Ministerul Educației al Republicii Moldova
Universitatea Tehnică a Moldovei
Facultatea de Calculatoare, Informatică și Microelectronică
Filiera Anglofonă „Computer Science”

Admis la susținere
Prof. dr. hab. Viorel Bostan,
Director Filieră Anglofonă

_____

„__”_____ 2017

Android Application for Personal Budget Management
Aplicație Android pentru gestiunea bugetului personal

Proiect de licență

Student:Alina Mocanaș (_____)
Conducător: Radu Melnic (_____)
Consultanţi: Viorel Bostan (_____)
Elena Gogoi (_____)

Chişinău 2017

Abstract

Rezumat

Table of contents

List of Tables

List of Figures

Listings

Introduction

Those first couple of years after university are an intensely action-packed, exciting time. The students have finished school,they've made plans to move out of their parents' house, and they've started a career and are getting paid.Maybe not as much as they'd like, and maybe not as much as some of their friends are making, but they're finally starting to see some cash roll in.Actually here is the beginning of that delicate time, beginning of the life as an adult, and it is the moment when you are finally in control of your money.The beginning of the financial life is no less delicate, and your wealth in your thirties and forties will be mainly determined by the financial decisions make right after university. A common problem encountered by young people at this time is wrong money management, this is a time when the desire for independence combined with optimism about the future drives everything.So, here appears the need of having more control over money, and a mobile application which will scan all kind of receipts and will make a kind of statistics of all expenditures accumulated may be will be the best choice of everyone who will care about the tracking budget correctly.

At the moment there a lot of applications helping you to track your expenditures but all of them require a manual input of all outgoings, which is somehow uncomfortable and requires more time to do. Also, given the fact that today's young generation is increasingly lazy and is always looking for easier solutions then such kind of application seems to be very attractive for them. Moreover it will be convenient to use it as well for older people, such as it's user interface will is very clear and simple to use for all groups of people in the society.

The main purpose of the Spending Tracker is scanning the receipts and showing a total amount spend by selecting a desired period,besides this it has a lot of other specific functionalities, such as displaying a statistic of expenditures by categories of stuff,categorizing expenditures by food, drinks, communal services, entertainment, fuel and others or categorizing by stores,advertising user when a product is cheaper in one store that in other,also it's possible to set a reminder of upcoming payment which serves people as a strong point in making themselves more responsible.

For users to access the application is necessary to install it on an Android platform, to open it and enjoy the user friendly interface which allows user to fast obtain a result.The first screen is a Welcome screen which informs users about the main functionalities of the application and get the user ready to start using it.Then the main fragment is opened which contains the total amount of expenditures for today by default, and the list of them.For scanning the receipt another action is required,by pressing the plus button there is opened another view which informs you about the process of scanning the receipt,for that is necessary to take a photo of the receipt,then all necessary information is stocked in a database for forward use.The application contains a spinner in the main fragment where can be fount all features: Spendings,Categories,Statistics and some History which includes the list with all the stuff and the amounts of them inside.

The current thesis consists of 4 chapters. Chapter 1 explains the application analysis, together with the general overview about the final product. A market research has been done, in order to define advantages/disadvantages of using the application and provide a better understanding over the resulting system. Chapter 2 contains a more complex description of the application, from archi-

tectural point of view. The application is examined from different approaches, using UML diagrams. Chapter 3 contains a brief explanation regarding the technologies used for the implementation process and how those technologies were linked to the application.Chapter 4 represents a review over the economical position of the product. It is presented an estimation of costs for developing the project and some economical prediction for the future use.

1 Description and Analysis of the Domain

Today,people live comfortable life,because of the favourable progress of technology.A lot of things that people had to do by themselves even few years ago become the work of machines today.It is essential to understand that during the next few decades or maybe sooner, the notion of work and whether it is handled by a human will be replaced by software,instead of using the ancient approach of doing everything individually.However someone believes that this progress make people lazy,saying people rely on machines too much,actually it makes human beings more active both physically and mentally,machines or high technology help people to complete tasks more quickly.

Time and money are those two key points which people are in rush for continuously and their right management are a priority nowadays.Here is the point of using technologies for repetitive operations, in such a way,mistakes are reduced or eliminated, and the time it takes to complete the task is greatly reduced.Diving deeper into the subject,an application which will manage the money and simultaneous will save the time by doing the action quickly is exactly what society needs.

Spending Tracker is a mobile application that comes with a solution for existing problem in the society, of managing right the budget and saving the time on doing that.Scheduling periodically the budget reviews and stick to a strict plan to bring the financial health of the family in line with the goals.So,such an application brings a lot of benefits to its users as:

- Gives control over user's money;
- Accelerates financial goals;
- Helps in organizing spending and savings;
- Enables saving for expected and unexpected costs;
- Enables to produce extra money
- Aligns priorities;
- Builds new habits;

1.1 The workflow of the system

In order to be able to access the Spending Tracker application,a user should first of all install it on an Android platform, which is made simply by pressing install button in Android Play Store.Once the application is opened,a Welcome screen is displayed.The Welcome screen serves as an introduction point for users such as there, in a pager are described the basic futures of the application, in such a way user is getting ready to use the application.Then by pressing start button, the main view is opened which contains the basic information about the expenditures of default today date.If to expend the spinner from the left top corner then can be observed the lists of all the features which the application offer to use.

The application itself offers some statistics about the amounts spend on different categories of stuff in different stores.User can see on what product he spends most and receive some advices

as an alert if it's case to save on something.The next action which can be performed is scanning of receipts which is made by pressing the bottom right corner plus button which opens another view with all necessary information about scanning the receipt.It's necessary to press another bottom button "TAKE A PHOTO OF YOUR RECEIPT" which starts the phone video camera and capture a photo for forward use. After capturing when clicking "OK" button for storing data, an alerting windows may appear in case one product is cheaper in the current store than in others.Another moment that gives an additional credit to the discussed product is the fact that the user can choose the period for showing the total amount spend,it's possible to see amount for today, for the week or an entire month.Assuming that project is in its early development stage,a lot of useful functionalities are going to be added further.

1.2 Similar products on the market

The idea of creating the discussed application came first of all from the society needs.Often are tackled subjects like saving for the future, paying off debt and protecting what people have got and everyone is complaining about lack of knowledge in how to do all of enumerated actions.

Spending is highly individual stuff.Beyond the golden rule—spend less than you earn—it's hard to be prescriptive because expenditures are the result of so many individual decisions, many of which feel not significant as people make them.So,in order to solve the frequently asked questions by the people "to spend on this" or "don't spend on that",today exists plenty of mobile and web applications which try somehow to track people's outflows.

Assuming that, in order to create a good software, it is necessary to make a market research and find similar solutions to the problems that are being solved by that software, making a connection between the system to be developed and the missing features in the existing similar applications.The Receipt Scanner application was inspired from another existing ones but with lack of the basic functionality - automated scanner of receipt through the phone video camera, and storing all the content of receipt in a digital text which allows to perform a lot of actions on it forward.According a Forbes research [1] almost all of existing similar applications require a manual input of the expenditures,which is somehow inconvenient and requires more time to do.

If to analyse some actual examples of such applications,can be observed that all of them require a manual input,besides this they have a lot of other nice functionalities which can be used as well for the receipt scanner further.

Dollarbird - Add past or future income and expenses to a calendar that calculates the impact on people balance, as well as spending by category. The basic features are:

- Price: Free
- Links to accounts: No
- Manual input: Yes
- Available on: iOS, Android

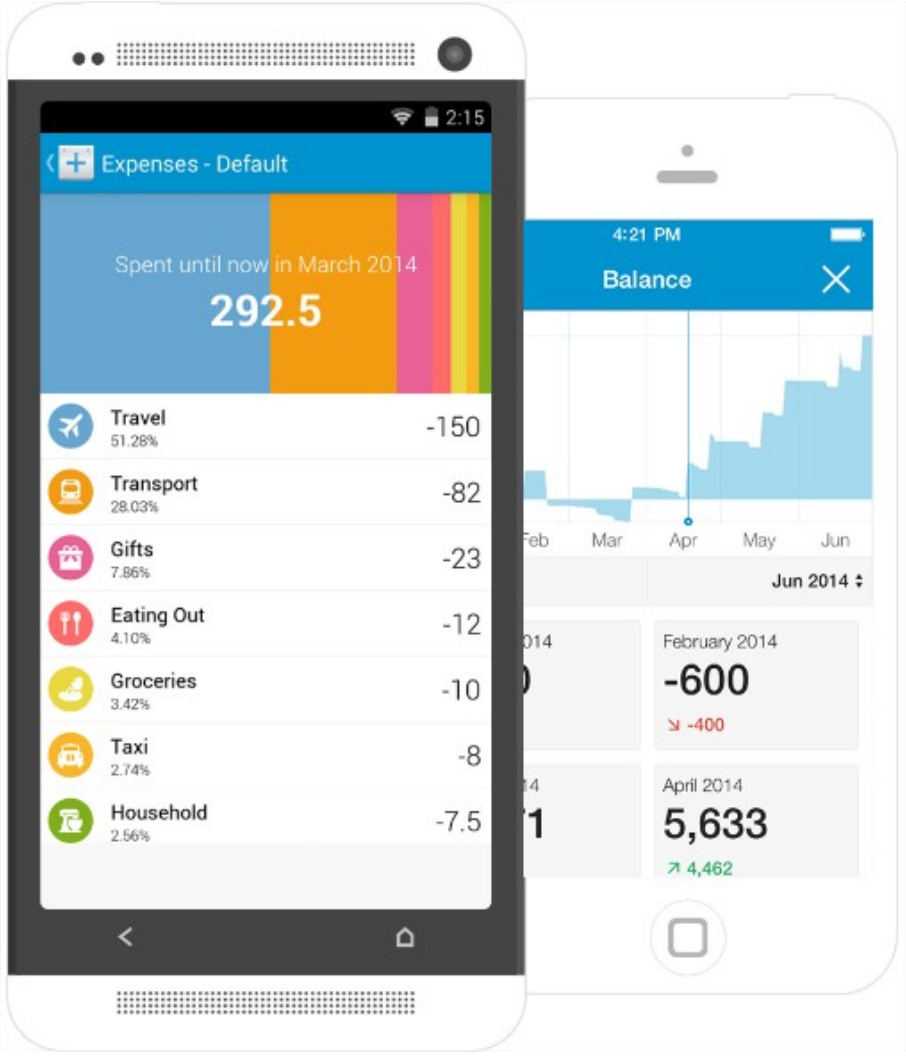

Figure 1.1– Dollarbird Application, [6]

Goodbudget - Recreates envelope budgeting of yesteryear for the digital world. Set a budget for each category and spend from that designated category. When the money run out stop spending.Bellow are represent some basic features:

- Price: Free; $45/year for unlimited categories, five years of history, to sync up to five devices
- Links to accounts: No
- Manual input: Yes
- Available on: iOS, Android

Figure 1.2– Goodbudget application, [7]

mvelopes - Like GoodBudeget, mvelopes is a spin on classic envelope budgeting but with account integration. Some basic features are enumerated bellow:

- Price: Free; $95/year to link unlimited accounts and to create more than 25 envelopes, as well as for a debt management tool
- Links to accounts: Yes
- Manual input: Yes
- Available on: iOS, Android

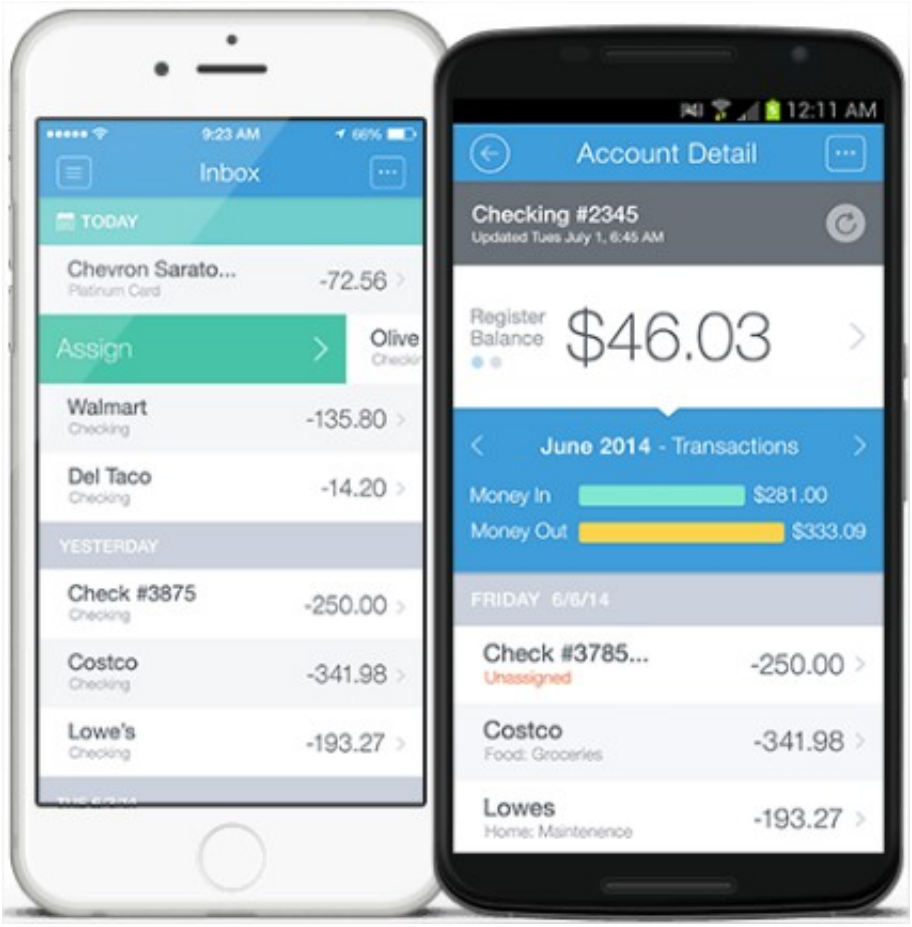

Figure 1.3– Mvelopes Application, [8]

So existing applications on market can serve as a prototype of the Receipt Scanner, with some improvement in that area of automating the process of manual input.

1.3 The development environment

Building an Android application comes down to two major skills or languages: Java and Android. Java is the language used in Android, but the Android part surround learning XML for the design of the application, learning the concepts of Android, and using the concepts programmatically with Java.The Mobile Vision Text API gives a powerful and reliable Optical Character Recognition (OCR) capability that works with most Android devices,that's why this is exactly technology which is used for recognizing easily all the text from a receipt.

When writing an Android application,there are some primary points which need to be taken in consideration:

– Describing and modelling the application's domain which represents the universe of the application is the key point before starting.In this case, the domain represents an application that would allow scanning the receipts and will return a budget report,weekly,monthly or even daily.At this point need to be established what's in it,how many views and actions contains the application and what is the relation between them.This is like modelling a database structure to keep the entities and their relationship.

– Specifying what can happen in this domain is another key point, it is necessary to identify all the possible scenarios or actions that the elements of the domain can participate in.

– Choosing the right IDE(Integrated Development Environment) is not less important point.The most common IDE for Android development and that used for the current project is named Android Studio,it comes direct from the Google itself.It gives the main UI when code is entered,it highlights things which are wrong,offers suggestions and lets running and testing creations conveniently.It creates files, provides basic layouts and generally it saves a lot of time and effort.

1.3.1 Back-end implementation: Java for Android

For implementing the idea of Spending Tracker, Java for Android language was chosen. Such as the application from the start point was designed to be a mobile application,then there was not a better choice than to use Java,it's really the only option for native applications.It is one from the most popular programming languages,namely for its important core features as are:

– easy to learn and understand;

– designed to be a platform-independent and secure,using virtual machines;

– it's object oriented.

Android relies intensely on these Java fundamentals. The Android SDK includes many standard Java libraries as well as special Android libraries, as is Mobile Vision Text API for Android, the basic library used in developing Spending Tracker.

Spending Tracker application is based generally on OCR (Optical Character Recognition) used with Mobile Text API for Android. OCR gives a computer the ability to read text that appear in an image. The Mobile Vision Text API gives Android developers a powerful nd reliable OCR capability that works with most Android devices and won't increase the size of the application.[2]

Text recognition is the process of detecting text in images, video streams and recognizing the text contained in.When it is detected,than the recognizer determines the actual text in each block and segments into lines or words.The Text API recognize text in any Latin based language. [3]

In software applications, it is mostly required to store user's and application data locally, so that was stored for Spending Tracker as well.**Realm** is one of the way of storing data provided by Android SDK. Realm is is a mobile database and a replacement for SQLite.Realm store data in a universal, table-based format by a C++ core. This is what allows Realm to allow data access from multiple languages as well as a range of ad hoc queries. Some key points for understanding better Realm are enumerated bellow:[9]

– faster than SQLite (up to 10x speed up over raw SQLite for normal operations)

– easy to use

– object conversion handled for user

– convenient for creating and storing data on the fly

– very responsive team

1.3.2 Model-View-Controller architecture

This chapter can be started with the best expression about MVC architectural patterns: "The only way to make the deadline—the only way to go fast—is to keep the code as clean as possible at all times.",said by *Robert C. Martin, Clean Code: A Handbook of Agile Software Craftsmanship*[**?**].

MVC is an architectural pattern which means **Model,View,Controller** and its aim is to provide kind of a "template" for organization of software system.The idea of MVC is that of dividing the software into three independent components:

– The component that stores system's state (whether this state persistent or not). This component is referred to as **Model**.

– The component that handles input-output from/to the client (the client might be a human user, but it doesn't have to). This component is referred to as **View**.

– The component that represents the logical functionality of the system, holds system's "business rules". This component is referred to as **Controller/Presenter**.

Bellow are represented interaction diagrams for that two similar architectural patterns MVC and MVP,the differences are not such significant,but for Android development, MVP is more suitable and next will be explained why so.

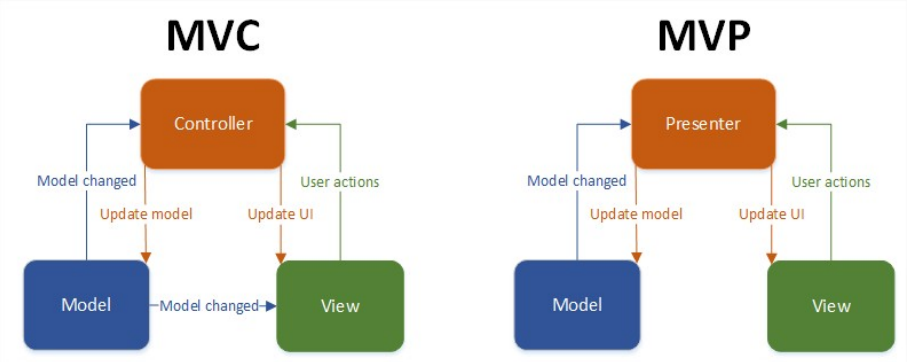

Figure 1.4– MVC pattern integration with Android, [4]

MVP was proved to be more suitable for Android development due to very tight coupling between Activities and Fragments (which are MVP presenters) and various parts of Android framework. The "Model" part is fine - it is straightforward to implement a standalone model functionality in Android. The problems arise if to try to separate view and controller functionality.For example *onCreate()* method,which is called by Android framework in response to framework's internal state transitions (these transitions might be related to user's actions in the app, but the app never calls onCreate() directly). So can be created an impression that this method belongs to the "view" part of MVC because setContentView() is called along the way, but, in fact, setContentView() is a standard binding of a view performed by the controller.

In MVP, the view knows nothing about the model, and it is presenter's job to fetch the up to date data from the model, understand whether the view should be updated and bind a new data to the view. In MVP, the logic contained in MVC view, is located in the presenter, which makes the views very "stupid" – their sole purpose becomes rendering data that was bound to them by the presenter.Finally, good MVP model tends to just manage application's data state and notify presenters of any change, nothing more.

1.3.3 Front-end implementation in Android

Android front-end implementation could be defined in two ways:

- **Declare UI elements in XML.** Android provides a straightforward XML vocabulary that corresponds to the View classes and subclasses, such as those for widgets and layouts.
- **Instantiate layout elements at runtime.**The application can create View and ViewGroup objects (and manipulate their properties) programmatically.

It is possible to use either or both of these methods for declaring and managing the application's UI.The advantage of declaring the UI in XML is that it enables to better separate the presentation of the application from the code that controls its behaviour.The UI descriptions are external to the application which means that application can be modified and adapted without modifying the source code and recompiling.Also, declaring the UI using XML makes it easier to visualize structure of UI,it's easier to debug problems.

In general, the XML vocabulary for declaring UI elements closely follows the structure and naming of the classes and methods, where element names correspond to class names and attribute names correspond to methods. [5]

2 Analysis of the application from architectural point of view

In the requirements phase, earlier,was built an abstract model of the problem,which identifies what objects,entities,views are involved,how they relate to one another and how they look like.Probably now is the moment to describe the software architecture more thoroughly, from the implementation point of view,describing all the most important interfaces,classes, interaction between user and system and last but not least - the interaction between system components.In order to characterize the Spending Tracker application system, a collection of models were used:

- **Use Case model**.It represents a set of use cases, actors, and their relationships.A use case represents a particular functionality of a system and it is used to describe the relationships among the functionalities and their controllers known also as **actors**.
- **Activity model**.It describes the flow of control in a system. It consists of activities and links.Activities are nothing but the functions of a system. Numbers of activity diagrams are prepared to capture the entire flow in a system.
- **Sequence model**.A sequence diagram is an interaction diagram.It is clear that the diagram deals with some sequences, which can be sequence of messages flowing from one object to another.
- **Database model**.It represents a static structure diagram which underlines which are the entities where data are stored in the system,and the relationships which exist between them. [13]

This is the chapter where the Spending Tracker application is analysed using the above described diagrams.This will help to get a more clear image and a better understanding about the entire system and about the components interaction.

2.1 Representation of the system via Use Case Diagrams

2.2 System Analysis using Activity Diagrams

2.3 Process interaction analysis using Sequence Diagram

2.4 The Database Model of the system

2.5 Final stage of product development: Deployment Diagram

3 Implementation and development methodologies

System Requirments

For the user to run and have a nice use of Spending Tracker application,it's necessary to have:

- A mobile device with an Anroid system with minimum version of 4.0.3, which means minimum API Level to be 15.
- A reliable internet connection for downloading application from the Play Store.

In this chapter the Spending Tracker application is analysed from the implementation point of view.For the beginning some quick notes about the most important libraries used will be mentioned,in order to make an overview of the entire implementation environment of the application.

Spending Tracker Android application is based on four main libraries:

- **Realm** java [9]
  A mobile database which is convenient for creating and storing data on the fly.
- **MpAndroidChart** [10]
  MpAndroidChart is a powerful and easy to use chart library for Android.
- **Android support libraries** [11]
  The Android Support Library is not actually a single library, but rather a collection of libraries which provide a wide range of classes for building app.Many of the classes are backward compatible implementations, but some of them are new features in their own right.
- **OCR Document API / Mobile Vision Text API**, [2]
  Using OCR Document API,the feature of reading text and numbers from any image,in the Spending Tracker app from the receipt, is easily added to any application without prior knowledge about OCR(Optical Character Recognition) technology.

More details about the implementation of each functionality of Spending Tracker application are described in the next subsections.

3.1 Realm Java

Realm Java gives the possibility to efficiently write the application's model layer in a safe, persisted and fast way.It is installed as a Gradle plugin as follows:

```
buildscript {
  repositories {
    jcenter()
  }
  dependencies {
    compile 'io.realm:realm-android:0.82.2'
  }
}
```

Listing 1 – Realm gradle plugin

Realm is a mobile first database that is built from the ground-up to run directly inside phones and tablets.It is kind of what user see is what is saved workflow, changes to the object in the user interface are automatically saved to the database if that object is a Realm managed object. Realm managed objects are equivalents of SQLite tables. For Java object to become a Realm managed, the class must either extend RealmObject or implement RealmModel interface. The Spending Tracker application have two entities which are managed by Realm object, Expense and Category. Bellow is represented the model of the Expense class:

```
public class Expense extends RealmObject {

  @PrimaryKey
  private String id;
  private String description;
  private Date date;
  private @IExpensesType int type;
  private Category category;
  private float total;

  public Expense() {
  }

  public Expense(String description, Date date, @IExpensesType int type, Category
     category, float total) {
    this.description = description;
    this.date = date;
    this.type = type;
    this.category = category;
    this.total = total;
  }
}
```

Listing 2– Realm Expense model

Actually this is equivalent with creating a table in SQLite, but with Realm the data is in motion or live auto updated object.So when is updating any field of the class,is needed to open a Realm transaction at the point where the value is retrieved from the TextView or another entity.For example bellow is the method of getting expenses from the corresponding fields:

```
public static RealmResults<Expense> getExpensesList(Date dateFrom, Date dateTo,
    @IExpensesType Integer type, Category category) {
  RealmQuery<Expense> realmQuery = RealmManager.getInstance().getRealmInstance()
  .where(Expense.class);
  if (dateTo != null) {
    realmQuery.between("date", dateFrom, dateTo);
  }else {
    realmQuery.equalTo("date", dateFrom);
  }
  if (category != null) realmQuery.equalTo("category.id", category.getId());
```

```
10    if (type != null) realmQuery.equalTo("type", type);
11    return realmQuery.findAll();
12  }
```

Listing 3– Realm queries example

3.2 Android Support Libraries

Spending Tracker application is based on Android Support Libraries,which have few distinct uses.

- **Backward Compatibility for newer APIs** - many support libraries provide backward compatibility for newer framework classes and methods. For example, the **Fragment** support class which is used in Spending Tracker as well, provides support for fragments on devices running versions earlier than Android 3.0 (API level 11).
- **Convenience and Helper Classes** - also support libraries provides a number of helper classes, especially for user interface development. For example the **RecyclerView** class used in Spending Tracker app as well,provides an user interface widget for displaying and managing very long lists, useable on versions of Android from API level 7 and up.

For a better understanding about the usage of these libraries,some of them are described bellow each in a separate subsection, using the pieces of code from the Spending Tracker application.

3.2.1 Fragments in the Support Libraries

Fragments represent some modular building blocks which help in creating a responsive Android UI.They help in targeting different devices and screen sizes.Actually they constitute a part of the Activity presented to the user,they are used to display some section of the screen and to react to events that happen within this section.

Spending Tracker application is based on support library Fragment of the package:

$$com.android.support : support - v4 : 25.3.1$$

Almost each entity in the application is based on a Fragment,there are fragments for: categories, expenses,history,login screen and for date selecting as well.To explain a little lifecycle of fragments let get the Expense Detail Fragment example.This fragment inflates a view.But it does not do so in the *onCreate()* method like you would do in an activity, but within the *onCreateView()* method.Android passes the inflater needed to use for view inflation into the method, as well as the container for the view.

```
1  public View onCreateView(LayoutInflater inflater, ViewGroup container,
2    Bundle savedInstanceState) {
3    View rootView = onCreateFragmentView(R.layout.fragment_expense_detail, inflater
       , container, true);
4    bcWeekExpenses = (BarChart) rootView.findViewById(R.id.bc_expenses);
5    return rootView;
```

```
}
```

Listing 4 – View inflating in Expense Fragment

To return a layout from *onCreateView()*,it is inflated from a layout resource defined in XML, in the example above fragment_expense_layout.xml file is loaded by the Fragment.

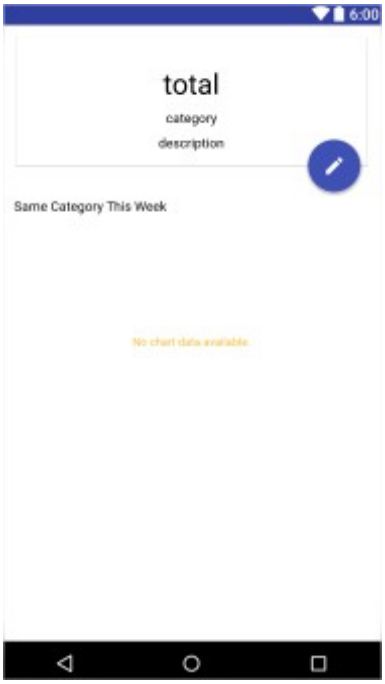

Figure 3.1 – Expense Detail Fragment layout

Since fragments can be attached or detached from an activity, there also exist onAttach() and onDetach() methods which are located in the MainFragment class in Spending Tracker app. After the fragment has been attached,can be used getActivity() within the fragment to get to the enclosing activity.This can be observed inside the *onClick()* method in the Expenses Fragment class.

```
public void onClick(RecyclerView.ViewHolder vh, int position) {
  if (!expenseContainerListener.isActionMode()) {
    Expense expenseSelected = (Expense) vh.itemView.getTag();
    Intent expenseDetail = new Intent(getActivity(), ExpenseDetailActivity.class)
    ;
    expenseDetail.putExtra(ExpenseDetailFragment.EXPENSE_ID_KEY, expenseSelected.
   getId());
    startActivity(expenseDetail);
  } else {
    toggleSelection(position);
}
```

Listing 5 – Getting to the enclosing activity

The same it works for all other fragments in the project.

Adding a fragment to an activity

There are two ways of adding a fragment to the activity layout, such as a fragment contributes a portion of UI to the host activity, which is embedded as a part of the activity's overall view hierarchy.

– Declare the fragment inside the activity's layout file.

– Or, programmatically add the fragment to an existing ViewGroup.

In the Spending Tracker application was used the second way.To make fragment transaction in the activity (such as add, remove, or replace a fragment), the APIs from FragmentTransaction must be used.The instance of FragmentTransaction from the Activity was obtained in the *replaceFragment()* method like this:

```
public void replaceFragment(int containerId, Fragment fragment, boolean
    addToBackStack) {
  FragmentTransaction transaction = getSupportFragmentManager().beginTransaction
    ();
  String tag = fragment.getClass().getSimpleName();
  transaction.replace(containerId, fragment, tag);
  if(addToBackStack) transaction.addToBackStack(null);
  transaction.commit();
}
```

Listing 6– Instance of Fragment Transaction

The Fragment is replaced then in another method which has the same name,specifying the fragment to add and the view in which to insert it.

```
public void replaceFragment(Fragment fragment, boolean addToBackStack) {
  replaceFragment(R.id.main_content, fragment, addToBackStack);
}
```

Listing 7– Fragment replacing

By calling addToBackStack(), the replace transaction is saved to the back stack so the user can reverse the transaction and bring back the previous fragment by pressing the Back button. Once the changes with FragmentTransaction were made, commit() must be called for the changes to take effect.

3.2.2 Activities

The Activities are crucial components of an Android application.Generally, one activity implements one screen in an app.Most apps contain multiple screens, which means they comprise multiple activities as happens in Spending Tracker application as well. For the app to be able to use activities,the activities and certain of their attributes must be declared in the manifest.For example,here are declared two activities from Spending Tracker app:

```
<activity
  android:name=".ui.login.LoginActivity"
  android:configChanges="orientation"
  android:label="@string/app_name"
  android:screenOrientation="portrait"
  android:theme="@style/AppTheme.NoActionBar.TransparentStatusBar" >
  <intent-filter>
```

```
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
  </intent-filter>
</activity>
<activity android:name=".OcrCaptureActivity">
  <intent-filter>
    <category android:name="android.intent.category.LAUNCHER"/>
  </intent-filter>
</activity>
```

Listing 8– AndroidManifest.xml

The only required attribute for this element is android:name, which specifies the class name of the activity. The rest of attributes define activity characteristics such as label, icon, or UI theme.There are also implicit requests which are declared using **¡intent-filter¿** attribute, in the **¡activity¿** element.The definition of this element includes an **¡action¿** element and, optionally, a **¡category¿** element.These elements combine to specify the type of intent to which your activity can respond.In the above example from Spending Tracker, for the first activity **android.intent.action.MAIN** means that this activity is the entry point of the application,when the application is launched, this activity is created.Category parameter gives additional information about the action to execute,**category.LAUNCHER** means it should appear in the Launcher as a top-level application.

Over its lifetime,an activity goes through a number of states.A series of callbacks can be used for handling transactions between states,such as are: onCreate(), onStart(), onResume(), onPause(),onStop(), onRestart(), onDestroy().The most trivial one is **onCreate()** callback.Bellow is presented an example from the MainActivity of Spending Tracker application.

```
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  initUI();
  setUpDrawer();
  setUpToolbar();
  if ( savedInstanceState != null) {
    int menuItemId = savedInstanceState.getInt(NAVIGATION_POSITION);
    mainNavigationView.setCheckedItem(menuItemId);
    mainNavigationView.getMenu().performIdentifierAction(menuItemId, 0);
  } else {
    mainNavigationView.getMenu().performIdentifierAction(R.id.nav_expenses, 0);
  }
}
```

Listing 9– onCreate() callback of MainActivity

This callback must be implemented,it fires when system creates the activity.This callback initializes the essential components of the activity.This is where **setContentView()** must be called to define the layout for the activity's user interface.

3.3 Optical Character Recognition on Android – OCR

Android can perform OCR very efficiently and correctly using the official Optical Character Recognition API of Android and the Mobile Vision library.Google has introduced recently **Mobile Vision APIs** [12], which provide a very easy to use programming interfaces through which can be scanned Faces, Barcodes,QR Codes and Text without writing huge amount of code.And the best part is that these can be used offline as well.To enable the application to use Mobile Vision APIs, this dependency is needed to be added in the build.gradle file:

$$compile'com.google.android.gms : play - services - vision : 9.4.0+'$$

Introducing an Android OCR Library – Text Recognition API
This Google powered API contains features like multiple language recognition,the Text API recognize text in any Latin based language. Also the text can be parsed from a stream of frames i.e. a video and displayed on the screen in real time.In Spending Tracker application the things are kept simple and the text is scanned from the photo of the receipt.All this is done offline by the Google Play services itself, i.e. no internet connection is required after once it has been set up in the application. The Text Recognizer segments text into blocks, lines, and words.

- a **Block** is a contiguous set of text lines, such as a paragraph or column;
- a **Line** is a contiguous set of words on the same vertical axis;
- a **Word** is a contiguous set of alphanumeric characters on the same vertical axis

Bellow is represented an example of each of these in descending order:

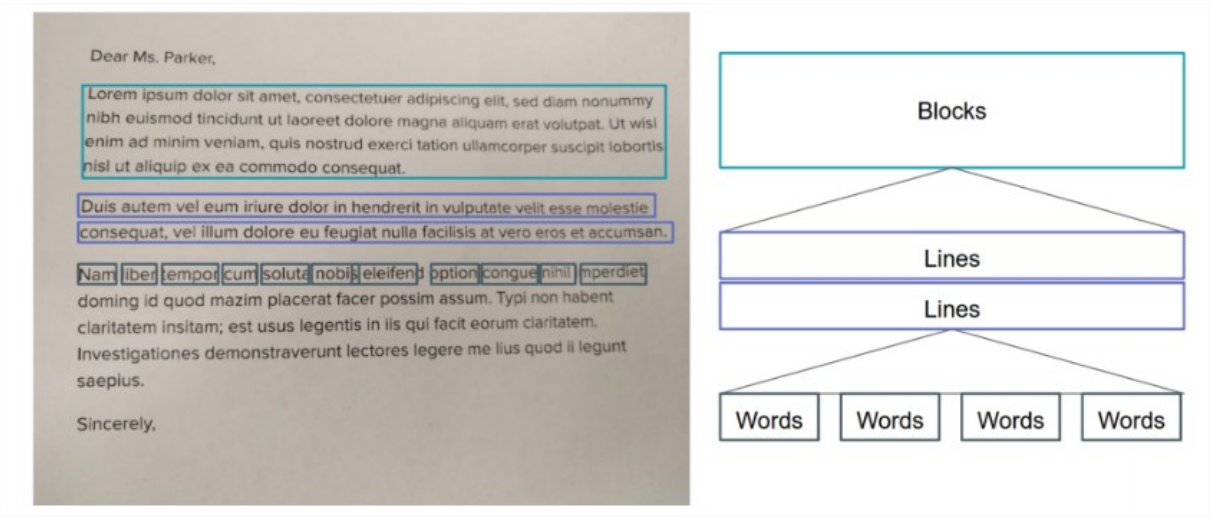

Figure 3.2– Text Recognizer Example [3]

3.3.1 Tesseract library

3.3.2 Easy OCR library

3.4 User's experience handling the product

4 Economic Analysis

4.1 Project description

Spending Tracker is an application developed for different target groups,for young people which face problems at the chapter of correct money management and as well for older people,who want to monitor all expenses,such as it's user interface is very clear and simple to use for all groups of people in the society.The main task of the application is scanning the receipts and displaying the total expenses for different periods,depending on user's choice,and for different stores,which present categories in the statistics bar chart.Such as it does not require a manual input introducing, it presents a major advantage for its users.This application ensures managements of two most important values in the life: money and time.It saves time by offering that simple possibility of scanning the receipt, and it saves money of course because the main purpose of the application is tracking right the budget,by monitoring day by day all the expenses,and making some decisions according the correctness of their spending.

It is worth mentioning that such applications already exist on the market but they're lack of the most functionality the Spending Tracker has,automatic receipt scanner by taking a photo of the receipt.Also,it's strong point is the simplicity in use,it does not require connection to a bank account or other dubious actions as can be observed in other existing applications on market.It offers exactly what needs every person,nothing more,a user friendly interface and a quick result displaying.

Before proceeding to the implementation of the system,it is necessary to analyse the project budget,this will help to find a justifiable economical point of view over the system.In order to understand and decide if the product is workable on the national market,it would be better to elaborate an overview of the expenditures and incomes after releasing the application.Given that fact,the economic analysis should be one of the first steps in starting development of any product,whose final result would be a general overview on the elaboration of the system.

This chapter represents the analysis of all expenses necessary to elaborate the application,starting from materials/non-materials used for long term, time schedule establishment and indirect expenses. After that,the days necessary for development should be estimated,should be established the working team and of course each team member salary.Only after this starting analysis will be clear what steps should be performed further in order to have a gain.Even this chapter is last,it is in fact the most important one in real life,because economic analysis is a means to help bring about a better allocation of resources that can lead to enhanced incomes for investment or consumption purposes,it is best undertaken at the early stages of the project cycle to enable decision makers to make an informed decision on whether to undertake a particular investment given various alternatives and their corresponding costs.

4.2 Project time schedule

Taking into account that the Spending Tracker application is quite complex and have in the requirements list many features to be implemented,it would be better to plan a time schedule.In order to get the better results faster,and also a considerable income,will be tried to be used **Agile Software Development**.In such a way the product will be delivered faster,maybe not implemented at all,a process of frequent feedbacks will be adopted for excluding risks of a single-pass development when all the requirements are implemented.The future modifications can be treated as improvements for the base application which represents a benefit of Agile.

4.2.1 SWOT Analysis

4.2.2 Defining objectives

4.2.3 Time schedule establishment

4.3 Economic motivation

4.3.1 Tangible and intangible asset expenses

4.3.2 Salary expenses

4.4 Individual person salary

4.4.1 Indirect expenses

4.4.2 Wear and depreciation

4.4.3 Product cost

4.4.4 Economic indicators and results

4.5 Economic conclusions

Conclusions

References

1 Forbes, *12 Free Apps To Track Your Spending*,
https://www.forbes.com/sites/samanthasharf/2016/03/02/12-free-apps-to-track-your-spending-and-how-to-pick-the-best-one-for-you/#357a13005445

2 Google Codelabs, *See and Understand Text using OCR with Mobile Vision Text API for Android* ,
https://codelabs.developers.google.com/codelabs/mobile-vision-ocr/#0

3 Text Recognition API Overview,
https://developers.google.com/vision/text-overview

4 MVP and MVC Architectures in Android, *Which architectural pattern is more suitable for Android development – MVC or MVP?*
http://www.techyourchance.com/mvp-mvc-android-1/

5 Android Layouts,
https://developer.android.com/guide/topics/ui/declaring-layout.html

6 Dollarbird, http://www.dollarbird.co/

7 Goodbudget, https://goodbudget.com/

8 Mvelopes, https://www.mvelopes.com/

9 Android Working with Realm Database,
http://www.androidhive.info/2016/05/android-working-with-realm-database-replacing-

10 MpAndroidChart, https://github.com/PhilJay/MPAndroidChart

11 Android Support Libraries,
http://developer.android.com/intl/es/tools/support-library/features.html

12 Mobile Vision APIs https://developers.google.com/vision/

13 Structural UML Diagrams, https://www.tutorialspoint.com/uml/uml_standard_diagrams.htm