Roger That Project

Software Engineering

Architecture Document

Denis Garabajiu *(S4142551)*
Constantin Cainarean *(S4142152)*
Valeria Mavceanscaia *(S3673952)*
Gasan Rzaev *(S3553213)*

Contents

1 Introduction

An actual economical and social problem in the Netherlands is a relatively big number of people, especially young adults (18 - 35 y.o) with debts or bad financial situations. A lot of people do not even know how they are doing at the moment. **RogerThat Project** is a tool for organizing financial information of an individual and providing feedback on the user's financial overview based on the user's transactions.
We decided to split our program into 3 different parts: an online interface, a back end and a data base.
In this document we will describe how our platform works.

2 General Overview of the Architecture

The Front end is the interface the user will use to access his data, upload his files with the transactions, transform the report into PDF format etc.
The Back end is the part where all the checks and operations related to the functionality of the Web App will be processed and executed.
The DB will be used for storing important data that is necessary for our platform.

3 Front End

All calculations and needed information will be retrieved from the back end, meanwhile the front end is just a user-friendly interface for easier usage of our platform. It communicates with the back end via http requests and responses. It is divided into several parts which will be discussed in details in the sections below. Our front end is based on Angular Framework with NG-Zorro library.

3.1 Login and Register

The first thing that the user sees when he enters on the platform is the form that will be used to login or register to the platform. All credentials will be encrypted and sent via an http request to the back end. If the login/register action was performed with success, the user will be redirected to the main panel of platform, else an error message will appear describing what went wrong.

3.2 Main Panel

Main Panel is the working space of the user and it consists of various sections, in each of them the user will be able to perform different actions. The panel is a folding slide menu, which contains all the functionality of the interface.

3.2.1 My Profile Section

Here the user will be able to view own personal information. The information displayed is the one that is kept in the database, which was added after the user has registered to the app and has input the information in the register form.

3.2.2 Settings

The settings section is meant to improve the usability of our web application. On this page the user is able to change the current e-mail that is registered to the account to a new one. Also the user has the possibility to enter a new password that will be replaced with the old one for the authentication process. To confirm the choice of the new password, it should be entered twice, and in both fields the passwords should be the same. In this section the user is also able to bind a new phone number to the account, the number should be entered with the plus sign(+),the country-code and the actual number itself.

3.2.3 Status Section

This section is the most important part of our front end in terms of app functionality. The user will upload a CSV file containing his transactions. This file will be sent as a byte stream to the back end and will be stored in the root of the program. The following operations done on the uploaded file will be discussed in the Back end section. In this section the user will be able to see his short current financial status (a bar from red to green) and will be able to download a PDF file containing a detailed overview of his financial situation.

4 Back End

As mentioned previously all the computation and data processing is done in the back end of the application. The computations and data processing are divided into smaller parts of the back end. Additionally, there are several endpoints which are there for the purpose of communication with the front end. The following parts will be discussed in the subsections below.

4.1 CSV Parser

CSV Parser is the part of the application that accepts CSV files from the endpoint, parses the file and persists it to the database. CSV Parser accepts the line of CSV file, converts it into the token list. Afterwards, the token list's elements are passed on to the Transaction object created specifically for the Database, with its attributes being equivalent and in right order to the columns of the database table for each transaction. Finally, every transaction is being persisted, using Panache ORM method $persist()$ to the database right after passing the values of the line are read.

4.2 Database

MySQL is used as a service for the database development and manipulation. And the main purpose of it is to store all the bank transactions of all users. However, apart from transactions the database will hold a purpose of holding the rest of the information about users. For instance, user's personal information, log-in information is going to be stored in different tables. In future, database will also hold information about the transactions that were generated by analyser, such as spending and income classification.

4.3 Data analyser

Data analyser is one of the vital parts of the back end, since its purpose would fulfill the main feature of this project. One of the first and simplest tasks that it solves is the check of income and spending recurrence. The check will mainly test the ratio between two in the past $1 - 3$ months. Moreover, the classification of each transaction is going to be introduced to make it easier for the PDF generator to sum the financial overview up.

4.4 PDF generator

For later iteration(s).

5 Technology Stack

Due to the fact that this project involves a web application, a full functional back end and a DB for storing data we have a vast stack of technologies. As mentioned above for front end we use Angular + ng-zorro library. We chose this combination of technology because it gave us the possibility to develop a qualitative web application without any design skills. Also because it uses typescript language it was fast in development. For back end we used Java with Quarkus. This stack was chosen because we were all familiar with Java and it is a language that fits perfectly for this kind of requirements. Quarkus was chosen because it has a lot of functionality required for our project, for example Panache ORM(a tool that makes the work with the DB much easier). For the Data Base we chose MySQL because it ensures the functionality we need for the project, and every member of our team is familiar with it.

6 Design of the project

6.1 Front End Design decisions

As mentioned above, we have built the front end of our application using the Angular framework, this decision was reasoned by the previously accumulated knowledge in operating with it of some members of our team, also this framework is component-based, which ensures a higher quality of code, and the TypeScript language used in it is very simple, clean, and scalable. One of the strong points of this framework was that it's detailed documentation is easily accessible, allowing us to operate with framework, and in the future, the code can be easily modified/maintained the Roger That team. This section will be described more detailed in our future iterations, when the project will be in a finished state.

6.2 Back End Design decisions

Behind the "scenes", on the back end side of our application, we are using Java Framework Quarkus - which acts as a compiler, that helps us to run our project. This framework is working in combination with ORM Panache which is an utility that helps us to connect and maintain the connection between the Data Base and the Java code, which in the context of our project makes the operations on user's transactions much easier. Quarkus has a lot of libraries that will later on be used by us on connecting the front end with the back end using end-points. This section will be described more detailed in our future iterations, when the project will be in a finished state.

7 Sequence Diagrams

Figure 1: Graphical SSD for Logging in

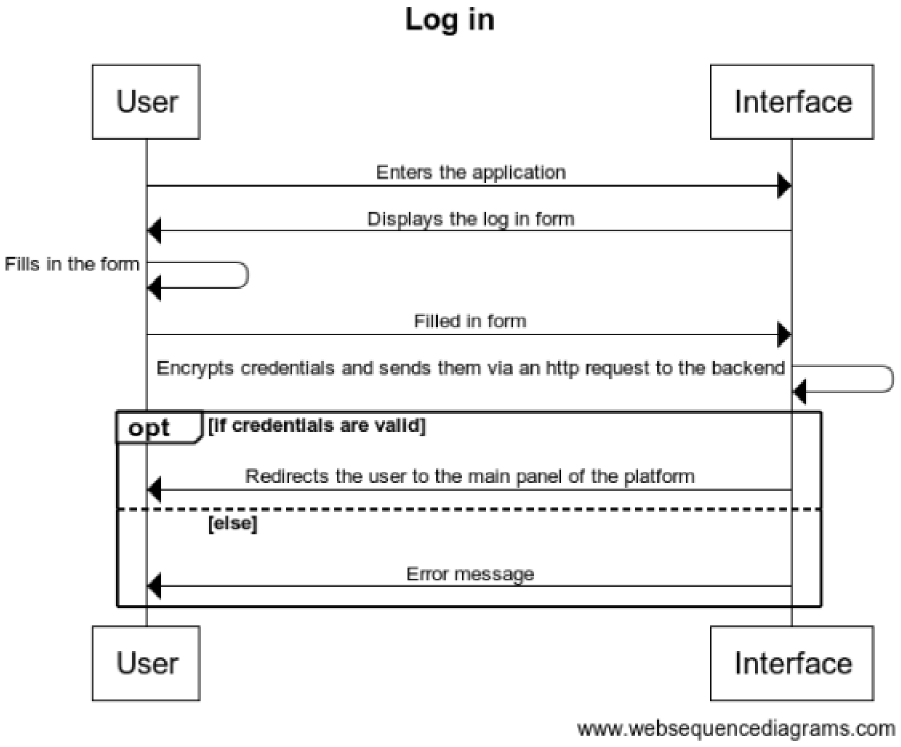

Log in
User
Interface
Enters the application
Displays the log in form
Fills in the form
Filled in form
Encrypts credentials and sends them via an http request to the backend
opt
[If credentials are valid]
Redirects the user to the main panel of the platform
[else]
Error message
User
Interface
www.websequencediagrams.com

Figure 2: Graphical SSD for File upload

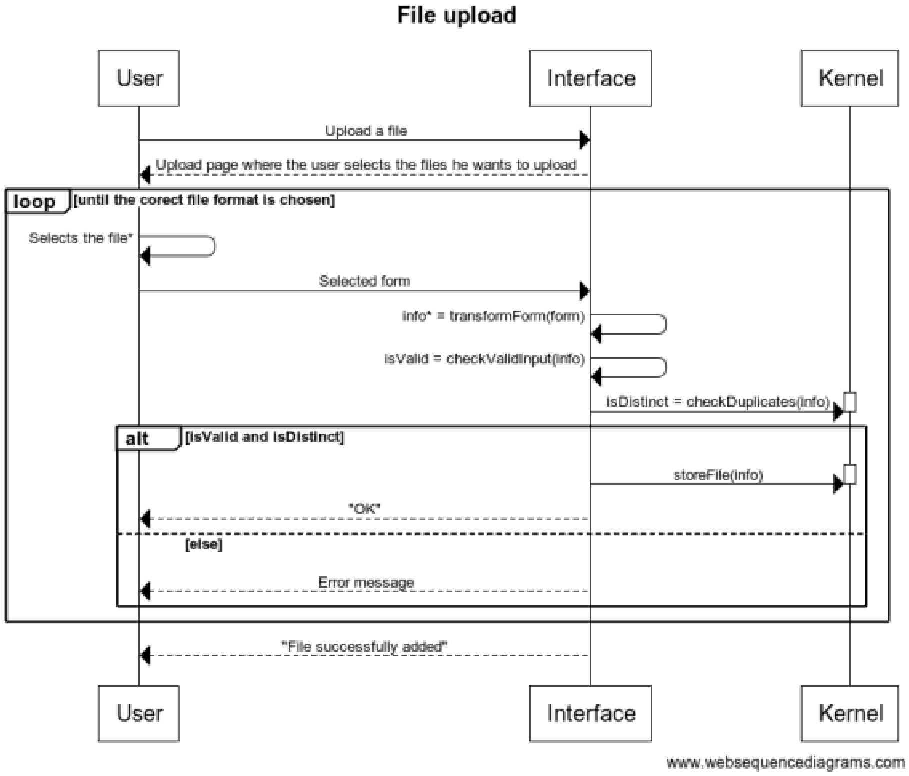
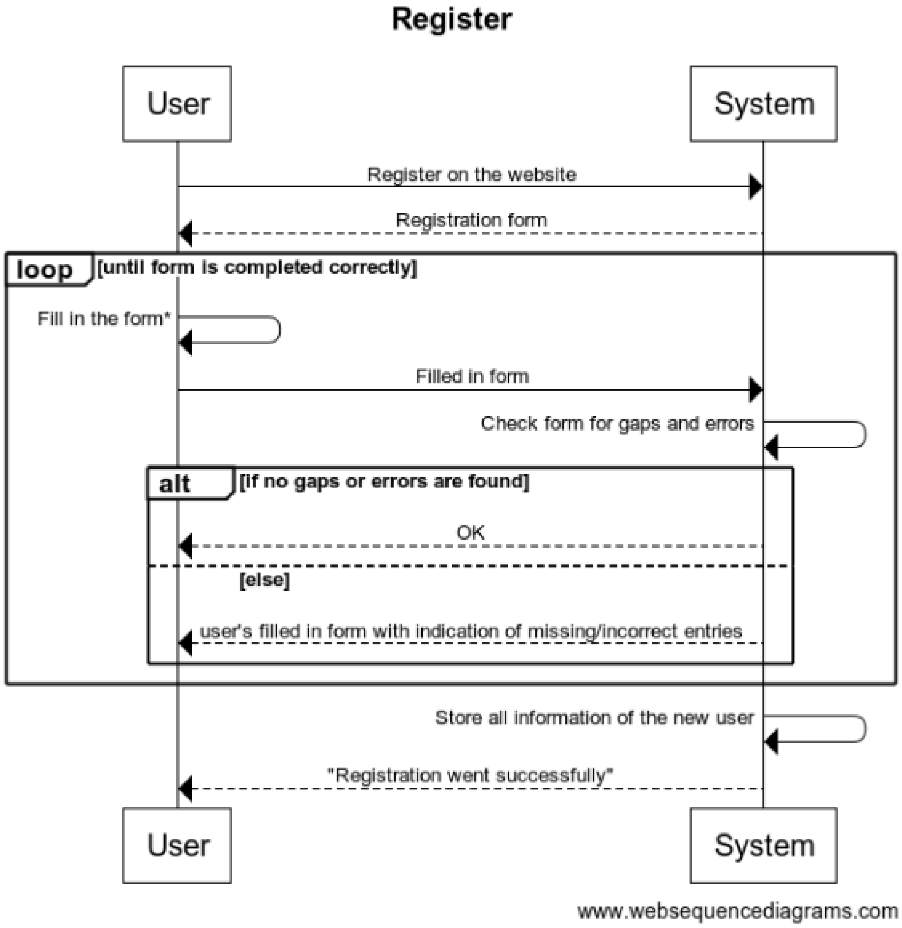

File upload
User
Interface
Kernel
Upload a file
Upload page where the user selects the files he wants to upload
loop [until the corect file format is chosen]
Selects the file*
Selected form
info* = transformForm(form)
isValid = checkValidInput(info)
isDistinct = checkDuplicates(info)
alt [isValid and isDistinct]
storeFile(info)
"OK"
[else]
Error message
"File successfully added"
User
Interface
Kernel
www.websequencediagrams.com

Figure 3: Graphical SSD for Register


Register
User
System
Register on the website
Registration form
loop [until form is completed correctly]
Fill in the form*
Filled in form
Check form for gaps and errors
alt [if no gaps or errors are found]
OK
[else]
user's filled in form with indication of missing/incorrect entries
Store all information of the new user
"Registration went successfully"
User
System
www.websequencediagrams.com

8 Change Log

Date	Name	Changes
25.04	Cainarean Constantin	Introduction Part
27.04	Cainarean Constantin	General Overview, Front end, Login and Register
28.04	Valeria Mavceanscaia	Login diagram
28.04	Valeria Mavceanscaia	Register diagram
28.04	Gasan Rzaev	Backend, CSV Parser
29.04	Gasan Rzaev	Database, Data analyser
01.05	Cainarean Constantin	Main Panel, My Profile Section, Status Section
01.05	Denis Garabajiu	Settings section
03.05	Cainarean Constantin	Technology Stack
03.05	Valeria Mavceanscaia	File upload diagram
03.05	Valeria Mavceanscaia	General corrections/additions to the document
03.05	Denis Garabajiu	Improvements on previously added sections
03.05	Denis Garabajiu	Design section with the decisions on front and back end