eCare Project

Technical Solution Description

Olga Andreeva

Content

Introduction	3
Technologies and Frameworks	3
Additional Features	3
DataBase	4
Description of the Model	5
Users Interface	6
Services	7
Entities	10
Repositories	12
Appendix 1 (Screenshots)	14
Appendix 2 (ePromo)	18

Introduction

eCare is a web application for a mobile operator. This project gives permission to manage the contracts, tariffs, options and clients profiles. Administrator has got privileges to create a new contract, edit and block customers' contracts, add and change tariffs with options. Usual users have a right to manage their personal contract, inspect options and tariffs.

Technologies and Frameworks

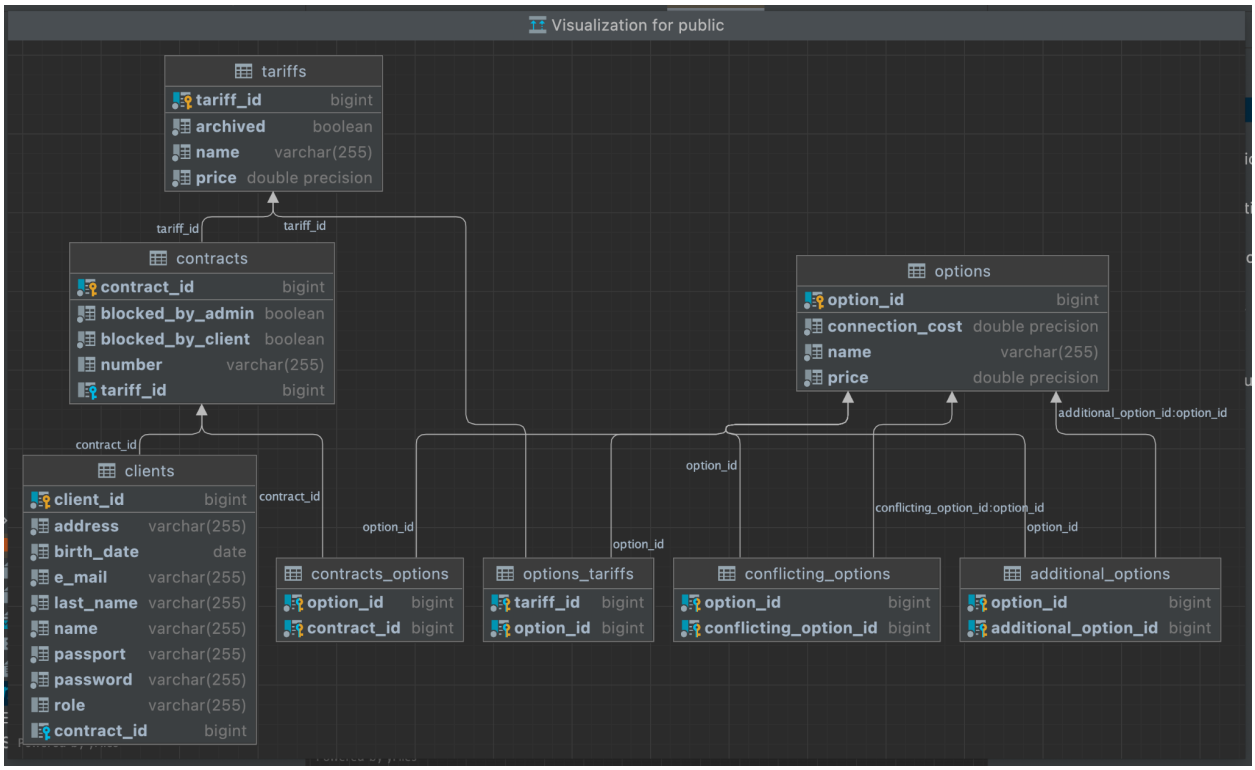
- Spring Boot
- Spring Data JPA
- Spring Security
- PostgreSQL
- JSP
- Maven

Additional Features

SonarQube tool for checking code quality and test coverage;

Aspects (AOP) for separation of core application logic and common tasks;

DataBase



Main Tables

Clients	Client's information. Contains name, email, etc.
Contracts	Contract's information. Contains number, blocking level, etc.
Tariffs	Tariff's information. Contains name, price, etc.
Options	Option's information. Contains name, cost, etc.

Relationships Tables

Contracts_options	Contracts-Options many-to-many relationship. Defines options chosen for the contract.
Options_tariffs	Options-Tariffs many-to-many relationship. Defines options chosen for the tariff.
Conflicting_options	Options-Options many-to-many relationship. Defines “incompatible” relationship between options.
Additional_options	Options-Options many-to-many relationship. Defines “required” relationship between options.

Description of the Model

Client is a registered user of the service. A regular user can only manage his/hers contract. A user with administrative rights can create a new contract, edit and block customers’ contracts, add and change tariffs with options.

Contract combines the related tariff and options. It contains the user's contract number. A contract can only have one user. A contract can be blocked/unblocked by its user or by the administrator.

Tariff is the main “rule” of a contract. It defines options that can be related to its contract. Every contract has exactly one tariff.

Option is an additional “rule” of a contract. It is related to the contract’s tariff. Some options have relationships “required” and “incompatible” between them.

Modules Interaction

<table>
<tr><td rowspan="2">Data as DTO</td><td>Client</td></tr>
<tr><td>Controller</td></tr>
<tr><td rowspan="3">Data as Entity</td><td rowspan="2">Service</td></tr>
<tr><td>Repository</td></tr>
<tr><td>Database</td></tr>
</table>

Users Interface

List of Available Web Pages:

- Login – login to service;
- Registration - register a new client;
- Admin pages:

    - Clients' list – table of all clients with their contracts; allows to search and switch to a client.
    - Tariffs' list – table of all tariffs; allows to archive a tariff, add a new one.
    - Options' list – table of all options; allows to add a new option.
    - Contract profile – editing a contract; allows to block/unblock a contract, update its tariff and options.
    - Tariff profile – editing a tariff; allows to change a tariff's name, price and set its options.
    - Option profile – editing an option; allows to change an option's name, price and sets its additional and conflicting options.
    - New contract – creating a new contract; allows to set a contract number.
- Customer pages:
    - Contract profile – editing a contract; allows to block/unblock a contract, update its tariff and options.

Services

The application consists of four services with implementation for each one:

1. Client service
2. Contract service
3. Option service
4. Tariff service

List of Methods:

Client service

Method	Description
findClientByIdClient	finds a client in DB by id
findClientByEmail	finds a client in DB by email
findClientByContract	finds a client in DB by contract
getClients	gets a list of all clients from DB
registerNewClient	adds a new client to DB

Contract service

Method	Description
findContractByIdContract	finds a contract in DB by id
getContracts	gets a list of all contracts from DB
blockByAdmin/ unblockByAdmin	sets (un)block status for a contract
blockByClient/ unblockByClient	sets (un)block status for a contract
updateContractOptions	sets new options to a contract
getGeneratedNumbers	gets a list of new unique numbers
generateNumber	generates a random contract number
updateContractTariff	sets a new tariff to a contract
addNewContract	adds a new contract to DB

Option service

Method	Description
findOptionByIdOption	finds an option in DB by id
getOptions	gets a list of all options from DB
updateOption	sets new required and incompatible options to the main one
validateOptions	checks if there are all required options and no incompatible ones in selection

Tariff service

Method	Description
findTariffByIdTariff	finds a tariff in DB by id
getTariffs	gets a list of all tariffs from DB
archiveTariff	sets archived status for a tariff so that it stops being available for selection
prepareTariffOptionsForUpdate	checks if there are all required options and no incompatible ones in selection
updateTariffOptions	sets new options for tariff after their validation
addNewTariff	adds a new tariff to DB

Entities

Entities package contains four classes:

1. Client
2. Contract
3. Option
4. Tariff

1. **Client** is a service user; a client has only one contract.

Field	**Description**
idClient	user's id, autogenerated by DB
name	user's first name
lastname	user's last name
birthDate	user's birthdate
passport	user's passport, must be unique in DB
address	user's address
email	user's email, must be unique in DB
password	user's password, hashed by BCrypt
role	a role set for a user
contract	a contract set for a user

2. **Contract** represents a contract; a contract has only one client and tariff, and a unique contract number.

Field	Description
idContract	contrac's id, autogenerated by DB
number	contract's number, must be unique in DB
blockedByClient	defines a contract's blocking status
blockedByAdmin	defines a contract's blocking status
client	a client related to the contract
tariff	a tariff set for a contract
contractOptions	options set for a contract's tariff

3. **Option** represents an option; it has one or more tariffs and contracts.

Field	Description
idOption	option's id, autogenerated by DB
name	option's name
price	option's price
connectionCost	option's connection cost
conflictingOptions	options that cannot be added together to one tariff
additionalOptions	options that must be added together to one tariff
tariffs	tariffs related to the option
contracts	contracts related to the option

4. **Tariff** represents a tariff; it has one or more options and contracts.

Field	Description
idTariff	tariff's id, autogenerated by DB
name	tariff's name
price	tariff's price
archived	defines a tariff's status for deletion
markedForUpdate	defines a tariff's status for update after any changes of its options
options	options set to the tariff
contracts	contracts related to the tariff

Repositories

All repository package is represented by the interfaces for the above mentioned entities:

Client Repository:

- findClientByContract;
- findClientByIdClient;
- findClientByEmail;
- findClientByPassport;
- getClientByEmail;

Contract Repository:

- findContractByNumber;
- findContractByIdContract;
- existsContractByNumber;

Option Repository:

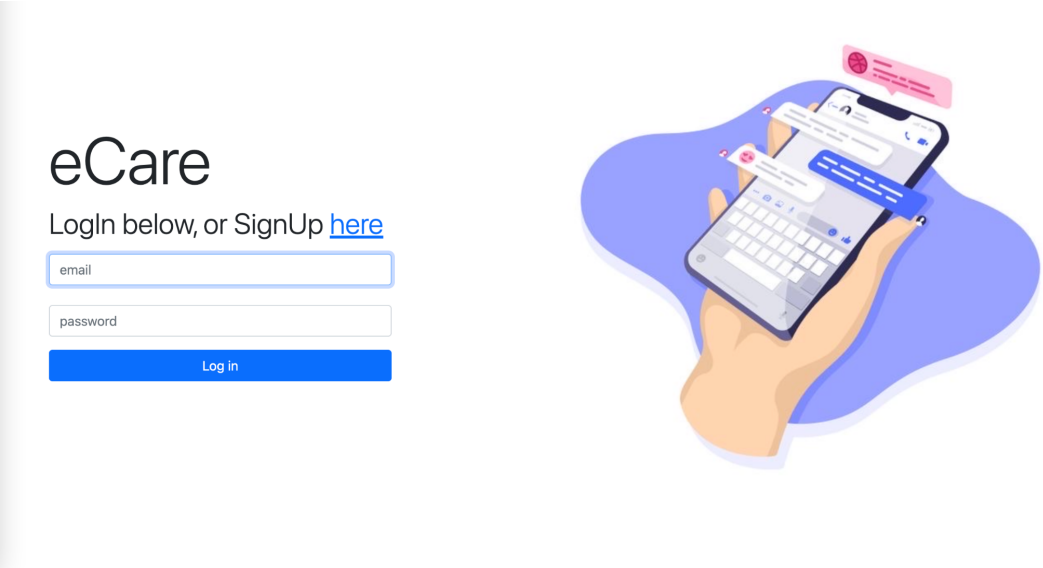
- findOptionByIdOption;

Tariff Repository:

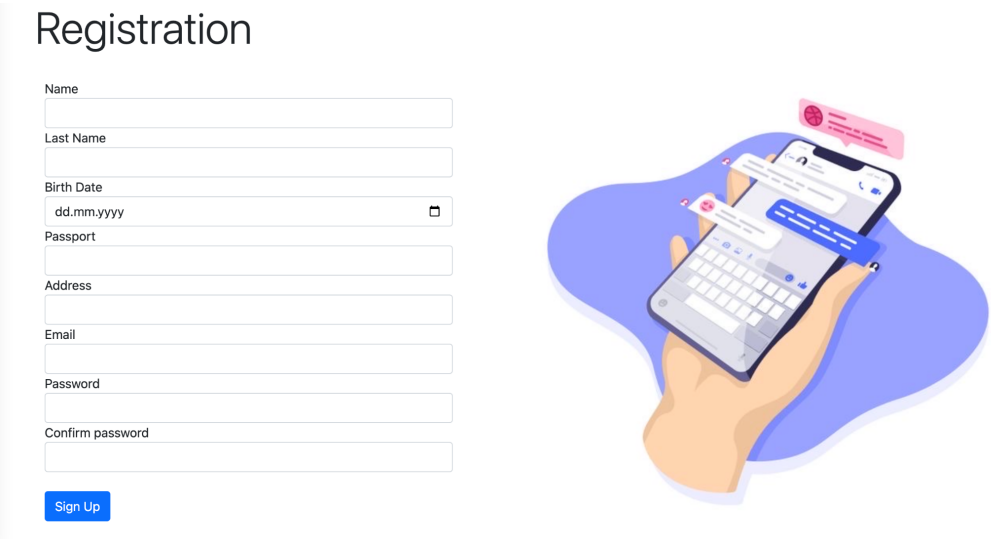
- findTariffByIdTariff;
- findTariffByName;

Appendix 1 (Screenshots)

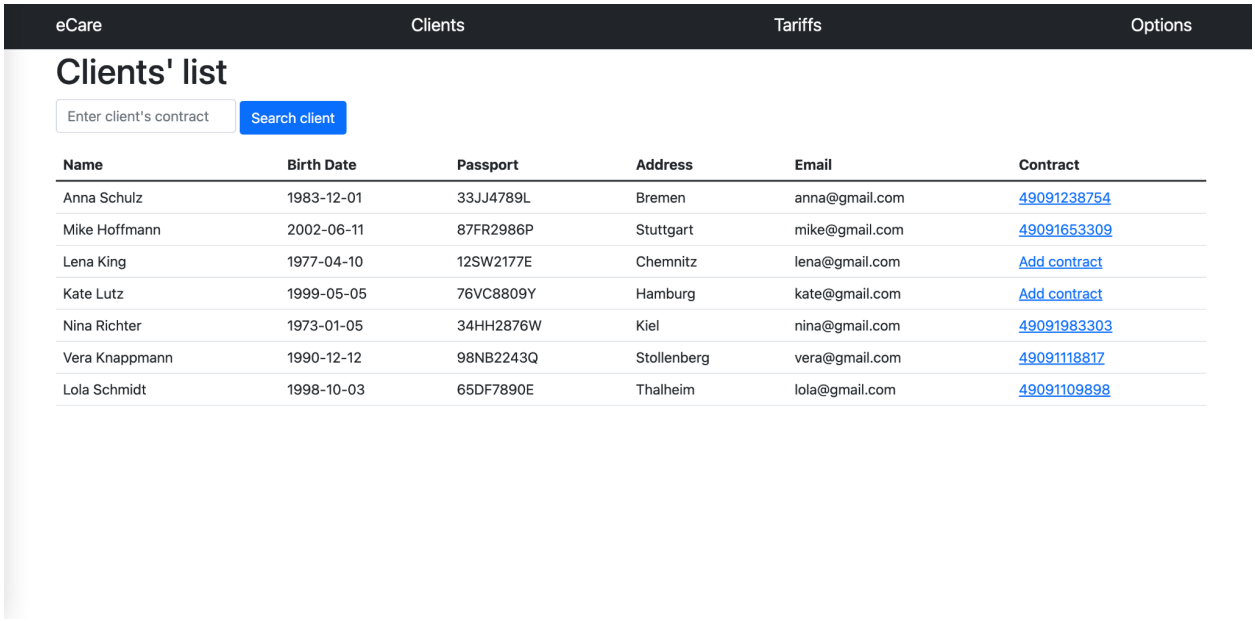
- LogIn form



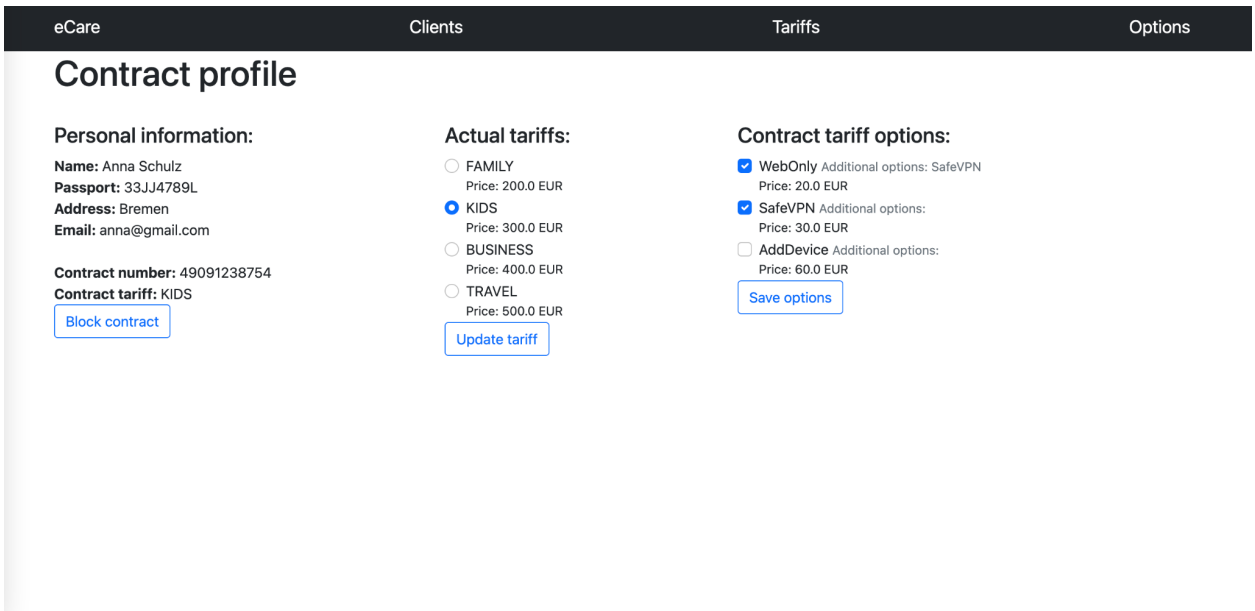
- Registration form



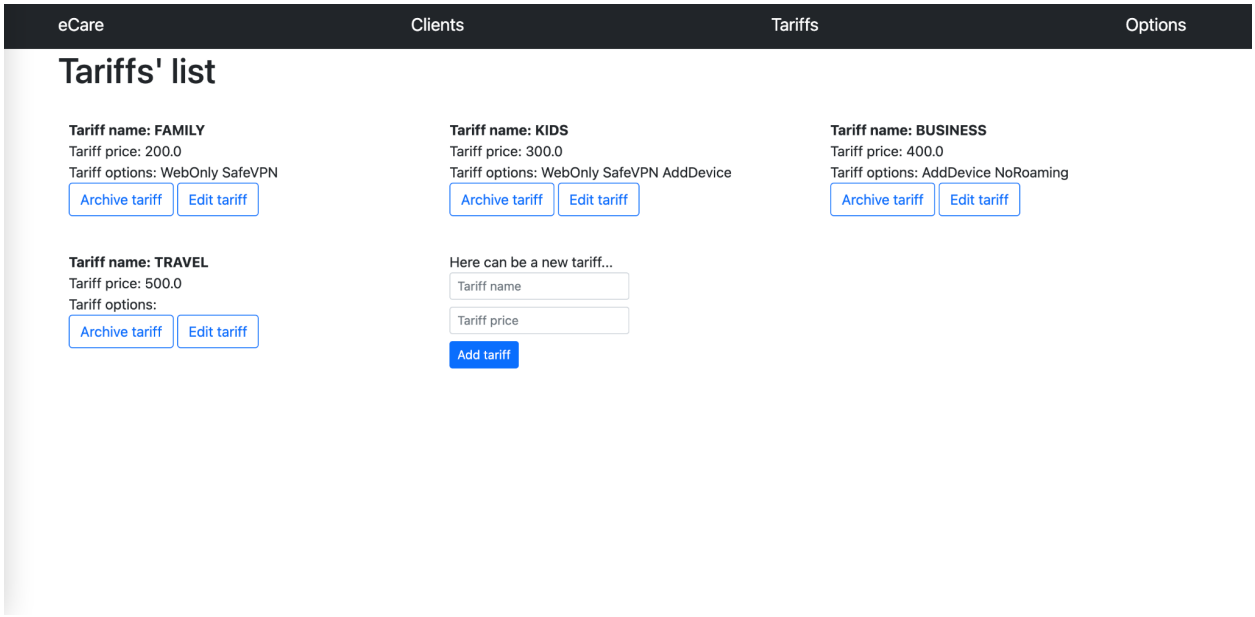
- Clients' list



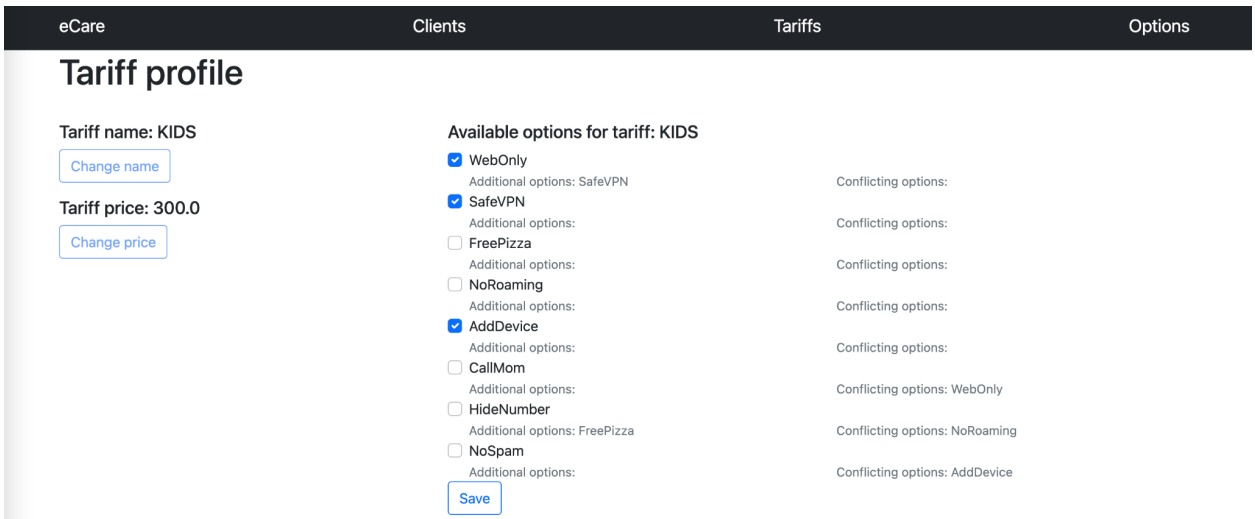
- Contract profile



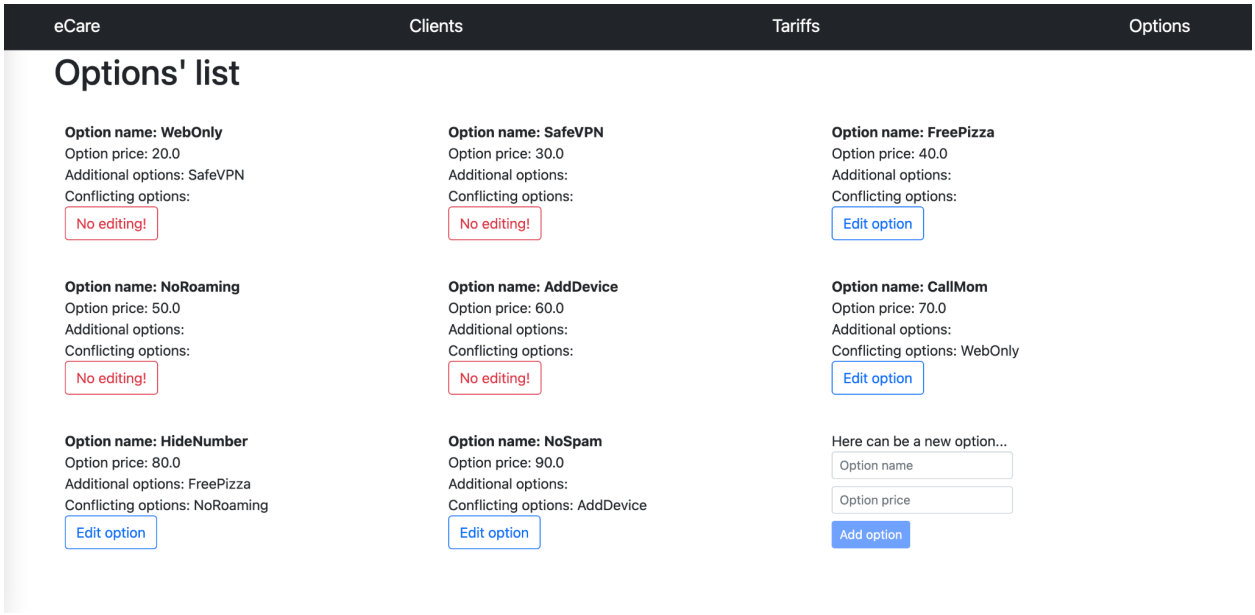
- Tariffs' list



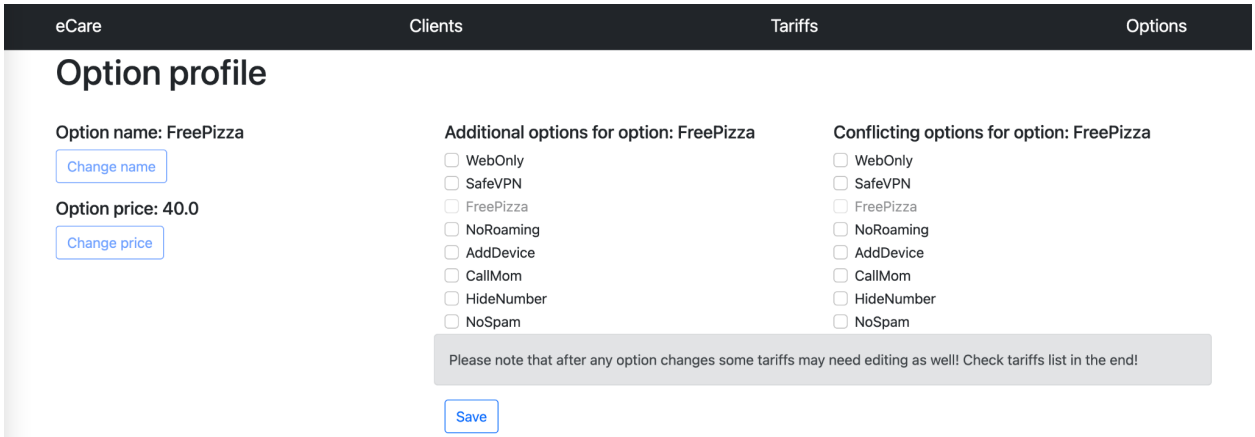
- Tariff profile



- Options' list



- Option profile



Appendix 2 (ePromo)

Introduction:

ePromo is a microservice for eCare web application. It functions independently, and allows to display the most popular tariff of eCare together with its options as an advertisement.

Technologies and Frameworks:

Spring MVC

RabbitMQ

JavaScript

WebSocket

Maven

Description:

1. By the start of eCare application: eCare creates MQ connection to ePromo, and sends its most popular tariff (with its options) to ePromo as a message. Further eCare will send any other messages every time after updating tariffs by any contract.
2. By the start of ePromo application: ePromo creates the connection to eCare, receives the message, and saves the message into the newly created text file. Further the connection will also send the message to a websocket session as soon as any session is open.
3. By the opening of a websocket session: a session reads a message from the text file and displays it. Further the session will display any other messages receiving them directly from the connection.