EXPENSIO – AN EXPENSE MANAGER

A PROJECT REPORT

Submitted in fulfilment of

MOBILE APPLICATION DEVELOPMENT (ITE1016)

By

NEHA BALASUNDARAM	**16BIT0040**
SURYA K	**16BIT0029**
ACHYUTH R	**16BIT0237**

Under the guidance of

PROF. VENKATESH P

SCHOOL OF INFORMATION TECHNOLOGY

APRIL, 2018

ACKNOWLEDGEMENTS

We thank our professor PROF. VENKATESH P for guiding us through this project and giving feedback on its strengths and flaws.

We also thank the VIT University Management and the management at SITE for giving us the opportunity to carry out our project to carry out our studies at the university.

NEHA BALASUNDARAM (16BIT0040)

ACHYUTH R(16BIT0237)

SURYA K(16BIT0029)

DECLARATION BY THE CANDIDATE

We hereby declare that the project report entitled "***EXPENSIO – AN EXPENSE MANAGER"*** being submitted by
"***SURYA K, NEHA BALASUNDARAM, ACHYUTH R***" for Mobile Application Development (ITE1016) to VIT University, Vellore in partial fulfilment of the requirement for the award of the degree of B.Tech.(Information Technology) is a record of J component of project work carried out by us under the guidance of PROF VENKATESH P. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Date: 7th APRIL, 2018

Name of Students:

NEHA BALASUNDARAM

SURYA K

ACHYUTH R

CERTIFICATE

This is to certify that the project work entitled "***EXPENSE MANAGER"*** being submitted by "***NEHA BALASUNDARAM, SURYA K, ACHYUTH R***" for MOBILE APPLICATION DEVELOPMENT (ITE1016) is a record of bonafide work done under my supervision. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted for any other CAL course.

Place: Vellore

Date: 7th APRIL, 2018

Name of Students:

NEHA BALASUNDARAM

SURYA B

ACHYUTH R

Name of Faculty:

PROF. VENKATESH P

TABLE OF CONTENTS

1. INTRODUCTION

1.1.ABSTRACT

Expensio is an easy to use, intuitive application to create and maintain weekly expenses. It classifies various expenses under different categories, and helps the user track the various expenses he or she may incur during the course of the week. It also keeps track of the various debts that we may incur by taking money from people, and the small loans that we might give them. A summary of this information is also provided as the various out standings we owe to respective people or vice versa. The expenses are highlighted in various clean and simple to understand demographics.

1.2.AIM

The aim of this project is to create an expense manager which will help the user keep track and cut down on his/her expenses but setting a weekly budget and pointing when they are over it. It should display the expenses in asthetically pleasing graphs. It should also provide functionality to categorize various expenses.

1.3.PROBLEM STATEMENT

To create an android application which will keep a record of the various transactions performed by the user through the week. The application should also be able to extract and show useful information like spending trends in the form the data in the form of simple and easy to understand graphs. The application should also be able to handle a small scale debt/loan system that should keep track of the date debts were issued and outstanding amount.

2. DESCRIPTION OF THE PROJECT

2.1.FEATURES

- Manage your everyday expenses. Create, add and monitor your expenses
- Add category labels for your expenses. Helps classify your expenses

- Set a budget for a particular time period and follow how much of your planned budget you have left or have exceeded
- Follow your debts and loans by adding and removing them
- Fluid material design to enhance UI/UX
- View an analysis of the trends in your expenses with date, categories etc with the help of charts and graphs

2.2.USER INTERFACE

- We have used the material design library which offers the user physical edges and surfaces to work with – with seams and shadows giving context to what parts of a digital design can be touched.
- We have implemented the card-view and recycler-view as it is more efficient by defaulta and the layout is separated and we have more possibilities over the data set inside the adapter.
- Use uniform colour scheme throughout the application since colour plays a big part in visitors' psychological responses.

2.3.TOOLS USED

- Android Studio - The Official IDE for Android.
- Kotlin - Statically typed programming language for modern multiplatform applications. It is a concise language reducing the amount of boilerplate code, safe and tool friendly language. It is also interoperable with java which allowed extending java's power and its library supports
- MPAndroidChart – A powerful and easy to use chart and graph library for Android. It supports scaling, dragging and animations and provide a clean user experience
- Realm - A database built for mobile. Offline first functionality, Fast queries, safe threading, cross-platform apps, secured with encryption, reactive architecture are some of the highlights
- Android asset studio – A collection of tools that easily generate assets such as launcher icons for your Android app. It allows app designers

create launcher icons, app shortcut icons, action bar icons and splash screens

3. SAMPLE CODE

The entire codebase is now open source and can be viewed in the following link.
https://github.com/SuryaThiru/expensio-WINSEM18

Here are a few samples of our project:

A database model

```
package model

import io.realm.RealmList
import io.realm.RealmObject
import io.realm.annotations.PrimaryKey
import java.util.*

/**
 * Created by surya on 3/19/18.
 */
open class User(
        @PrimaryKey var name: String = "John Doe",
        var budget: Int = 0,
        var budgetStartDate: Date? = null,
        var budgetEndDate: Date? = null,
        var expenses: RealmList<Expense> = RealmList(), // fancy one-to-many definitions
        var loans: RealmList<Loan> = RealmList()
) : RealmObject()
```

Manifest file:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.nas.android.expensio">

    <application
        android:name=".GlobalState"
        android:allowBackup="true"
        android:icon="@mipmap/launcher_logo_round"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/launcher_logo_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".SplashScreen"
            android:theme="@style/SplashTheme">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".MainNavigationActivity"
            android:label="@string/todays_expenses"
            android:theme="@style/AppTheme.NoActionBar" />
        <activity
            android:name=".SetBudget"
            android:label="@string/setBudget" />
        <activity
            android:name=".AddCategory"
            android:label="@string/add_category" />
        <activity
            android:name=".DebtsAndLoansList"
            android:label="@string/debts_and_loans" />
        <activity
            android:name=".ViewSpendTrends"
            android:label="@string/view_spend_trends" />
        <activity android:name=".AddDebtOrLoan"
            android:label="@string/Add_debt_loan"
            />
        <activity android:name=".CategoryList"
            android:label="@string/Category_list"
            />
        <activity android:name=".BudgetList"
            android:label="@string/Budget_list"
            />
        <activity android:name=".AddExpense"
            android:label="@string/Add_expense"
            />
    </application>

</manifest>
```

Main Activity

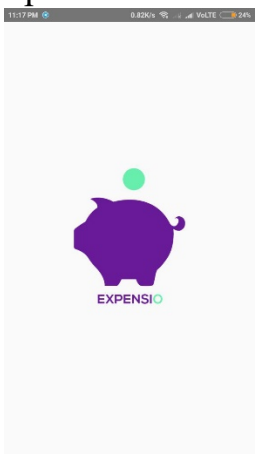
```
package com.nas.android.expensio

import android.content.Intent
import android.os.Bundle
import android.support.design.widget.NavigationView
import android.support.design.widget.Snackbar
import android.support.v4.view.GravityCompat
import android.support.v7.app.ActionBarDrawerToggle
import android.support.v7.app.AppCompatActivity
import android.support.v7.widget.LinearLayoutManager
import android.util.Log
import android.view.Menu
import android.view.MenuItem
import io.realm.Realm
import kotlinx.android.synthetic.main.activity_main_navigation.*
import kotlinx.android.synthetic.main.content_main_navigation.*
import kotlinx.android.synthetic.main.app_bar_main_navigation.*
import model.*
import java.util.*


class MainNavigationActivity : AppCompatActivity(), NavigationView.OnNavigationItemSelectedListener {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main_navigation)
        setSupportActionBar(toolbar)

        // create the recycler view
        main_recyclerview.layoutManager = LinearLayoutManager(this)
        main_recyclerview.adapter = MainAdapter()

        fab_main.setOnClickListener {
            val intent = Intent(applicationContext, AddExpense::class.java)
            startActivity(intent)
        }

        val toggle = ActionBarDrawerToggle(
                this, drawer_layout, toolbar, R.string.navigation_drawer_open, R.string.navigation_drawer_close)
        drawer_layout.addDrawerListener(toggle)
        toggle.syncState()

        nav_view.setNavigationItemSelectedListener(this)

        // test new user
        val realm = Realm.getDefaultInstance()
        if (getUser(realm) == null) {
            val newUserFragment = NewUserFragment()
            newUserFragment.show(supportFragmentManager, "Enter user name")
        }

        // test more db
//        val realm = Realm.getDefaultInstance()
//        createUser(realm, "developer")
//        getUser(realm, "test")
//        addCategory(realm, "test category", "#fff")
//        addExpense(realm, 500, "spend em all", Date(), "name")
//        var exp = getExpenses(realm)
//        Log.i("Realm query result", "exp - ${exp[0]?.amount} , ${exp[0]?.remarks} , ${exp[0]?.category}, ${exp[0]?.date}'
//        var cat = getCategories(realm)
//        Log.i("Realm query result", "category - $cat")
    }
```

Continued –

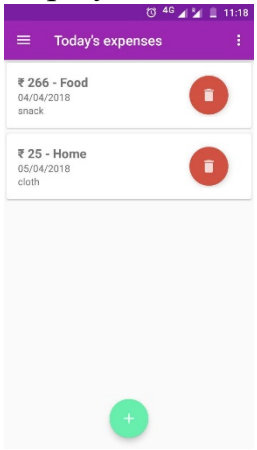
```
    override fun onResume() {
        super.onResume()

        Log.d("Main activity", "on resume triggered")
        main_recyclerview.layoutManager = LinearLayoutManager(this)
        main_recyclerview.adapter = MainAdapter()
    }

    override fun onBackPressed() {
        if (drawer_layout.isDrawerOpen(GravityCompat.START)) {
            drawer_layout.closeDrawer(GravityCompat.START)
        } else {
            super.onBackPressed()
        }
    }

    override fun onCreateOptionsMenu(menu: Menu): Boolean {
        // Inflate the menu; this adds items to the action bar if it is present.
        menuInflater.inflate(R.menu.main_navigation, menu)
        return true
    }

    override fun onOptionsItemSelected(item: MenuItem): Boolean {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        return when (item.itemId) {
            R.id.action_settings -> true
            else -> super.onOptionsItemSelected(item)
        }
    }

    override fun onNavigationItemSelected(item: MenuItem): Boolean {
        // Handle navigation view item clicks here.
        when (item.itemId) {
            R.id.add_category -> {
                // Handles the action
                val intent = Intent(applicationContext, CategoryList::class.java)
                startActivity(intent)

            }
            R.id.set_budget -> {
                val intent = Intent(applicationContext, BudgetList::class.java)
                startActivity(intent)

            }
            R.id.view_spend_trends -> {
                val intent = Intent(applicationContext, ViewSpendTrends::class.java)
                startActivity(intent)

            }
            R.id.debts_and_loans -> {
                val intent = Intent(applicationContext, DebtsAndLoansList::class.java)
                startActivity(intent)
            }

        }

        drawer_layout.closeDrawer(GravityCompat.START)
        return true
    }
}
```

4. SCREENSHOTS OF THE APPLICATION

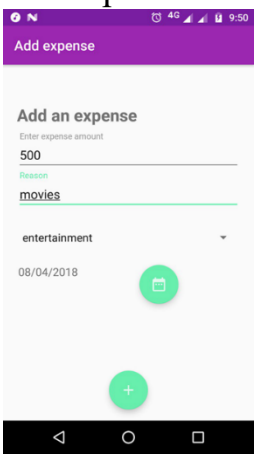
- Splash screen.



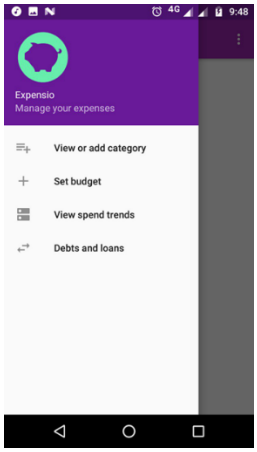
- Expenses list (home page)– this is where the list of all expenses is displayed.



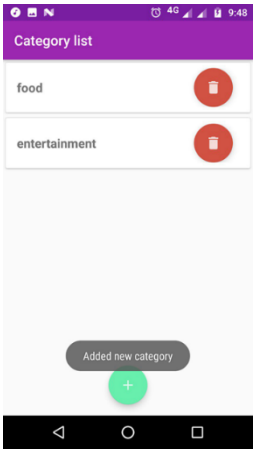
- Add expense – a form.



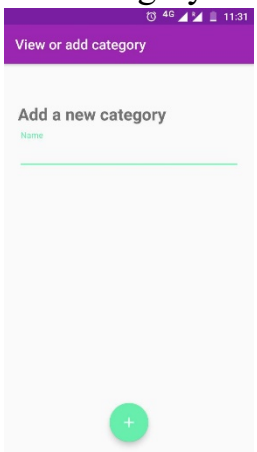
- Navigation drawer.



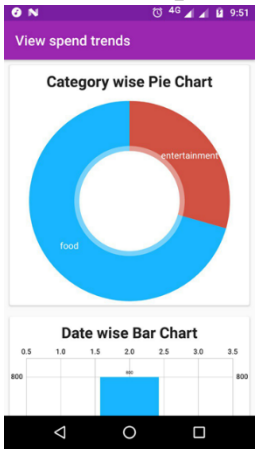
- Category list – the user defined categories are stored here.

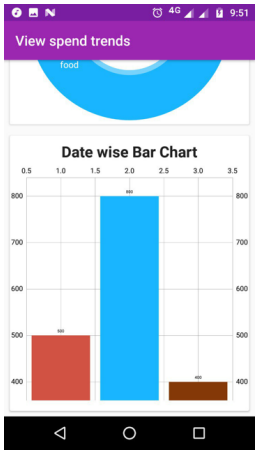


- Add category – a form.

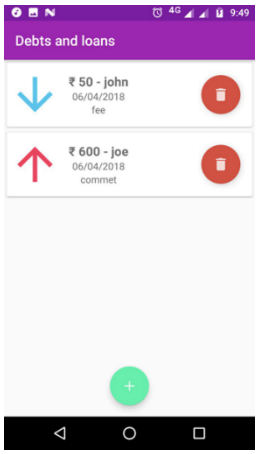


- View spend trends – where the spend trend for the past week can be viewed – a pie and a bar chart.

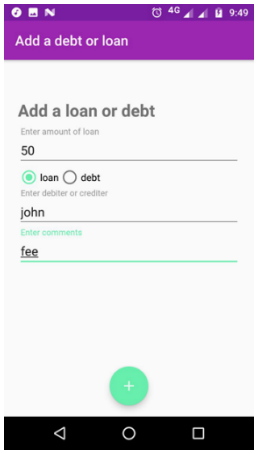




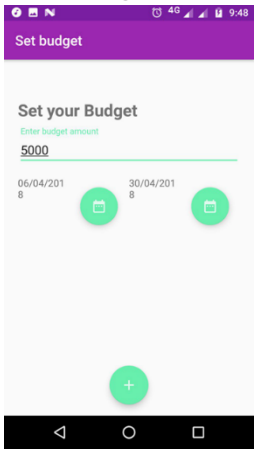
- Debts and loans – a list of people who either borrowed from you or you borrowed from



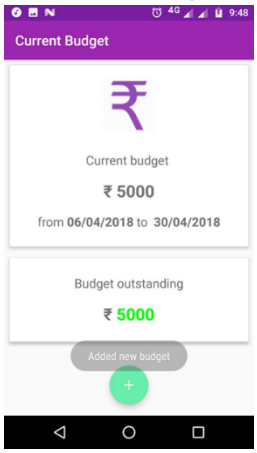
- Add debt or loan – a form.



- Set budget – set the budget for the week



- Current budget – shows current budget and outstanding expenses



5. CONCLUSION

In conclusion, we have developed an application which in the best of our knowledge serves to better the spending tendencies and Patten's of various people by imposing a tangible limit on the amount of money that they can spend

in the future we are looking into adding notification and widget support, If we can acquire permission we are also looking into adding UPI functionality so that we can directly Access bank accounts and provide better budget allotments.