**Bachelor degree**

**Accountability of Utilities**

**(ACCU)**

Scientific coordinator Student

**Lect. Dan Cojocar Alexandru-Vasile Moldovan**

# **Table of contents**

## 1. Table of contents1

## 2. Introduction2

## 2.1 Problem approach2

## 2.2 Solution developed4

## 3. About Android1

3.1 General information5

3.2 Android features and architecture5

## 4. Accountability5

## 5. Accountability of Utilities (ACCU)

## 5.1 Description of the application5

## 5.2 Technologies used6

## 5.3 Android code structure7

## 5.4 Functional and Non-functional requirements11

## 5.5 Server side12

## 5.6 Mobile client side13

## 5.7 Use cases14

## 6. User manual15

## 7. Testing and validation16

## 8. Conclusions18

## 9. Bibliography19

1. **Introduction**

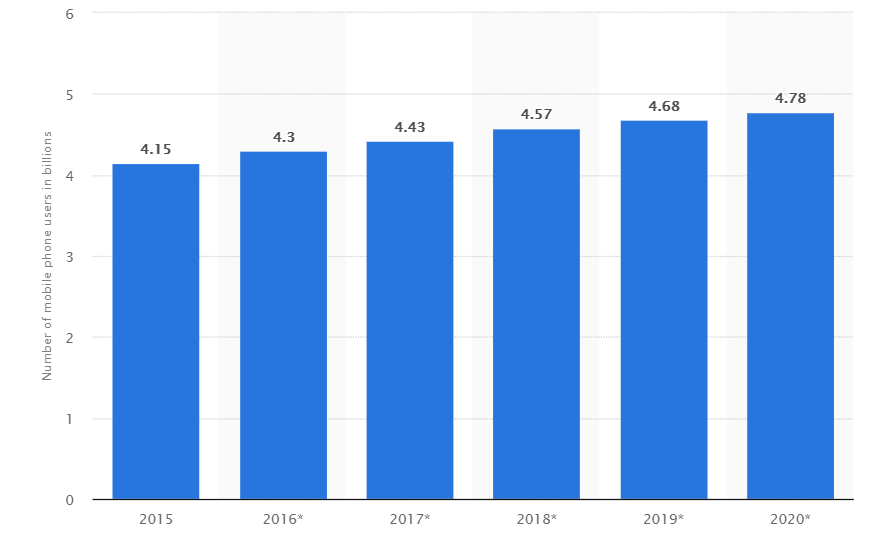
**2.1 Problem approach**

Nowadays, mobile applications are used in many fields of activity, even in accountability. Mobile features have developed a lot over the past years such that everything it is now possible on an android device.

Mobile phones have become a necessity in everyone's life. Being used as an alternative for a computer, a phone can also be used for personal entertainment (playing games, navigating through social media platforms, listening to music, etc.) besides the functionalities that it was originally created for (communication between people).

According to “Statista” [11], at the moment, there are 4.68 billion phone users worldwide and the number is increasing year by year. In 2020 it is presumed that there will be a growth of 0.10 billion phone users.

*Graph of phone users worldwide*



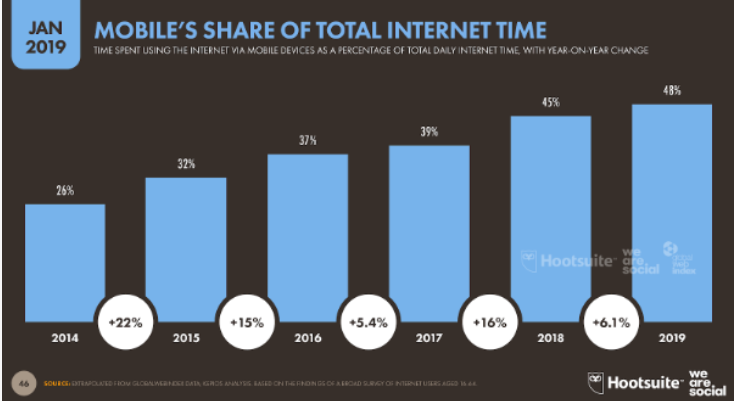
*Figure 2.1.1 [11]*

Accessing the internet has also become easier, even if you are not at home, having a smartphone or a laptop with you, a simple touch on the screen or a click on the browser can do the trick. The Internet plays a big role in our life, it helps us find a huge amount of useful information on every existing subject and “there is no question that the Internet has, and will continue to have, a major impact on consumer information search behavior” [15].

According to “We Are Social” [3], the total number of mobile internet users it is 3.98 billion and on average the world’s internet users spend around 6 hours and 42 minutes online every day.

The graph below shows us the time spent on the internet by mobile users.

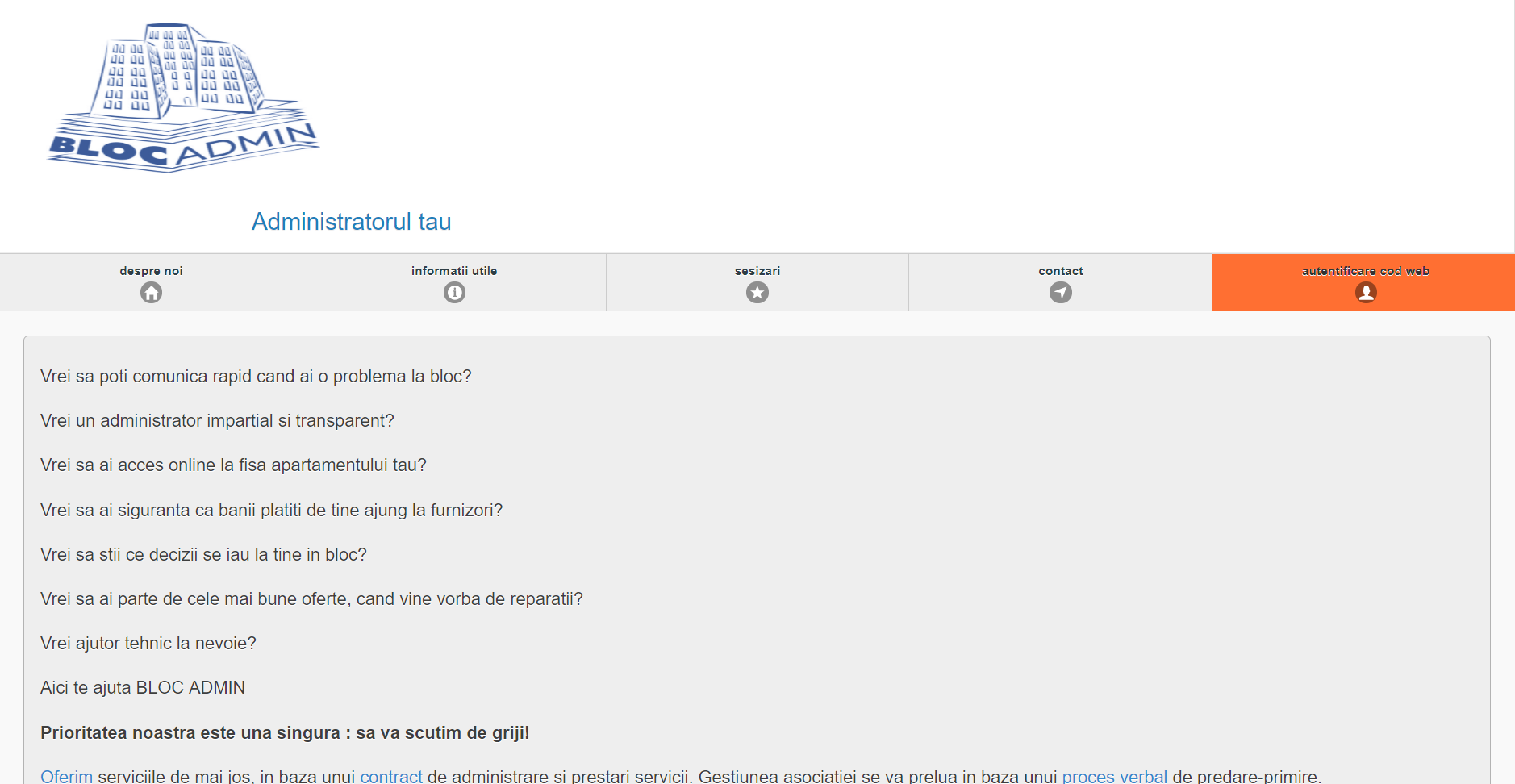
*Mobile’s share of total internet time*



*Figure 2.1.2 [3]*

Having access to so much information so easily, one would think that about the work an administrator of a block of flats has to do every month, and how easy could this be for the administrator but also for the residents if there was a mobile application that can simplify all the work. As far as I know there a web application that has implemented functionalities that manage data about the utilities of the people and allows them to pay their bills already exists. The site name is “Bloc-Admin” [13], but I do not know any mobile application related or similar with the functionalities of the mentioned website above. The website users have to authenticate using the code from the utilities bill and after that they can send all the data they have to send to the administrator. They can pay by credit card by sending money in a banking account, all of this can save precious time for the user but also for the administrator.

*Bloc-Admin website*



*Figure 2.1.3 [13]*

So, I came up with the idea of an accountability mobile application that stores the data of used utilities by the residents from a block of flats. The application has an administrator flow and a user (resident) flow, each of those two have specific functionalities depending on the type of account. Each account type has certain permissions and actions that can be done; a summary of the solution developed will be presented in the next chapter.

**TODO: write more? If yes, what?**

**2.2 Solution developed**

The problem I approached was how to make the job of an administrator of block of flats easier. With the details given in the previous chapter, I developed an application, application split in two flows, an administrator parts which can read only the data collected from the residents, and a user part from where he sends data to the administrator.

**The administrator part**

The administrator features were developed such that he can receive data sent by residents, for example: water meter, electricity meter, gas meter or how many people lived in an apartment that month. He is also able to create events to which users can respond, saying if they attend or not, for example a meeting with all the residents of a particular block of flats. The administrator can update his profile and create accounts for residents but also can set up prices for reparations that are made in the building, price for the usage of a garage (if the building has garages) and a price for cleaning the building.

**Resident part**

The resident features were developed such that the resident can send data to the administrator about their usage of utilities (water meter, electricity meter, gas meter or how many people lived in an apartment that month). The residents can receive notifications about events created by the administrator, and can choose to attend or not on a specific event; the users (residents) can see a prediction for their usage of utilities for the next month also they can update their profile, and can see an archive of the previous reports of their utilities. They can also see the cost of cleaning the building, repairing different things from the building or if they are using a parking spot in garage how much is the fee per month.

**TODO: write more?**

1. **About Android**

**3.1 General information**

“Android is a mobile operating system that is based on a modified version of Linux.” [1] “The development of Android operating system all started in 2003 by the pioneer owners, Android, Inc.” [2] “In 2005, as part of its strategy to enter the mobile space, Google purchased Android and took over its development work (as well as its development team).

Google wanted Android to be open and free; hence, most of the Android code was released under open source Apache License, which means that anyone who wants to use Android can do so by downloading the full Android source code Moreover, vendors (typically hardware manufacturers) can add their own proprietary extensions to Android and customize Android to differentiate their products from others. This simple development model makes Android very attractive and has thus piqued the interest of many vendors.

The main advantage of adopting Android is that it offers a unified approach to application development. Developers need only develop for Android, and their application should be able to run on numerous different devices, as long as the devices are powered using Android. In the world of smartphones, applications are the most important part of the success chain.

In the picture below are all the Android versions (except the Android Pie operating system which was released in 7 March 2018). It can be seen that after Android Beta verison each operating system that was released, received the name of something sweet in alphabetic order.” [1]

*Android versions*

****

*Figure 3.1.1 [2]*

**3.2 Android features and architecture**

“Because Android is open source and freely available to manufacturers for customization, there are no fixed hardware or software configurations. However, Android itself supports the following features:

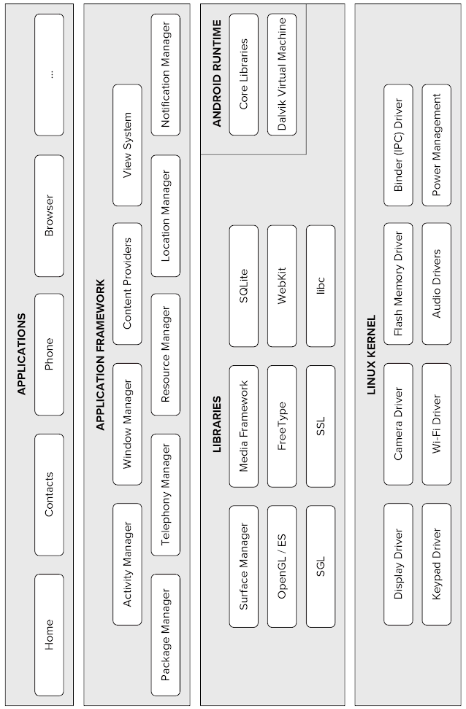
* **Storage** – Uses SQLite, a lightweight relational database, for data storage.
* **Connectivity** – Supports GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth (includes A2DP and AVRCP), Wi-Fi, LTE, and WiMAX.
* **Messaging** – Supports both SMS and MMS.
* **Web** **browser** – Based on the open source WebKit, together with Chrome’s V8 JavaScript engine.
* **Media** **support** – Includes support for the following media: H.263, H.264 (in 3GP or MP4 container), MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF and BMP.
* **Hardware support** – Accelerometer Sensor, Camera, Digital Compass, Proximity Sensor and GPS.
* **Multi**-**touch** – Supports multi-touch screens.
* **Multi**-**tasking** – Supports multi-tasking applications.
* **Flash support** – Android 2.3 supports Flash 10.1.
* **Tethering** – Supports sharing of Internet connections as a wired/wireless hotspot.

The Android OS is roughly divided into five sections in four main layers:

* **Linux kernel –** this is the kernel on which Android is based. This layer contains all the low-level device drivers for the various hardware components of an Android device.
* **Libraries –** These contain all the code that provides the main features of an Android OS. For example, the SQLite library provides database support so that an application can use it for data storage. The WebKit library provides functionalities for web browsing.
* **Android runtime –** At the same layer as the libraries, the Android runtime provides a set of core libraries that enable developers to write Android apps using the Java programming language. The Android runtime also includes the Dalvik virtual machine, which enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine (Android applications are compiled into Dalvik executables). Dalvik is a specialized virtual machine designed specifically for Android and optimized for battery-powered mobile devices with limited memory and CPU.
* **Application framework –** Exposes the various capabilities of the Android OS to application developers so that they can make use of them in their applications.
* **Applications –** At this top layer, you will fin application that ship with the Android device (such as Phone, Contacts, Browser, etc.), as well as applications that you download and install from Android Market. Any applications that you write are located at this layer.

In order to understand how Android works, take a look at Figure 3.2.1, which shows the various layers that make up the Android operating system (OS).” [1]

*Layouts of Android operating system*



*Figure 3.2.1 [1]*

# **Accountability**

**TODO: Find article and write about accountability**

1. **Accountability of Utilities (ACCU)**

**5.1 Description of the application**

Accountability of Utilities, ACCU, is an android mobile application developed in Android Studio using Java a programming language. The mobile application communicates with a server written in PHP programming language, the server allows CRUD operations to be made on the database called “utilities” created in MySQL.

The application has two user flows, the first flow is the administrator flow and the second one is the resident flow. Each flow starts on the log in screen where the user can insert its log in credentials but also, he is able to create a new account and after the user fills in all the necessarily data, he is asked to choose the address of his residence.

Depending on the inserted credentials for log in the application checks if it is an administrator or a resident. Depending on the received response, the application starts the administrator flow or the resident flow.

**Administrator flow** TODO

**Resident flow** TODO

As an innovation the application has implemented an artificial intelligence that will suggest a consumption of the utilities for the next month based on the consumption from the previous months.

**5.2 Technologies used**

**Android Studio IDE**

Android Studio is an Integrated Development Environment(IDE) created by Google. “Android Studio IDE can be used to create applications for different kinds of devices such as phones, phablets, tablets, watches and other wearables, Google Glass, TV applications and even auto applications. Android Studio is the recommended IDE to develop Android applications and is available for free for anyone who develops professional Android applications. Android Studio is based on the JetBrains IntelliJ IDEA software, which might explain why even the preview and beta versions of Android Studio were already better than Eclipse and why many Android developers were using it as their IDE from the beginning.

The first stable build of Android Studio was released in December 2014 and has replaced Eclipse (with Android Development Tools) as the primary IDE for Android development. Now with Android Studio, we do not just have a more stable and faster IDE, but it also comes with cool stuff such as Gradle, better refactoring methods, and much better layout editor to name just a few of them.

There are a lot of Android versions, resulting in fragmentation. Because of this, you cannot expect that all devices will run on the latest Android version, in fact, most do not. Many devices still run on Android 4.x (or even older versions), for example.” [16]

For being usable, Android Studio needs a **wizard** **tool** like **Java Development Kit** (JDK). As for my application, I used in Android Studio the Java Development Kit version 1.8 and the application is designed mostly for the Android Kit Kat operating system version, it can be used also on other android versions but inconsistency can occur between them and the Kit Kat version. Also, the application needs permissions in order to be able to access different tasks, like permissions for camera, internet or folder access. For my application there is need only for network state and internet.

**Java**

“The **JavaTM** programming language is a general-purpose, concurrent, class-based, object-oriented language. It is designed to be simple enough that many programmers can achieve fluency in the language. The Java programming language is related to C and C++ but is organized rather differently, with a number of aspects of C and C++ omitted and a few ideas from other languages included. It is intended to be a production language, not a research language, and so, as C. A. R.

The Java programming language is strongly typed. This specification clearly distinguishes between the compile-time errors that can and must be detected at compile time, and those that occur at run time. Compile time normally consists of translating programs into a machine-independent byte code representation. Run-time activities include loading and linking of the classes needed to execute a program, optional machine code generation and dynamic optimization of the program, and actual program execution.

The Java programming language is a relatively high-level language, in that details of the machine representation are not available through the language. It includes automatic storage management, typically using a garbage collector, to avoid the safety problems of explicit deallocation (as in C’s *free* or C++’s *delete*). High-performance garbage-collected implementations can have bounded pauses to support systems programming and real-time applications. The language does not include any unsafe constructs, such as array accesses without index checking, since such unsafe constructs, would cause a program to behave in an unspecified way

Classes support single implementation inheritance, in which the implementation of each class is derived from that of a single superclass, and ultimately from the class *Object.* Variables of a class type can reference an instance of that class or of any subclass, allowing new types to be used with existing methods, polymorphically.

Classes support concurrent programming with *synchronized* methods/ Methods declare the checked exceptions that can arise from their execution, which allows compile-time checking to ensure that exceptional conditions are handled. Objects can declare a *finalize* method that will be invoked before the objects are discarded by the garbage collector, allowing the objects to clean up their state.

For simplicity, the language has neither declaration ‘headers’ separate from the implementation of a class nor separate type and class hierarchies.

Although the language does not include parametrized classes, the semantics of arrays are those of a parametrized class with some syntactic sugar. Like the programming language Beta, the Java programming language uses a run-time type check when sorting references in arrays to ensure complete type safety.” [17]

At the moment, Android Studio supports two programming languages, Java and Kotlin. The code of Accountability of Utilities (ACCU) application is fully written in Java language.

**MySQL**

“MySQL (pronounced *My-Ess-Que-Ell*) is a very fast, robust, *relational database management system* (RDBMS). A database enables you to efficiently store, search, sort, and retrieve data. The MySQL server controls access to your data to ensure that multiple users can work with it concurrently to provide fast access to it, and ensure that only authorized users can obtain access. Hence, MySQL is a multi-user, multi-threaded server. It uses SQL (*Structured Query Language*), the standard database query language worldwide. MySQL has been publicly available since 1996, but has a development history going back to 1979. It has now won the *Linux Journal* Reader’s Choice Award on a number of occasions.

MySQL is now available under an Open Source license, but commercial licenses are also available if required.” [18]

In order to create the database for this application I used phpMyAdmin, which is a free and open source tool for MySQL, to store all the data of the application.

**PHP**

“PHP is a server-side scripting language specifically for the Web. Within an HTML page, you can embed PHP code that will be executed each time the page is visited. Your PHP code is interpreted at the Web server and generates HTML or other output that the visitor will see.

PHP was conceived in 1994 and was originally the work of one man, Rasmus Lendorf. It was adopted by other talented people and has gone through three major rewrites to bring us the broad, mature product we can see today. As of October 2002, it was in use on more than nine million domains worldwide, and this number is growing rapidly.

PHP is an open source product. You have to access to the source code. You can use it, alter it, and redistribute it all without charge.

PHP originally stood for *Personal Home Page,* but was changed in line with the GNU recursive naming convention (GNU = Gnu’s Not Unix) and now stands for *PHP Hypertext Preprocessor.*

PHP has many strengths, including the following:

* High performance
* Interfaces to many different database systems
* Built-in libraries for many common Web tasks
* Low cost
* Ease of learning and use
* Portability
* Availability of source code” [18]

In order to create the server-side for this application I used the PHP programming language to make CRUD operations in the database.

**TensorFlow**

“TensorFlow is an open source software library used to implement machine learning and deep learning systems.

Behind these two names are hidden a series of powerful algorithms that share a common challenge: to allow a computer to learn how to automatically recognize complex patterns and make the smartest decisions possible.

Machine learning algorithms are supervised or unsupervised; simplifying as much as possible, we can say that the biggest difference is that in supervised learning the programmer instructs the computer how to do something, whereas in unsupervised learning the computer will learn all by itself.

Deep learning is instead a new area of machine learning research that has been introduced with the objective of moving machine learning closer to artificial intelligence goals. This means that deep learning algorithms try to operate like the human brain.

With the aim of conducting research in these fascinating areas, the Google team developed TensorFlow.” [19]

“The main features include:

* Defining, optimizing and efficiently calculating mathematical expressions involving multi-dimensional arrays (tensors).
* Programming support of deep neural networks and machine learning techniques.
* Transparent use of GPU computing, automating management and optimization of the same memory and the data used. You can write the same code and run it either on CPUs or GPUs. More specifically, TensorFlow will figure out which parts of the computation should be moved to the GPU.
* High Scalability of computation across machines and huge data sets.” [19]

In order to be able to make a suggestion about the consumption for the next month, based on the data that already exists in the database, the application uses an artificial intelligence trained with TensorFlow to give the prediction.

**Volley**

“It proposed by Google in 2013, packaging of the Http asynchronous request API, providing an elegant and robust request queue, with better cache mechanism, allows network access to Android applications easier and more efficient. Volley provides a NetworkImageView class that inherits the ImageView class. It not only provides network communications, but also provides automatically decoding function and image cache. Also, It can process image in batches when diplaying many images on UI at a time. However, it uses a separate image view and image loader objects because its main feature is network communication. This library is difficult to use than others. However, there is an advantage to provide an efficient image downloads.” [20]

In this android mobile application, Volley was used to call methods implemented in the PHP code to make CRUD operations over the database. Volley it is used in the application with a RequestQueue and a StringRequest

**XAMPP**

“Is a small and light Apache distribution containing the most common web development technologies in a single package. Its contents, small size, and portability make it the ideal tool for students developing and testing applications in PHP and MySQL. XAMPP is available as a free download in two specific packages: full and lite.” [21]

In this application, XAMPP tool was used to create a database and run the server written in PHP programming language.

**5.3 Android code structure**

The Android code of the Accountability of Utilities application, is structured in three big packages:

* manifests – contains AndroidManifest.xml file
* java – contains all the Java files of the application
* res – contains all the resources used in the application

**Manifests** package contains only one file of .xml type (AndroidManifest.xml) where is declared the startup screen which would open when the application is accessed and all the created activities, also, in this file are declared all the permissions the application needs, for example the permissions needed by this application are the following:

* permission for opening network sockets (*android.permission.INTERNET*)
* permission for accessing information about networks (android.permission.ACCESS\_NETWORK\_STATE)

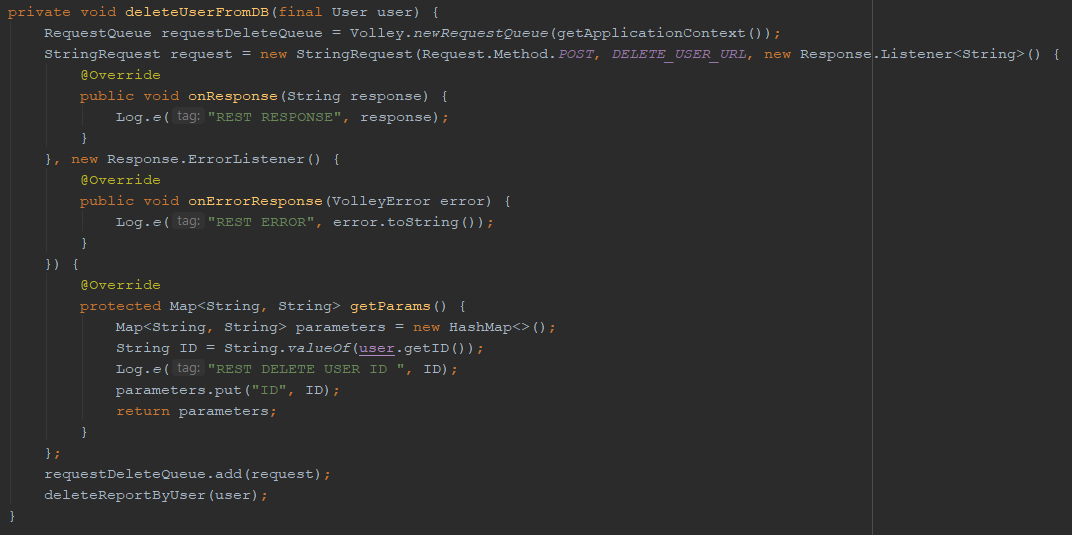
**Java** package contains all the .java files used to create the mobile application. For this project, the java package contains five packages. The packages are the following:

* **Admin package**

Contains two more packages, Activities and Fragments, both of them contains files that are used in the administrator flow of the application.

Activities package contains a java file ActivityAdmin.java that contains methods used to make operations over the database using Volley framework but also contains methods that allows the navigation through the application on different screens. For example, the above method is used to delete a user from the database.

*Delete user from database method*

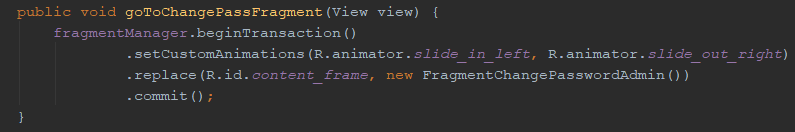


*Figure 5.3.1*

The ‘deleteUserFromDB’ method has as parameter the user that the administrator wants to delete. The request queue creates a new request queue for the current activity and a string request is made specifying in its parameters: the **request method**, in this case it is a POST method, the **URL** over which the request is made is saved in a constant DELETE\_USER\_URL which is equal with "http://" + IP + ":8012/UtilitiesAPI/User/deleteUser.php", the IP variable is of type string which is equal with the IP of the machine where the server runs followed by the port number, UtilitiesAPI/User/deleteUser.php is the path used by XAMPP, of the PHP file where the method that deletes the user can be found, a **response listener** for a valid request which makes a log with the response and a **response error listener** for an invalid request which makes a log with the error that occurred. After those parameters were specified, the method ‘getParams’ of the **StringRequest** class is overridden, in it are send all the needed data in the POST method from the server, in our application is sent only the ID of the user wanted to be deleted. The string request is placed in the request queue in order to make the request.

An example for a method that allows the user to go in other screen of the application would be the navigation to the screen where the administrator can change his account password, the image above represents the code that makes the transition from its current screen to the change password screen:

*Navigate to Change Password Screen method*



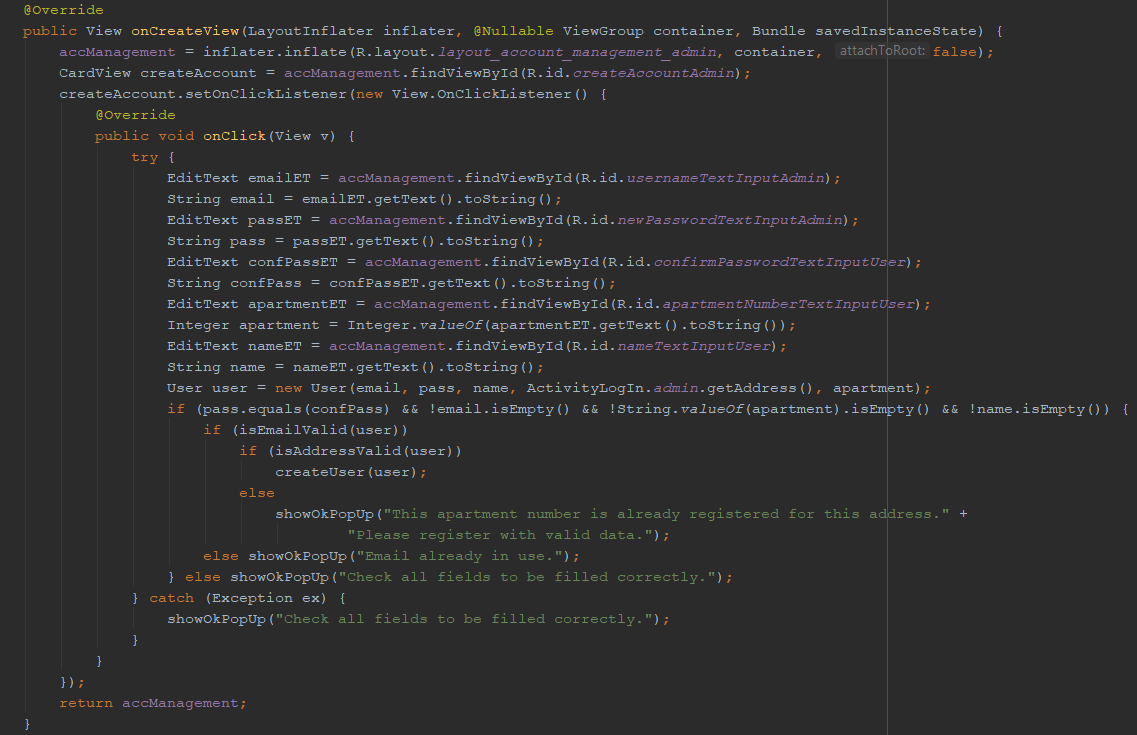
*Figure 5.3.2*

The ‘goToChangePassFragment’ method has as parameter the current view from admin activity. The variable ‘fragmentManager’ it is of type FragmentManager and when the transition between screen is wanted to be made, it starts with ‘beginTransaction’. For this screen transition, is specified using ‘setCustomAnimation’ and the animator will tell how the transition will be, so, the current screen is closing in left side of the screen while the new screen opens from right side of the screen. The application will open the screen specified in the ‘replace’, in this case will be the Change Password screen (class equivalent FragmentChangePasswordAdmin).

The second package from Admin package, is the Fragments package, where are placed all the fragments used in the administrator flows. The following java classes belongs to the Fragments package: FragmentAccountManagementAdmin, FragmentArchiveAdmin, FragmentArchiveMonthsAdmin, FragmentArchiveMontsDetailsAdmin, FragmentChangePasswordAdmin, FragmentChargesAdmin, FragmentCreateEventAdmin, FragmentEditEventAdmin, FragmentEventsManagementAdmin, FragmentEventsReportsAdmin, FragmentEventsReportsDetailsAdmin, FragmentHomeAdmin, FragmentMonthReportsAdmin, FragmentMonthReportsDetailsAdmin, FragmentProfileAdmin and FragmentSettingsAdmin.

As an example for fragments we will choose FragmentAccountManagementAdmin. On this screen the administrator can create a resident account and in this class is made all data validation for the inputs and the request is made if all the inputs are valid.

*Android onCreateView method*



*Figure 5.3.3*

The method ‘onCreateView’ is one of the needed methods in a fragment besides ‘onCreate’. In this method it is made the set up of the screen and its “listens” for a touch over the button with id “createAccountAdmin”. When the administrator presses that button, all input data from the fields are saved as strings. If all the inputs are valid, not empty, the email is not already in use and the address is a valid one, the request for account creation is sent to the server. If there are errors in the validation, properly dialogs are displayed to inform the administrator that something went wrong.

* **AppUtils package**

Contains .java files with methods that are frequently used in the application so there will not be duplicated code.

* **Common** **package**

Contains activities and fragments that are common for the administrator and also for the resident flows.

* **Models** **package**

Contains all the objects that were necessarily in order to be able to implement the mobile application

* **Resident** **package**

Contains two more packages Activities and Fragments, both of them contains files that are used in the resident flow of the application

**Res** package is a resources directory and it contains seven packages:

* anim – contains .xml files that are designed to add animation to the layouts of the activities.
* animator – contains .xml files that are designed to add animations to the layouts of the fragments.
* drawable – contains .xml files but also .png files. The .xml files are icons imported from android studio from Image and Vector Assets.
* layout – contains .xml files that are used in the design of the application.
* menu – contains .xml files that are used in the design of a left menu for the application.
* mipmap – contains .xml files that are used in the design of the icon launcher of the application
* values – contains .xml files where are declared the colors, hardcoded strings, dimensions, background of the icon launcher, styles used in the application

# **7. Bibligraphy**

[1] Lee, Wei-Meng. *Beginning android 4 application Development*. John Wiley & Sons, 2012.

[2] Unwana Johnson, *Android versions, their names and their evolution through the years, Tech Informat*, 29 July 2018*.*

[3] We Are Social, Mobile’s share of total internet time.

[4] Siraj Raval, A Guide to Running TensorFlow Models on Android.

[5] Amit Shekhar, Android TensorFlow Machine Learning Example.

[6] Shulin, Yang, and Hu Jieping. "Research and implementation of Web Services in Android network communication framework Volley." *2014 11th International Conference on Service Systems and Service Management (ICSSSM)*. IEEE, 2014.

[7] Olson, Gene H. "Network terminal server with full API implementation." U.S. Patent No. 6,047,319. 4 Apr. 2000.

[8] GONG, Chengying, Jinghong XING, and Yinbao HU. "Data communication of Android mobile terminal and PHP and MySQL based on JSON [J]." *Industrial Instrumentation & Automation* 1 (2013): 021.

[9] Dobjanschi, Virgil. "Developing Android REST client applications." *Google I/O 2010* (2010).

[10] Android official website.

[11]Statista 2018, Number of mobile phone users worldwide from 2013 to 2019.

[12] Android, *Android Studio.*

[13] Bloc-Admin official website

[14] Microsoft, *Microsoft Visual Code*

[15]Peterson, Robert A., and Maria C. Merino. "Consumer information search behavior and the Internet." *Psychology & Marketing* 20.2 (2003): 99-121.

[16] Van Drongelen, Mike. *Android studio cookbook*. Packt Publishing Ltd, 2015.

[17] Gosling, James, et al. *The Java language specification*. Addison-Wesley Professional, 2000.

[18] Welling, Luke, and Laura Thomson. *PHP and MySQL Web development*. Sams Publishing, 2003.

[19] Zaccone, Giancarlo. *Getting Started with TensorFlow*. Packt Publishing Ltd, 2016.

[20] SONG, Yoo-jeong, Soo-bin OU, and Jong-woo LEE. "An Analysis of Existing Android Image Loading Libraries: Picasso, Glide, Fresco, AUIL and Volley." *DEStech Transactions on Engineering and Technology Research* imeia (2016).

[21] Dvorski, Dalibor D. "Installing, configuring, and developing with Xampp." *Skills Canada* (2007).