Disclaimer

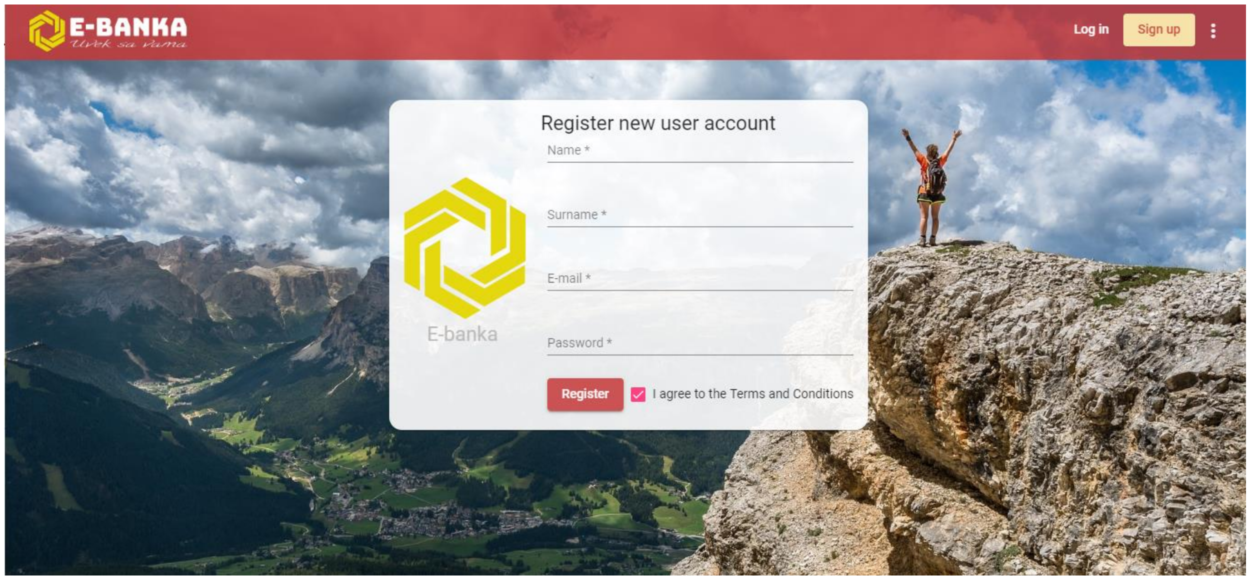
This project and its source code should be used for educational purposes only. A fictional names "Ebank", "E-bank", "Ebanka", "E-banka" as far as fictional bank logos have been used for illustration purpose only.

Ebank-Web-App

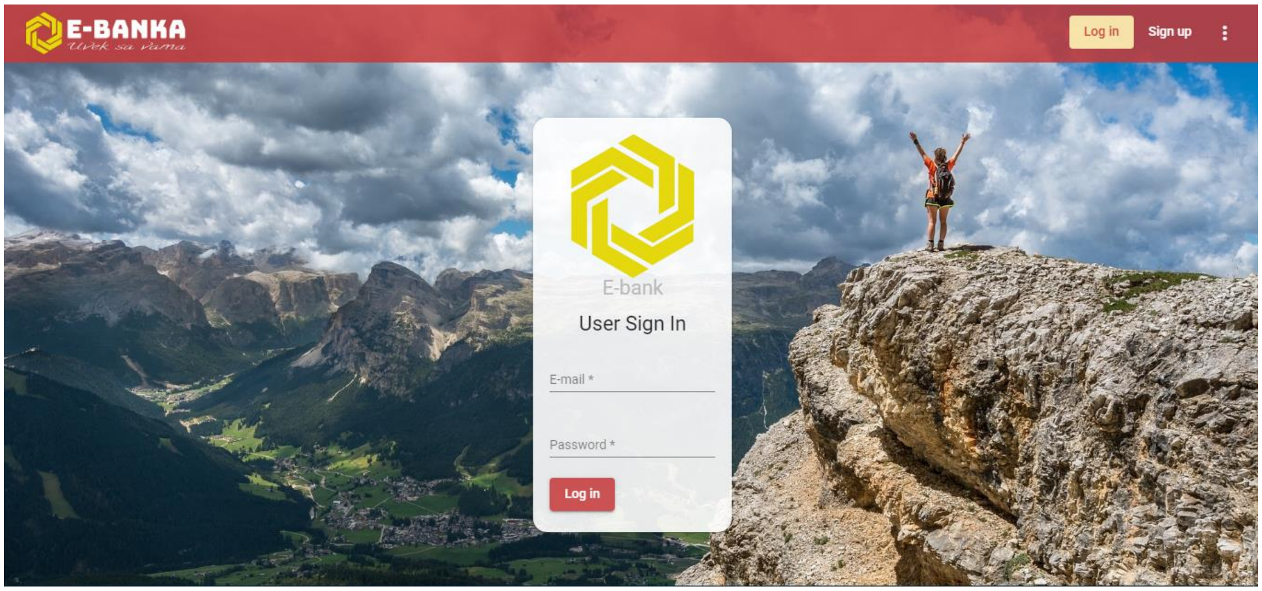
This is a web application for online banking with all essential features. It allows registered users to manage their bank accounts, transfer funds, get a list of all past transactions, as well as to pay their utility bills.

User interface pictures

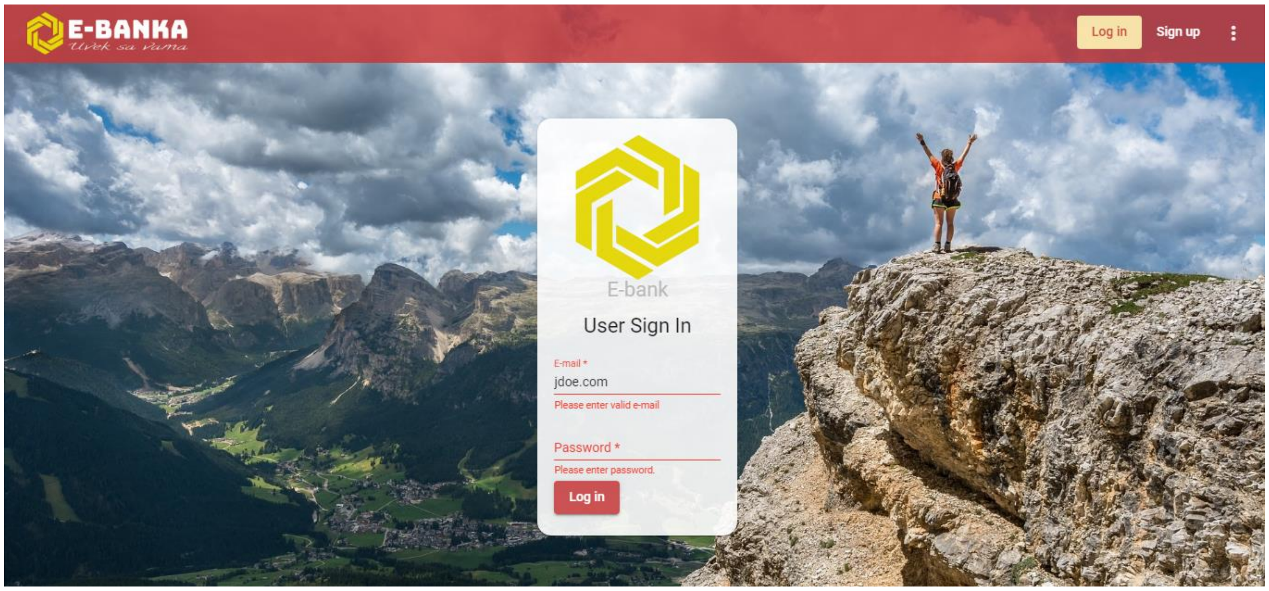
- Sign Up page



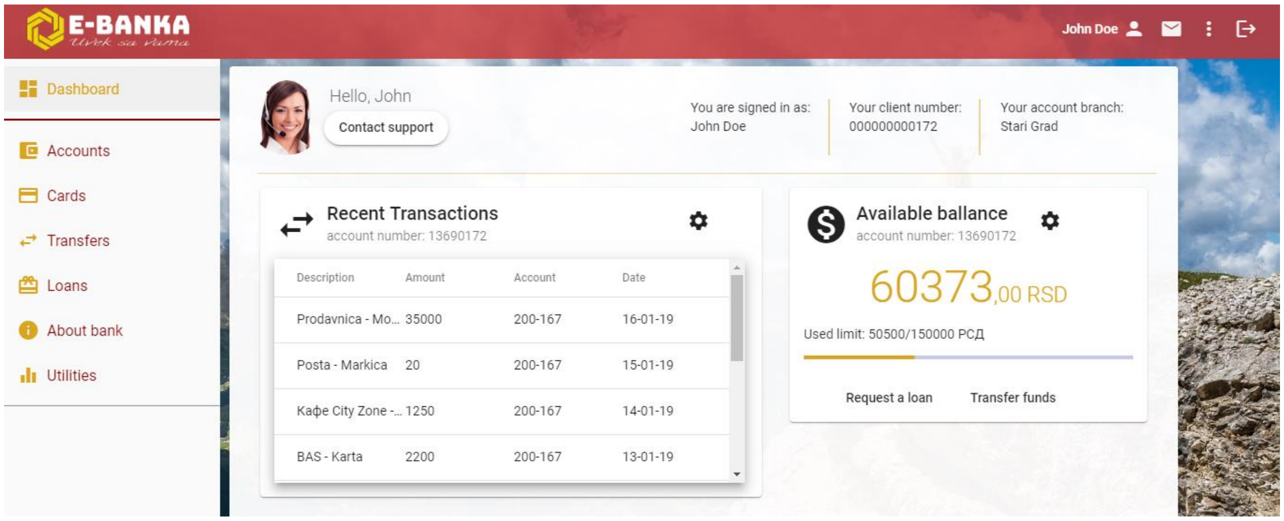
- Log in page



- Log in page invalid email/no password



- Dashboard page



Hello, John

You are signed in as: John Doe

Your client number: 000000000172

Your account branch: Stari Grad

Recent Transactions
account number: 13690172

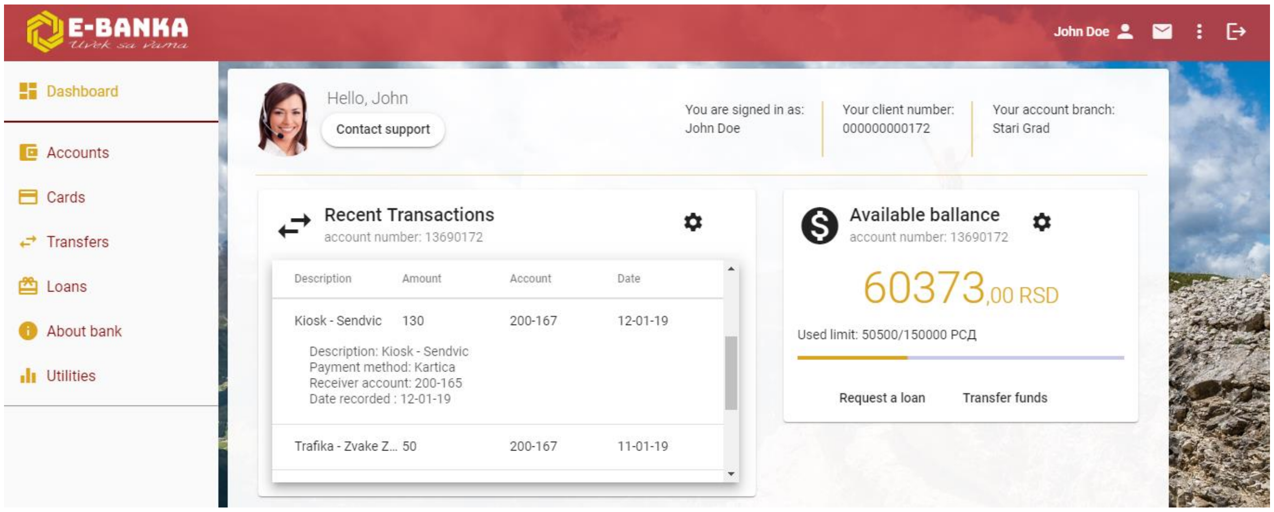
Description	Amount	Account	Date
Prodavnica - Mo...	35000	200-167	16-01-19
Posta - Markica	20	200-167	15-01-19
Кафе City Zone -...	1250	200-167	14-01-19
BAS - Karta	2200	200-167	13-01-19

Available ballance
account number: 13690172

60373,00 RSD

Used limit: 50500/150000 РСД

- Transaction details



Hello, John

You are signed in as: John Doe

Your client number: 000000000172

Your account branch: Stari Grad

Recent Transactions
account number: 13690172

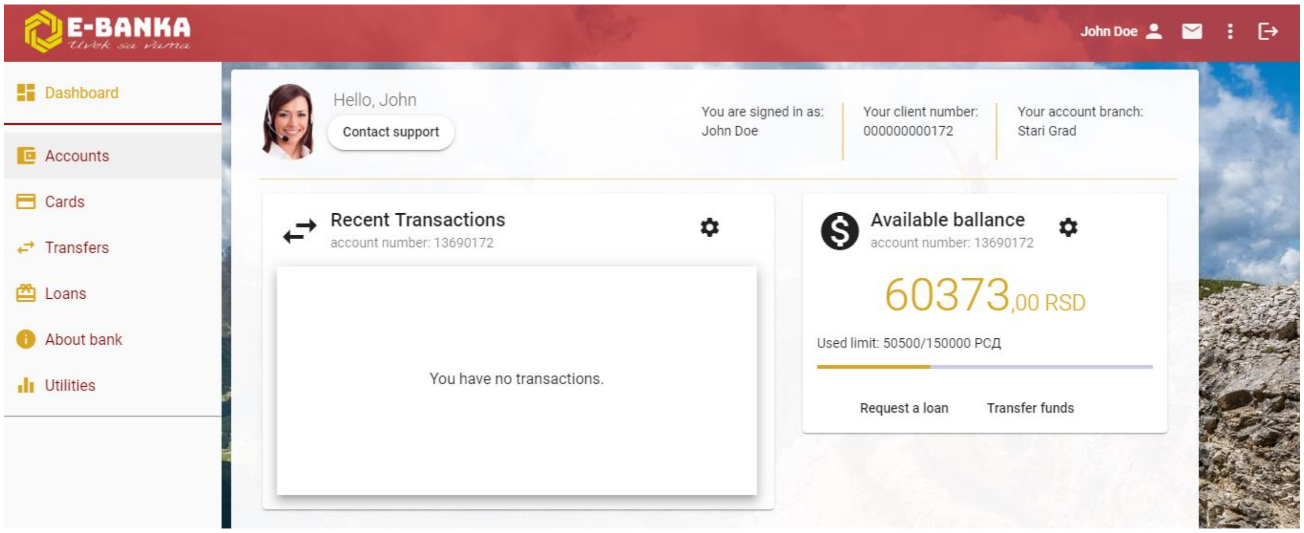
Description	Amount	Account	Date
Kiosk - Sendvic	130	200-167	12-01-19
Description: Kiosk - Sendvic Payment method: Kartica Receiver account: 200-165 Date recorded : 12-01-19			
Trafika - Zvake Z...	50	200-167	11-01-19

Available ballance
account number: 13690172

60373,00 RSD

Used limit: 50500/150000 РСД

- No transcations



Hello, John

You are signed in as: John Doe

Your client number: 000000000172

Your account branch: Stari Grad

Recent Transactions
account number: 13690172

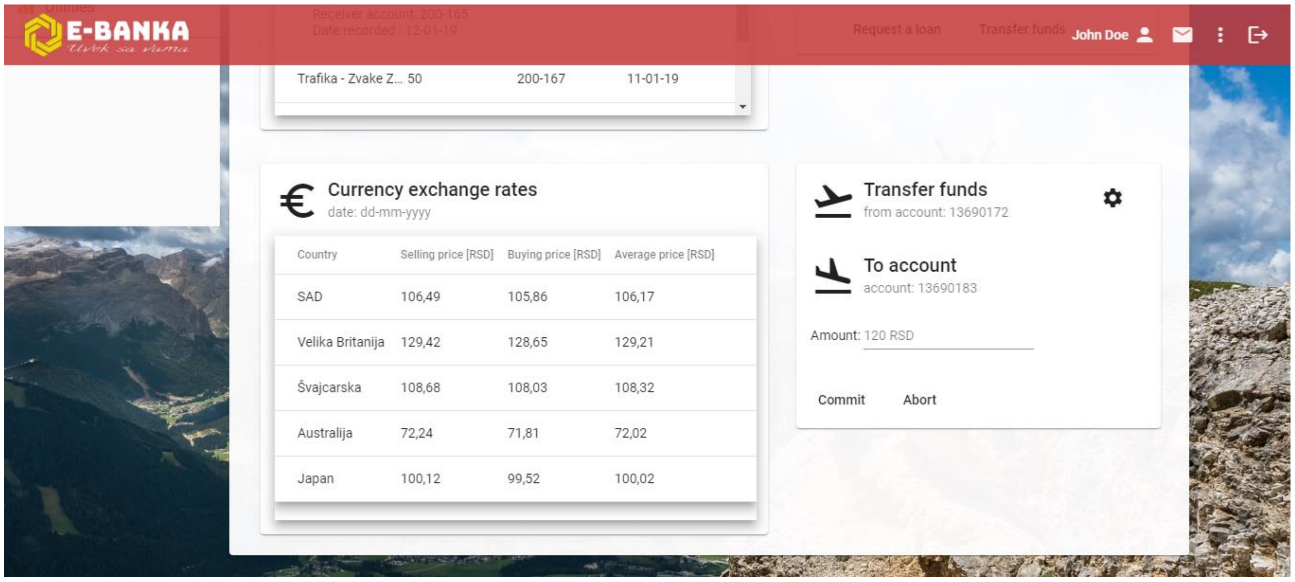
You have no transactions.

Available ballance
account number: 13690172

60373,00 RSD

Used limit: 50500/150000 РСД

- Exchange Rates list (dashboard scrolled down)



Receiver account: 200-165
Date recorded : 12-01-19

Trafika - Zvake Z... 50 200-167 11-01-19

Currency exchange rates
date: dd-mm-yyyy

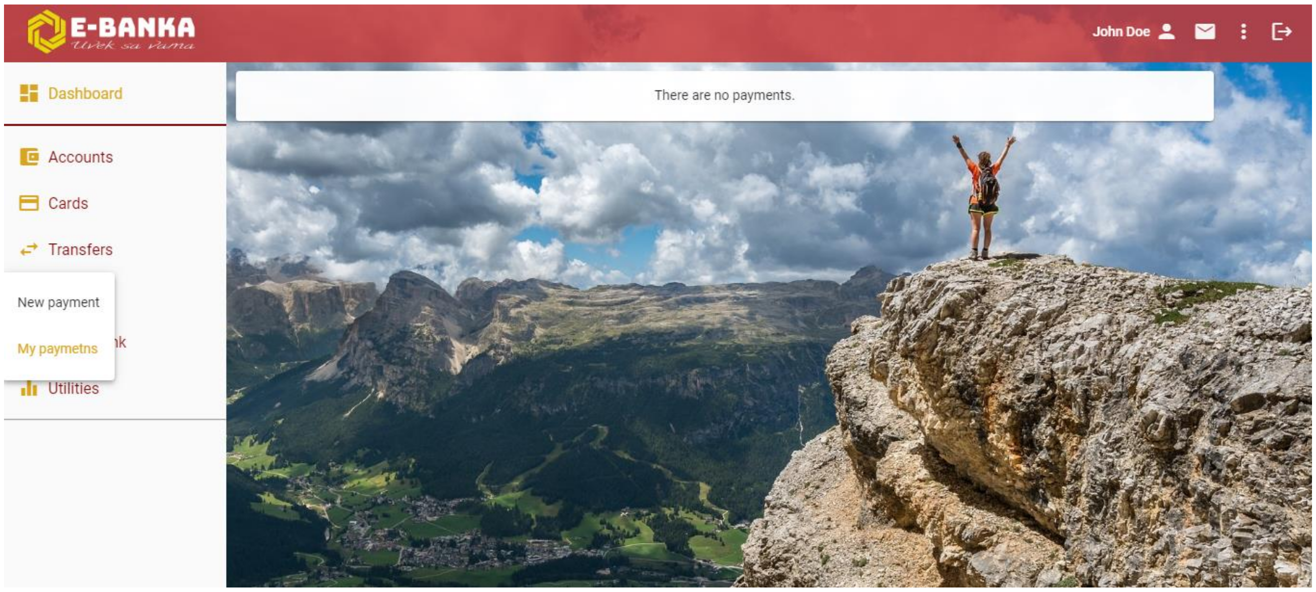
Country	Selling price [RSD]	Buying price [RSD]	Average price [RSD]
SAD	106,49	105,86	106,17
Velika Britanija	129,42	128,65	129,21
Švajcarska	108,68	108,03	108,32
Australija	72,24	71,81	72,02
Japan	100,12	99,52	100,02

Transfer funds
from account: 13690172

To account
account: 13690183

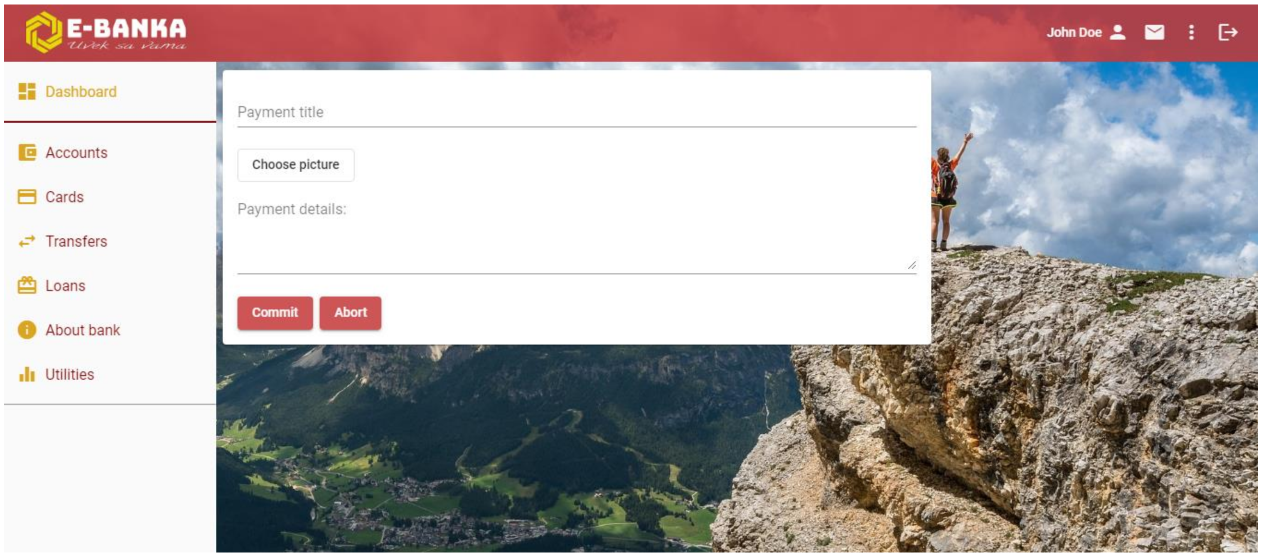
Amount: 120 RSD

- List of all utility payments

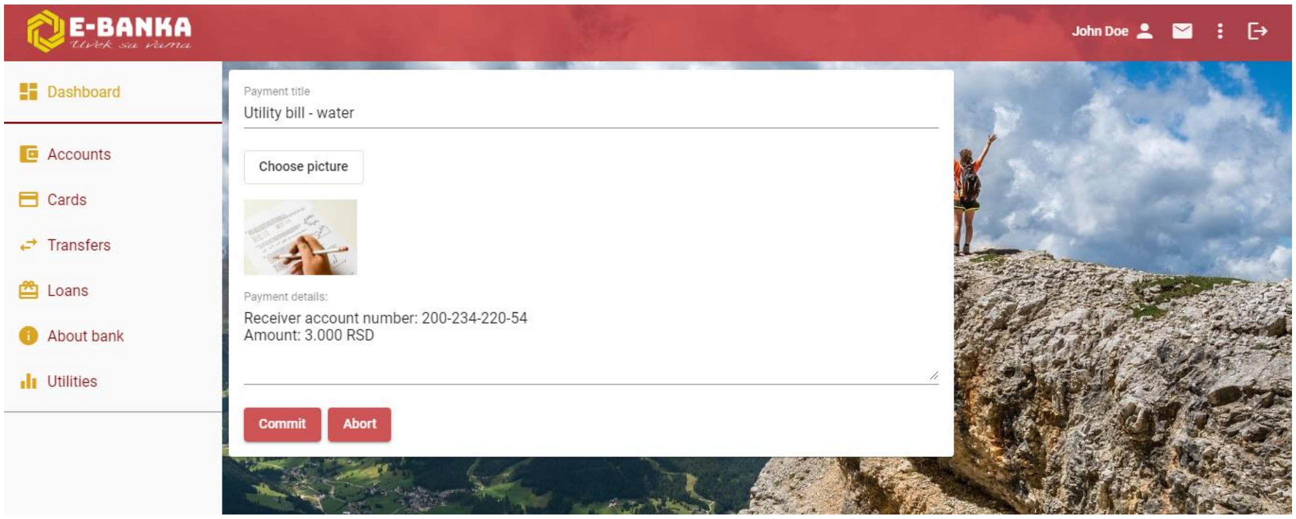


There are no payments.

  - New utility payment



    - Utility payment checkout



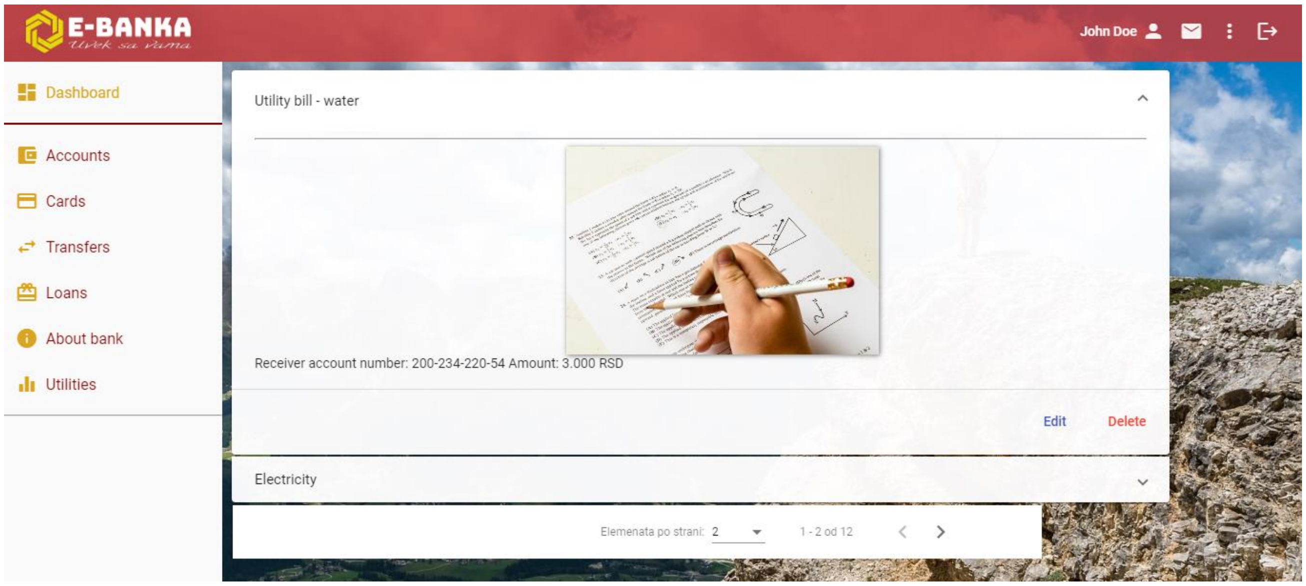
Payment title

Utility bill - water

Payment details:

Receiver account number: 200-234-220-54
Amount: 3.000 RSD

- Utility payments list and details



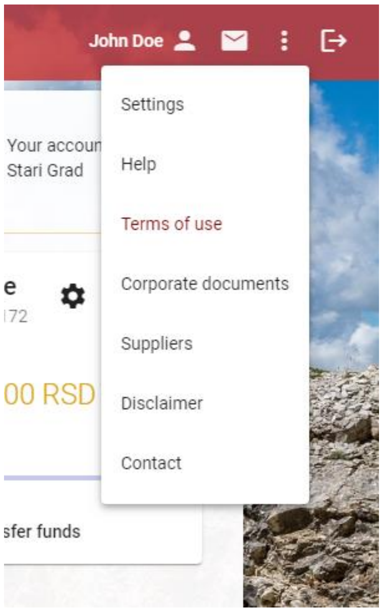
Utility bill - water

Receiver account number: 200-234-220-54 Amount: 3.000 RSD

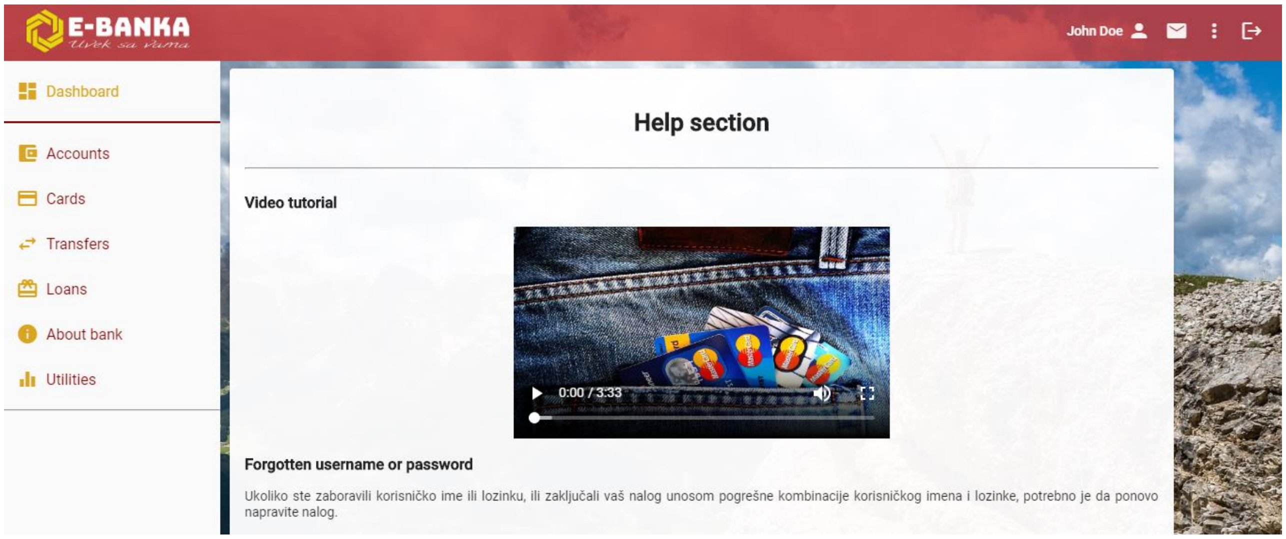
Electricity

Elemenata po strani: 2 1 - 2 od 12

- Drop-down menu



- Help section



Help section

Video tutorial

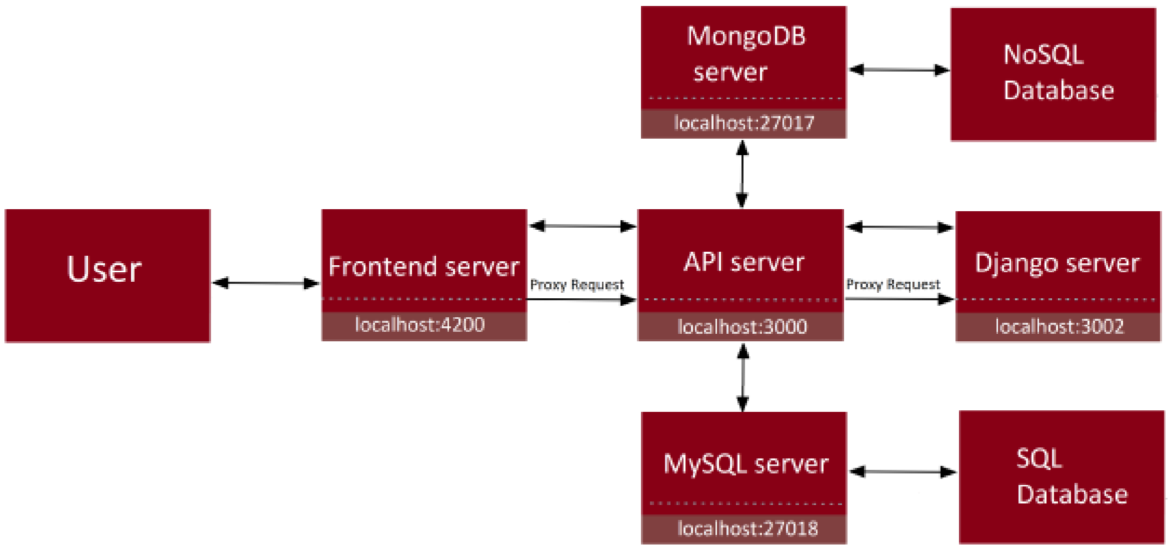
Forgotten username or password

Ukoliko ste zaboravili korisničko ime ili lozinku, ili zaključali vaš nalog unosom pogrešne kombinacije korisničkog imena i lozinke, potrebno je da ponovo napravite nalog.

Project details

Logical structure

From a logical point of view, the system has a 3-tiered REST application architecture. It is a modular client-server architecture that consists of a presentation tier, an application tier and a data tier.



- **Presentation tier**
  Communicates with other two tiers and holds GUI. It is built with Angular, HTML, CSS, and a TypeScript as a front-end logic. Communication with the other tiers is established through API calls.

- **Application tier**
  Handles application logic. It consists of two separate servers, Node.js server (written in JavaScript) and a Django server (written in Python). Their primary purpose is to support the application's core functions and fetch/post user data from databases throughout database server calls.

- **Data tier**
  Stores user related information. The data tier consists of a two database servers:

  - MongoDB server and its database, as far as Robo 3T DBMS
  - MySQL server and corresponding database, as far as MySQL DBMS

MongoDB database is, due to its non-relational nature, used for storing user access credentials, reference to a user bank account number and other user-centered information.

MySQL database is used to store users bank account details and all related transaction details.

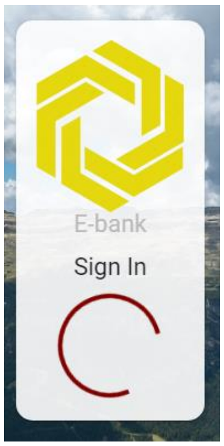
Presentation tier

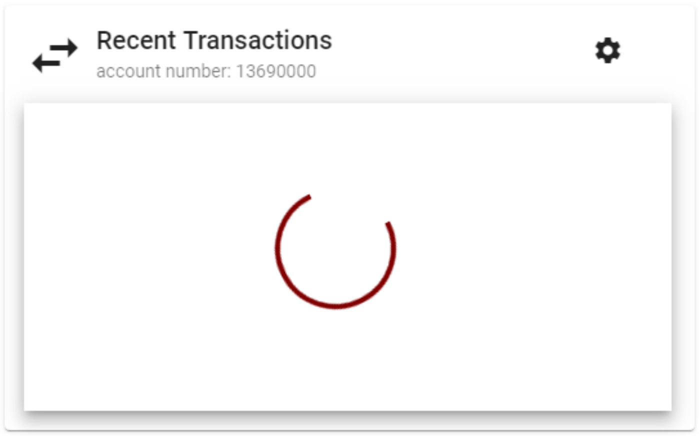
Front-end

- **Design principles**
  User interface design was conducted according to the usability guidelines given in 10 Usability Heuristics for User Interface Design by Jacob Nielsen. It recommends following heuristics for UI design:

  - 1: Visibility of system status
    The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

    Examples (left picture - Sign In visibility status, right picture - Recent Transactions visibility status):

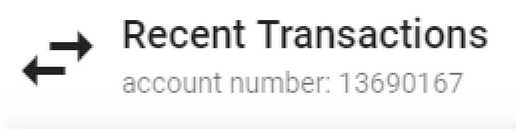




  - 2: Match between system and the real world
    The system should speak the users' language, with words, phrases and concepts familiar to the user.

-

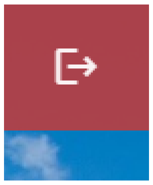
    One way to accomplish this requirement is to use metaphor to symbolically represent abstract idea of a real world experience. For example, opposite arrows icon can be a good real life metaphor for banking transactions:

    
    Recent Transactions
    account number: 13690167

- 3: User control and freedom
- 4: Consistency and standards
  Users should not have to wonder whether different words, situations, or actions mean the same thing.

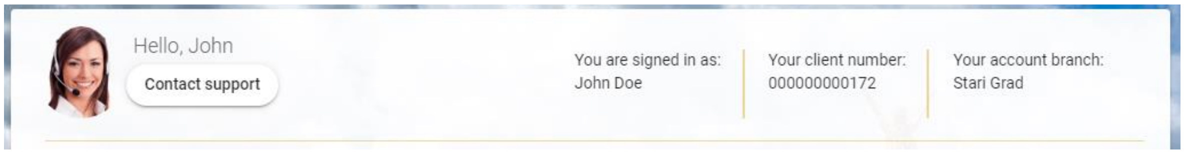
  During front-end development, Google Material Design principles were followed.
- 5: Error prevention
- 6: Recognition rather than recall
  Minimize the user's memory load by making objects, actions, and options visible.

  Stylized door icon can be a good replacement for *Sign out* label:

  
- 7: Flexibility and efficiency of use
  Accelerators — unseen by the novice user — may often speed up the interaction for the expert user.
  Three out of six Gestalt grouping principles of visual perception were used during development:
- Similarity
- Symmetry

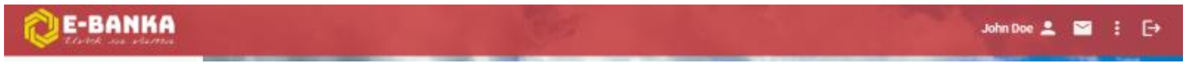
- Closure

  Example of the closure and similarity principles:

  You are signed in as: John Doe | Your client number: 000000000172 | Your account branch: Stari Grad

  Example of the symmetry principle:

- 8: Aesthetic and minimalist design
  Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

- 9: Help users recognize, diagnose, and recover from errors
  Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

- 10: Help and documentation
  Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search and focused on the user's task.
  Help section is mainly focused on a video help instructions, because they are:

  - Easy to understand
  - Short
  - Easily seen
  - Contain minimum amount of task specific information

- **Component organisation and architecture**
  Each Angular front-end component consists of three files:

  - HTML template
  - CSS template
  - TypeScript controller

TypeScript controller's task is to store all variables relevant for component HTML part as far as functions to make calls for service controllers (service controllers are TypeScript modules which initiate API server calls).

- **Services**
    - Authentication service
    - Dashboard service
    - Posts service
    - Auth guard
    - Auth interceptor

Application tier

Node.js server

This is the main application server. It communicates with database and Django servers through API server calls.

Server is capable of executing following tasks:

- User Sign Up/Sign In
- Tokenization service (auto authentication if user authenticated user close or reloads web page, auto user sign out after token expiration)
- Handling HTTP PUT, POST and GET requests
- Getting all user information from databases (make database queries)
- MIME type validation and image upload handling
- Formatting and preparing data in convenient format to be shown on front end
- Creating, editing and deleting of user transactions

Django server

This server has two implemented API services:

- **Dummy data service**
  When a new user signs up, this service sends random JSON object to Node.js sever upon request. JSON object contains random data (mobile phone number, branch name, and home address) pulled out of .csv file (or as a random number

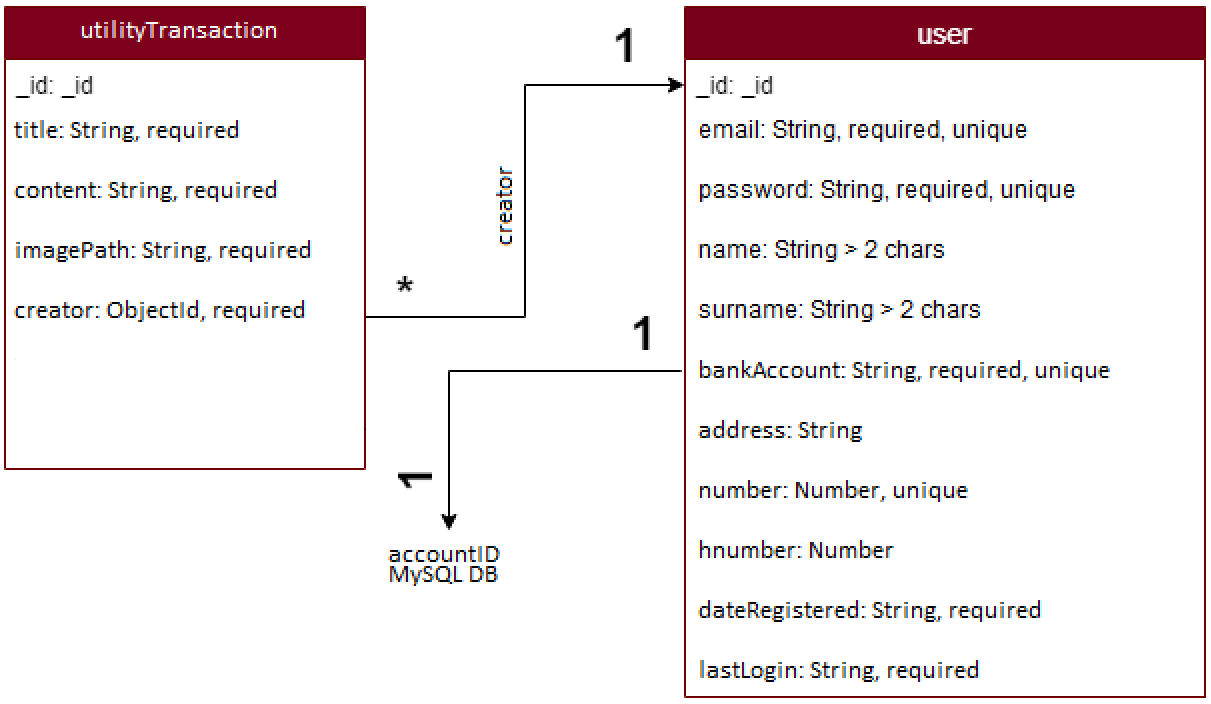
in case of phone number) on Django server. After data receiving, Node.js server updates corresponding user data in MongoDB with the data provided.

- **Exchange rates service**
  This service sends API calls to the free Exchange rates API, formats retrieved data and sends it back to the Node.js server.

Data tier

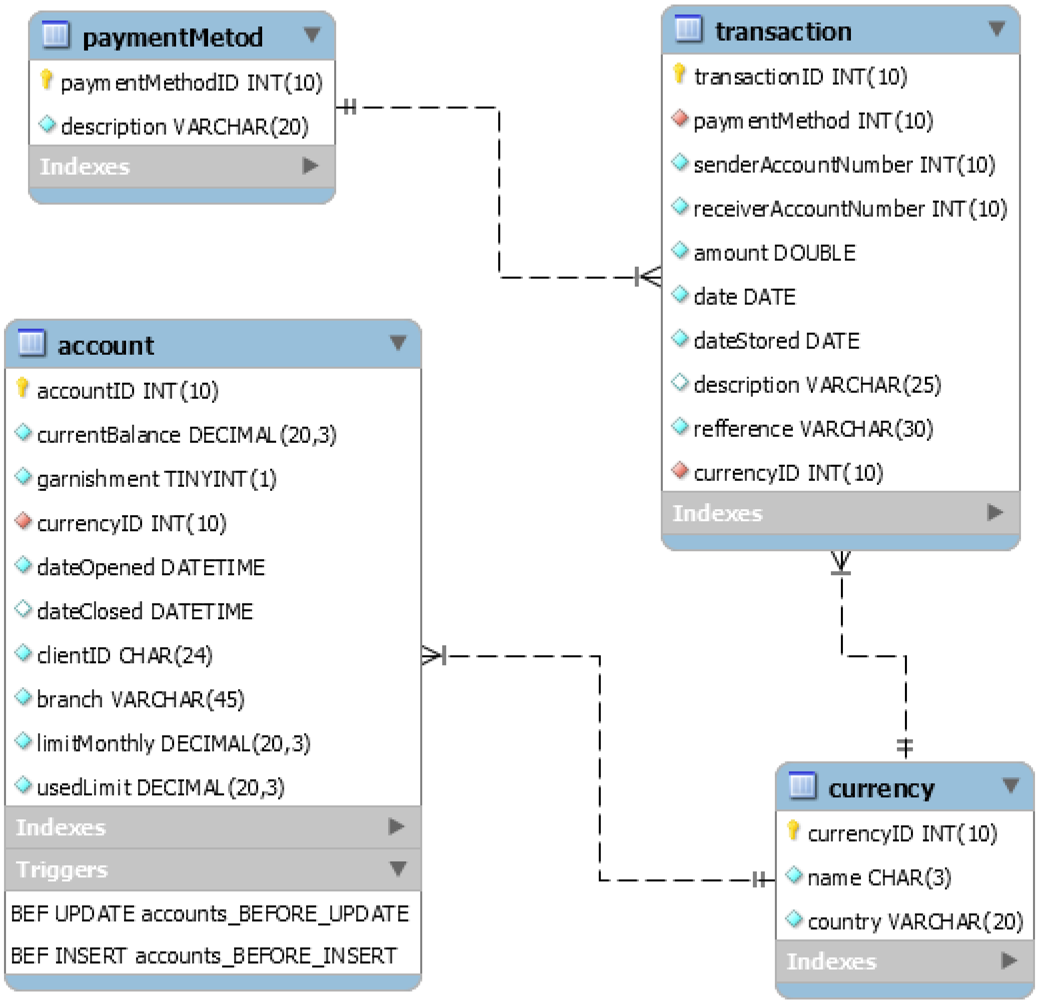
MongoDB database

Database holds data related to a user account and user transactions. User transactions are stored only temporarily in this database (such data should be held in relational database). Also, it is connected with 1-1 relation through foreign key (bankAccount filed) with MySQL table account (private key accountID). In the further development it will be shifted to a MySQL Database, and will be connected through _ id field with customerID table in MySQL. Here is the model of database:


utilityTransaction
_id: _id
title: String, required
content: String, required
imagePath: String, required
creator: ObjectId, required
user
_id: _id
email: String, required, unique
password: String, required, unique
name: String > 2 chars
surname: String > 2 chars
bankAccount: String, required, unique
address: String
number: Number, unique
hnumber: Number
dateRegistered: String, required
lastLogin: String, required
creator
*
1
1
1
accountID
MySQL DB

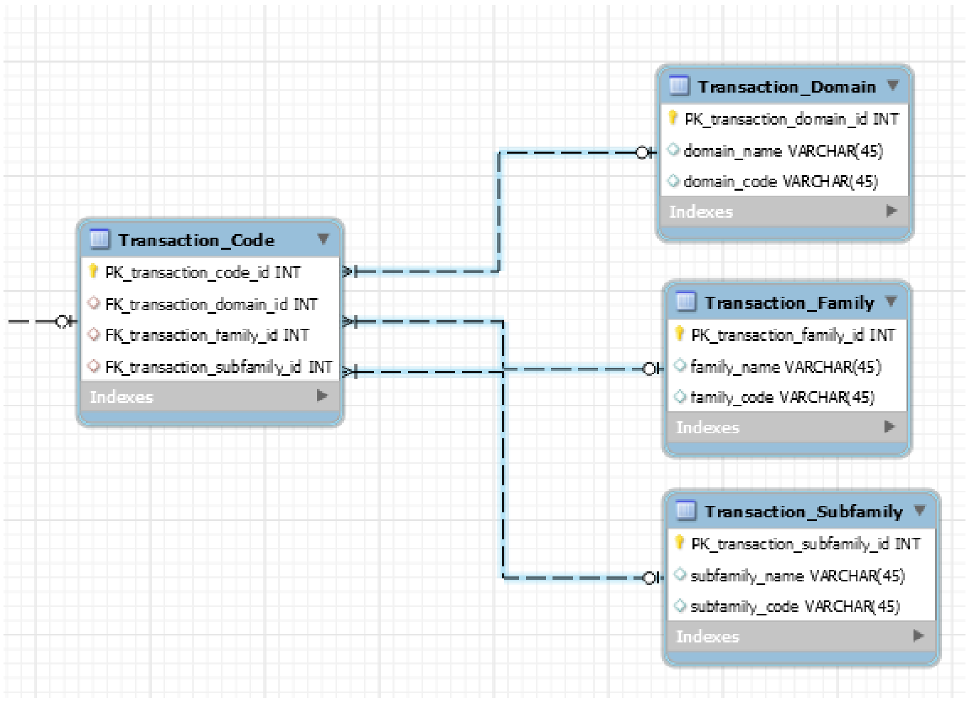
MySQL database

- **Testing database**
  At the moment, this database is used only for testing purposes. It has 4 tables and it is also connected with MongoDB through foreign key field clientID.



- **Real-world database (work in progress)**
  This database has been developed according to the several international standards used in an IT banking systems:

  - **ISO 20022 Financial Services - Universal financial industry message scheme**
    As a result of guidelines and schemes presented in this standard, following tables were modeled:

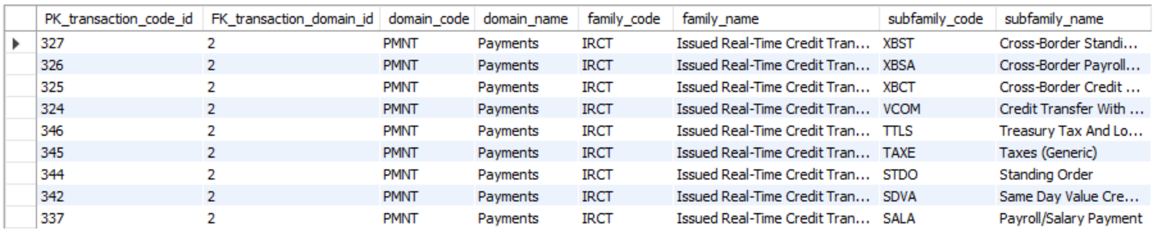


Transaction_Domain
PK_transaction_domain_id INT
domain_name VARCHAR(45)
domain_code VARCHAR(45)
Indexes
Transaction_Code
PK_transaction_code_id INT
FK_transaction_domain_id INT
FK_transaction_family_id INT
FK_transaction_subfamily_id INT
Indexes
Transaction_Family
PK_transaction_family_id INT
family_name VARCHAR(45)
family_code VARCHAR(45)
Indexes
Transaction_Subfamily
PK_transaction_subfamily_id INT
subfamily_name VARCHAR(45)
subfamily_code VARCHAR(45)
Indexes

Example:
Suppose we want to list last nine possible transaction codes and its corresponding names related to Issued Real-Time Credit Transfer payments (as defined in ISO 20022).
Then, we would form the following SQL query:

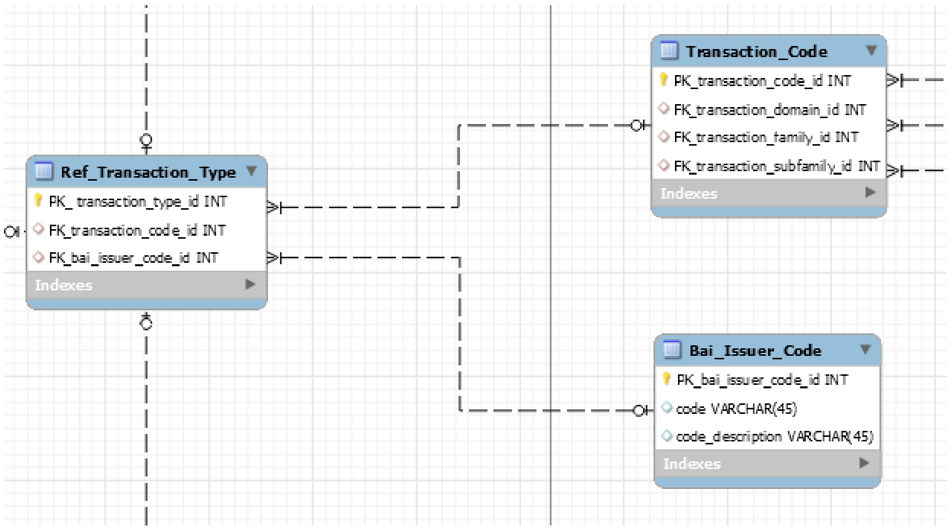
```
`SELECT
   PK_transaction_code_id,
   FK_transaction_domain_id,
   domain_code, domain_name,
   family_code, family_name,
   subfamily_code,
   subfamily_name

FROM
   ref_transaction_code,
   ref_transaction_domain,
   ref_transaction_family,
   ref_transaction_subfamily

WHERE
   domain_code = "PMNT" &&
   family_code = "IRCT" &&
   FK_transaction_domain_id = PK_transaction_domain_id &&
   FK_transaction_family_id = PK_transaction_family_id &&
   FK_transaction_subfamily_id = PK_transaction_subfamily_id

ORDER BY subfamily_code DESC

LIMIT 9`
```

and would get the next query result:

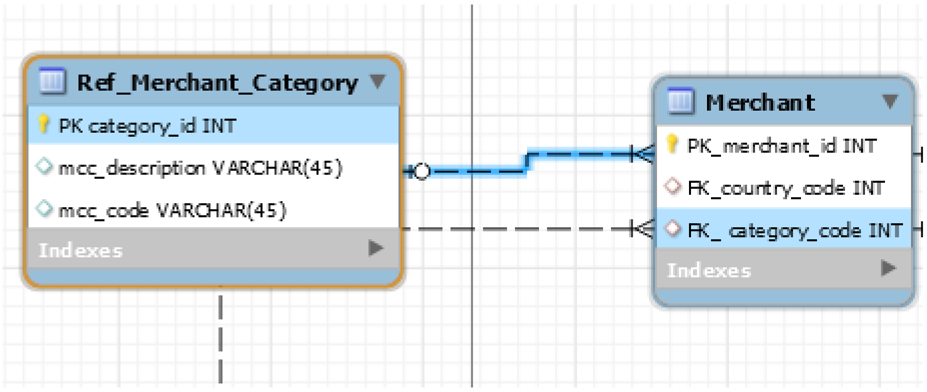


PK_transaction_code_id	FK_transaction_domain_id	domain_code	domain_name	family_code	family_name	subfamily_code	subfamily_name
327	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	XBST	Cross-Border Standi...
326	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	XBSA	Cross-Border Payroll...
325	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	XBCT	Cross-Border Credit ...
324	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	VCOM	Credit Transfer With ...
346	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	TTLS	Treasury Tax And Lo...
345	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	TAXE	Taxes (Generic)
344	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	STDO	Standing Order
342	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	SDVA	Same Day Value Cre...
337	2	PMNT	Payments	IRCT	Issued Real-Time Credit Tran...	SALA	Payroll/Salary Payment

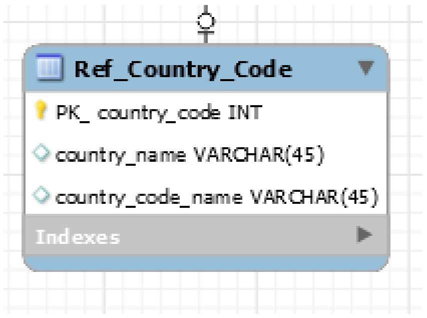
- **BAI2 codes**
  ISO 20022 suggests using the BAI code appended to the transaction code. So, the following table is created, and together with transaction code table uniquely identifies transaction type:


Transaction_Code
PK_transaction_code_id INT
FK_transaction_domain_id INT
FK_transaction_family_id INT
FK_transaction_subfamily_id INT
Indexes
Ref_Transaction_Type
PK_ transaction_type_id INT
FK_transaction_code_id INT
FK_bai_issuer_code_id INT
Indexes
Bai_Issuer_Code
PK_bai_issuer_code_id INT
code VARCHAR(45)
code_description VARCHAR(45)
Indexes

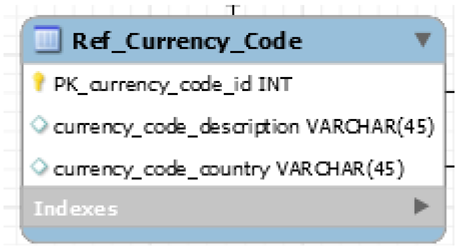
- **ISO 18245:2003 Retail financial services** - Merchant category codes (MCC)
  This standard was used for the merchant category table modeling. This table stores MCC codes upon which all electronic payments (card payments) are standardized and categorized.



- **ISO 3166-1 alpha-3** - Three-letter country codes



- **ISO 4217** - Currency designators
  This standard delineates currency designators, country codes (alpha and numeric), and references to minor units. It was used for the creation of currency codes table.


Ref_Currency_Code
PK_currency_code_id INT
currency_code_description VARCHAR(45)
currency_code_country VARCHAR(45)
Indexes

Complete EER diagram of the database can be seen here.

Build

Run `ng build` to build the project. The build artifacts will be stored in the `dist/`directory. Use the `--prod` flag for a production build.
To get more help on the Angular CLI use `ng help` or go check out the Angular CLI README.

Servers

Front-end development server setup

Run `ng serve` to start a development server. Navigate to `http://localhost:4200/`. The app will automatically reload if you change any of the source files.

Back-end servers setup

Node server

Run `npm run start:server` to start a Node.js server.

MongoDB server

Download and install:

- MongoDB Community Server 4.2.0 from its official website
- Robo 3T DBMS software from its official website

Run `mongod` to start a MongoDB database server. Make sure you have added **mongod** daemon to the `$PATH` variable.
If everything goes well you should see following message in CLI:

```
I NETWORK [initandlisten] waiting for connections on port 27017
```

MySQL server

Download MySQL Installer from its official website and install:

- MySQL Community Server 8.0.17
- MySQL Workbench 8.0.17.
  If everything goes without errors, MySQL server daemon should be automatically started.

Django server

Download and install:

- Python 3.7.4 from its official website
- Django framework with `pip install Django==2.2.4`
- Install virtual environment with `pip install virtualenv`
  - Launch virtualenv env in cmd
  - Start new environment `cd your_project virtualenv env`
  - Activate virtualenv (on Windows) `cd \env\Scripts\activate.bat`
  - Now install following Python libraries:
    `pip install numpy pandas request`

In order to run a Django server, do the following:

- Change working directory to `cd \server-python\env\pythonserver`
- Start a server with `python manage.py runserver 3002`
- To check if server works, head to http://127.0.0.1:3002/randomUserData/.

Further development

Front-end

- Adding user ability to get an graphical report on statistics of its bank account with plotly
- If you have any questions, feel free to write me at matejavulic@gmail.com

Back-end

- Adding two new types if web app authentication:
    - Mobile device token
    - Fingerprint biometric authentication
    - QR code payments

Contact

If you have any questions, feel free to write me an e-mail at matejavulic@gmail.com .