TODO LISTS

A MINI PROJECT REPORT

Submitted by

MANOJ KUMAR J **220701524**

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS) THANDALAM

CHENNAI-602105

MAY-2025

BONAFIDE CERTIFICATE

Certified that this project report "**MOVIE TICKET BOOKING SYSTEM**" is the bonafide work of **MANOJ KUMAR J(220701524)**

who carried out the project work under my supervision.

Submitted for the Practical Examination held on _____

SIGNATURE

Mr. V.Karthick
Associate professor
Computer Science and Engineering,
Rajalakshmi Engineering College,
(Autonomous),
Thandalam, Chennai - 602 105

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

This project presents a **To-Do List Application** built using **Kotlin** and the **Android Room Database**. The application enables users to efficiently create, manage, and organize their daily tasks. Users can add new tasks by specifying details such as the title, description, category, date, and time. Tasks are displayed in a dynamic and colorful **RecyclerView**, with functionalities that allow users to mark tasks as completed or delete them via intuitive swipe gestures. Data persistence is achieved using **Room**, ensuring that all tasks are stored locally within an SQLite database, enabling offline access. Users can easily search for tasks by their titles through an integrated search functionality. Furthermore, the application provides a separate **History section** for users to view finished tasks, ensuring better task tracking and management. The application's architecture is modular, employing best practices such as **LiveData** observation, **Coroutines** for background operations, and **MVVM-like** separation of concerns. This ensures a responsive, maintainable, and user-friendly experience suitable for personal productivity enhancement.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 IMPLEMENTATION

2. SYSTEM SPECIFICATION

2.1 HARDWARE SPECIFICATION

2.2 SOFTWARE SPECIFICATION

3. SOURCE CODE

4. SNAPSHOTS

5. CONCLUSION

6.REFERENCES

CHAPTER 1

1. INTRODUCTION:

In today's fast-paced world, managing tasks effectively is crucial for personal productivity. This To-Do List Application provides an intuitive platform for users to create, update, and organize their tasks. It utilizes Android's Room Database for reliable local data storage and Kotlin for smooth app performance. With features like task categorization, reminders, and swipe actions, the app simplifies everyday task management.

1.1 IMPLEMENTATION:

The To-Do List Application is developed using **Kotlin** and **Android Studio**, following modern Android development practices. It leverages the **Room Persistence Library** to manage local data storage, providing a structured and efficient way to perform database operations such as insertion, retrieval, update, and deletion of tasks. The application interface is built with components like **RecyclerView**, **CardView**, and **Material Design elements** to create an engaging and responsive user experience. Tasks are displayed dynamically, and **swipe gestures** (implemented using ItemTouchHelper) allow users to delete tasks or mark them as completed easily. he **TaskActivity** enables users to create new tasks by filling out fields such as title, description, category, date, and time using intuitive date and time pickers. **Search functionality** is incorporated into the main activity, enabling real-time filtering of tasks based on keywords entered by the user. The project uses **Coroutines** to perform database operations asynchronously, ensuring that the UI remains smooth and responsive even during background tasks. Additionally, **LiveData** is used to observe changes in the database and update the UI automatically, without manual refreshes. By following a modular coding approach, with a clear separation between UI (Activities, Adapters), data handling (DAO, Database, Model),

and business logic, the application ensures scalability, maintainability, and ease of future development or enhancement.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATION:

PROCESSOR	- Intel® core™ i5-6006U @ 2.00 GHz
RAM	- 4GB
OPERATING SYSTEM	- Microsoft Windows 11
HARD DISK	- 850 GB of free space
SYSTEM TYPE	- 64-bit operating system , x64 based processor

2.2 SOFTWARE SPECIFICATION:

PROGRAMMING LANGUAGE: kotlin

OPERATING SYSTEM	:	Microsoft Windows 11
SOFTWARE	:	Android Studio

CHAPTER 3

Source Code

```
package com.example.todoapp

import android.content.Intent
import android.graphics.*
import android.os.Bundle
import android.view.Menu
import android.view.MenuItem
import android.view.View
import androidx.appcompat.app.AppCompatActivity
import androidx.appcompat.widget.SearchView
import androidx.lifecycle.Observer
import androidx.recyclerview.widget.ItemTouchHelper
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import kotlinx.android.synthetic.main.activity_main.*
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.GlobalScope
import kotlinx.coroutines.launch

class MainActivity : AppCompatActivity() {

    val list = arrayListOf<TodoModel>()
    var adapter = TodoAdapter(list)

    val db by lazy {
        AppDatabase.getDatabase(this)
    }

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        setSupportActionBar(toolbar)
        todoRv.apply {
            layoutManager = LinearLayoutManager(this@MainActivity)
            adapter = this@MainActivity.adapter
        }

        initSwipe()
```

```
        db.todoDao().getTask().observe(this, Observer {
            if (!it.isNullOrEmpty()) {
                list.clear()
                list.addAll(it)
                adapter.notifyDataSetChanged()
            }else{
                list.clear()
                adapter.notifyDataSetChanged()
            }
        })


    }

    fun initSwipe() {
        val simpleItemTouchCallback = object : ItemTouchHelper.SimpleCallback(
            0,
            ItemTouchHelper.LEFT or ItemTouchHelper.RIGHT
        ) {
            override fun onMove(
                recyclerView: RecyclerView,
                viewHolder: RecyclerView.ViewHolder,
                target: RecyclerView.ViewHolder
            ): Boolean = false

            override fun onSwiped(viewHolder: RecyclerView.ViewHolder, direction:
Int) {
                val position = viewHolder.adapterPosition

                if (direction == ItemTouchHelper.LEFT) {
                    GlobalScope.launch(Dispatchers.IO) {
                        db.todoDao().deleteTask(adapter.getItemId(position))
                    }
                } else if (direction == ItemTouchHelper.RIGHT) {
                    GlobalScope.launch(Dispatchers.IO) {
                        db.todoDao().finishTask(adapter.getItemId(position))
                    }
                }
            }

            override fun onChildDraw(
                canvas: Canvas,
```

```
            recyclerView: RecyclerView,
            viewHolder: RecyclerView.ViewHolder,
            dX: Float,
            dY: Float,
            actionState: Int,
            isCurrentlyActive: Boolean
        ) {
            if (actionState == ItemTouchHelper.ACTION_STATE_SWIPE) {
                val itemView = viewHolder.itemView

                val paint = Paint()
                val icon: Bitmap

                if (dX > 0) {

                    icon = BitmapFactory.decodeResource(resources,
R.mipmap.ic_check_white_png)

                    paint.color = Color.parseColor("#388E3C")

                    canvas.drawRect(
                        itemView.left.toFloat(), itemView.top.toFloat(),
                        itemView.left.toFloat() + dX, itemView.bottom.toFloat(), paint
                    )

                    canvas.drawBitmap(
                        icon,
                        itemView.left.toFloat(),
                        itemView.top.toFloat() + (itemView.bottom.toFloat() -
itemView.top.toFloat() - icon.height.toFloat()) / 2,
                        paint
                    )


                } else {
                    icon = BitmapFactory.decodeResource(resources,
R.mipmap.ic_delete_white_png)

                    paint.color = Color.parseColor("#D32F2F")

                    canvas.drawRect(
                        itemView.right.toFloat() + dX, itemView.top.toFloat(),
```

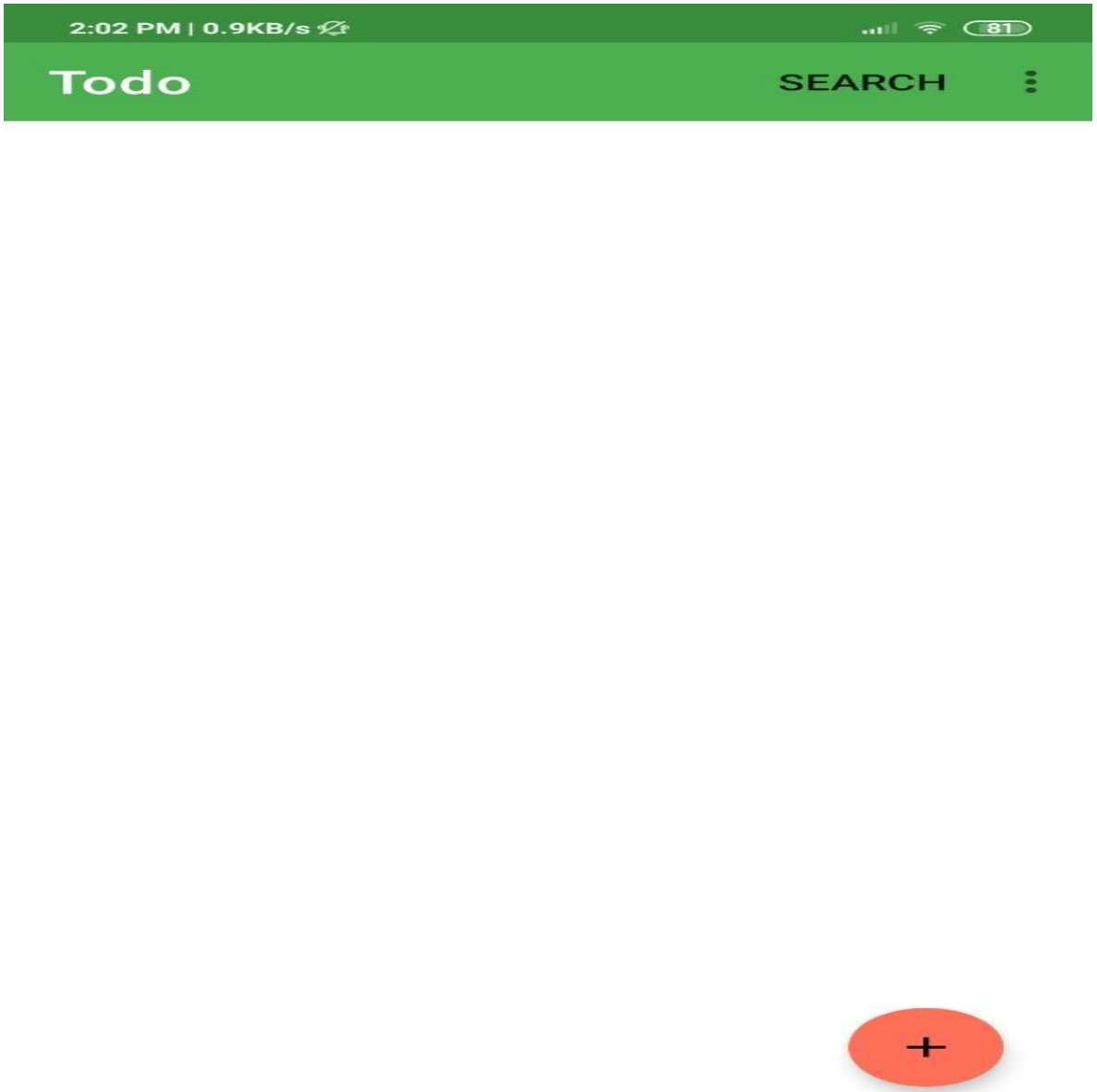
```
                    itemView.right.toFloat(), itemView.bottom.toFloat(), paint
                )

                canvas.drawBitmap(
                    icon,
                    itemView.right.toFloat() - icon.width,
                    itemView.top.toFloat() + (itemView.bottom.toFloat() -
itemView.top.toFloat() - icon.height.toFloat()) / 2,
                    paint
                )
            }
            viewHolder.itemView.translationX = dX


        } else {
            super.onChildDraw(
                canvas,
                recyclerView,
                viewHolder,
                dX,
                dY,
                actionState,
                isCurrentlyActive
            )
        }
    }


}

val itemTouchHelper = ItemTouchHelper(simpleItemTouchCallback)
itemTouchHelper.attachToRecyclerView(todoRv)
}

override fun onCreateOptionsMenu(menu: Menu): Boolean {
    menuInflater.inflate(R.menu.main_menu, menu)
    val item = menu.findItem(R.id.search)
    val searchView = item.actionView as SearchView
    item.setOnActionExpandListener(object :MenuItem.OnActionExpandListener{
        override fun onMenuItemActionExpand(item: MenuItem?): Boolean {
            displayTodo()
            return true
```

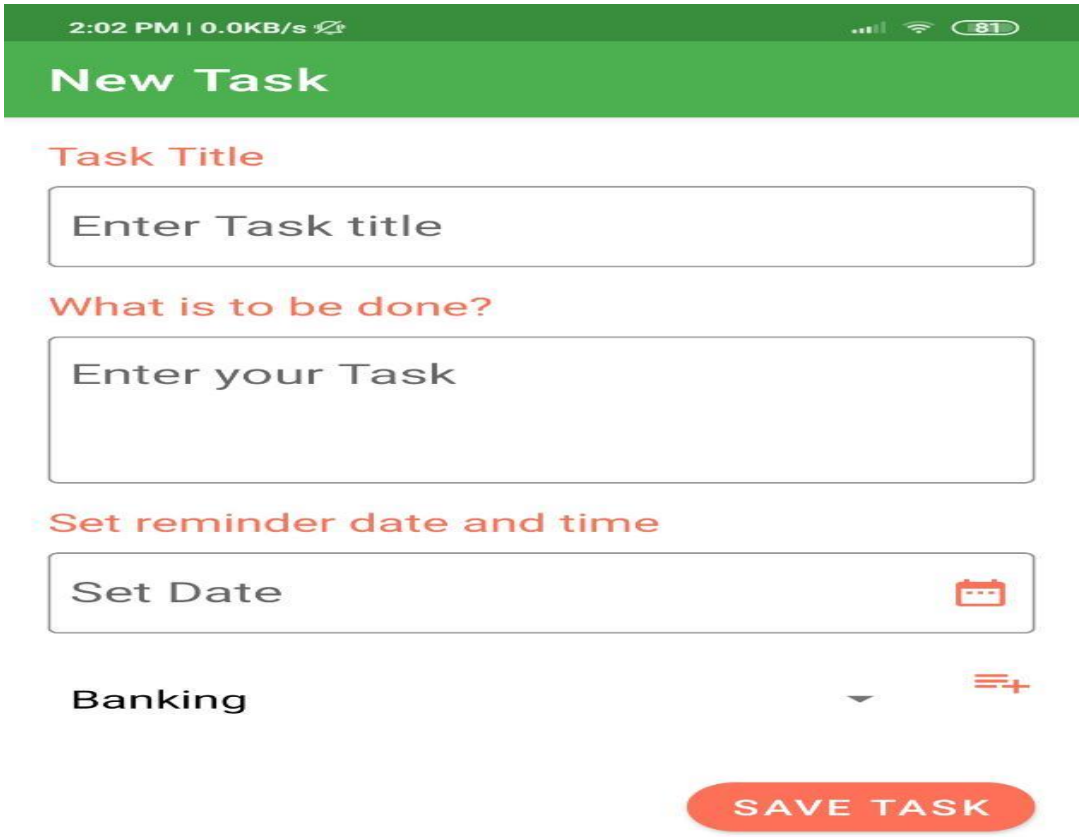
```
        }

        override fun onMenuItemActionCollapse(item: MenuItem?): Boolean {
            displayTodo()
            return true
        }

    })
    searchView.setOnQueryTextListener(object :
SearchView.OnQueryTextListener{
        override fun onQueryTextSubmit(query: String?): Boolean {
            return false
        }

        override fun onQueryTextChange(newText: String?): Boolean {
            if(!newText.isNullOrEmpty()){
                displayTodo(newText)
            }
            return true
        }

    })

    return super.onCreateOptionsMenu(menu)
}

fun displayTodo(newText: String = "") {
    db.todoDao().getTask().observe(this, Observer {
        if(it.isNotEmpty()){
            list.clear()
            list.addAll(
                it.filter { todo ->
                    todo.title.contains(newText,true)
                }
            )
            adapter.notifyDataSetChanged()
        }
    })
}

override fun onOptionsItemSelected(item: MenuItem): Boolean {
    when (item.itemId) {
```

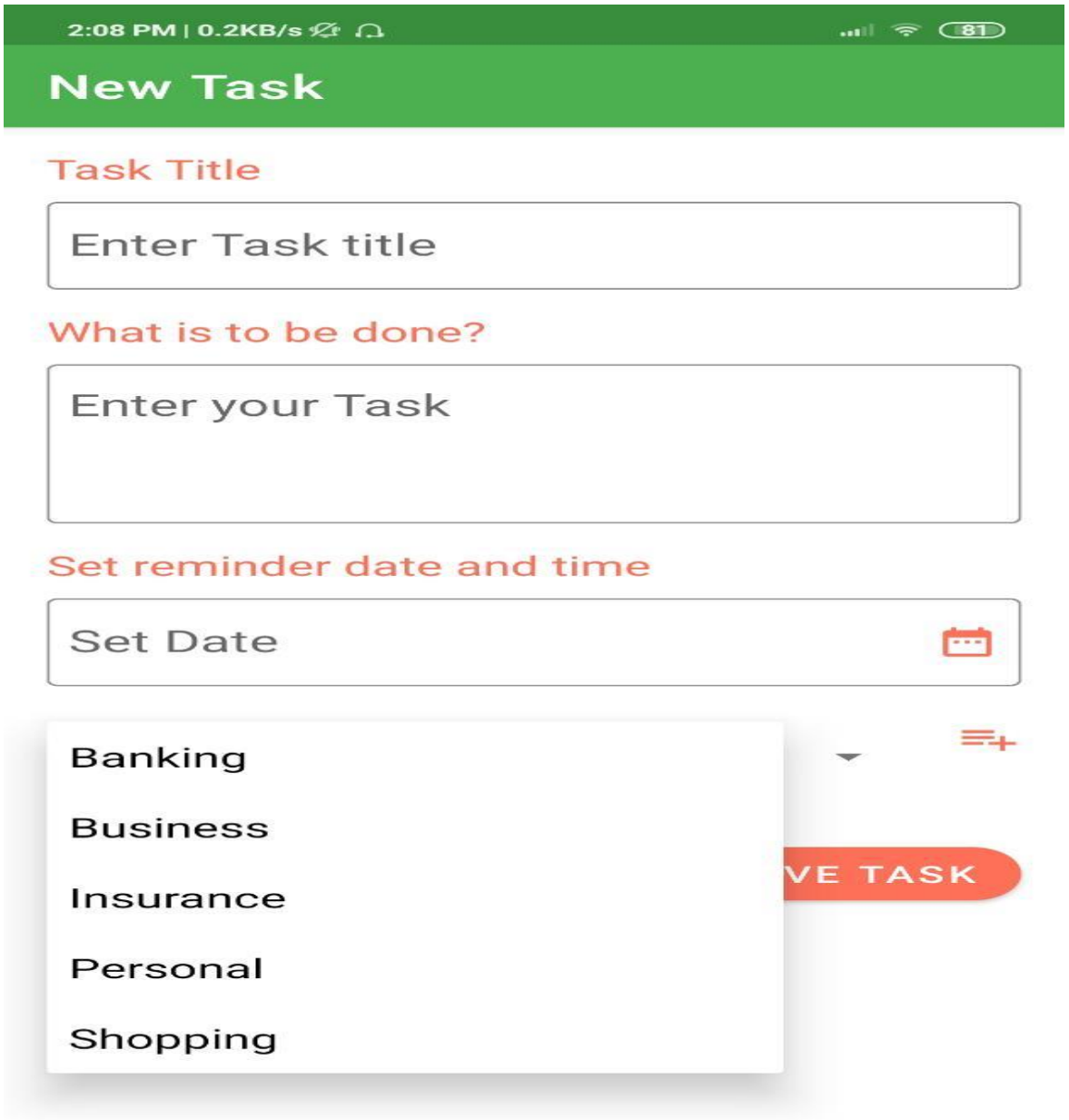
```
            R.id.history -> {
                startActivity(Intent(this, HistoryActivity::class.java))
            }
        }
        return super.onOptionsItemSelected(item)
    }

    fun openNewTask(view: View) {
        startActivity(Intent(this, TaskActivity::class.java))
    }
}
```

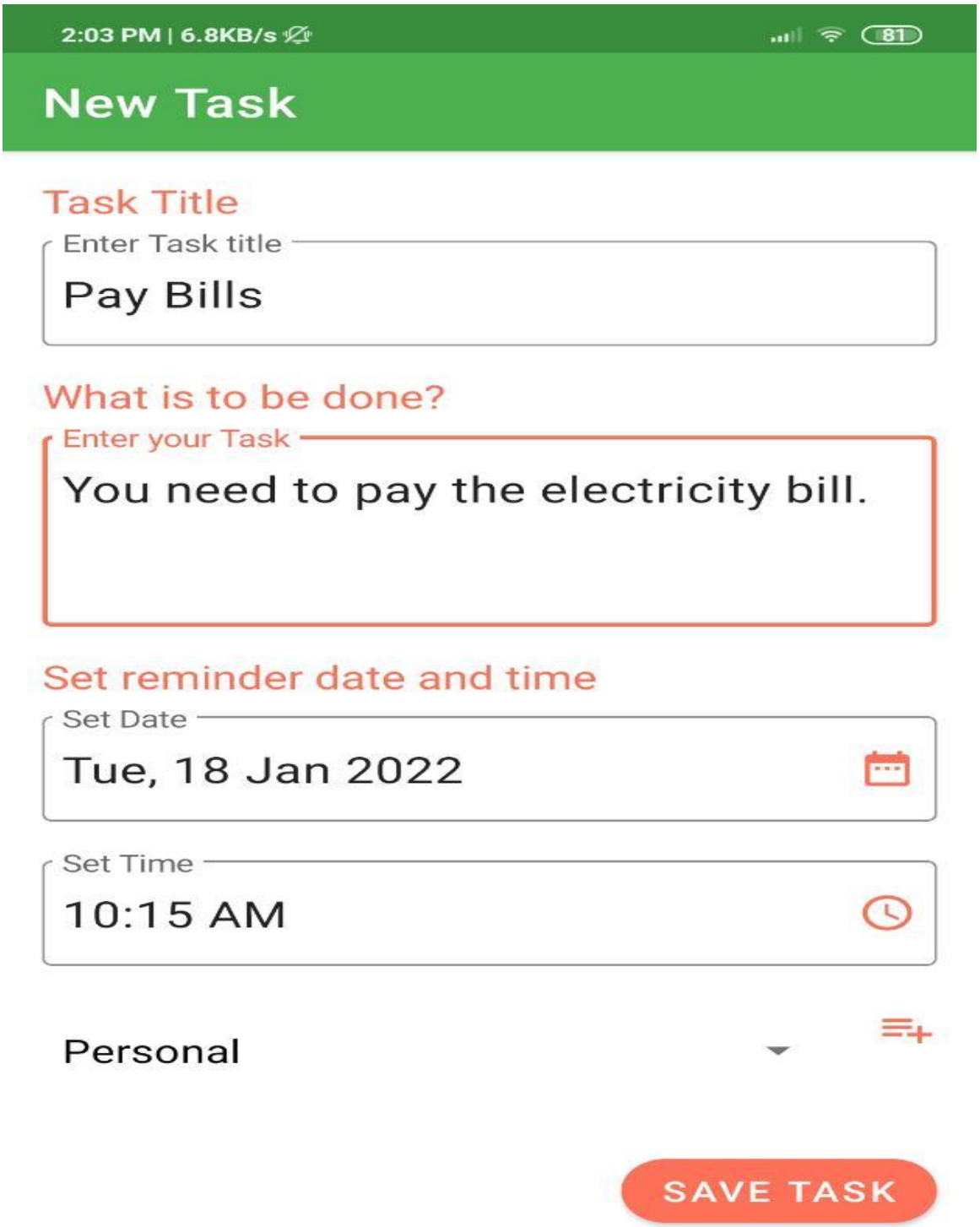
CHAPTER 4

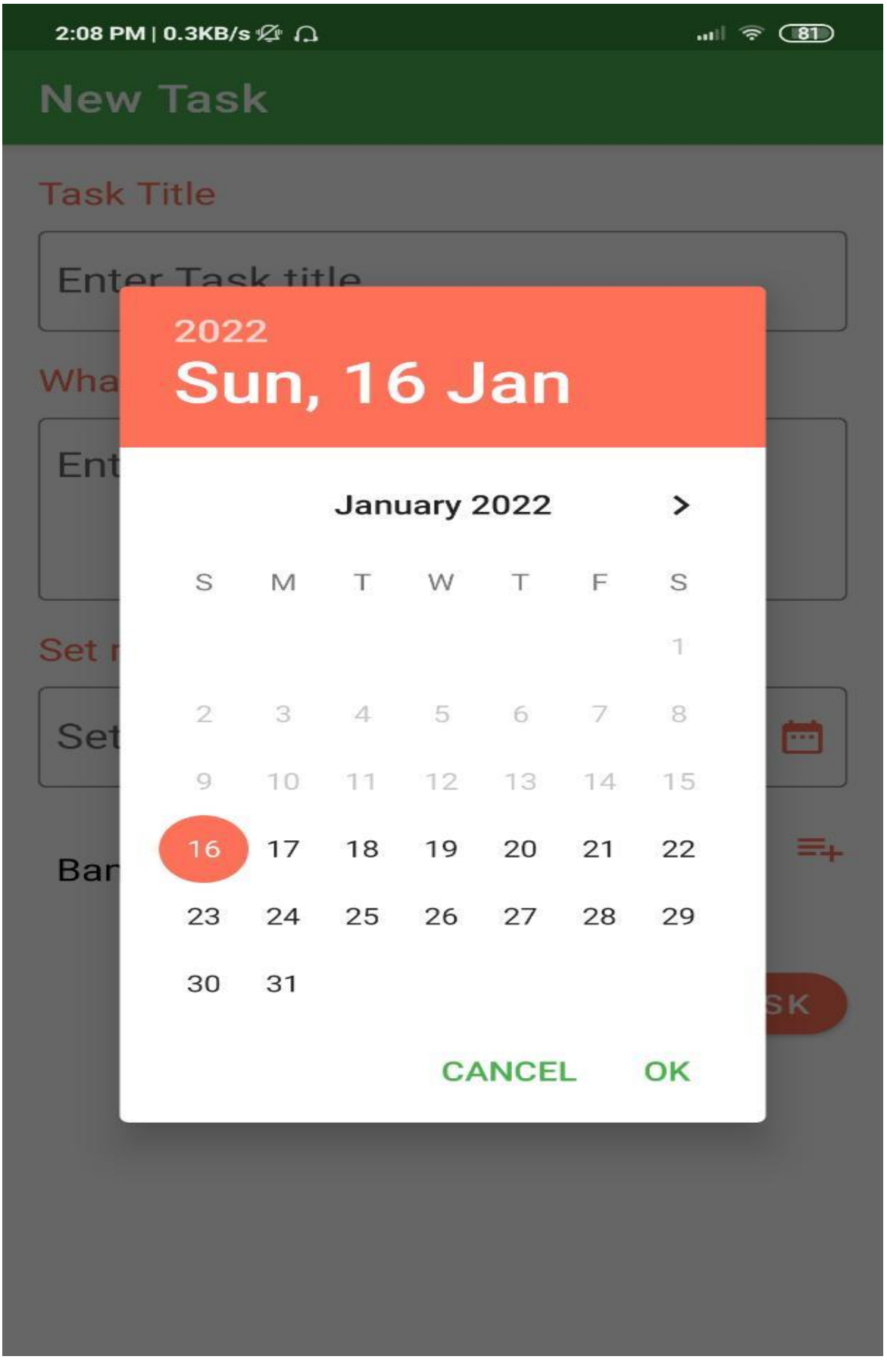
SNAP SHOTS

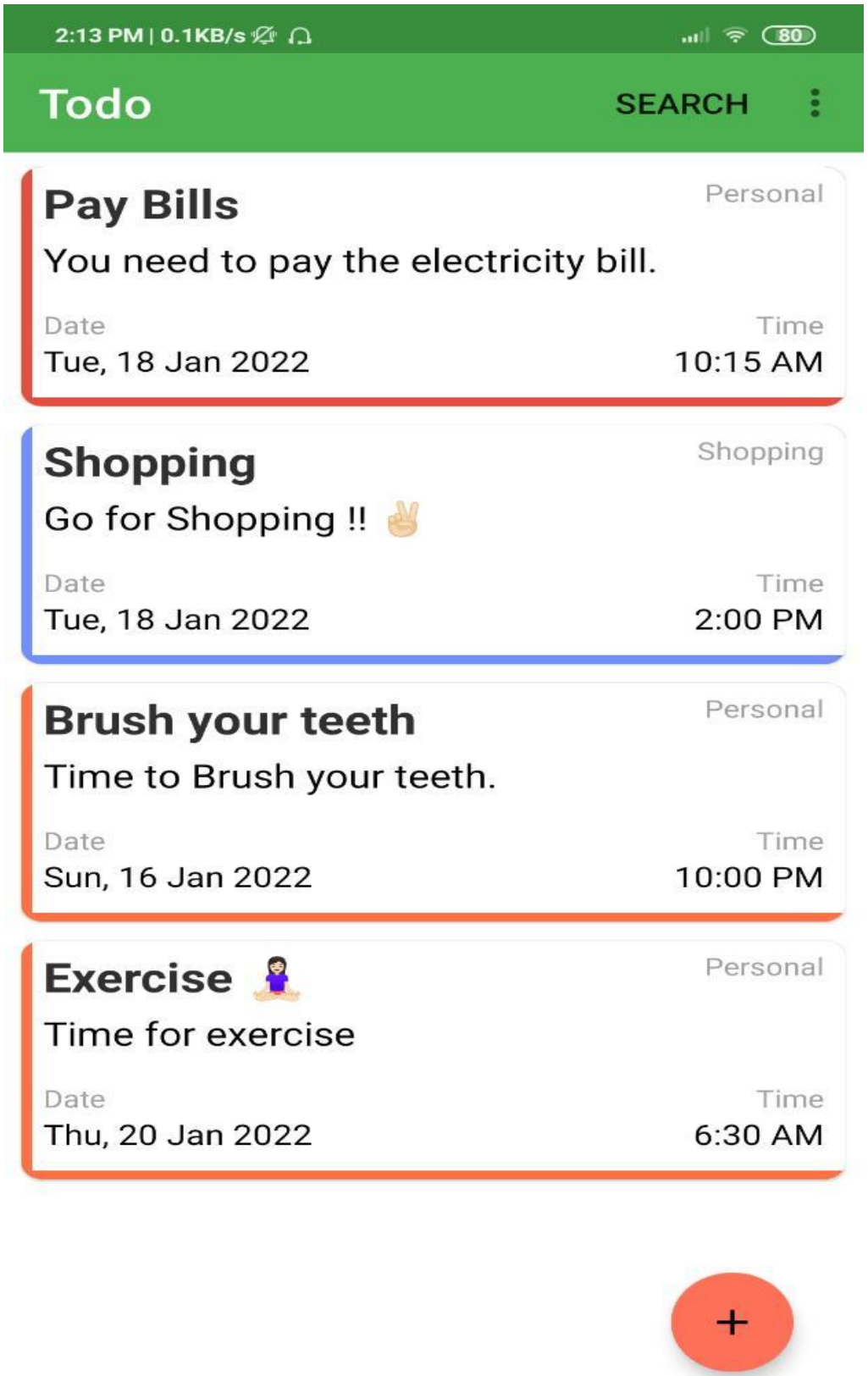



2:02 PM | 0.0KB/s
New Task
Task Title
Enter Task title
What is to be done?
Enter your Task
Set reminder date and time
Set Date
Banking
SAVE TASK


2:08 PM | 0.2KB/s
New Task
Task Title
Enter Task title
What is to be done?
Enter your Task
Set reminder date and time
Set Date
Banking
Business
Insurance
Personal
Shopping
VE TASK


2:03 PM | 6.8KB/s
81
New Task
Task Title
Enter Task title
Pay Bills
What is to be done?
Enter your Task
You need to pay the electricity bill.
Set reminder date and time
Set Date
Tue, 18 Jan 2022
Set Time
10:15 AM
Personal
SAVE TASK


2:08 PM | 0.3KB/s
81
New Task
Task Title
Enter Task title
Wha
Ent
Set r
Set
Bar
SK
2022
Sun, 16 Jan
January 2022
S M T W T F S
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
CANCEL
OK


2:13 PM | 0.1KB/s
80
Todo
SEARCH
Pay Bills
Personal
You need to pay the electricity bill.
Date
Tue, 18 Jan 2022
Time
10:15 AM
Shopping
Shopping
Go for Shopping !!
Date
Tue, 18 Jan 2022
Time
2:00 PM
Brush your teeth
Personal
Time to Brush your teeth.
Date
Sun, 16 Jan 2022
Time
10:00 PM
Exercise
Personal
Time for exercise
Date
Thu, 20 Jan 2022
Time
6:30 AM
+

CHAPTER 5

CONCLUSION

This program has been created successfully to create an TODO LISTS using Kotlin