eCare project

Technical Solution Description

Nikita Sugorovskiy

Content

Introduction 3

Technologies and frameworks 3

Additional features 3

Database 4

Description 4

Description of the model 5

User interface 7

List of available pages 7

Additional features 8

Services 8

List of methods 9

Entities and repository 11

Entities 11

Repository 14

Screenshots 15

# Introduction

eCare project – web application for mobile operator. This project gives permission to manage the contracts, customer, tariffs and options. Users with administrator privileges can create new customers, editing and blocking of customers contracts. Usual users have a right to manage their personal contract, inspect options and tariffs.

## Technologies and frameworks

* Spring
  + Spring MVC
  + Spring Data JPA
  + Spring AOP
  + Spring security
* Hibernate
* jQuery, css.
* EJB
* JSF
* Maven
* WildFly

# Additional features

Password recovery via email with sending a temporary code.

Archiving rates and options to remove them from access.

# Database

## Description

Main table

|  |  |
| --- | --- |
| **Table** | **Description** |
| Customers | Customer’s information. Contains name, surname, email, password, etc. |
| Contracts | Contracts information. Contains number, balance, block level, etc. |
| Tariffs | Tariffs information. Contains name, cost and description. |
| Options | Options information. Contains name, cost, description, etc. |
| Roles | Contain name of role and id |

Relationships table

|  |  |
| --- | --- |
| **Table** | **Description** |
| Possible\_options\_of\_tariffs | Options – Tariffs many-to-many relationship.  Define options, that available for tariff. |
| Required\_options\_relationships | Options – Options many-to-many relationship. Define “required” relationship between options. |
| Forbidden\_options\_relationships | Options – Options many-to-many relationship. Define “incompatible” relationship between options. |
| Used\_options\_of\_tariff | Options – Contracts many-to-many relationship. Define chosen options by contract. |
| Users\_roles | Roles – customer many-to-many relationship.  Сontains information about user roles |

# Description of the model

**Customer -** registered user of the service. A normal user can only manage their contract. A user with administrative rights can register new users, create and manage contracts, tariffs, and options

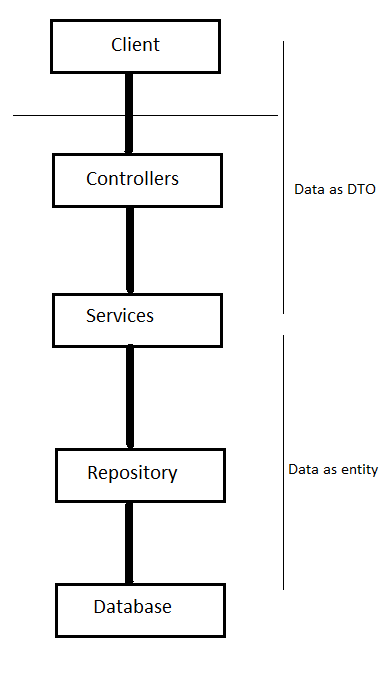
**Contract -** Combines the used tariff and options. Contains the user's phone number. A contract can only have 1 user. The contract can be blocked by the user or by an employee eCare.

**Tariff -** Main “rule” of contract. Defines options, that can be used by contract. Every contract has exactly one tariff.

**Option -** additional “rule” for contract. Provided by used tariff. Every option provided by one or more tariffs. Some options have relationships “required” and “incompatible” between them.

**Role** - defines the user's role. Depending on the role, the registered user has different rights.

Modules interaction



# User interface

# List of available pages

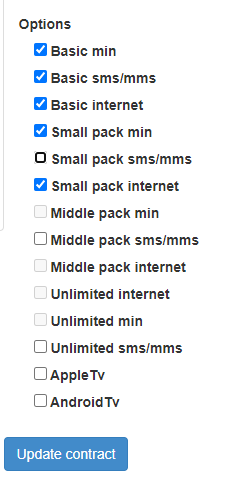
* login – login to service
* admin pages
  + Show all customers – table of all customers. Searching and switch to specific customer. Show contract of customer.
  + Show all tariffs – table of all tariffs.
  + Edit tariff - change page for tariff.
  + Show all options – table of all options.
  + Show all contracts – table of all contract.
  + Customer contracts - switch to customer with chosen contract
  + Add new options – add new option. Defining name, cost, tariff, relationships etc.
  + Add new tariff – add new tariff. Defining name, cost, options etc.
  + Add new customer – add new customer. Defining name, surname, passport data, new contract data, etc
  + Specific customer page – page of customer. Allows to add, edit, block and delete used contracts.
* Customer pages
  + Me – page of customer. Allows to edit and block contracts.
  + Change tariff – list of available options.
  + Change options – table of all options.

## Additional features

Web interface provided smart options relationships resolving system.

If one option is chosen, all options that are required would be chosen. All Incompatible options would be disabled and unchecked.

This rules are defined for each case and prevent wrong option configuration.



# Services

The application contains five services and one generic service

* Generic service
* Customer service
* Contract service
* Tariff service
* Options service
* Role service

All services have interfaces and it’s implementation.

In service layer take place DTO – Entity convention.

All services are extended by GenericService.

## List of methods

**GenericService**

|  |  |
| --- | --- |
| Method | Description |
| loadByKey | Load entity from database by id |
| remove | Remove entity from database |
| loadAll | Load all entities from database |

**ContractService**

|  |  |
| --- | --- |
| Method | Description |
| setBlock | Set block level for contract |
| updateContract | Update contract with new option and tariffs |
| addNew | Add new entity to database |

**CustomerService**

|  |  |
| --- | --- |
| Method | Description |
| findByEmail | Find a customer by email |
| add | Add new entity to database |

**OptionService**

|  |  |
| --- | --- |
| Method | Description |
| getOptionsOfTariff | Find a list of options available for a tariff |
| addNew | Add new entity to database |

**TariffService**

|  |  |
| --- | --- |
| Method | Description |
| update | Changes fields and options for a tariff |
| addNew | Add new entity to database |

**RoleService (Not extended by GenericService)**

|  |  |
| --- | --- |
| Method | Description |
| findByName | Change password by old password |
| getRoleById | Change password by temporary code |

# Entities and repository

## Entities

Entities package contains 5 classes

* Contract
* Customer
* Option
* Role
* Tariff

**Customer** – service user.

|  |  |
| --- | --- |
| Field | Description |
| id | ID of customer. Auto generated by database |
| name | First name of user |
| surname | Last name of user |
| email | Email of user |
| password | Password of user. Hashed with BCrypt |
| dateOfBirth | Birthday of customer |
| passportNumber | Passport number of customer. Must be unique in system |
| passportData | Additional passport data |
| address | Address of user |
| isBlocked | Define block level of user |
| contracts | Set of user’s contracts |
| roles | Set of user’s roles |

**Contract** – represents contract. Has unique phone numbers. Has only one customer.

|  |  |
| --- | --- |
| Field | Description |
| id | ID of contract. Auto generated by database |
| number | Contract phone number. Must be unique |
| isBlocked | Define block level of contract |
| balance | Balance of contract |
| customer | Contract’s customer |
| tariff | Used tariff of contract |
| usedOptions | Set of used options |

**Option –** represents options. Have one or more contracts.

|  |  |
| --- | --- |
| Field | Description |
| id | ID of option. Auto generated by database |
| name | Name of option. Must be unique |
| cost | Cost per month of option |
| connectCost | Cost of option switching-on |
| description | Description of option |
| required | Set of options. Define “required from” relationships |
| forbidden | Set of options. Define “incompatible” relationships |
| possibleTariffsOfOption | Set of tariffs, that use this option |
| contractsThoseUseOption | Set of contracts, that use this option |

**Tariff** – represents tariffs. Has zero of more options.

|  |  |
| --- | --- |
| Field | Description |
| id | ID of tariff. Auto generated by database |
| name | Name of tariff. Must be unique |
| cost | Cost per month of tariff |
| description | Description of tariff |
| possibleOptions | Set of options. Define available options for tariff. |

**Role** – represents roles.

|  |  |
| --- | --- |
| Field | Description |
| id | ID of role. Auto generated by database |
| roleName | Name of role. Must be unique |

## Repository

Application created with Spring framework and JPA Hibernate implementation.

Data source defined by JNDI on server.

Full repository package represented as interfaces with implementation of interfaces. All interfaces extended by GenericDao.

Some interfaces have additional methods.

**GenericDao**

* add
* loadAll
* loadByKey
* remove
* update

**ContractDao**

No specific methods

**CustomerDao**

* findByEmail

**OptionDao**

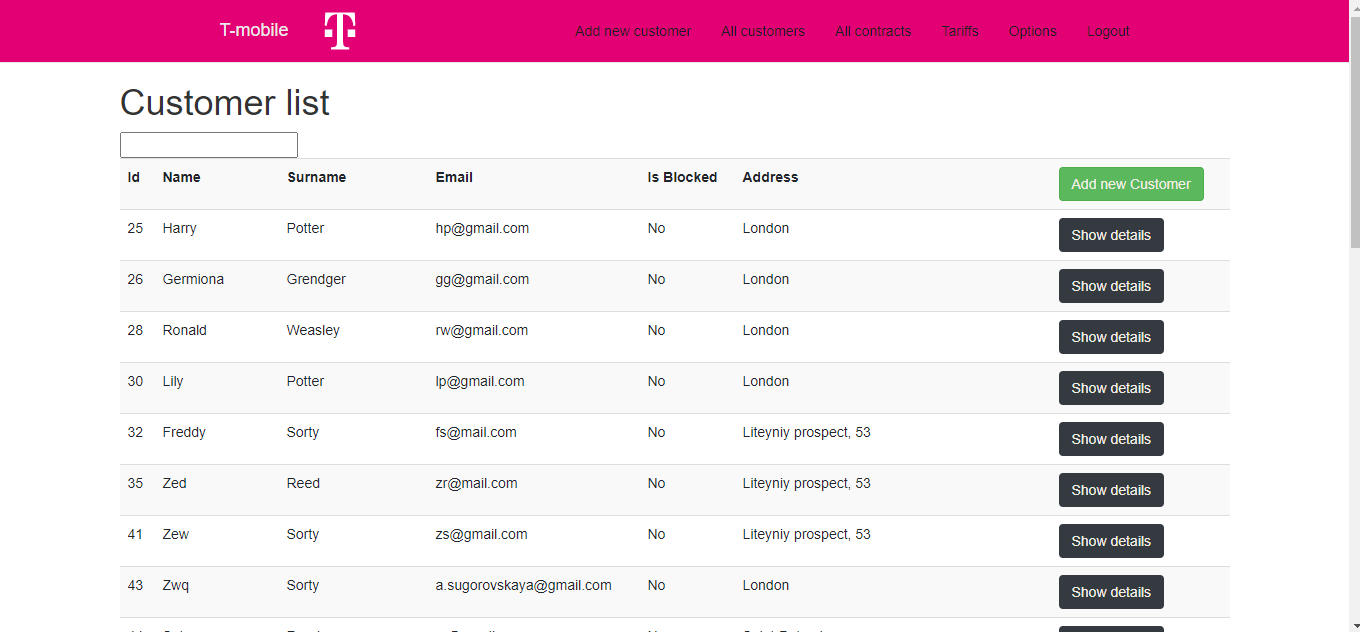
* getOptionsForTariff

**RoleDao**

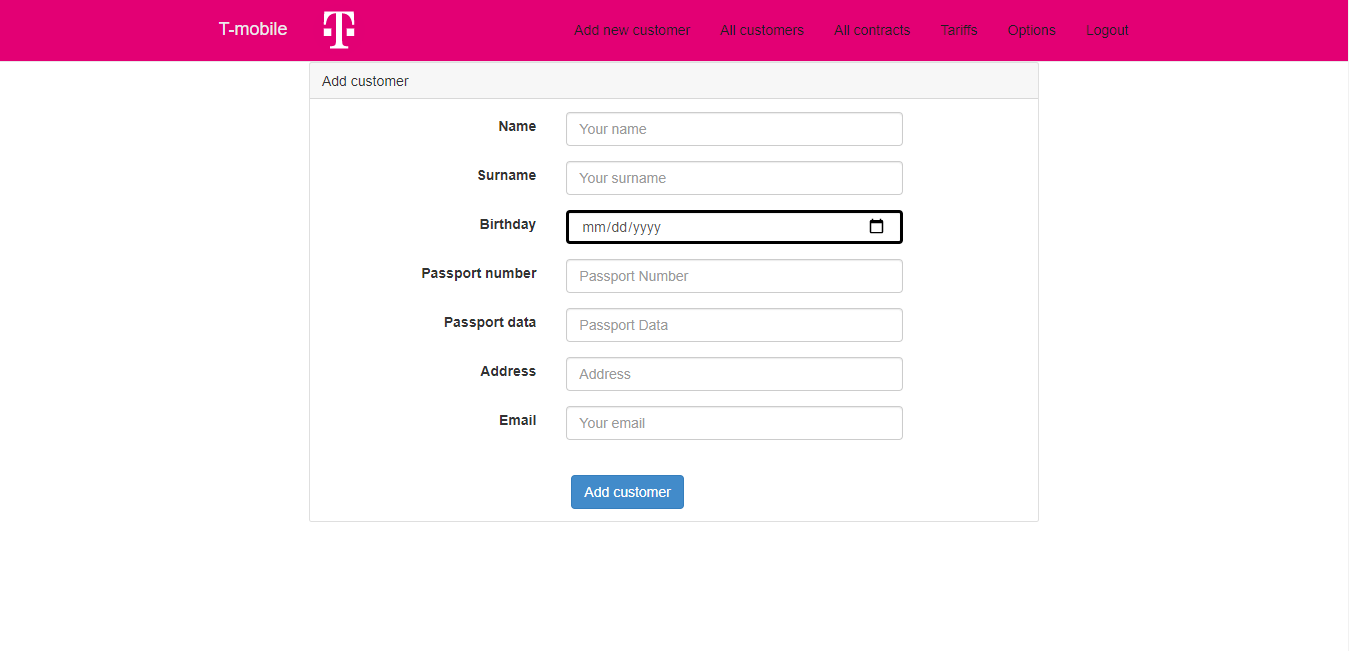
* getRoleById
* findByName

# Screenshots

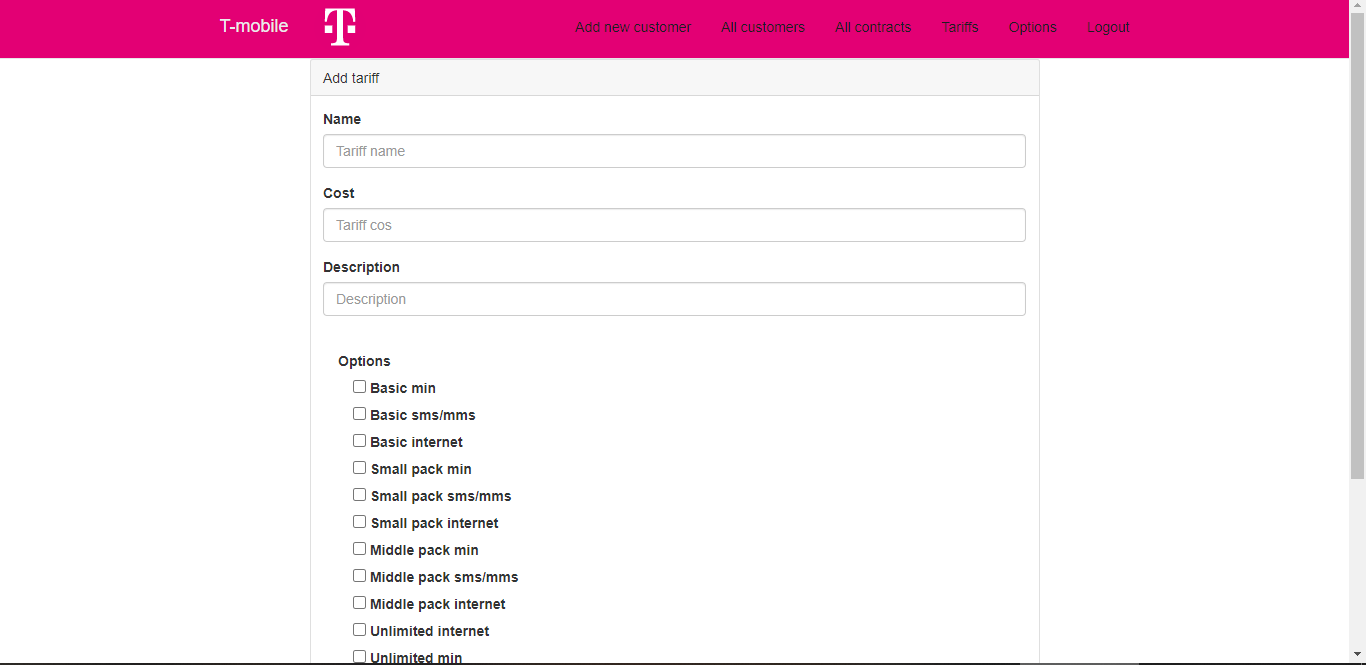
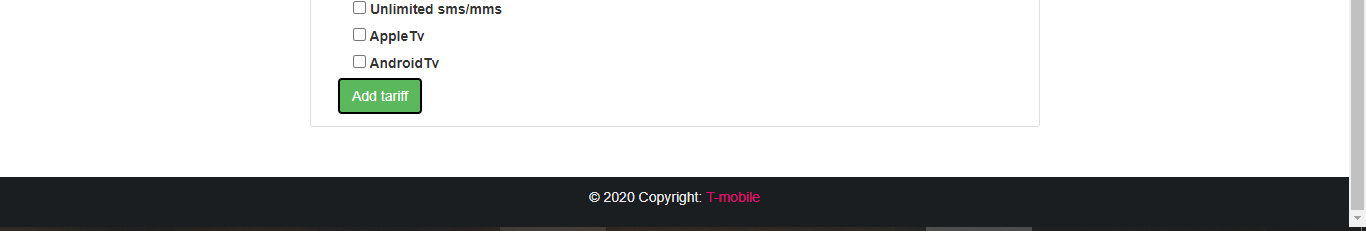
Customers table



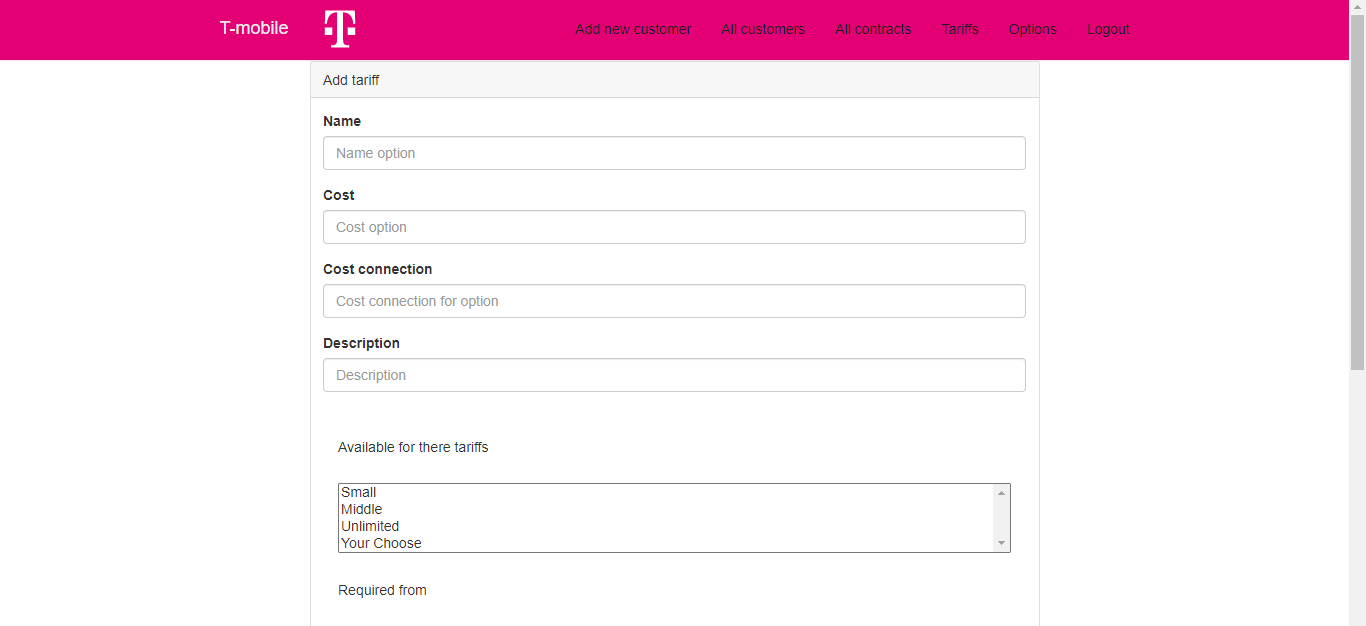
Customer register

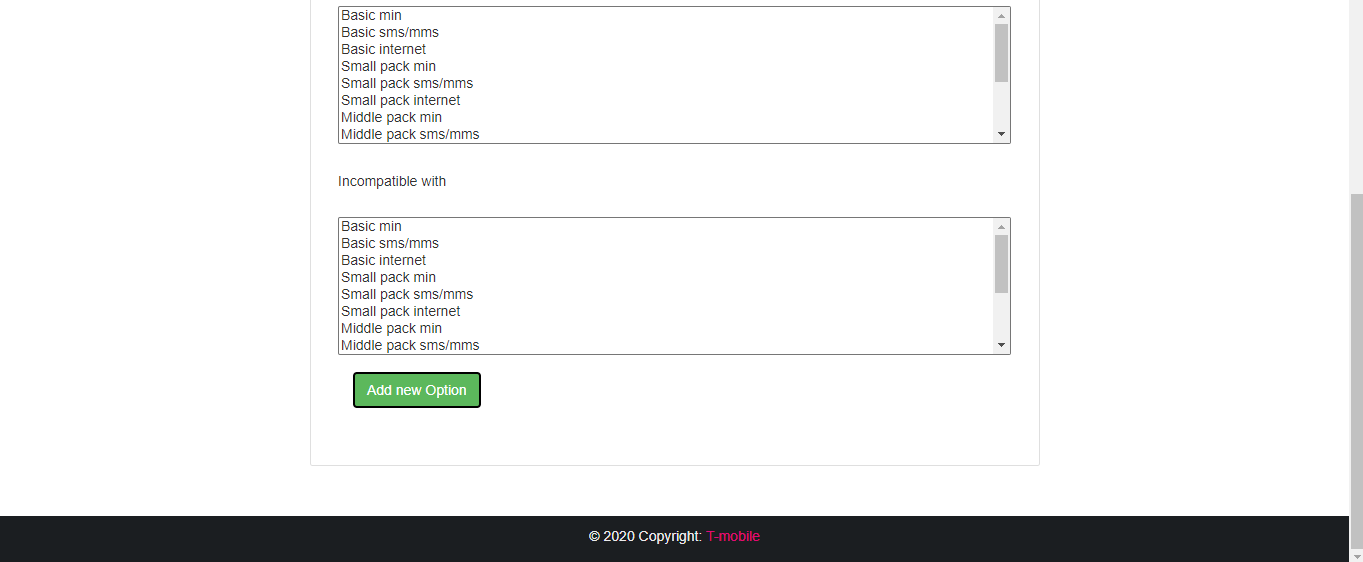


Tariff adding

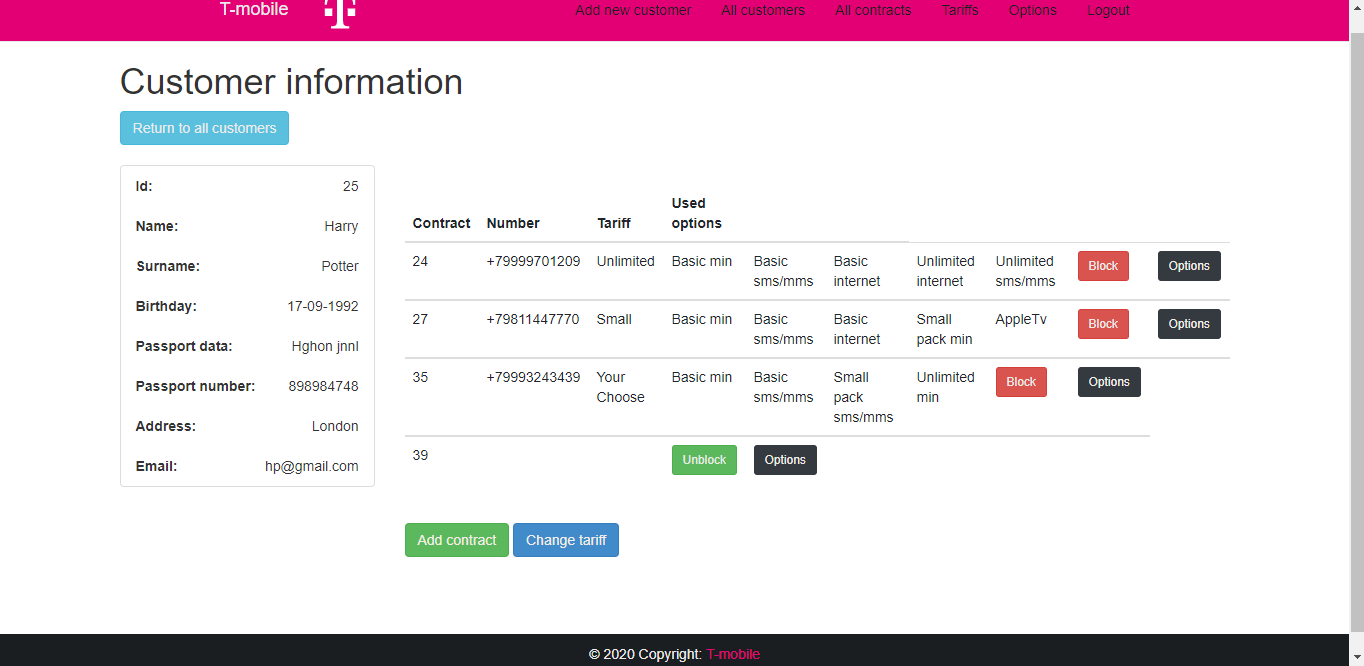


Option adding





Customer page



Customer page (customer view)

