**Software Requirements**

**Specification**

**for**

**<MAI Bank>**

***Copyright © 1999 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document.***

# Table of Contents

**Introduction 1**

**Purpose 1**

**Document Conventions 1**

**Intended Audience and Reading Suggestions 2**

**Product Scope 2**

**References 2**

**Overall Description 3**

**Product Perspective 3**

**Product Functions 3**

**User Classes and Characteristics 3**

**Operating Environment 4**

**Design and Implementation Constraints 4**

**User Documentation 4**

**Assumptions and Dependencies 4**

**External Interface Requirements 5**

**User Interfaces 5**

**Hardware Interfaces 6**

**Software Interfaces 6**

**Communications Interfaces 6**

**System Features 7**

**The Login 7**

**The Register 7**

**The Contact 8**

**The Home 8**

**The Accounts 8**

**The Payments 9**

**The Info 10**

**The Logout 10**

**The Graphics 10**

**Other Nonfunctional Requirements 12**

**Performance Requirements 12**

**Safety Requirements 12**

**Security Requirements 12**

**Software Quality Attributes 12**

**Business Rules 14**

**Other Requirements 15**

# Introduction

## Purpose

This document should give a perspective of the desktop application. It will present the interface, the constraints of the system and the purpose for developing the system. This document is primarily intended to be proposed to a customer for its approval and a reference for developing the first version of the system for the development team.

## Document Conventions

|  |  |
| --- | --- |
| **Term** | **Definition** |
| **UI** | user interface |
| **QA** | quality assurance |
| **JDBC** | java database connection [1] |
| **user** | someone who interacts with the mobile application |
| **CRUD** | create, read, update, delete |
| **SRS** | software requirements specification [2] |
| **TAG** | a unique, persistent identifier contained in a PLanguage statement [3] |
| **GIST** | a short, simple description of the concept contained in a PLanguage statement [3] |
| **SCALE** | the scale of measure used by the requirement contained in a PLanguage statement [3] |
| **METER** | the process or device used to establish location on a SCALE contained in a PLanguage statement [3] |
| **MUST** | the minimum level required to avoid failure contained in a PLanguage statement [3] |
| **PLAN** | the level at which good success can be claimed contained in a PLanguage statement [3] |
| **WISH** | a desirable level of achievement that may not be attainable through available means contained in a PLanguage statement [3] |
| **TITLE** | title |
| **DESC** | description |
| **RAT** | rational |
| **DEP** | dependency |

* 1. **Intended Audience and Reading Suggestions**

The document will have the development team to make more clear the task for them and it will also be useful for the QA [4] team when testing the functionality of the application. It will also represent a high end perspective for the client and make the development path to be more clear. The users may also read this SRS document in order to be more familiar with the application.

## Product Scope

This product will automate the banking transaction process.As the article [5] shows soon everything will become automated. Considering that nowadays everyone has a phone at hand, we choose our app to be one for Mobile. With this product making transfers to other accounts is just a click away. The user can both view the transactions made [6] and information about their account. Even opening a new account is easier with this new application.

We know how hard it is to get to the bank and how long it takes, so we try to make customer’s lives easier.

## References

1. <https://docs.oracle.com/javase/tutorial/jdbc/basics/index.html>
2. Software Requirements Speciﬁcation - Amazing Lunch Indicator (PDF) Sarah Geagea 881024-4940 ; Sheng Zhang 850820-4735 ; Niclas Sahlin 880314-5658 etc
3. Feldt R,”re\_lecture5b\_100914”, unpublished.
4. <https://www.guru99.com/all-about-quality-assurance.html>
5. The future of work: will everything become “automatable”? - Gautam Jaggi ( Nov, 2 2017)
6. <https://www.freeagent.com/glossary/bank-transaction/>

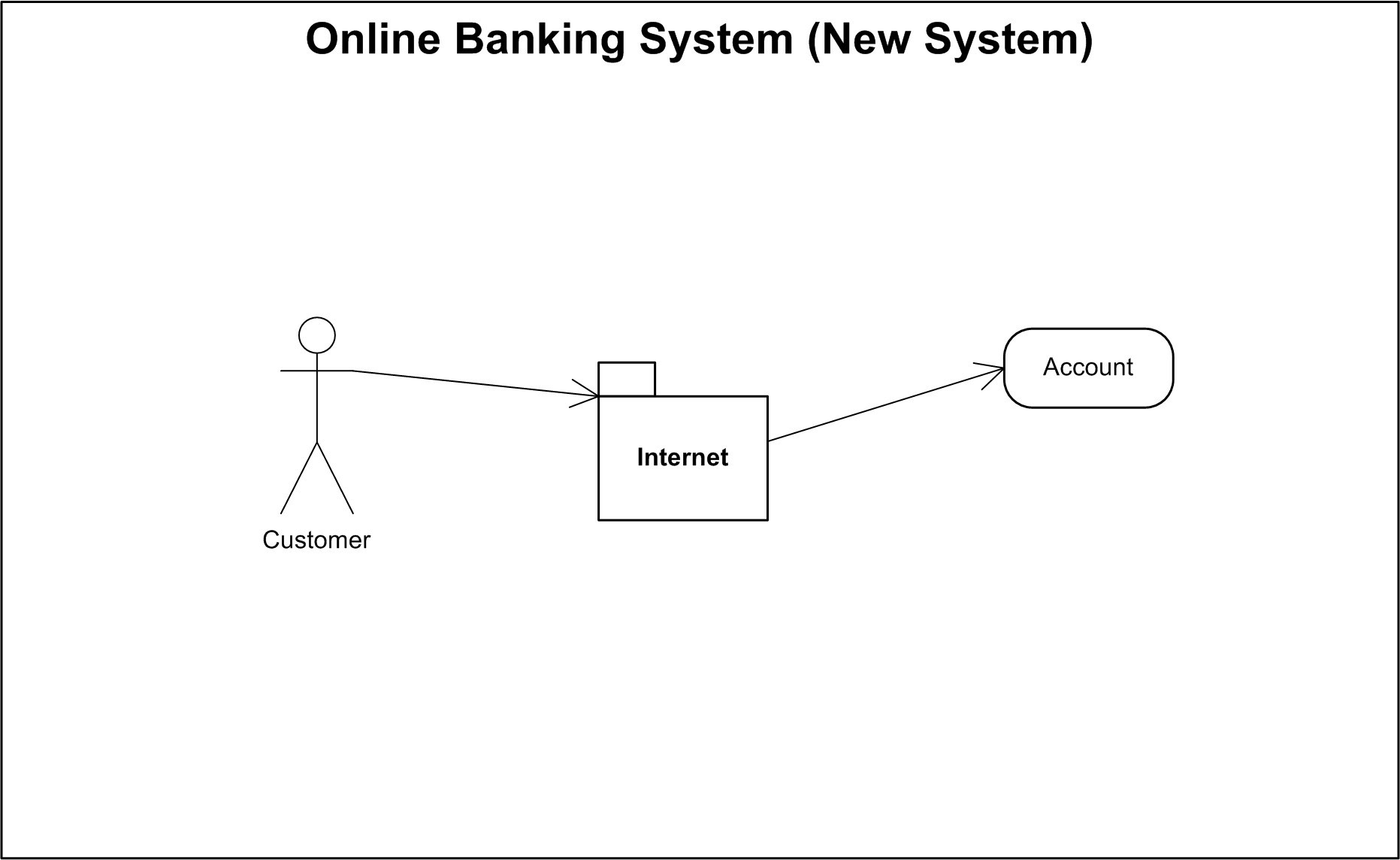
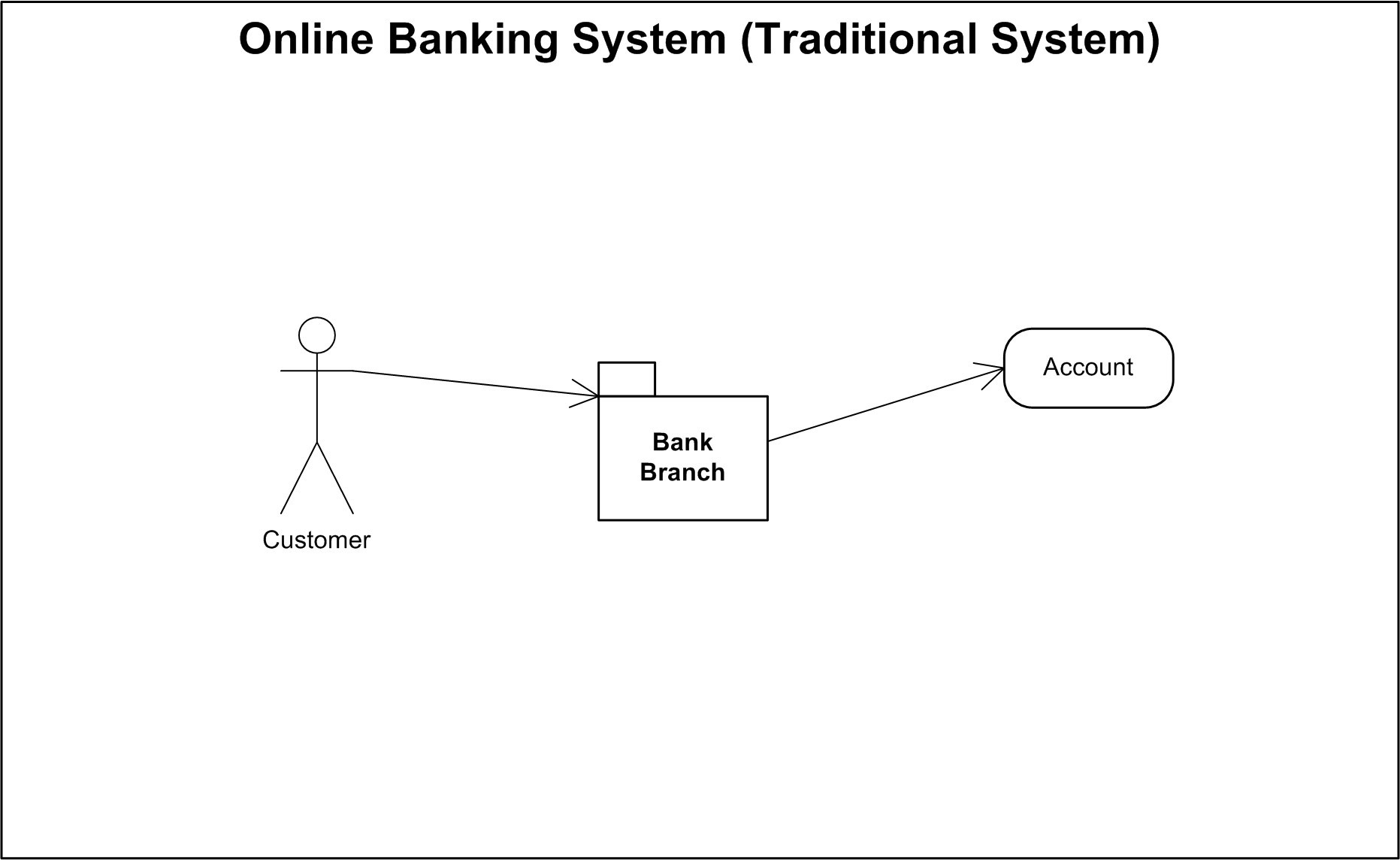
# Overall Description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

## Product Perspective

SRS defines directly the product,there is no large family to belong to. The client does not need other helpful applications, it is enough to have this.

In the traditional system, customers should have to visit the Bank branch physically for the transactions or some other task. It wastes time. After implementing the online banking system, customers will be able to connect to this account through the internet only with a user and a password. The use of time will be minimized, the task will be done quickly instead of waiting for someone else to complete the task.



## Product Functions

The main functionality of our application are:

* + - User can modify her/his address
    - User can change their password
    - Our customers will be able to pay the bills from their account to various companies (Electrica, Rds-Rcs, etc)
    - The possibility to see all the accounts for the current user
    - The possibility to create a new account (we have 3 types of accounts: EUR, USD and RON)
    - Customers will be able to see the list of recent transaction

## User Classes and Characteristics

The only type of users that interact with the system are the clients of the bank.

The mobile application users can only use the application to manage their accounts. This means that the user has to be able to transfer money, to make payments, to open another account, to view

transactions history, information about her accounts and also, edit the information about his/her person.

## Operating Environment

Our application will run on all android devices with version above API 23 (Android 6.0 Marshmallow) with an internet connection.

## Design and Implementation Constraints

The application is developed with the help of Android Studio (version 4.1.2), SqlLite database API, JDBC, MicrosoftSqlServerManagementStudio 18 and Eclipse IDE. We use Java as a programming language and for testing our application we utilize JUnit 4 framework.

## User Documentation

To help users to work with our application we make a help page where we give some useful information. Also, when we will deliver the product, we will give the client this documentation about the application.

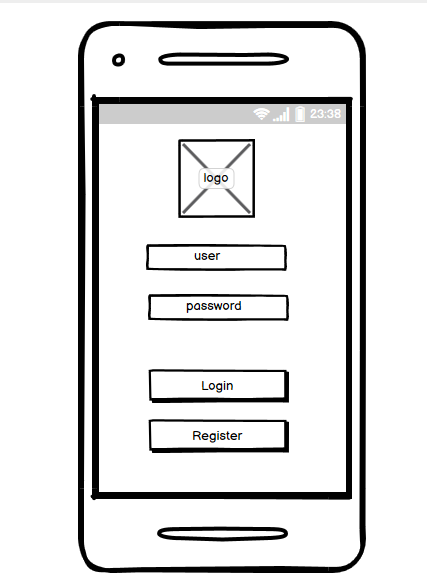
## Assumptions and Dependencies

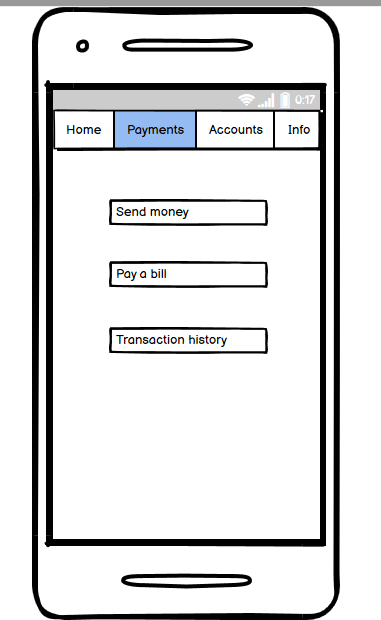
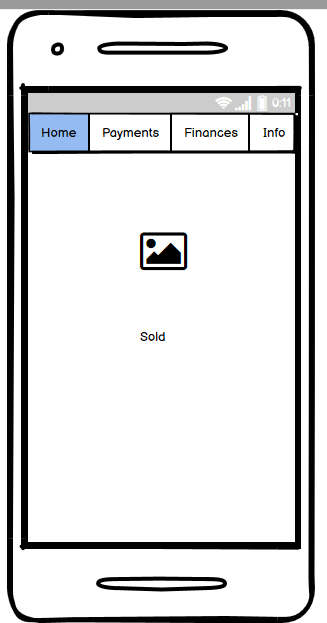
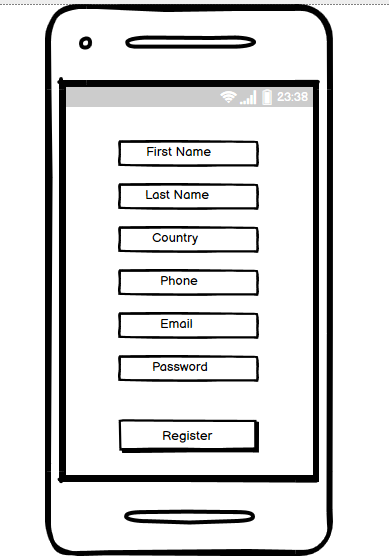
The client should have a good connection (min requirement 3G approx 0,5 Mbps) to the internet, a telephone with android version 6 or higher.

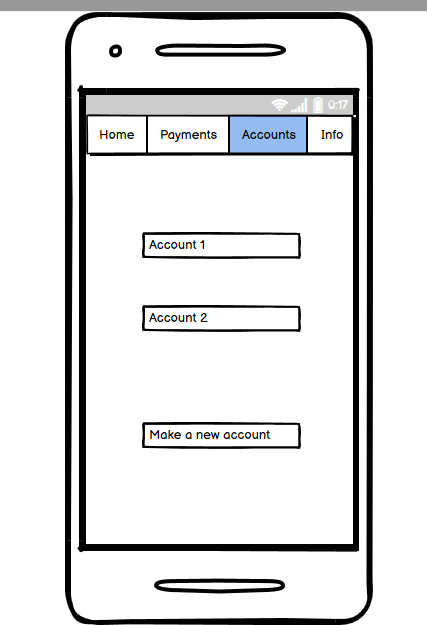
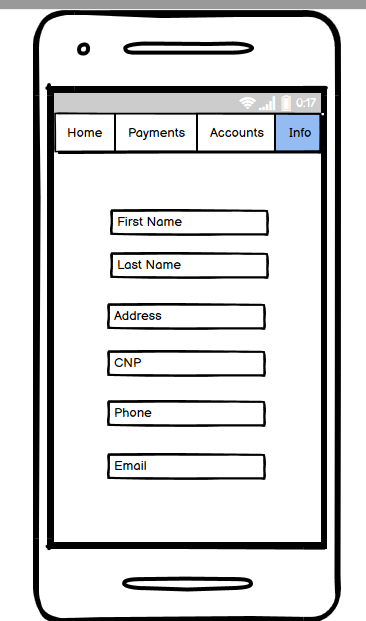
# External Interface Requirements

## User Interfaces

First of all we have a mock-up to Login page:



If the user or the password is incorrect a toast will appear. Also, if the login fails, a message to the user will be shown. Pressing the “Register” button the corresponding page for this action will be opened

## Hardware Interfaces

*Not applicable*

## Software Interfaces

We develop an API to communicate with the database which will respect the CRUD operation: create, read, update and delete.

## Communications Interfaces

*Not applicable*

# System Features

## The Login

* + 1. *Description and Priority*

To use this application, the client must be logged in, using credentials. This feature has a high priority.

* + 1. *Stimulus/Response Sequences*

To login, it is required to introduce a user and a password. After that, the client will be able to use the functionality of the application.

* + 1. *Functional Requirements*

REQ-1: User must insert the username and the password. REQ-2: Push the button “Login”

REQ-3: Validation of the credentials. We search in the database to verify if the client has an account

REQ-4: If the client has found, the user will enter in the application Home Page REQ-5: If the username is incorrect, we display a warning message.

REQ-6: If the password is incorrect, also we display a warning message.

## The Register

* + 1. *Description and Priority*

Like I said above, to use our application, the client must have an account.If he/she does not have an account, they have the possibility to register. This feature also has a high priority.

* + 1. *Stimulus/Response Sequences*

To register, it is required to complete the fields with the required data. After that, the client will be able to use the functionality of the application.

* + 1. *Functional Requirements*

REQ-7: Users must insert the required data. REQ-8: Push the button “Register”

REQ-9: Validation of the inserted data. We verify if all the required field are not empty and correspond with the standards

REQ-10: If the client insert a valid data, the sign up process will be completed and the user will enter in the application Home Page

REQ-11: If any fields are not completed, we display a specific warning message.

REQ-12: If the client does not know how to complete the fields or has some questions, the user has the option to contact us.

## The Contact

* + 1. *Description and Priority*

From the desire to help our clients, we create a Contact Page. For this feature, the priority is low.

* + 1. *Stimulus/Response Sequences*

On the Contact Page, users can find our addresses, a telephone number and an email for support.

* + 1. *Functional Requirements*

REQ-13: Clients have 2 possibilities to contact us: via email or they can call

us..

REQ-14: If they press to our email address, they will be redirected to an

application where they can write us an email.

REQ-15: If they press to our telephone number, they will be redirected to Contact Phone on their mobile.

## The Home

* + 1. *Description and Priority*

On this page, the users can view their current sold. Priority will be medium.

* + 1. *Stimulus/Response Sequences*

The above page is a navigation bar, where we offer some functionality to our clients, like make a transfer, pay a bill or update their personal information.

* + 1. *Functional Requirements*

REQ-16: Clients can view their current sold REQ-17: They can navigate to other pages

REQ-18: If they press to Accounts Page they will be redirected here REQ-19: If they press to Payments Page they will be redirected here REQ-20: If they press to Info Page they will be redirected here

## The Accounts

* + 1. *Description and Priority*

This page offers users control over their accounts. The priority is medium.

* + 1. *Stimulus/Response Sequences*

The clients can view all their accounts or have the possibility to make another account.

* + 1. *Functional Requirements*

REQ-21: Clients can manage and view their accounts

REQ-22: If the client should want to make other account, they can press to the button “Make a new account”

REQ-23: They need to complete the field with the required data.

REQ-24: That data will be verified. We verify in the database if the account already exists.

REQ-25: If the accounts already exist, we display a warning message.

REQ-26: If the verification was successfully done, we display a fit message and add this account to the client.

## The Payments

* + 1. *Description and Priority*

On this page, the clients can find all they need to make a payment, view a list of transactions or send money. This section has a high priority.

* + 1. *Stimulus/Response Sequences*

To pay a bill, send money or view a list of all the transactions, the clients just click on the corresponding option.

* + 1. *Functional Requirements*

REQ-27: Users have 3 main options: pay a bill, send money and view transaction history.

REQ-28: If they press to “Pay a bill” option, they will be redirected to specific page, where they must select a service company (Electrica, Rds-Rcs, etc) and complete some fields

REQ-29: We check if the client has enough money in their account

REQ-30: If validation was successful, then we display a notify message. REQ-31: If validation has failed, then we display a warning message.

REQ-32: If they press the “Send Money” option, they will be redirected to a specific page, where they must complete some field with specific data.

REQ-33: We check if the destination account is correct.

REQ-34: If the destination account is correct, we check if the client has enough money in their account

REQ-35: If the sold was insufficient, we display a warning message

REQ-36: If the destination account does not exist, we display an error message.

REQ-37: If all the validation steps are successfully complete, we will approve the transfer and update the solds.

REQ-38: If they press the “Transaction List” option, they will be redirected to a specific page, where they can view a list of all their transactions.

REQ-39: Also here is an option to download the list of the tranzactions

## The Info

* + 1. *Description and Priority*

This page contains a list of the personal information about the client. The priority is low, because these data are about the client.

* + 1. *Stimulus/Response Sequences*

To view their personal information, the clients should press to the ‘’Info” from the Navigate Bar.

* + 1. *Functional Requirements*

REQ-40: Clients should view information about their person. REQ-41: They can edit their data, pressing the “Edit” option. REQ-42: The clients make the changes.

REQ-43: We verify any field if the data changed respect a correct pattern. (eg if the CNP number has 13 digits)

REQ-44: If a field does not respect the pattern, we display a warning message. REQ-45: If all the fields are correctly completed, we display a notification.

message and update this data into the database.

## The Logout

* + 1. *Description and Priority*

This option logs out a client. The priority for this feature is a low one.

* + 1. *Stimulus/Response Sequences*

To log out, a client just should click the “Logout” option

* + 1. *Functional Requirements*

REQ-46: Clients should press the “Logout” button. REQ-47: They will be redirected to the Login Page.

## The Graphics

* + 1. *Description and Priority*

We want to help you to reach your goals. In order to do this, we add a graphic on the Home Page. The priority of this feature is low, because this section is an informative one. The application can exist and function correctly and without it.

* + 1. *Stimulus/Response Sequences*

Under the graphic we add an “Edit” button. Pressing this button, you will be able to set a minimum and maximum value of your sold. Then on the Home Page we will display where actually are on the graphic based on your current sold. Here you also have the possibility to share your account information.

* + 1. *Functional Requirements*

REQ-48: Users should press the “Edit” button.

REQ-49: Then they must insert the minimum and the maximum value. REQ-50: We check if the minimum value is less than maximum value.

REQ-51: If the 2 sums are correct, we display a specific message, else we display an error message.

REQ-52: If users press the “Share” button, they can redirect their information account.

# Other Nonfunctional Requirements

## Performance Requirements

If the user happens to have a bad connection (speed under 0,5 Mbps), the functionality may take a longer time, maximum 30 seconds. In the unfortunate case, where there is no signal at all on the internet, the application will not start. It should load as fast as possible. The application layout changes depending on the screen size and supports displays of different sizes.

## Safety Requirements

*Not applicable*

## Security Requirements

The access will be provided using an user and a password. Also, the password will be encrypted using MD5, which is stored in the database.

## Software Quality Attributes

### Reliability

* 1. TAG: SystemReliability
  2. GIST: The reliability of the system.
  3. SCALE: The reliability that the system gives the right result on a search.
  4. METER: Measurements obtained from 1000 searches during testing.
  5. MUST: More than 98% of the searches.
  6. PLAN: More than 99% of the searches.
  7. WISH: 100% of the searches.

### Availability

* 1. TAG: SystemAvailability
  2. GIST: The availability of the system when it is used.
  3. SCALE: The average system availability (not considering network failing).
  4. METER: Measurements obtained from 1000 hours of usage during testing.
  5. MUST: More than 98% of the time.
  6. PLAN: More than 99% of the time.
  7. WISH: 100% of the time.
  8. TITLE: Internet Connection
  9. DESC: The application should be connected to the Internet.
  10. RAT: In order for the application to communicate with the database.
  11. DEP: none

### Security

* 1. TAG: CommunicationSecurity
  2. GIST: Security of the communication between the system and server.
  3. SCALE: The messages should be encrypted for log-in communications, so others cannot get user-name and password from those messages.
  4. METER: Attempts to get user-name and password through obtained messages on 1000 log-in sessions during testing.
  5. MUST: 100% of the Communication Messages in the communication of a log-in session should be encrypted.
  6. TAG: UserCreateAccountSecurity
  7. GIST: The security of creating accounts for users of the system.
  8. SCALE: If a user wants to create an account and the desired username is occupied, the user should be asked to choose a different user name.
  9. METER: Measurements obtained on 1000 hours of usage during testing.
  10. MUST: 100% of the time.

### 4 Maintainability

* 1. TITLE: Application extensibility
  2. DESC: The application should be easy to extend. The code should be written in a way that it favors implementation of new functions.
  3. RAT: In order for future functions to be implemented easily to the application.
  4. DEP: none

### Testability

* 1. TITLE: Application testability
  2. DESC: Test environments should be built for the application to allow testing of the applications' different functions.
  3. RAT: In order to test the application.
  4. DEP: none

### Reusability

* 1. TITLE: Code Reusability
  2. DESC: We should be able to reuse code.
  3. RAT: In order to reduce duplicate code and maintenance of the application
  4. MUST: 50%
  5. PLAN: 70%

6.6 WISH: 100%

### Modifiability

* 1. TITLE: Easy replacement of the code logic
  2. DESC: Determines how many common changes need to be made to the system to make changes to each individual item.
  3. RAT: for flexibility and maintenance
  4. DEP: none

### Performance

* 1. TITLE: Application Performance
  2. DESC: Shows the response of the system to performing certain actions for a certain period of time.
  3. RAT: In order to avoid unresponsive application and for the responsiveness of the system
  4. DEP: none

### Interoperability

* 1. TITLE: System Interoperability
  2. DESC: Consist in the responsibility for its operation and the transmission of data and its exchange with other external systems.
  3. RAT: Through create a server for database

### Correctness

* 1. TITLE: System Correctness
  2. DESC: Is asserted when it is said that the functionality is correct with respect to a desired behaviour

10.3 MUST: 100%

10.4 PLAN: 100%

10.5 WISH: 100%

## 5.5 Business Rules

*Not applicable*

# 6. Other Requirements Appendix A: Glossary Appendix B: Analysis Models

*Not applicable*

# Appendix C: To Be Determined List

*Not applicable*