# Technical description


Web-portal for mobile network operator «K-Mobile»

Version <2.0>

Artyom Karnov
T-systems

Saint-Petersburg
18.10.2016

T-Systems

# Content

# Revision history

| Date | Version | Description | Author |
|---|---|---|---|
| 15.09.2016 | 1.0 | Creating document | Artyom Karnov |
| 18.10.2015 | 2.0 | Updating document | Artyom Karnov |

# 1 Task

To develop web-application for mobile network operator which will be modeling information system of K-Mobile company. The application have to perform the required user's cases.

Users cases :

- For clients
  - To view contracts;
  - To view contracts options;
  - To change contract, contracts options;
  - To block/unblock number;
- For managers
  - To create new contracts, contract options.
  - To view all clients and their personal data;
  - To block/unblock number;
  - To find clients
  - To change client data;
  - Option control;

Additionally to develop co-application for report generating from main application.

# 2 Project goals

- The robust, useful and reliable system for mobile operator
- Cohesive data model
- Intuitive, user-friendly interface
- Scalable architecture
- Separate access to different system’s part

# 3 Application description

Web-application has two type of user: clients and managers. Clients could view their connected contracts, tariffs, options, add new tariff options, block and unblock numbers. Managers could add new tariff and their options, change them, correct user-list, clarify joint and incompatible options. There is authentication mechanism in system that controls access to portal. Each user in application has access level that display what information he could get and what couldn't. Data of users and their options store in reliable data base.

# 4 Used technologies

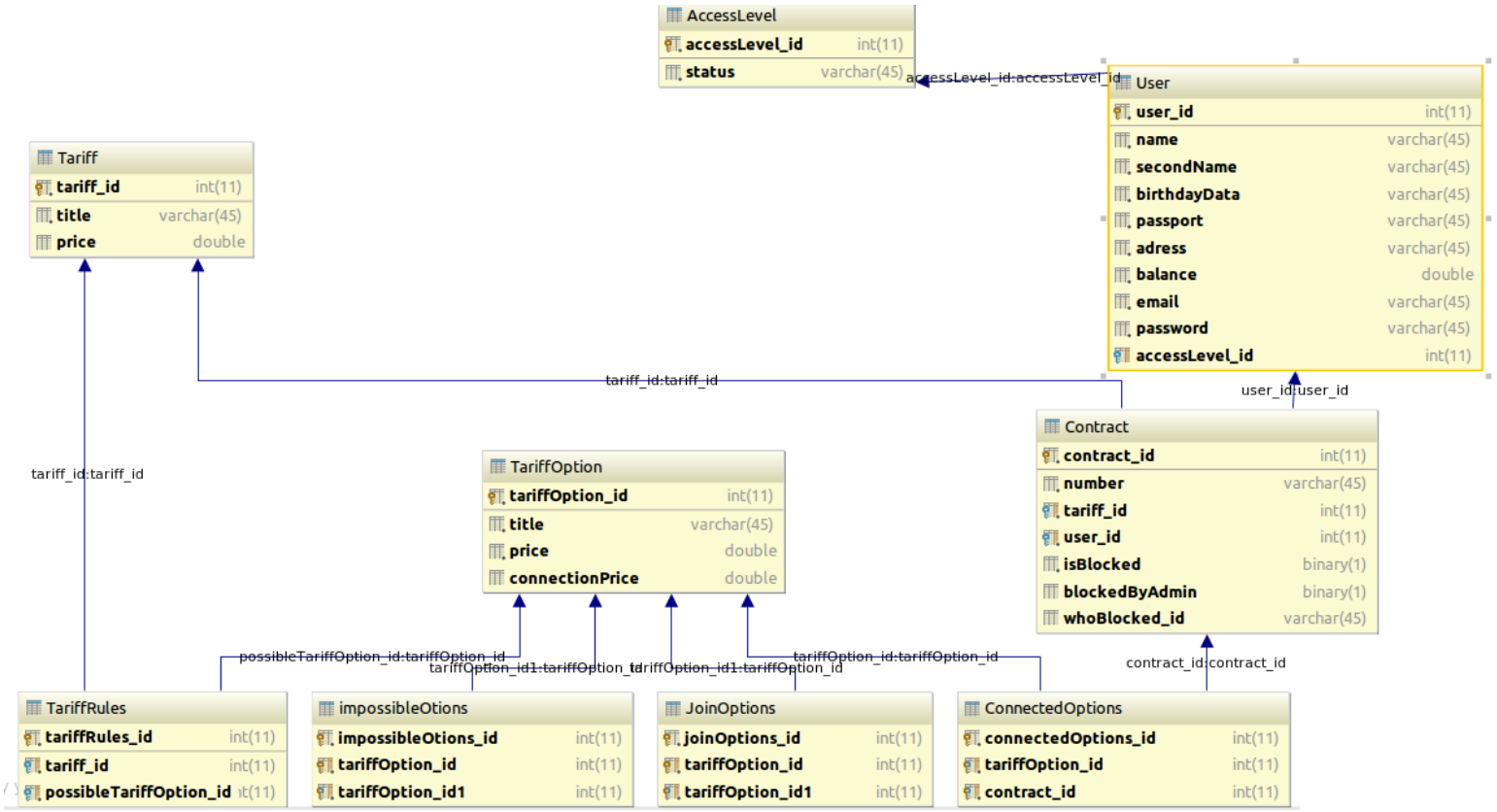
Instruments

- DataGrip 2016.2.1
- IDE – Intellij IDEA 2016.2.2
- Maven 3.1
- SQL WorkBench 6.3

Technologies

- Bootstrap 3.0
- DB – MySQL 5.8
- EJB 3
- ElasticSearch 0.90.5
- GSON
- ItxtPDF 4.1
- Hibernate 5.2.2
- Java 8
- JavaScript
- JSF 2.2.13
- JSP 2.1
- JUnit 2.12
- Log4j2 2.0.2
- Mockito 2.0.2
- Wildfly 10.1.0
- Postfix 3.1.1
- REST
- SmsC 1.0
- SonarQube 5.3
- Spring 4.3.3
- Spring security 4.1.3

# 5 Data base diagram



# 6 System architecture

## 6.1 System infrastructure

- Front-end (browser presentation level):

  1) Web-page structure - *HTML*
  2) Page-design - *CSS*
  3) Dynamic content – *JavaScript* (e.g. *AJAX*)

- Back-end (server based level):

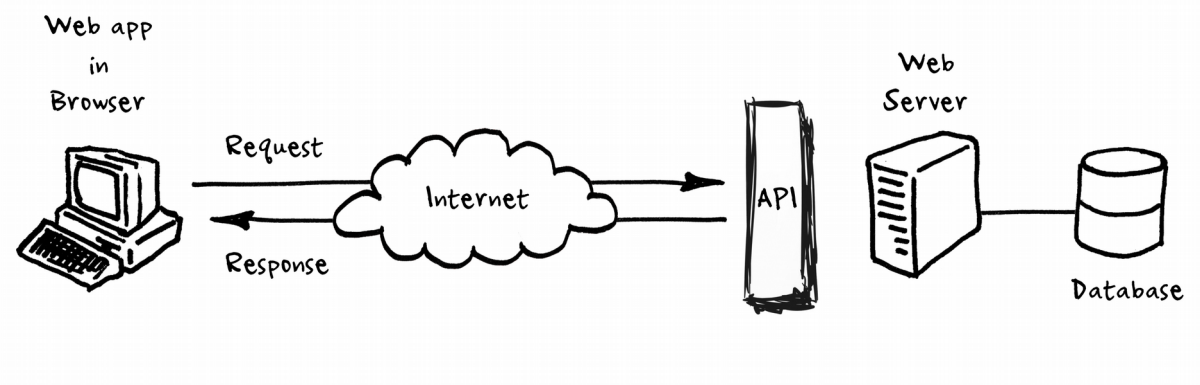
  1) Application server -*WildFly*
  2) Database – *MySQL*
  3) ORM - *Hibernate*
  4) Server logic - *Spring Framework*

- Rest client (transaction server level):

  1) Web-pages - *JSF*
  2) Server logic - *EJB*

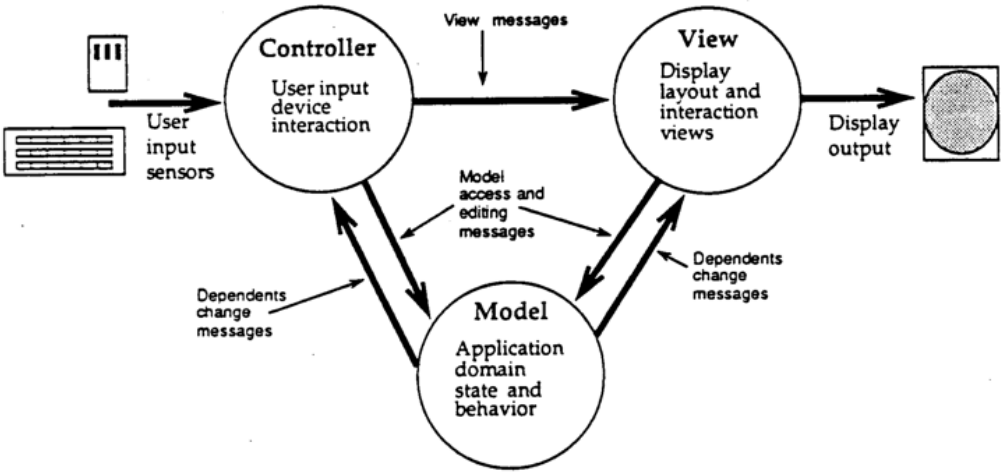
## 6.2 System ideology

Ideology of system – client-server architecture. There are server based part (back-end) and client-oriented (front-end). (Pic. #1.1)



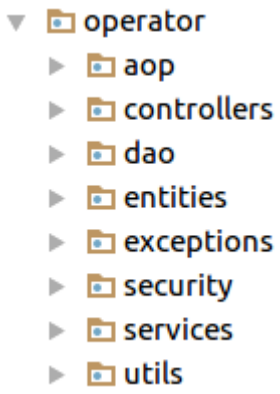

Picture 1.1 – System ideology

Architecture of server-based part presented by MVC - design pattern. (Pic. #1.2)

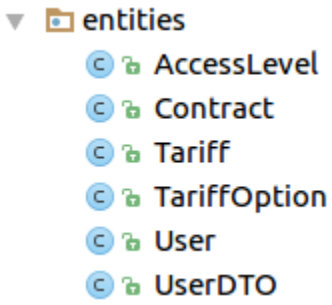



Picture 1.2 – System architecture

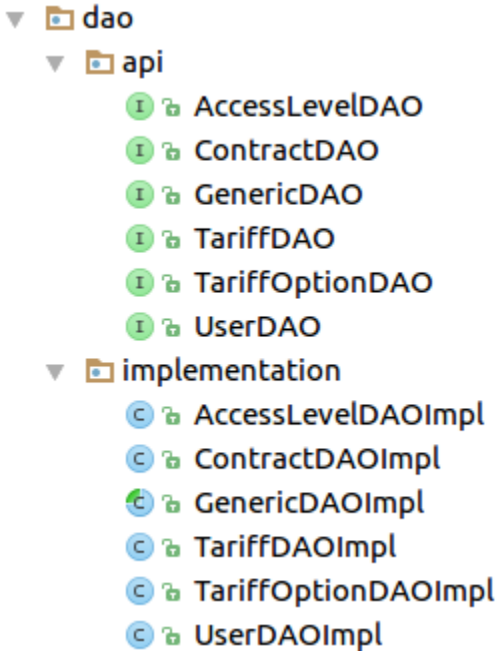
According MVC-pattern application has next structure:

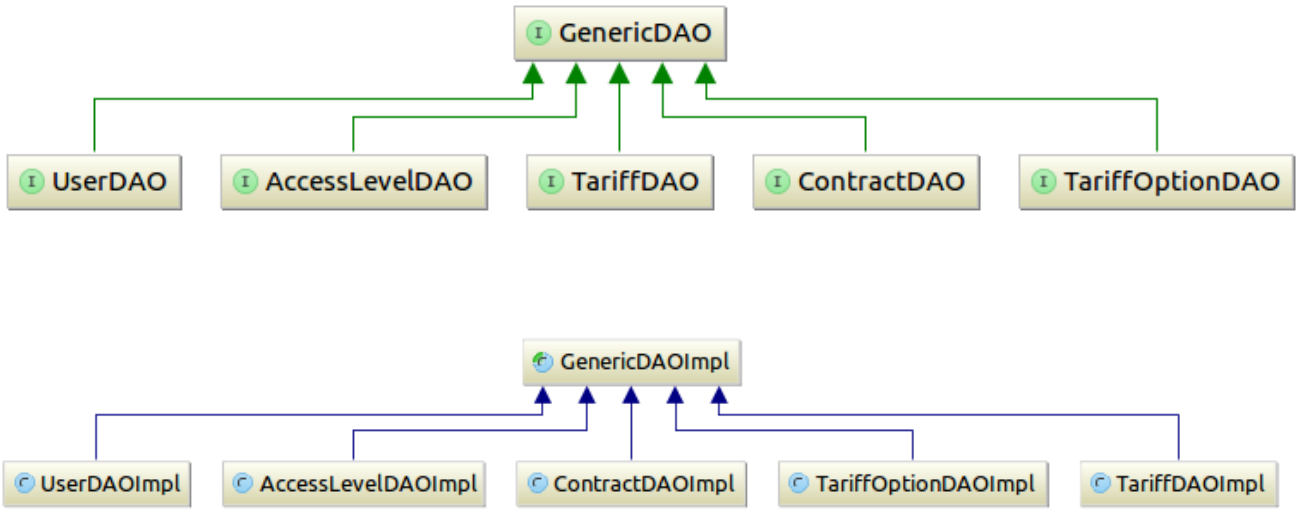



Model level:

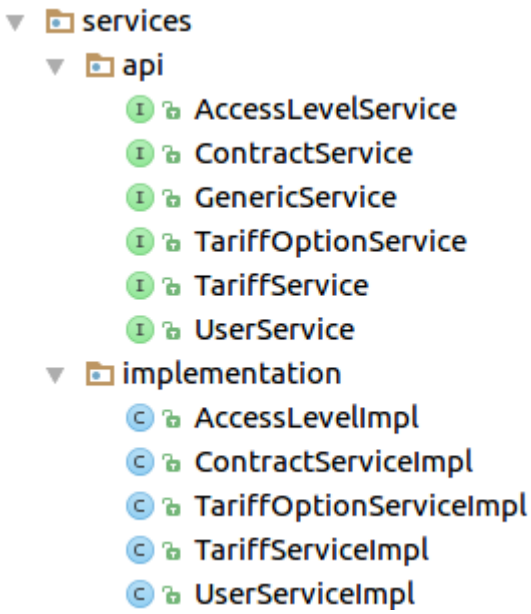


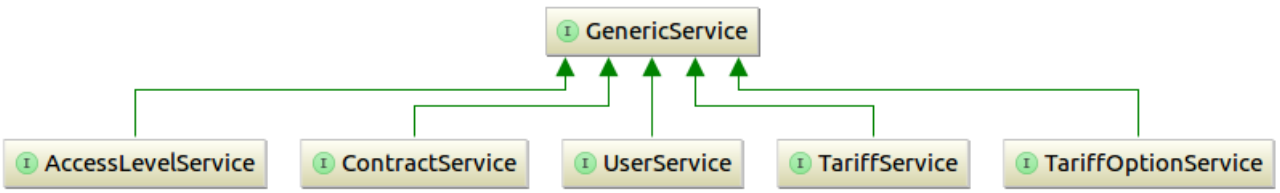

Model-service level:

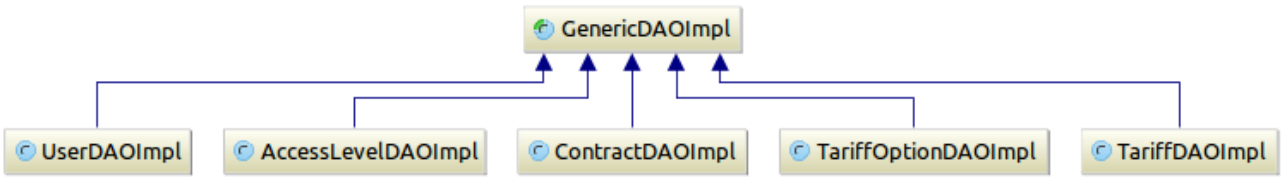




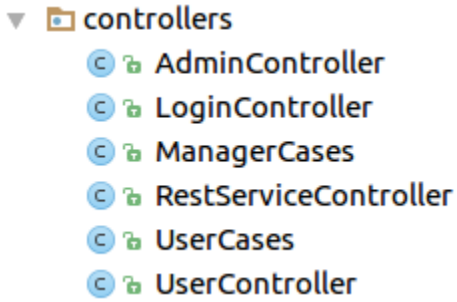

Service level:



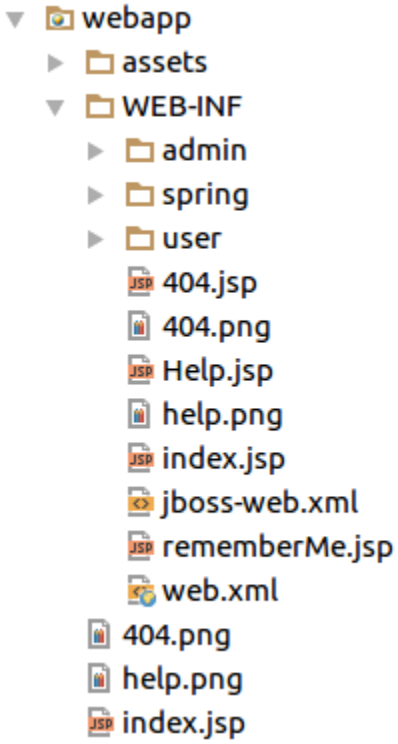





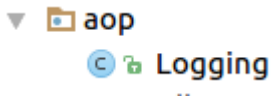

View-service level:



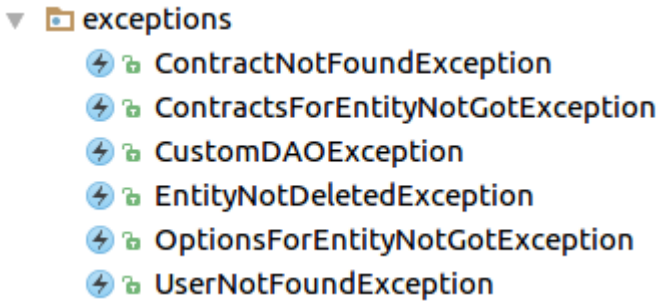

View level:



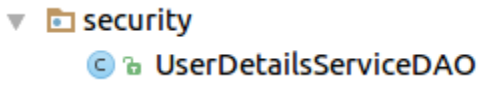
For logging is using AOP – programming style:



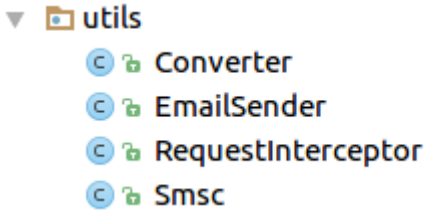

Exception handling presented:




Security mechanisms:




Support utilities:

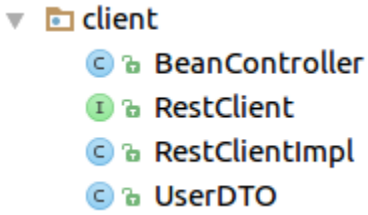



The rest client integrated in main application interface and built on EJB and JSF technologies.

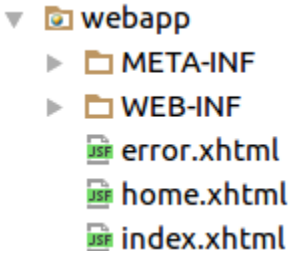
Architecture:

Bean controller dispatches all request from view, rest client perform operation, user data transfer object transports information between applications.

Business logic:




View:



# 7 User cases

- **Log in**

Each registered user has login and password (store in data base). After entering correct personal data (email and password) users get access to their scope. Managers could register new clients, contracts, add, edit and remove new tariff, options.  Users could change their data and observe services.

- **Password remembering**

Users could change forgotten password. It will be send by SMS and email. It possible after clicking on forgot password button and entering correct personal data .

- **User cases**

Users (according their access level ) could perform actions like tariff changing and etc. All activities could perform on user-friendly interface with tool tips. Important actions change information stored in data base. Actions store in tier level logs.

*inf.log* – information about normal work flow

*debug.log* – information about system services

*error.log* – information about problems

- **Report generator**

Logically build in manager's interface, but in fact it is separate application. After user's request with tariff title it generates pdf report about clients who have adjusted tariff.

# 8 Additional features

1. Password generating

   After getting user personal information system generates safe-password for users.

2. Password encrypting

   All user's passwords store in data base in encrypted state. Only hash stores. This hash made by md5 + secret phrase algorithm encrypting.

3. SMS notification

   Generated user's passwords send via sms to adjusted mobile phone. Also sms-service works on restoring forgotten password cases.

4. Email notification

   Generated user's passwords send via email to adjusted email address. Also email-service works on restoring forgotten password cases.

5. Selenium auto tests

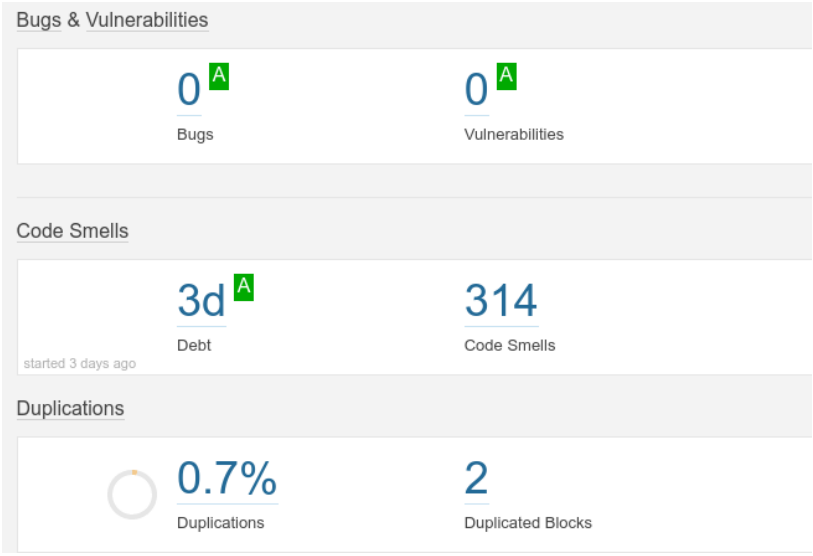
   For UI testing was used Selenium IDE software.

6. Site search engine

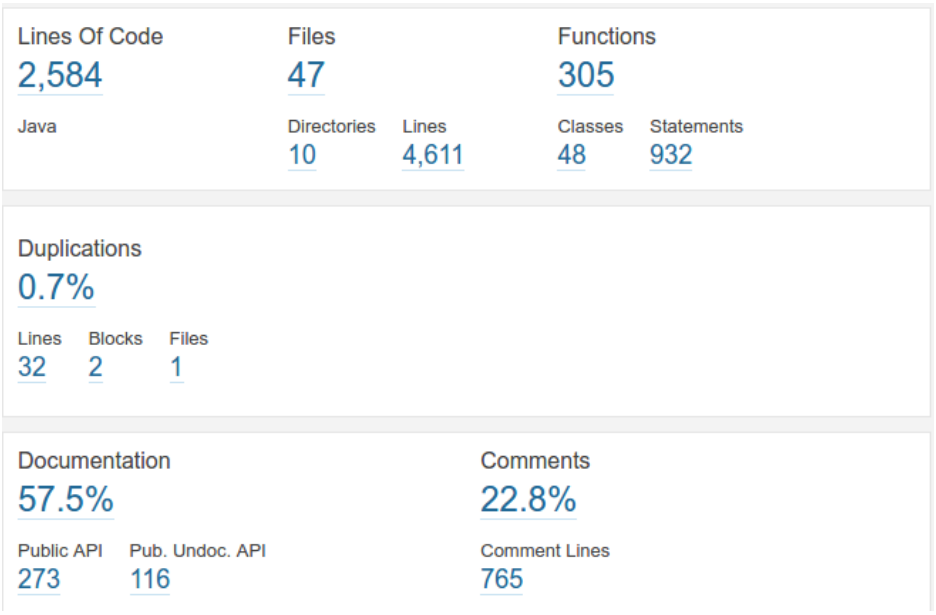
   ElasticSearch cluster software environment for outer-Internet independent search processes.

# 9 Code quality

SonarQube  measures presented bellow:



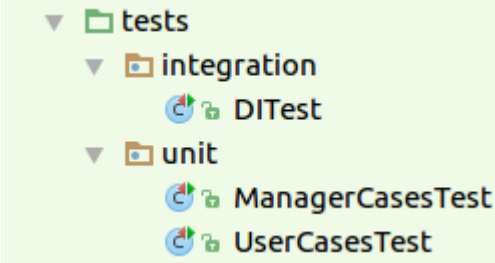

Picture 1.3 – SonarQube report



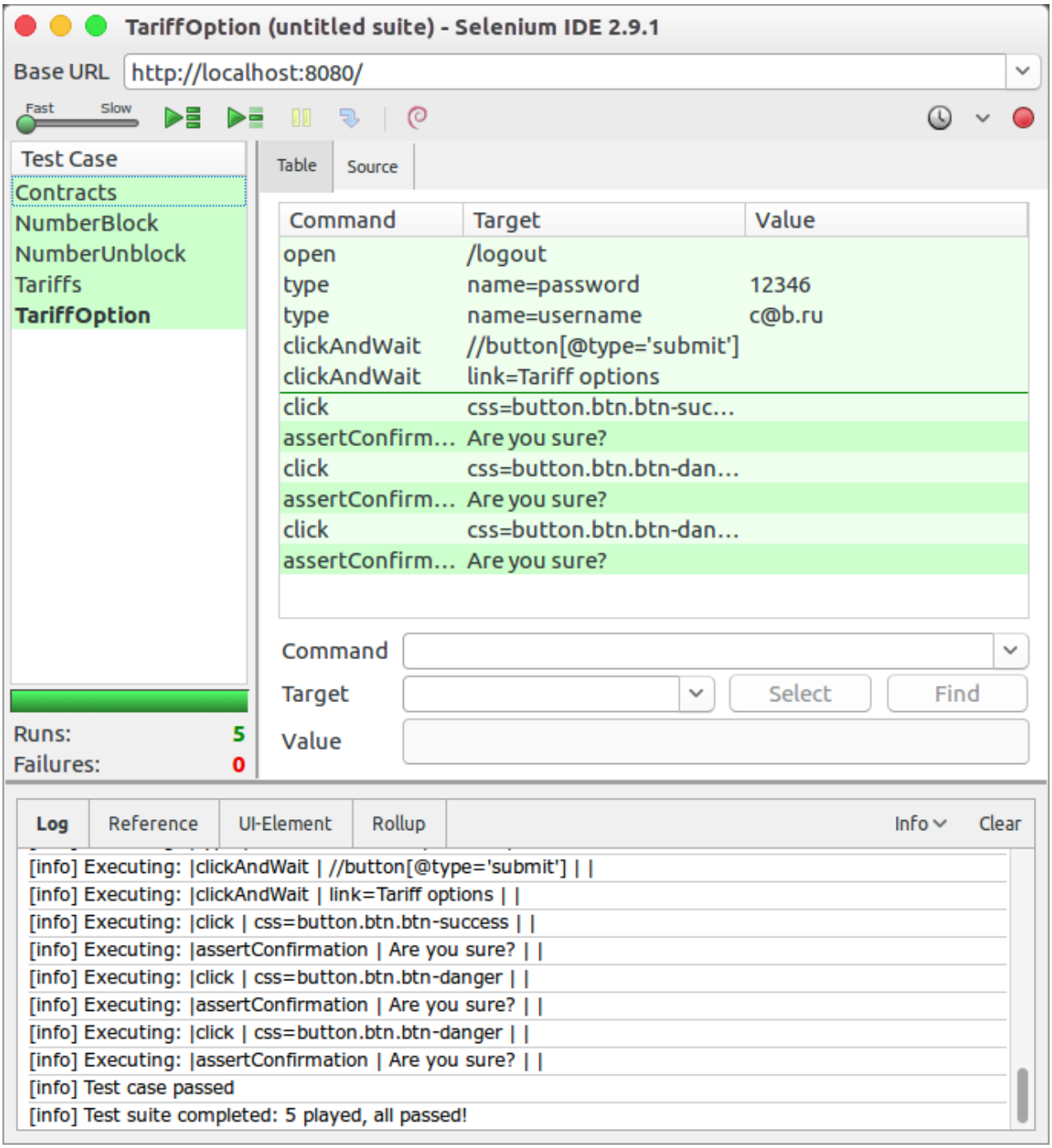

Picture 1.3 – SonarQube code analysis

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running services.TariffOptionServiceImplTest
Tests run: 10, Failures: 0, Errors: 0, Skipped: 10, Time elapsed: 0.106 sec
Running unit.controllers.ManagerCasesTest
Tests run: 13, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.648 sec
Running unit.controllers.UserCasesTest
Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.025 sec

Results :

Tests run: 28, Failures: 0, Errors: 0, Skipped: 10

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 9.820s
[INFO] Finished at: Tue Oct 11 00:02:16 MSK 2016
[INFO] Final Memory: 10M/113M
[INFO] ------------------------------------------------------------------------
```

Picture 1.4 – Test-results




Picture 1.5 – Test structure




Picture 1.6 – Selenium test results

# 10 Build and deploy

**Step 1:** Start Wildfly application server

*./standalone.sh*

***Step 2:*** Start *ElasticSearch* work cluster

*/etc/init.d/elasticsearch start*

**Step 3:** Main application installation (in *OperatorMainApp* directory)

*mvn clean install*

**Step 4:** Main application deploy (in *OperatorMainApp* directory)

*mvn wildfly:deploy*

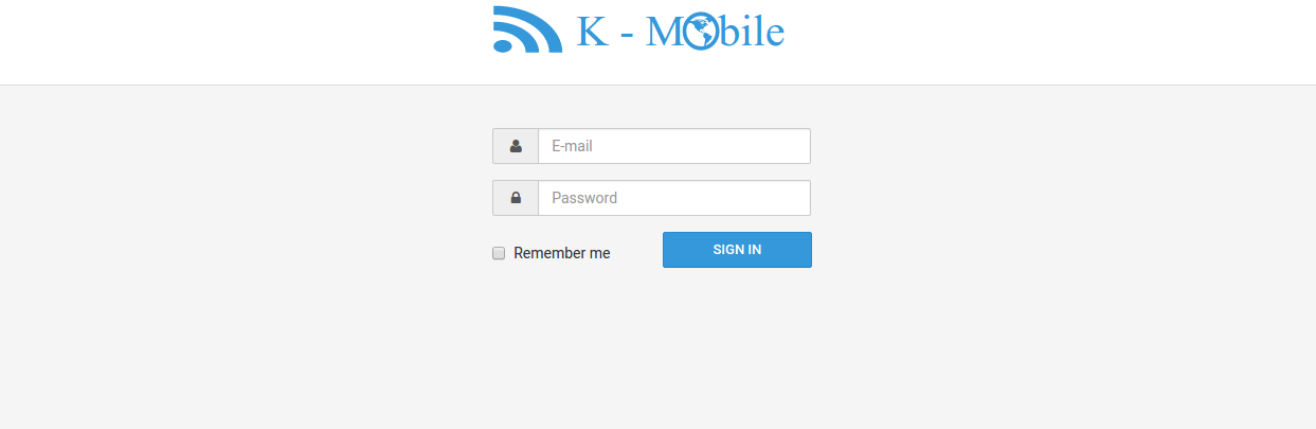
**Step 5:** REST client installation (in *RESTClient* directory)

*mvn clean install*

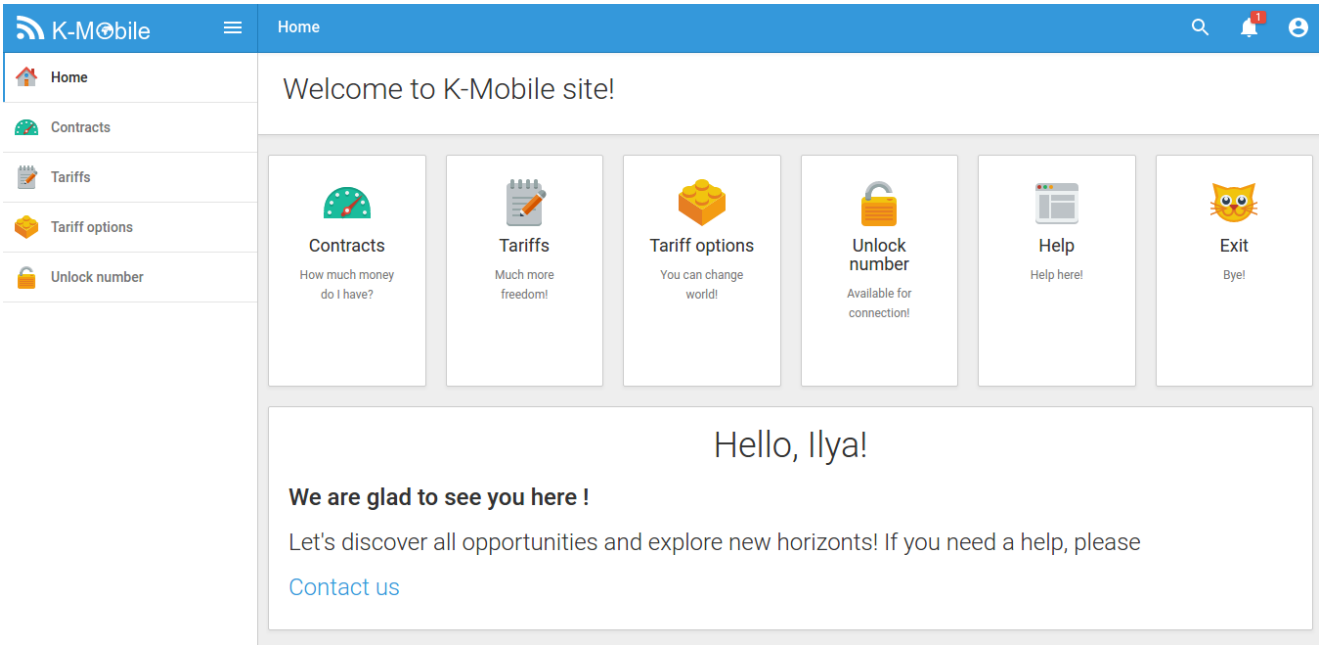
**Step 6:** REST client deploy (in *RESTClient* directory)

*mvn wildfly:deploy*

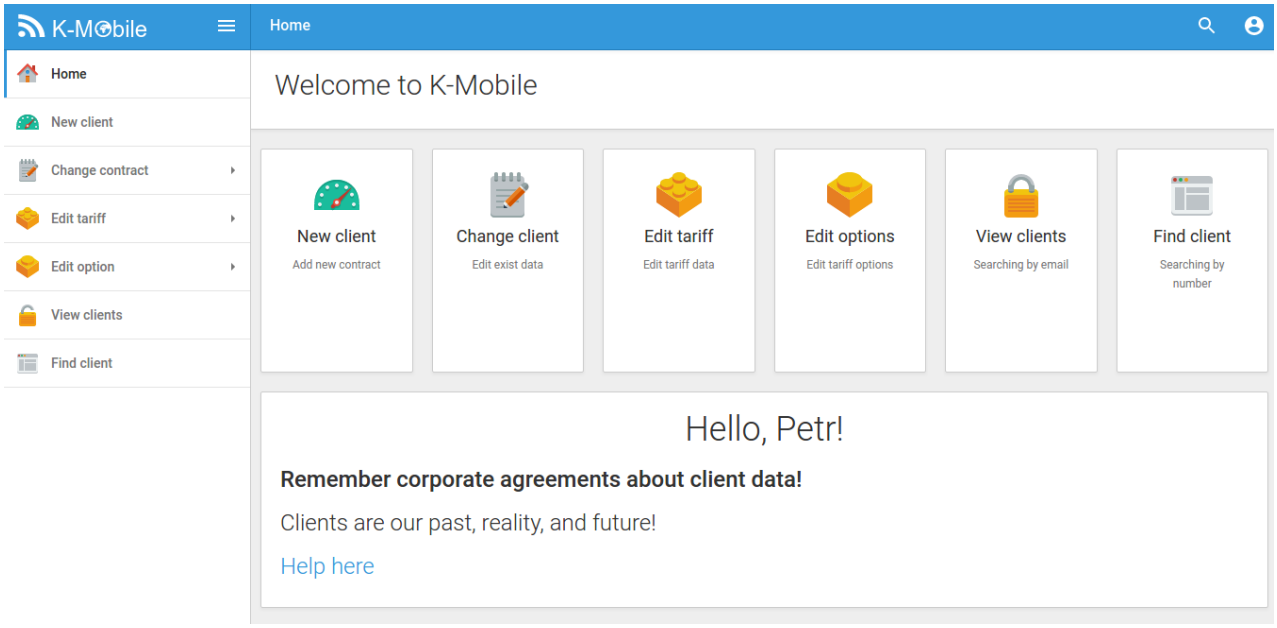
# 11 GUI



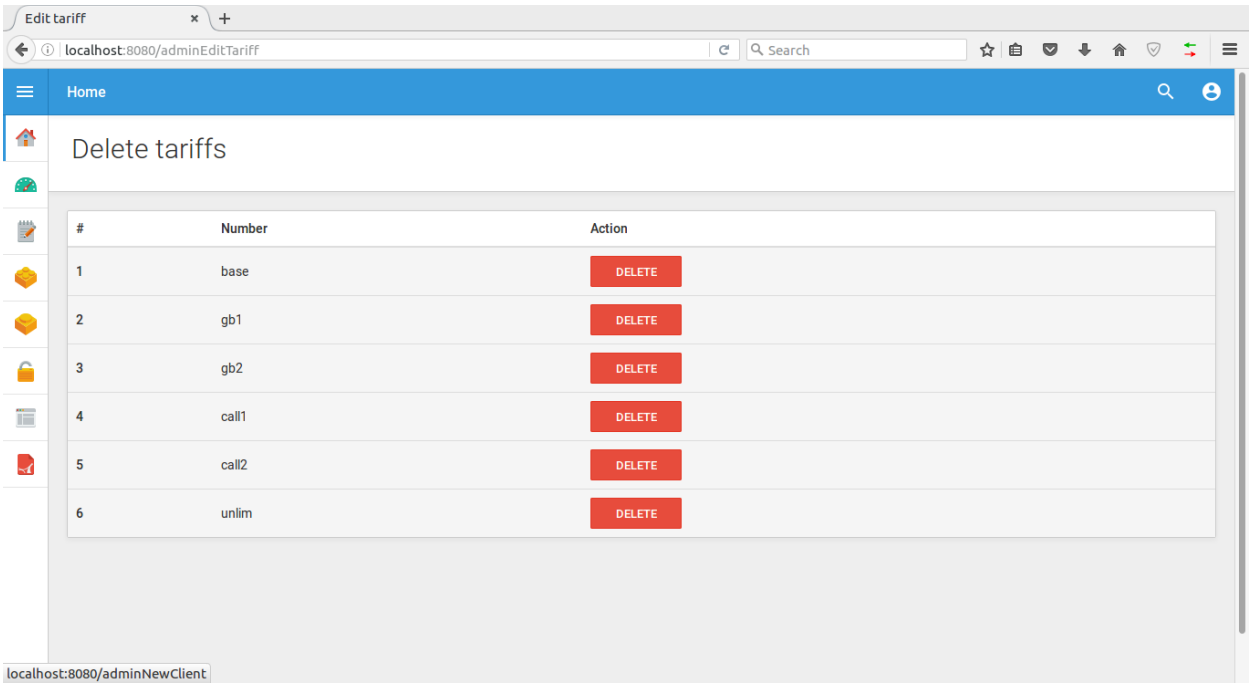

Picture 1.7 – Login page




Picture 1.8 – User's main page




Picture 1.9 – Manager's main page




Picture 1.10 – Manager's delete tariff page

# 12 Glossary

**Aspect-oriented programming** (**AOP**) is a programming paradigm that aims to increase modularity by allowing the separation of cross-cutting concerns.

**Authorization** or **authorisation** is the function of specifying access rights to resources related to information security and computer security in general and to access control in particular.

**Authentication** is the act of confirming the truth of an attribute of a single piece of data (a datum) claimed true by an entity. In contrast with identification which refers to the act of stating or otherwise indicating a claim purportedly attesting to a person or thing's identity, authentication is the process of actually confirming that identity.

**Login** refers to the process by which a user accesses to a computer system or other restricted area, or the credentials required during that process.

**Usability** is the ease of use and learn ability of a human-made object such as a tool or device

***User Friendly*** is a daily web comic about the staff of a small fictional Internet service provider, Columbia Internet. The strip's humor tends to be centered on technology jokes and geek humor.

# 13 Future improvement

I have 3 ways of improvement:

1) Optimization – improve performance by enhancement connection between level structures (service-view and etc.)

2) Adding new functionality. Two-factor authentication and etc.

3) Make scenarios for integration application into exists infrastructures. For future commercial expansion on market.