SIDDAGANGA INSTITUTE OF TECHNOLOGY

(Affiliated to VTU, Belgaum)

B.H. Road, Tumakuru - 572103

A MINI-PROJECT REPORT

ON

"Task Scheduler"

Submitted by

SIDDESH M G	**1SI16CS104**
SRILEKHA C B	**1SI16CS110**
KUMAR SHUBHAM	**1SI16CS132**

Department of Computer Science & Engineering

Siddaganga Institute of Technology

2019-2020

SIDDAGANGA INSTITUTE OF TECHNOLOGY

B.H. Road, Tumakuru – 572103

Department of Computer Science & Engineering

CERTIFICATE

This is to Certified that the Mini-project work entitled **"Task Scheduler"** is a bonafide work carried out by **SIDDESH M G (1SI16CS104), SRILEKHA C B (1SI16CS110), KUMAR SHUBHAM (1SI16CS132)** Of 6th semester computer science and engineering Siddaganga Institute of Technology, for the partial fulfilment of bachelor of engineering during the academic year 2019-2020

.....	.....
A.V. Krishna Mohan	**Mrs. NOUSHEEN TAJ**
Assistant Professor	Assistant Professor
Mini-Project Convener	Mini-Project Guide

.....

Dr. R. SUMATHI

Prof. and Head, Dept. of CSE

Name of the Examiners: **Signature with Date**

1.

2.

ACKNOWLEDGEMENT

We offer our humble pranams at the lotus feet of his holiness **Dr. Sree Sree Shivakumaraa Swamigalu**, Founder President Sree Siddaganga Education Societ and his holiness, **Sree Siddalinga Swamigalu,** President Sree Siddaganga Education Society and Sree Siddaganga Mutt for best owning upon their blessings.

We deem it as a privileged to thank **Dr. M N Channabasappa** director **Dr.K P Shivananda**, Principal, SIT, Tumakuru for fostering an excellent academic environment in this institution which made this endeavor fruitful.

We would like express our sincere gratitude to **Dr. R Sumathi**, Prof and Head, Department of CSE, SIT,Tumkuru for her encouragement and valuable suggestions.

We thank to convener **Mr. A V Krishnamohan**, Assistant Professor and our guide **Nousin Taj, Professor**,Department of Computer Science and Engineering, SIT, Tumakuru for the valuable guidance, advice and encouragement.

Above all, we would like to express thanks to our Parents for their support all along.

Project Associates

SIDDESH M G (1SI16CS104)

SRILEKHA C B (1SI16CS110)

KUMAR SHUBHAM (1SI16CS132)

ABSTRACT

The era of mobile technology opens the windows to the android app. The websites are vanishing and the mobile phones are emerging. It's the time to change from conventional websites to apps, which has become the part of our daily routine. We are introducing "ToDo.apk" the android application software which would be a "task scheduling app". It works not only as a normal task scheduler, but also it can work as a digital companion. We have developed a structured, optimal and efficient mobile application, which intelligently shows up the fixes tasks and the free slots, easy to use application which will get minimal user input and deliver highly smart and descriptive scheduled tasks for the user. This will be more of a personal application which stores all the user data in the local phone storage for ensuring user privacy.

Nowadays people find difficulty in managing their to-do list, like we college students have lot of work to do daily, and we face a problem in finding the free slots from our routine in prioritizing the tasks. In order to find the solution for this we designed this application which takes your fixed tasks of a week as input and designs a table which contains all your fixed tasks and highlights your free slots.so can get know when you will be free and you can schedule your task accordingly. There is a flexibility of rescheduling, editing, deleting the fixed tasks as well as scheduled task.

TABLE OF CONTENTS

Title **Page No.**

1) INTRODUCTION
2) LITERATURE SURVEY
3) REQUIREMENTS SPECIFICATION
4) DESIGN AND IMPLEMENTATION
5) RESULTS
6) CONCLUSION
7) REFERENCES

INTRODUCTION

ANDROID:

Android is a mobile Operating system developed by Google. It is based on a modified version of the Linux kernel and other open source software and it is primarily for touch screen mobile devices such as smart phones and tablets. Initially developed by android INC which Google brought in 2005, Android was unveiled in 2007 with the first commercial android device launched in September 2008. The OS has since gone through multiple major releases with the current version being "Pie" released in August 2018. Android OS Oreo has overtaken as the single most popular version at 20.82% and v.8.0 plus v.8.1 combined are most popular.

TODO APPS:

Studies have shown that people perform better when they have written down what they need to do.

What makes the to-do list such an effective productivity tool?

- They give us a structure, a plan that we can stick to
- They are proof of what we have achieved that day, week or month.

So it is better to schedule our tasks and work accordingly and be more productive than before.

MOTIVATION:

Nowadays people find difficulty in managing their to-do list, like we college students have lot of work to do daily, and we face a problem in finding the free slots from our routine in prioritizing the tasks. In order to find the solution for this we designed this application which takes your fixed tasks of a week as input and designs a table which contains all your fixed tasks and highlights your free slots.so can get know when you will be free and you can schedule your task accordingly. There is a flexibility of rescheduling, editing, deleting the fixed tasks as well as scheduled task.

EXPECTED OUTCOME:

There will be one Login activity to maintain the privacy of users. We are using local database SQLite so that the data is secure and safe. We are taking uses fixed tasks as input on the daily basis and preparing a table that contain these tasks and highlighted free slots. So the users can refer this table to find free slots and he can schedule his tasks here.There is also the flexibility of editing, deleting the fixed tasks as well as scheduled tasks. The user can add the categories for the task like whether the task is urgent, important, not urgent, not important likewise so he can be more alert when get notification about the scheduled task. The application will handle all the possible errors so that there is no crash while running the application. On the main screen users can keep the tasks that are already done or they can remove those as per their needs. The user will get notify about the fixed tasks so that they can perform tasks in time.

LITERATURE SURVEY

This chapter involves the literature survey of the project which gives the brief explanation of the existing system of the project.
There are plethora of application solution built on the category productivity and utilities.

Here are the examples of some of the to-do applications in the market with their features:

1. Google keep

Sticky notes are one of the most trusted to-do list managing methods around. Google Keep digitizes this approach into a friendly form for PCs and mobile devices.

You can jot down a reminder onto a single note or create basic lists with items that you can check off. If you don't want to forget about a particular task, you can tell Keep to shoot you a notification at a time of your choice.

A more specific application targeted at a specific kind of task like :

2. EveryDay

Sometimes you don't want to deal with the fuss. Do you really want to juggle yet another set of account credentials just to stay on top of your shopping list?

If you'd prefer something more basic, check out EveryDay. Creating an account with the app is optional. You only need to make one if you want to back up your tasks online or use the web app.

Without an account, you can still enjoy all the app's functionality. That includes making lists, creating sub-tasks, setting up reminders, adding notes to tasks, and tinkering with other tools that can help you stay organized.

Yet another specific application but related to task

3. Inbox by Gmail

Isn't inbox by Gmail for mail? Yes, it is. But unlike traditional email clients, it turns each message into a task. When you're finished with something, you don't mark it "read," you check it off as "done." You can even pin messages you want to act on later so that you don't forget them.

A funny name but a useful application but for the task management.

4. Remember the milk

All the features you expect—such as labels and folder-based hierarchies—are present. But it's the app's recent features that have allowed it to compete with services like Todoist.

For example, there's integration with Gmail, Google Calendar, Twitter, Evernote and more. You can dig into the official IFFFT page to find applets that'll link remember the Milk to other services.

PROPOSED SYSTEM

We have seen some generic and some specific applications which serves the purpose well

But our application is different from all the above applications because we are targeting on the productivity of users more specifically , on their ability to do tasks suppose students who are facing problem in doing tasks and they either forget or don't do the task on time, which is a dent on their productivity ball.

As stated earlier this application will take fixed time table from the users and then highlights free slots from the time table and there the new task to be done

can be inserted which helps the users to think how much of free time he has and how to utilize that .this is something new and exiting.

Requirement Specification

Android Studio:

Android studio is the official IDE (Integrated Development Environment) or tool for developing application exclusively for android platform.

It has a strong editor tool for developing creative UI and emulators for different versions to test and simulate sensors without having actual Android devices.

SQLite: SQLite is a self-contained high reliability embedded full-featured public-domain SQL database engine. It is the most used database engine in the world.

Advanced Java: It is used for Enterprise applications such as authentication using login and many more web and android applications defined in java are very useful and we can make use of this to develop our application in an efficient way.

Firebase: Firebase is a mobile and web app development platform that provides developers with a plethora of tools and services to help them develop high-quality apps, grow their user base, and earn more profit.

DESIGN AND IMPLEMENTATION

CODE

```
package com.example.lenovo.todolist;


public class MainActivity extends AppCompatActivity {


    int id;
    public static ListView mListView, checkListView;
    public static List<Item> items = new ArrayList<>();
    public static List<Item> tmp = new ArrayList<>();
    public static ArrayList<Categorie> cat = new ArrayList<>();



    TextView nb_tasks,tvTask;
    public static boolean aff_done, aff_todo, aff_passed, aff_ondate;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mListView =  findViewById(R.id.listView);
        checkListView =  findViewById(R.id.checkCat);
        nb_tasks =  findViewById(R.id.nb_tasks);
        tvTask=findViewById( R.id.tvTask );
        aff_done = true;
        aff_todo = true;
        aff_passed = true;
        aff_ondate = true;
        id = 0;
```

```
        CheckBox checkToDo = findViewById(R.id.switch_todo);
        checkToDo.setChecked(true);

        tvTask.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity( new Intent(
MainActivity.this,WeekWorkActivity.class ) );
            }
        } );


        checkToDo.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(CompoundButton compoundButton,
boolean b) {
                if (b)
                    aff_todo = true;
                else
                    aff_todo = false;
                affListCorresponding();
            }
        });

CheckBox checkDone = (CheckBox) findViewById(R.id.switch_done);
        checkDone.setChecked(true);
```

```
        checkDone.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            public void onCheckedChanged(CompoundButton buttonView,
boolean isChecked) {
                if (isChecked)
                    aff_done = true;
                else
                    aff_done = false;
                affListCorresponding();
            }
        });
        CheckBox checkPassed = (CheckBox)
findViewById(R.id.switch_passed);
        checkPassed.setChecked(true);
        checkPassed.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            public void onCheckedChanged(CompoundButton buttonView,
boolean isChecked) {
                if (isChecked)
                    aff_passed = true;
                else
                    aff_passed = false;
                affListCorresponding();
            }
        });
        CheckBox checkOnDate = (CheckBox)
findViewById(R.id.switch_ondate);
        checkOnDate.setChecked(true);
```

```
        checkOnDate.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            public void onCheckedChanged(CompoundButton buttonView,
boolean isChecked) {
                if (isChecked)
                    aff_ondate = true;
                else
                    aff_ondate = false;
                affListCorresponding();
            }
        });
        getData();
        getCatData();
        if (cat.size() == 0)
            cat.add(new Categorie("none", Color.parseColor("#262D3B")));

        mListView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view,
int position, long id) {
                Intent intentMain = new Intent(MainActivity.this, EditItem.class);
                Item item = (Item) mListView.getAdapter().getItem(position);
                String title = item.getTitle();
                String time = item.getTime();
                String txt = item.getText();
                String date = item.getDate();
                String categorie = item.getCategorie();
```

```
            intentMain.putExtra("position", String.valueOf(position));
            intentMain.putExtra("title", title);
            intentMain.putExtra("txt", txt);
            intentMain.putExtra("date", date);
            intentMain.putExtra("time", time);
            intentMain.putExtra("categorie", categorie);
            startActivityForResult(intentMain, 1);
        }
    });

    ItemAdapter adapter = new ItemAdapter(MainActivity.this, items);
    checkAdapter adapter1 = new checkAdapter(MainActivity.this, cat);
    checkListView.setAdapter(adapter1);
    mListView.setAdapter(adapter);
    checkDate();
  }
  public void getData() {
    List<Item> list = new ArrayList<>();
    Item tmp;
    SQLiteDatabase mydatabase = openOrCreateDatabase("todolist",
MODE_PRIVATE, null);
    mydatabase.execSQL("CREATE TABLE IF NOT EXISTS tasks(Titre
VARCHAR, Date VARCHAR, Status VARCHAR, Txt VARCHAR, Cat
VARCHAR);");
    Cursor resultSet = mydatabase.rawQuery("Select * from tasks", null);
    resultSet.moveToFirst();
    int count = 0;

 while (count < resultSet.getCount())
```

```
        {
            String title = resultSet.getString(resultSet.getColumnIndex("Titre"));
            String date = resultSet.getString(resultSet.getColumnIndex("Date"));
            String status = resultSet.getString(resultSet.getColumnIndex("Status"));
            String txt = resultSet.getString(resultSet.getColumnIndex("Txt"));
            String cat = resultSet.getString(resultSet.getColumnIndex("Cat"));
            Date d = new Date();
           // SimpleDateFormat newDateFormat = new SimpleDateFormat("EE d
MMM yyyyHH:mm","15");
            SimpleDateFormat newDateFormat =new SimpleDateFormat( "EEE,
MMM d, "yy" );
            try {
               d = newDateFormat.parse(date);
            } catch (ParseException e) {
               e.printStackTrace();
            }
            tmp = new Item(title, txt, d);
            if (status.equals(Item.Status.DONE.toString()))
               tmp.setStatus(Item.Status.DONE);
            else
               tmp.setStatus(Item.Status.TODO);
            tmp.setCategorie(cat);
            list.add(tmp);
            count++;
            resultSet.moveToNext();
        }
        items = list;
    }
    public void getCatData() {
```

```
        ArrayList<Categorie> list = new ArrayList<>();
        Categorie tmp;
        SQLiteDatabase mydatabase = openOrCreateDatabase("todolist",
MODE_PRIVATE, null);
        mydatabase.execSQL("CREATE TABLE IF NOT EXISTS cats(Name
VARCHAR, Color VARCHAR);");
        Cursor resultSet = mydatabase.rawQuery("Select * from cats", null);
        resultSet.moveToFirst();
        int count = 0;
        while (count < resultSet.getCount()) {
            String name = resultSet.getString(resultSet.getColumnIndex("Name"));
            String color = resultSet.getString(resultSet.getColumnIndex("Color"));
            tmp = new Categorie(name, Integer.parseInt(color));
            list.add(tmp);
            count++;
            resultSet.moveToNext();
        }
        cat = list;
    }
    public String addToDataBase(int i) {
        Item tmp = items.get(i);
        String query = """;
        query += tmp.getTitle() + "','";
        query += tmp.getDate() + tmp.getTime() + "','";
        query += tmp.getStatus().toString() + "','";
        query += tmp.getText() + "','";
        query += tmp.getCategorie() + """;
        return query;}
    public void saveData() {
```

```
        String query;
        SQLiteDatabase mydatabase = openOrCreateDatabase("todolist",
MODE_PRIVATE, null);
        mydatabase.execSQL("DROP TABLE IF EXISTS tasks");
        mydatabase.execSQL("CREATE TABLE IF NOT EXISTS tasks(Titre
VARCHAR, Date VARCHAR, Status VARCHAR, Txt VARCHAR, Cat
VARCHAR);");

        for (int i = 0; i < items.size(); i++) {
            query = addToDataBase(i);
            mydatabase.execSQL("INSERT INTO tasks VALUES(" + query +
");");
        }
    }

    public void saveCategory() {
        String query;
        SQLiteDatabase mydatabase = openOrCreateDatabase("todolist",
MODE_PRIVATE, null);
        mydatabase.execSQL("DROP TABLE IF EXISTS cats");
        mydatabase.execSQL("CREATE TABLE IF NOT EXISTS cats(Name
VARCHAR, Color VARCHAR);");
        for (int i = 0; i < cat.size(); i++) {
            query = "'" + cat.get(i).getName() + "'," +
String.valueOf(cat.get(i).getColor()) + "'";
            mydatabase.execSQL("INSERT INTO cats VALUES(" + query + ");");
        }
    }
    public void settings(View V) {
```

```
        DrawerLayout mDrawerLayout = (DrawerLayout)
findViewById(R.id.drawer_layout);
        mDrawerLayout.openDrawer(Gravity.LEFT);
    }
    public void add(View v) {
        Intent intentMain = new Intent(MainActivity.this, AddItem.class);
        startActivityForResult(intentMain, 1);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data)
{       if (requestCode == 1) {
          if (resultCode == Activity.RESULT_OK) {
              String title = data.getStringExtra("title");
              String txt = data.getStringExtra("txt");
              String date = data.getStringExtra("date");
              String delete = data.getStringExtra("delete");
              String category = data.getStringExtra("categorie");
              SimpleDateFormat newDateFormat = new SimpleDateFormat("EE d
MMM yyyy k:m");
              Date d = null;
              try {
                  d = newDateFormat.parse(date);
              } catch (ParseException e) {
                  e.printStackTrace();
              }
              if (data.getStringExtra("edit").equals("true")) {
                  int position = Integer.parseInt(data.getStringExtra("position"));
                  try {
```

```
                modifyItem(position, title, txt, d, delete, category);
            } catch (ParseException e) {
                e.printStackTrace();
            }
        } else {
            Item newItem = new Item(title, txt, d);
            newItem.setCategorie(category);
            try {
                addToList(newItem);
            } catch (ParseException e) {
                e.printStackTrace();
            }
        }
      }
      if (resultCode == Activity.RESULT_CANCELED) {
        //here goes nothing
      }
    }
    if (requestCode == 2) {
      if (resultCode == Activity.RESULT_OK) {
        saveCategory();
        checkCategories();
        affListCorresponding();
        ((checkAdapter)
checkListView.getAdapter()).notifyDataSetChanged();
      }


 if (resultCode == Activity.RESULT_CANCELED) {
```

```
            saveCategory();
            checkCategories();
            affListCorresponding();
            ((checkAdapter)
checkListView.getAdapter()).notifyDataSetChanged();
        }}}
    public void checkDate() {
        int i = 0;
        Date d;

        d = new Date();
        nb_tasks.setText(String.valueOf(items.size()) + " Tasks");
        while (i < items.size()) {
            if (!(items.get(i).getRealDate().after(d))) {
                items.get(i).setPassed(true);
                items.get(i).setDateColor("#FF0000");
            } else {
                items.get(i).setPassed(false);
                items.get(i).setDateColor("#121212");
            } i++;
        }}
        public void addToList(Item item) throws ParseException {
        items.add(item);
        checkDate();
        saveData();
        Date f = new Date();
        int c = 0;
        int color = Color.BLUE;
        while (c < cat.size()) {
```

```
            if (item.getCategorie().equals(cat.get(c).getName())) {
                color = cat.get(c).getColor();
            }
            c++;
        }
        int delay = (int) (item.getRealDate().getTime() - f.getTime());
        if (delay > 0)
            scheduleNotification(getNotification(item.getTitle(), item.getText(),
color), delay);
        affListCorresponding();
    }
    public void modifyItem(int position, String title, String txt, Date d, String
delete, String cate) throws ParseException {
        Item item = items.get(position);
        if (delete.equals("false")) {
            item.setTitle(title);
            item.setText(txt);
            item.setDueDate(d);
            item.setCategorie(cate);
        } else
            items.remove(item);
        checkDate();
        saveData();
        Date f = new Date();
        int delay = (int) (d.getTime() - f.getTime());
        int color = Color.BLUE;
        int c = 0;
        while (c < cat.size()) {
            if (item.getCategorie().equals(cat.get(c).getName())) {
```

```
            color = cat.get(c).getColor();
        }
        c++;
    }
    if (delay > 0)
        scheduleNotification(getNotification(title, txt, color), delay);
    affListCorresponding();
}
public boolean showCatForItem(Item item) {
    int i = 0;
    while (i < cat.size()) {
        if (cat.get(i).getName().equals(item.getCategorie())) {
            return cat.get(i).getShow();
        }
        i++;
    }
    return false;
}
public void affListCorresponding() {
    int nb_items = items.size();
    boolean t, p;
    int i = 0;
    tmp.clear();
    while (i < nb_items) {
        t = false;
        p = false;
        if (aff_done && items.get(i).getStatus() == Item.Status.DONE)
            t = true;
        if (aff_todo && items.get(i).getStatus() == Item.Status.TODO)
```

```
            t = true;
        if ((aff_passed && items.get(i).getPassed()))
            p = true;
        if ((aff_ondate && !items.get(i).getPassed()))
            p = true;
        if (t && p && showCatForItem(items.get(i)))
            tmp.add(items.get(i));
        i++;
      }
      ItemAdapter adapter = new ItemAdapter(MainActivity.this, tmp);
      mListView.setAdapter(adapter);
      if (tmp.size() > 1)
        ((TextView)
findViewById(R.id.nb_tasks)).setText(String.valueOf(tmp.size()) + " Tasks");
      else
        ((TextView)
findViewById(R.id.nb_tasks)).setText(String.valueOf(tmp.size()) + " Task");
      adapter.notifyDataSetChanged();
   }
   public void todoClick(View v) {
      final int position = mListView.getPositionForView((View) v.getParent());
      SwipeLayout s = (SwipeLayout) mListView.getChildAt(position);
      Item a = items.get(position);
      a.setStatus(Item.Status.TODO);
      affListCorresponding();
      saveData();
      s.close(true);
   }
```

```
  public void catCheck(View v) {
      final int position = checkListView.getPositionForView((View)
v.getParent());
      CheckBox checkBox = (CheckBox) v;
      if (checkBox.isChecked())
         cat.get(position).setShow(true);
      else
         cat.get(position).setShow(false);
      affListCorresponding();
   }
   public void doneClick(View v) {
      final int position = mListView.getPositionForView((View) v.getParent());
      SwipeLayout s = (SwipeLayout) mListView.getChildAt(position);
      Item a = items.get(position);
      a.setStatus(Item.Status.DONE);
      s.close(true);
      ItemAdapter b = (ItemAdapter) mListView.getAdapter();
      affListCorresponding();
      b.notifyDataSetChanged();
      saveData();
   }
   private void scheduleNotification(Notification notification, int delay) {

      Intent notificationIntent = new Intent(this, NotificationPublisher.class);
      notificationIntent.putExtra(NotificationPublisher.NOTIFICATION_ID,
1);
      notificationIntent.putExtra(NotificationPublisher.NOTIFICATION,
notification);
```

```
        PendingIntent pendingIntent = PendingIntent.getBroadcast(this, 0,
notificationIntent, PendingIntent.FLAG_UPDATE_CURRENT);

        long futureInMillis = SystemClock.elapsedRealtime() + delay;
        AlarmManager alarmManager = (AlarmManager)
getSystemService(Context.ALARM_SERVICE);
        alarmManager.set(AlarmManager.ELAPSED_REALTIME_WAKEUP,
futureInMillis, pendingIntent);
    }
    @RequiresApi(api = Build.VERSION_CODES.JELLY_BEAN)
    private Notification getNotification(String Title, String content, int color) {
        Notification.Builder builder = new Notification.Builder(this);
        builder.setContentTitle(Title);
        builder.setContentText(content);
        builder.setSmallIcon(R.drawable.ic_notification);
        if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.LOLLIPOP) {
            builder.setColor(color);
        }
        affListCorresponding();
        return builder.build();
    }
    public static ArrayList<Categorie> getCatA() {
        ArrayList<Categorie> tmp = new ArrayList<Categorie>();
        int i = 0;
        while (i < cat.size())
            tmp.add(cat.get(i++));
        return tmp;
    }
```

```
public void closeMenu(View v) {
    DrawerLayout d = ((DrawerLayout) findViewById(R.id.drawer_layout));
    d.closeDrawers();
}

public static ArrayList<Categorie> getCat() {
    return (getCatA());
}

public void checkCategories() {
    int i = 0;

    while (i < items.size()) {
        int c = 0;
        boolean found = false;
        while (c < cat.size()) {
            if (items.get(i).getCategorie().equals(cat.get(c).getName()))
                found = true;
            c++;
        }
        if (!found)
            items.get(i).setCategorie("none");
        i++;
    }
    affListCorresponding();}
public void addCategorie(View v) {
    Intent intentMain = new Intent(MainActivity.this, addCategory.class);
    startActivityForResult(intentMain, 2);
    checkCategories();}}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/drawer_layout"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="@drawable/bgpl"
   tools:context=".MainActivity">

<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      app:layout_behavior="@string/appbar_scrolling_view_behavior">
      <RelativeLayout
         android:id="@+id/top"
         android:layout_width="match_parent"
         android:layout_height="50dp"
         android:background="@color/colorPrimary">

         <TextView
            android:id="@+id/nb_tasks"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:gravity="center"
```

```
            android:text="nb Tasks"
            android:textAlignment="center"
            android:textColor="#FFFFFF"
            android:textSize="18dp"
            android:textStyle="bold" />

        <ImageView
            android:layout_width="40dp"
            android:layout_height="40dp"
            android:layout_centerVertical="true"
            android:contentDescription="menu_list"
            android:onClick="settings"
            android:paddingLeft="10dp"
            android:src="@drawable/menu" />

    </RelativeLayout>

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@+id/top">


<ListView
            android:id="@+id/listView"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true" />
```

```
        <ImageView
            android:id="@+id/add_btn"
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:layout_alignParentBottom="true"
            android:layout_alignParentEnd="true"
            android:layout_alignParentRight="true"
            android:layout_margin="10dp"
            android:onClick="add"
            android:src="@drawable/plus" />

        <TextView
            android:id="@+id/tvTask"
            android:layout_width="234dp"
            android:layout_height="wrap_content"
            android:layout_alignParentBottom="true"
            android:layout_alignParentStart="true"
            android:layout_marginBottom="23dp"
            android:text="Enter the fixed tasks"
            android:textAlignment="center"
            android:textColor="#f4ebec"
            android:textSize="20dp"
            android:textStyle="bold"
            android:typeface="serif" />
    </RelativeLayout>
</RelativeLayout>

<LinearLayout
```

```
android:id="@+id/left_drawer"
android:layout_width="200dp"
android:layout_height="fill_parent"
android:layout_gravity="start|left"
android:background="#FFFFFF"
android:fillViewport="true"
android:orientation="vertical">

<RelativeLayout
   android:id="@+id/top2"
   android:layout_width="match_parent"
   android:layout_height="50dp"
   android:background="@color/colorPrimary">

   <TextView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:gravity="center"
      android:text="Settings"
      android:textAlignment="center"
      android:textColor="#FFFFFF"
      android:textSize="18dp"
      android:textStyle="bold" />

   <ImageView
      android:layout_width="40dp"
      android:layout_height="40dp"
      android:layout_alignParentEnd="true"
      android:layout_centerVertical="true"
```

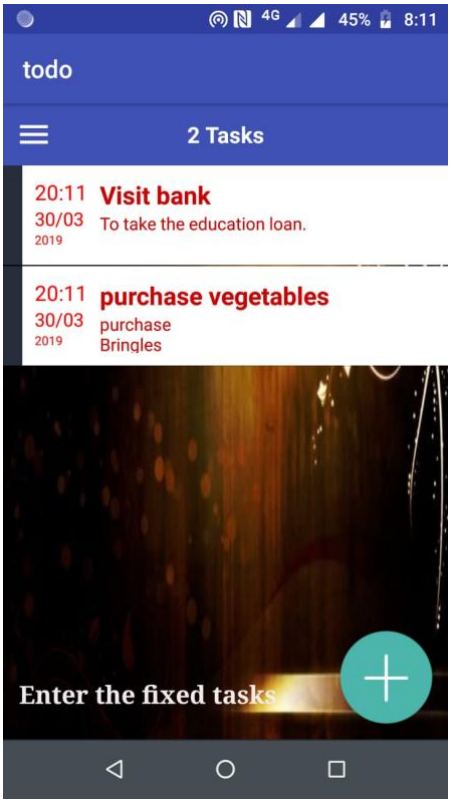
```
        android:onClick="closeMenu"
        android:padding="10dp"
        android:src="@drawable/close" />
</RelativeLayout>

<LinearLayout
    android:background="#EFEFEF"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:paddingLeft="2dp">

    <ImageView
        android:id="@+id/doneT"
        android:layout_width="20dp"
        android:layout_height="20dp"
        android:layout_gravity="center_vertical"
        android:src="@drawable/donetask" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:clickable="false"
        android:padding="10dp"
        android:text="Status"
        android:textSize="20sp" />
</LinearLayout>
<CheckBox
    android:id="@+id/switch_done"
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
    android:text="Done" />

<CheckBox
    android:id="@+id/switch_todo"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="To do" />
<LinearLayout
    android:background="#EFEFEF"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:paddingLeft="2dp">

    <ImageView
        android:layout_width="20dp"
        android:layout_height="20dp"
        android:layout_gravity="center_vertical"
        android:src="@drawable/ic_date_range_black_24dp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:clickable="false"
        android:padding="10dp"
        android:text="Date"
        android:textSize="20sp" />
</LinearLayout>
<CheckBox
```

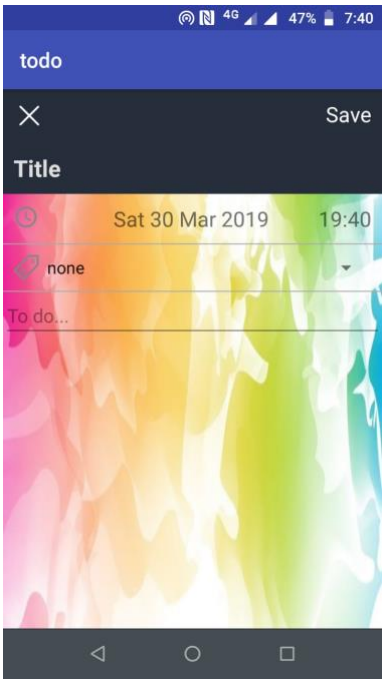
```
    android:id="@+id/switch_passed"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Passed" />

<CheckBox
    android:id="@+id/switch_ondate"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Up to Date" />
<LinearLayout
    android:background="#EFEFEF"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:paddingLeft="2dp">
    <ImageView
        android:layout_width="20dp"
        android:layout_height="20dp"
        android:layout_gravity="center_vertical"
        android:src="@drawable/label" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="addCategorie"
        android:padding="10dp"
        android:text="Categories"
        android:textSize="20sp" />
```

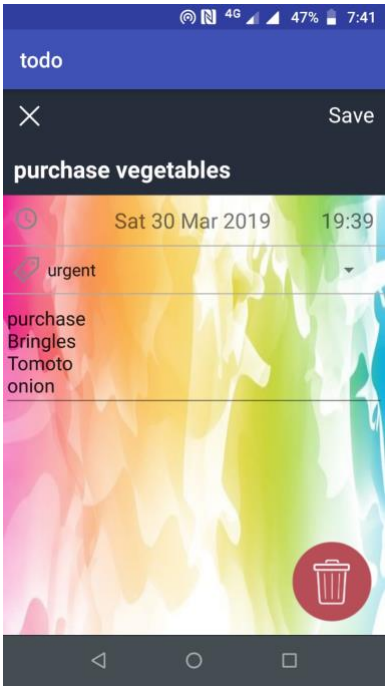
```
  <ImageView
            android:layout_width="20dp"
            android:layout_height="20dp"
            android:layout_gravity="center_vertical"
            android:onClick="addCategorie"
            android:src="@drawable/config" />
      </LinearLayout>
      <ListView
         android:id="@+id/checkCat"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:fillViewport="true">
      </ListView>
      <View
         android:layout_width="match_parent"
         android:layout_height="1dp"
         android:background="#DFDFDF" />
   </LinearLayout>
</android.support.v4.widget.DrawerLayout>
```

RESULTS



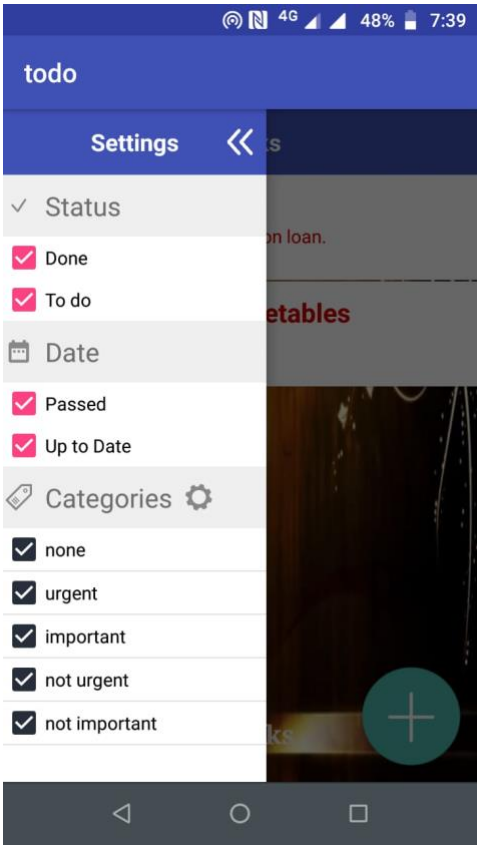
Our task scheduler application opens with a splash screen activity. Followed by that the above screen will appear. When we click on the plus mark then the new activity starts and there we can schedule the tasks. The toolbar shows the number of activities we have scheduled. When we click on “Enter the fixed tasks” another activity will opens where we can enter the fixed tasks as per the days of the week.

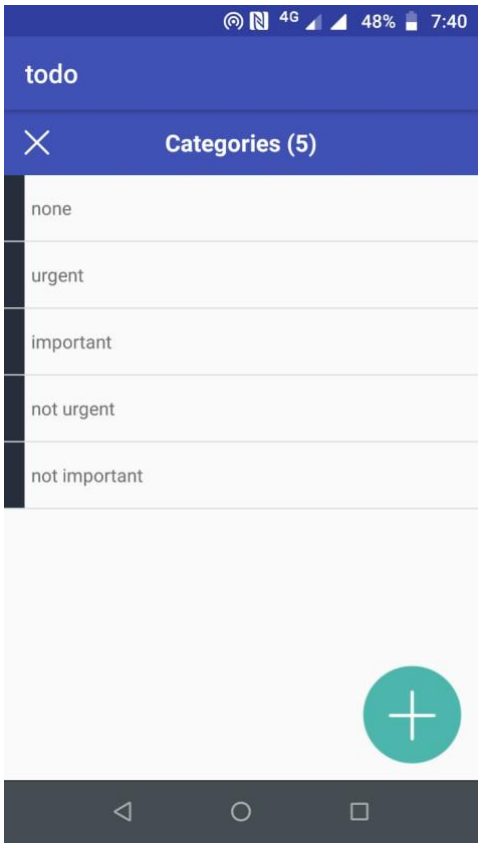




Here we can enter the tasks and its category. i.e. whether the task is important, urgent, not important, not urgent ,etc.here we also have the flexibility of the deleting or editing the task which has been scheduled already.

Here we can choose a day and add the fixed tasks corresponds to that day and the entered tasks will reflect on a table immediately.





This is navigation drawer activity. Here when we click on categories the screen beside it appears.in that when we click on plus mark we can add the categories. The toolbar above displays the number of categories added. We also provided the flexibility of deleting these categories added.

CONCLUSION

Task scheduler or to-do apps are very helpful. They are the proof of what we have achieving in our life .they gives the clarity of whether we are really making use of our time or we just wasting it. Keeping that in mind we developed this application which takes your fixed tasks of a week as input and designs a table which contains all your fixed tasks and highlights your free slots.so can get know when you will be free and you can schedule your task accordingly. There is a flexibility of rescheduling, editing, deleting the fixed tasks as well as scheduled task.

As a future scope we can extend this project to a extend where the appli do more things like by giving dataset to the machine and let the machine think for me and tell what to do now and what to do after so that my time is saving

REFERENCES

>Fundamentals of Database Systems by “Ramez Elmasri”

>GitHub

>Developer.Android.com

>Quora

> Stackoverflow