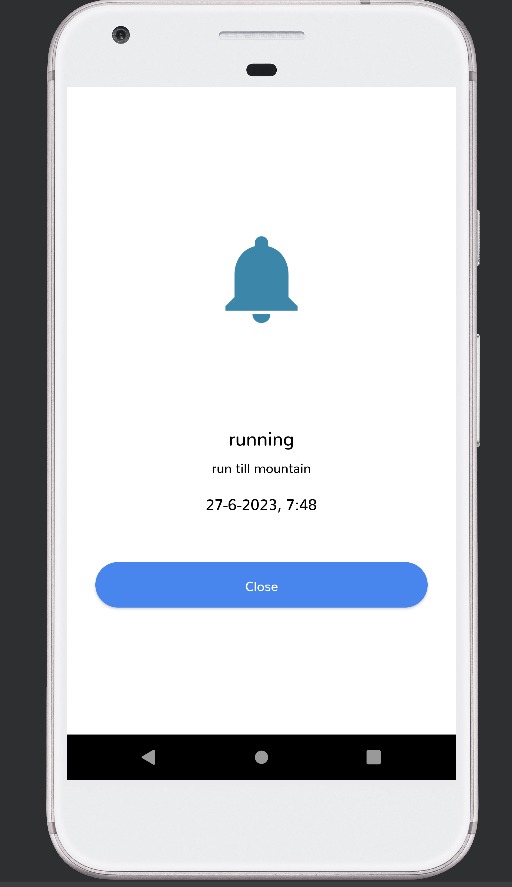
**Figure.5.4 Complete Task**

**(This Figure will show that the task is completed)**



**Figure.5.5 Calendar**

**(This figure will show the task dates that are yet to be done)**



**Figure.5.6 Notification**

**(This figure will give the notification)**

**Chapter 6**

# CONCLUSION

In conclusion, an online to-do list application with reminders offers a powerful and efficient solution for managing tasks in today's digital age. By leveraging the capabilities of the internet and cloud storage, these applications provide users with the convenience of accessing their tasks from various devices at any time. The integration of reminders within the application ensures that important deadlines and tasks are never overlooked. With an online to-do list application, users can easily add, modify, and delete tasks, allowing for seamless task management and organization. The ability to set reminders for tasks adds an extra layer of functionality, enabling users to stay on top of their responsibilities and deadlines. The reminder feature in these applications enhances productivity by sending notifications or alerts to users, prompting them to complete tasks or meet deadlines. This ensures that tasks are completed in a timely manner and reduces the chances of forgetting important assignments or commitments.

Furthermore, the centralized nature of online to-do list applications allows for efficient collaboration and sharing of tasks with team members or individuals, making them ideal for both personal and professional use. The ability to assign tasks and set reminders for team members further enhances coordination and ensures everyone stays on track.

## REFERENCES

1. <https://www.themuse.com/advice/10-todo-apps-for-10-kinds-of-people-your-life-just-got-a-whole-lot-more-productive>
2. <http://allthatsaas.com/roundup/best-reminder-apps>
3. http://developer.android.com/training/data-storage/room