Technical Solution Description

Online Store

Author: Sergey Nikolaychuk

2020

1. Task

There are two parts in the project.

First Part is Creating an application that simulates the operation of an online store. The application must provide the following functionality:

For Employers:

- Editing status of any order
- Add articles
- Creating and managing categories of catalog

For clients:

- View and filter the catalog
- Manage user’s own profile(information, address, password)
- Make an order, check history of orders

2. Application description

The main goal is to create a multi-user client-server application. All data are stored on a server side. Each client may load some data. After each modification operation, the data must be synchronized with a server.

3. Technologies

IDE:

- IntelliJ IDEA 2020.2.3 (Ultimate Edition)

Project build management tool:

- Apache Maven 3.6.3

Application Server:

- Wildfly-21.0.0.Final

Servlet Container:

- Tomcat 9.0.331

Database:

- MySQL 8.0.20

  Git

Backend:

- Apache ActiveMQ 5.16.0
- Jackson 2.11.2
- JPA 2.1
- JSF 2.3
- JSP 2.3
- EJB
- Hibernate
- Lombock 1.18.12

- Log4J 1.2.3
- MapStruct 1.2.17
- Slf4j 1.7.21
- Spring Boot 2

Frontend:

- Bootstrap 4
- React
- DataTables 1.10.21
- HTML/CSS
- JavaScript
- Primefaces 8.0

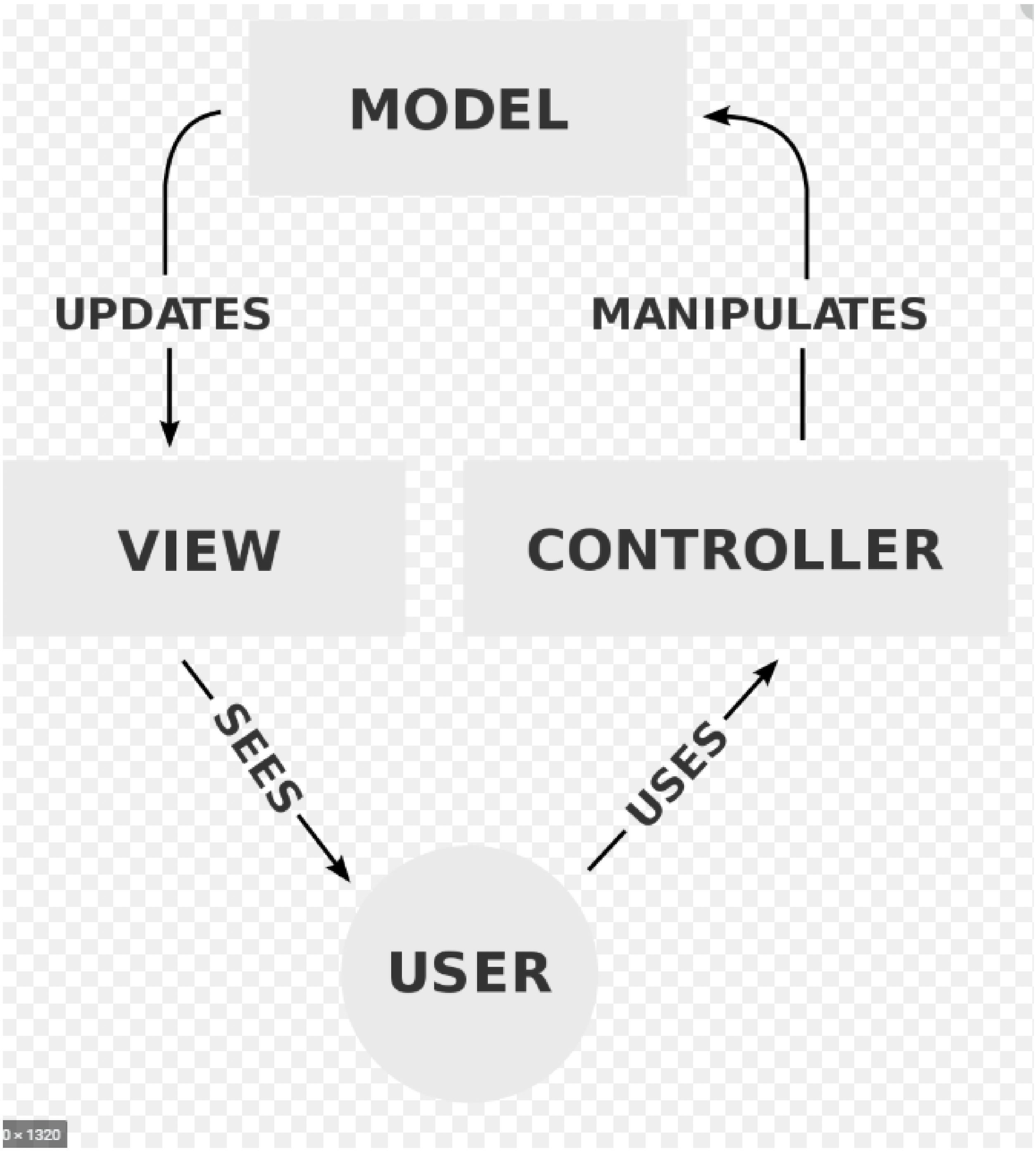
Testing instruments and libraries:

- JUnit 5
- Mockito 3.5.7
- JaCoCo 0.8.3
- SonarQube 8.5.0

4. Architecture

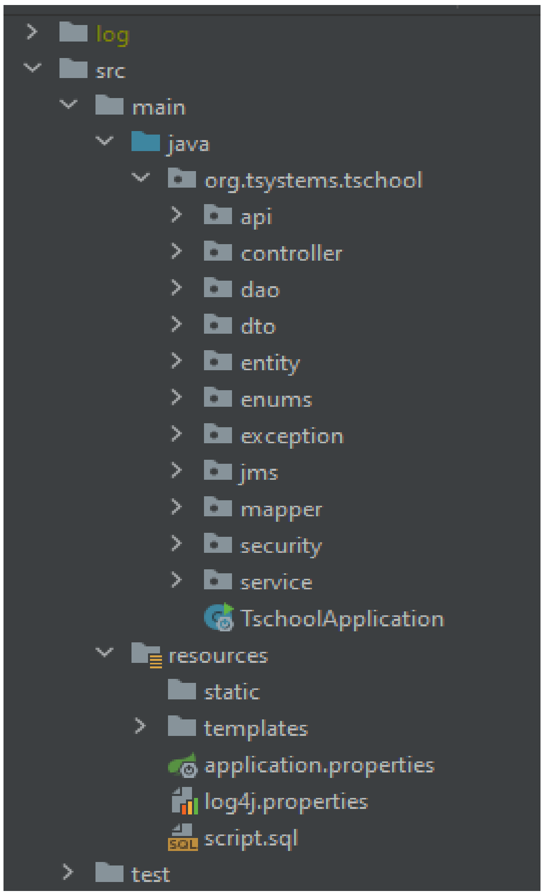
The application architecture is based on the implementation of the MVC design pattern.

- The Model contains only the pure application data, it contains no logic describing how to present the data to a user.
- The View presents the model’s data to the user.
- The Controller exists between the view and the model. It listens to events triggered by the view and executes the appropriate reaction to these events.



Picture 1 – MCV pattern visualization

The structure of the application is shown on the picture below:



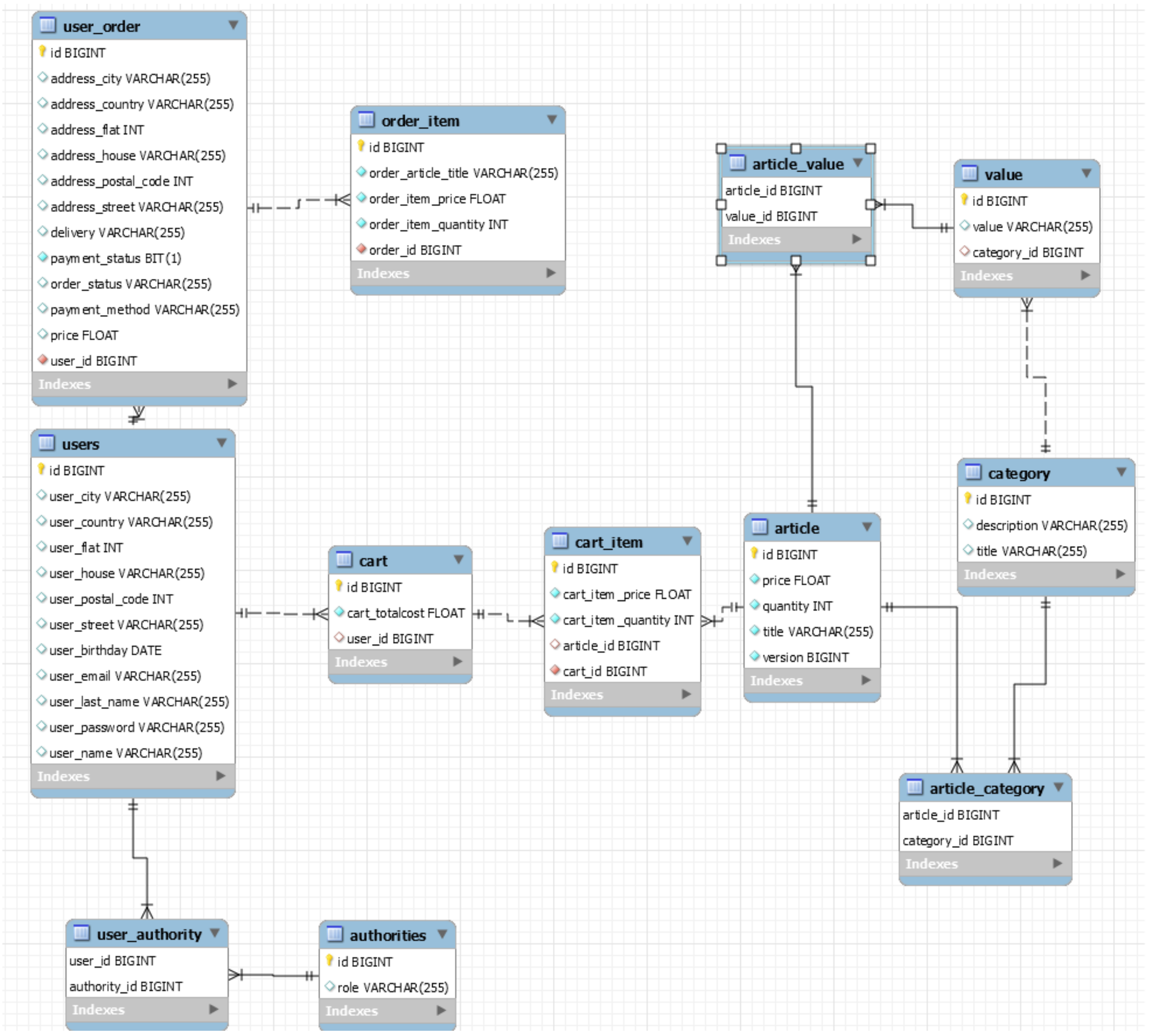
```
> log
v src
  v main
    v java
      v org.tsystems.tschool
        > api
        > controller
        > dao
        > dto
        > entity
        > enums
        > exception
        > jms
        > mapper
        > security
        > service
          TschoolApplication
    v resources
        static
      > templates
        application.properties
        log4j.properties
        script.sql
  > test
```

Picture 2 – Project structure

5. Database model

Table name	Description
users	Stores data about users such as client, employer and administrator. User's password is stored in encrypted form.
authorities	Stores data about users' roles.
user_order	Stores data about user's order. User can have many orders. Every order should have user, address, payment

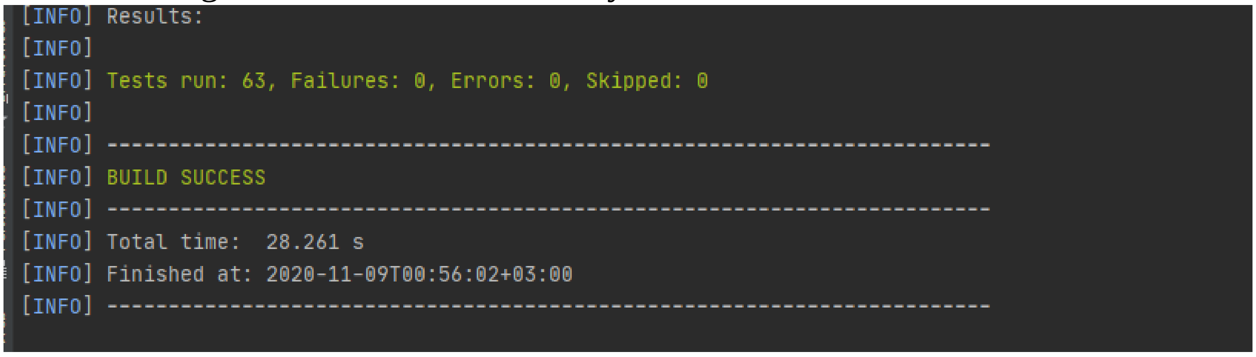
	method, delivery method, payment status, price.
order_item	Stores data about article in order. Every order item should have order, article’s title, quantity, price
cart	Stores data about user’s cart. Every cart should have user and it’s total price.
cart_item	Stores data about article in cart. Every cart item should have order, article, quantity, price
article	Stores data about article. Every article should have title, price, quantity,
value	Stores data about value. Every value should have title and category.
category	Stores data about category. Every category should have title and description.



Picture 3 – EER model

6. Testing environment

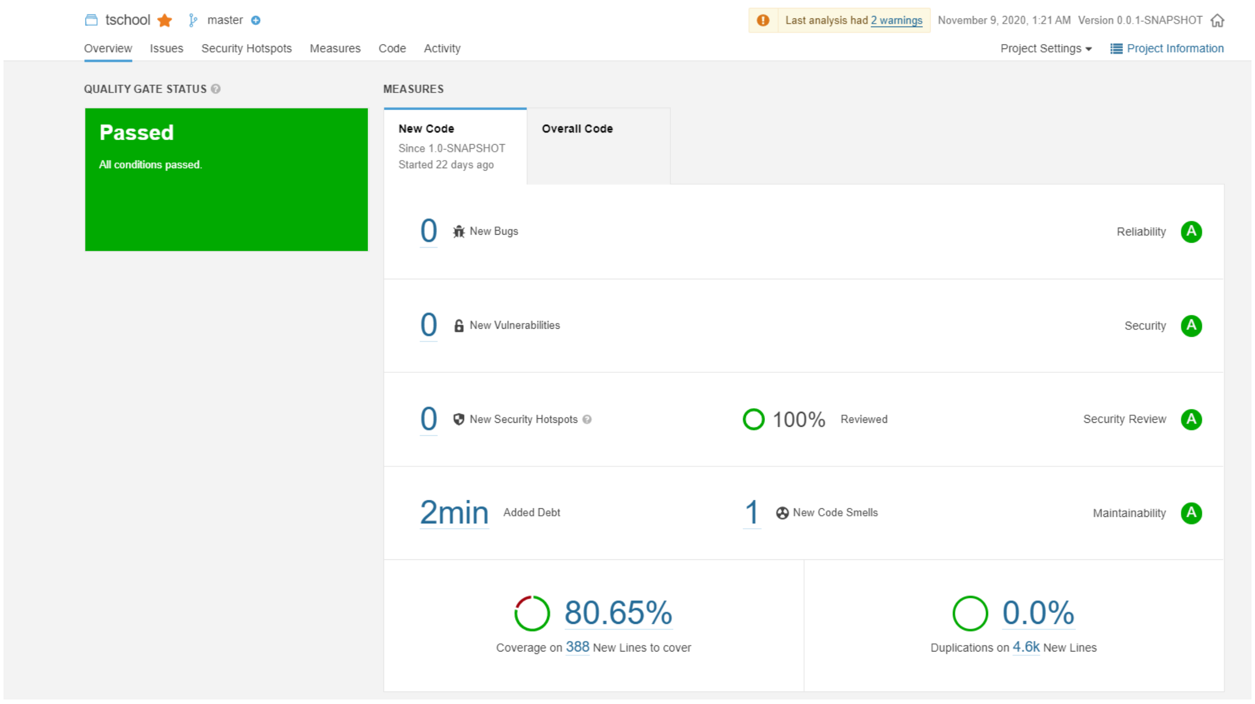
Unit-tests are written for business logic using Junit5 and Mockito. The coverage is 80% for the service layer.



```
[INFO] Results:
[INFO]
[INFO] Tests run: 63, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  28.261 s
[INFO] Finished at: 2020-11-09T00:56:02+03:00
[INFO] ------------------------------------------------------------------------
```

Picture 4 – Unit tests

7. SonarQube Report



tschool master

Last analysis had 2 warnings November 9, 2020, 1:21 AM Version 0.0.1-SNAPSHOT

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

QUALITY GATE STATUS

Passed

All conditions passed.

MEASURES

New Code
Since 1.0-SNAPSHOT
Started 22 days ago

Overall Code

0 New Bugs — Reliability A

0 New Vulnerabilities — Security A

0 New Security Hotspots — 100% Reviewed — Security Review A

2min Added Debt — 1 New Code Smells — Maintainability A

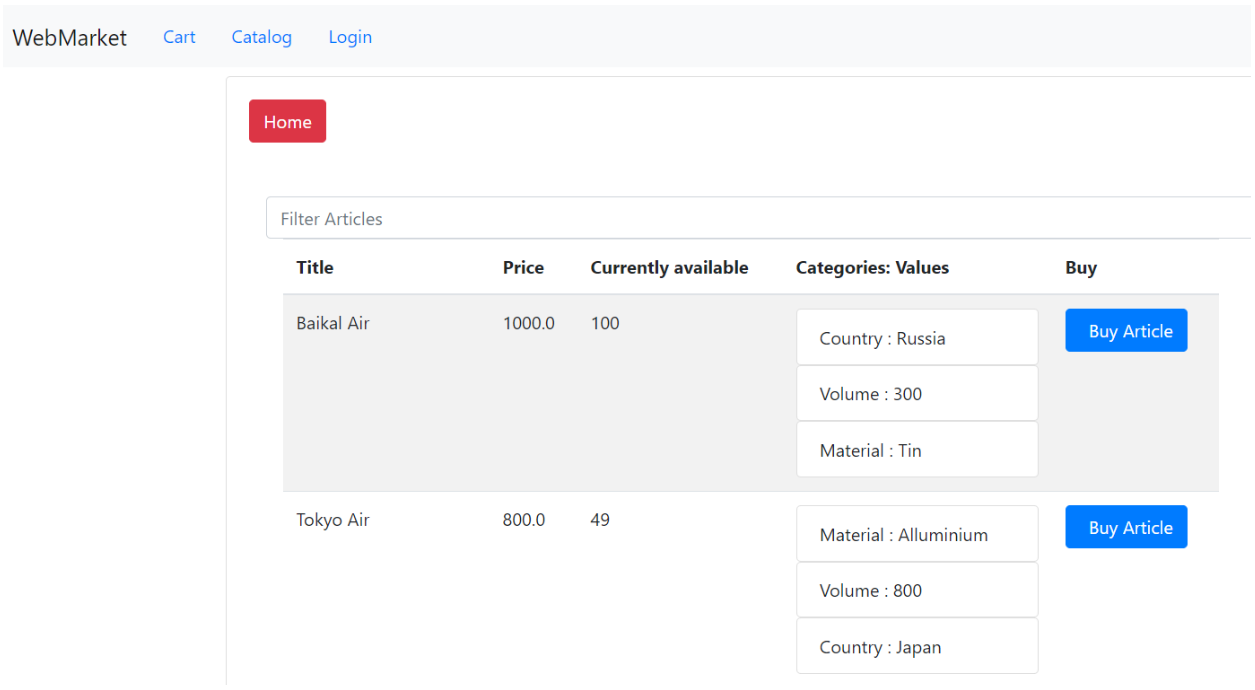
80.65% Coverage on 388 New Lines to cover

0.0% Duplications on 4.6k New Lines

Picture 5 – SonarQube report

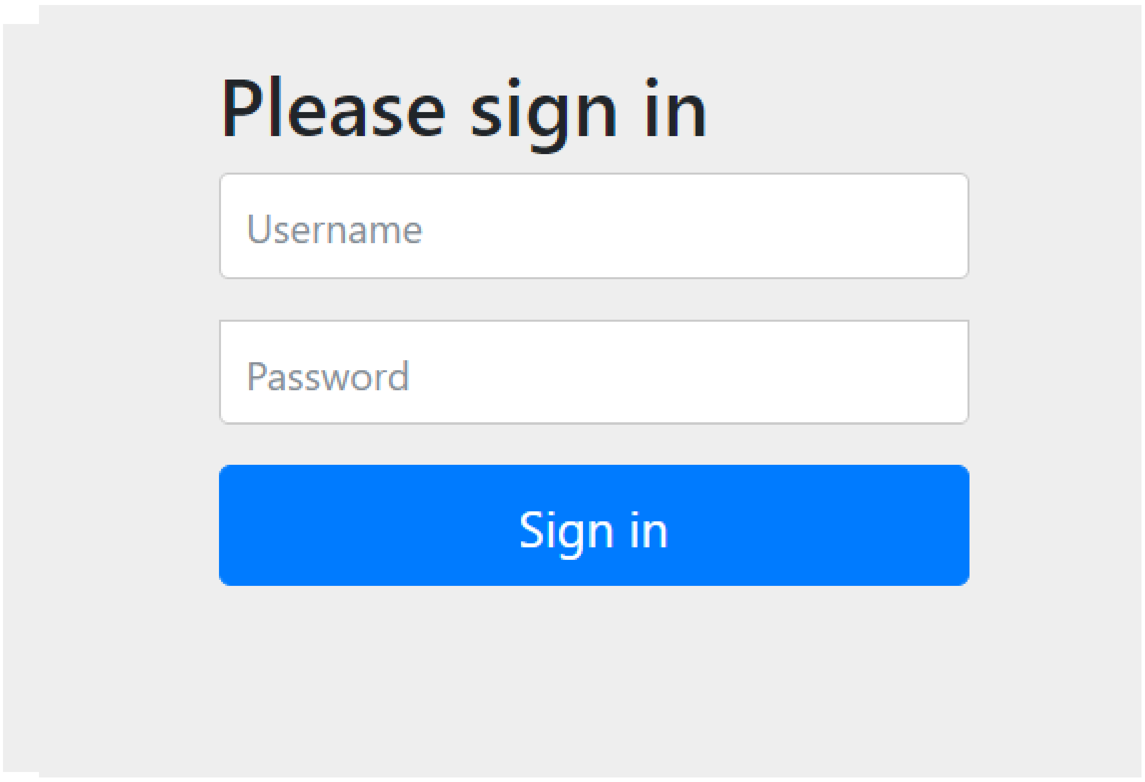
8. UI

There are examples of UI:

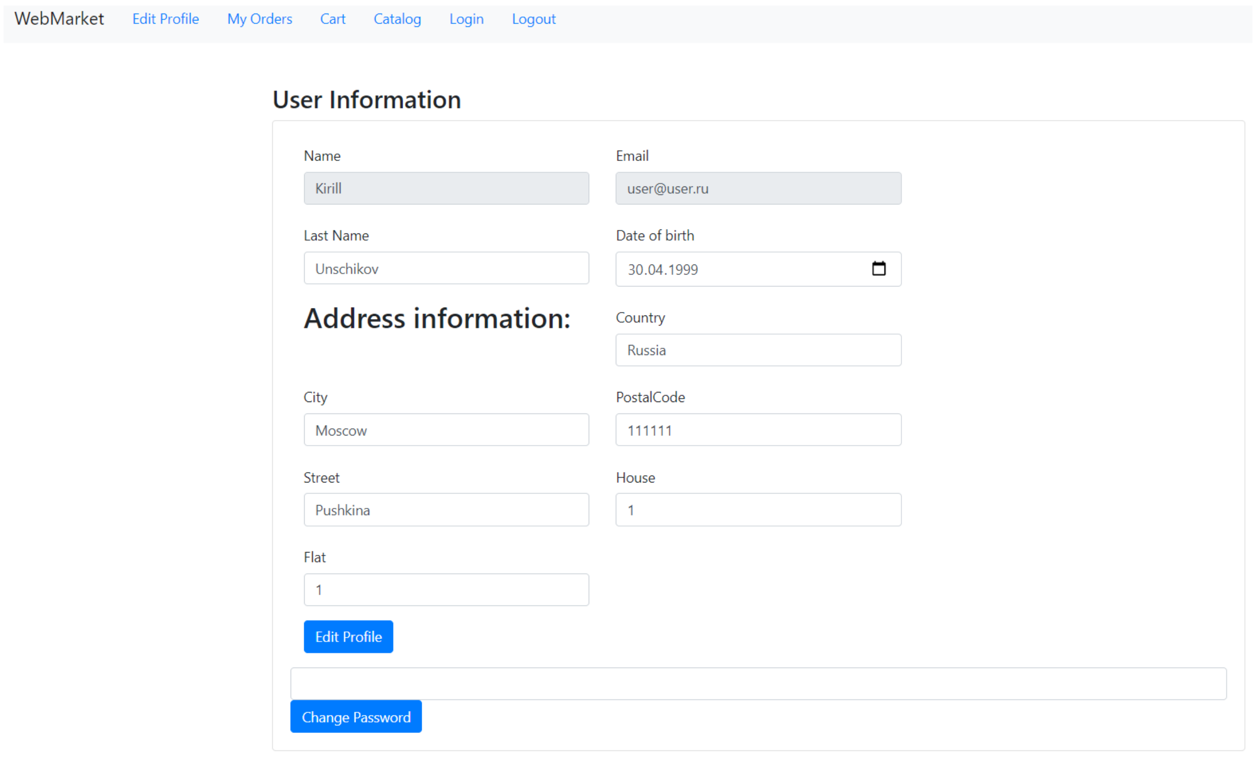


Title	Price	Currently available	Categories: Values	Buy
Baikal Air	1000.0	100	Country : Russia Volume : 300 Material : Tin	Buy Article
Tokyo Air	800.0	49	Material : Alluminium Volume : 800 Country : Japan	Buy Article

Picture 6 – Catalog interface



Picture 7 – Login form


WebMarket Edit Profile My Orders Cart Catalog Login Logout

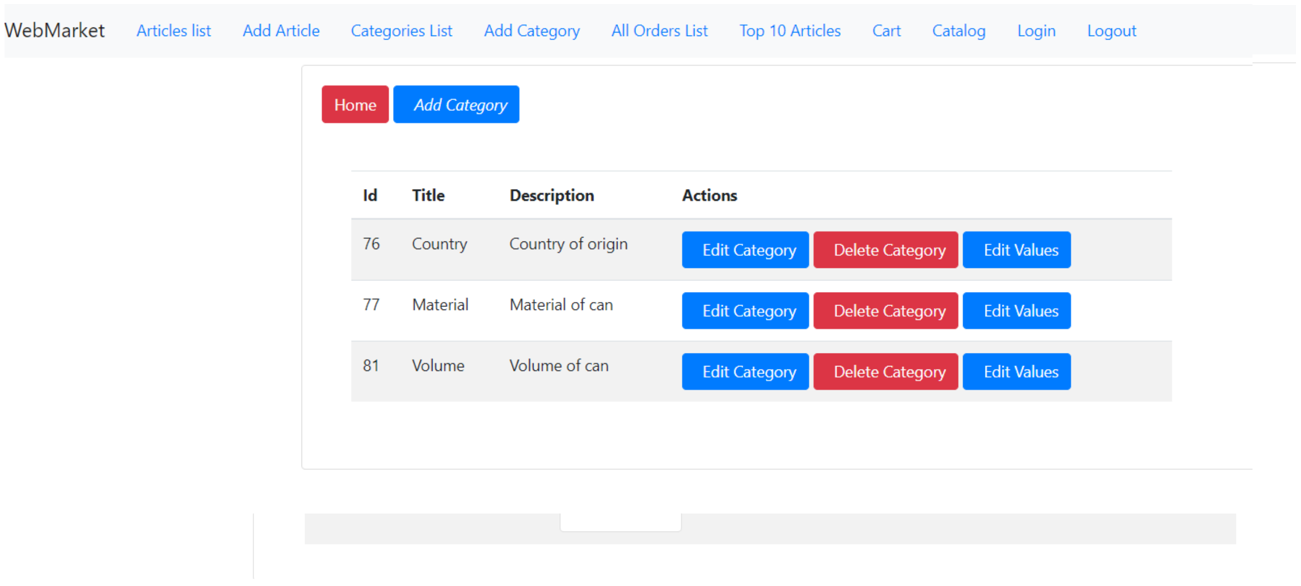
User Information

Name: Kirill	Email: user@user.ru
Last Name: Unschikov	Date of birth: 30.04.1999
Address information:	Country: Russia
City: Moscow	PostalCode: 111111
Street: Pushkina	House: 1
Flat: 1	

Edit Profile

Change Password

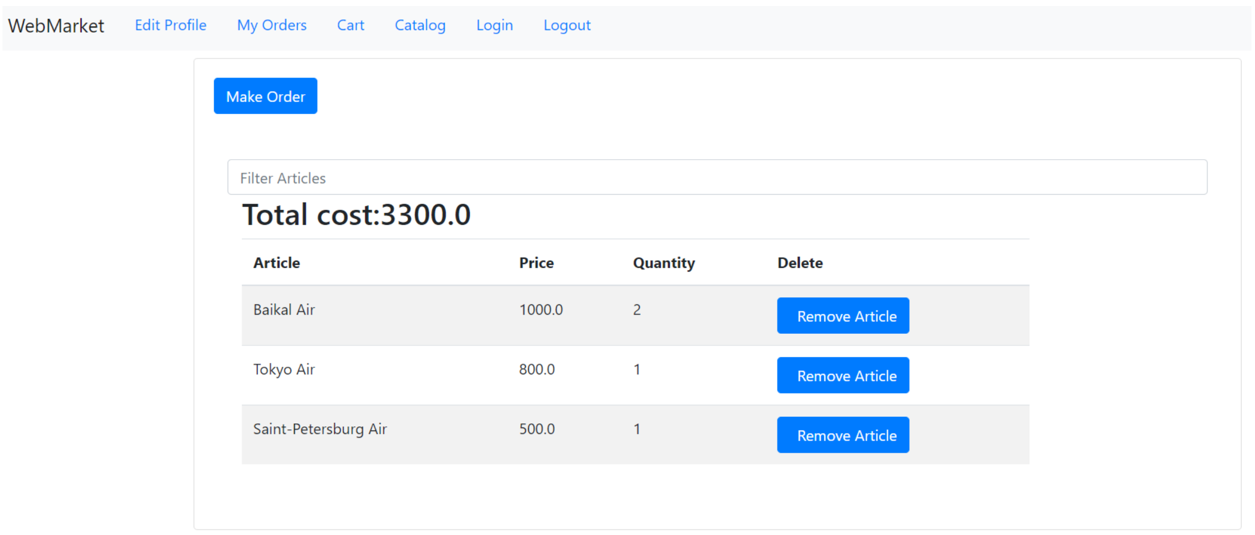
Picture 8 – User’s information


WebMarket Articles list Add Article Categories List Add Category All Orders List Top 10 Articles Cart Catalog Login Logout

Home *Add Category*

Id	Title	Description	Actions
76	Country	Country of origin	Edit Category Delete Category Edit Values
77	Material	Material of can	Edit Category Delete Category Edit Values
81	Volume	Volume of can	Edit Category Delete Category Edit Values

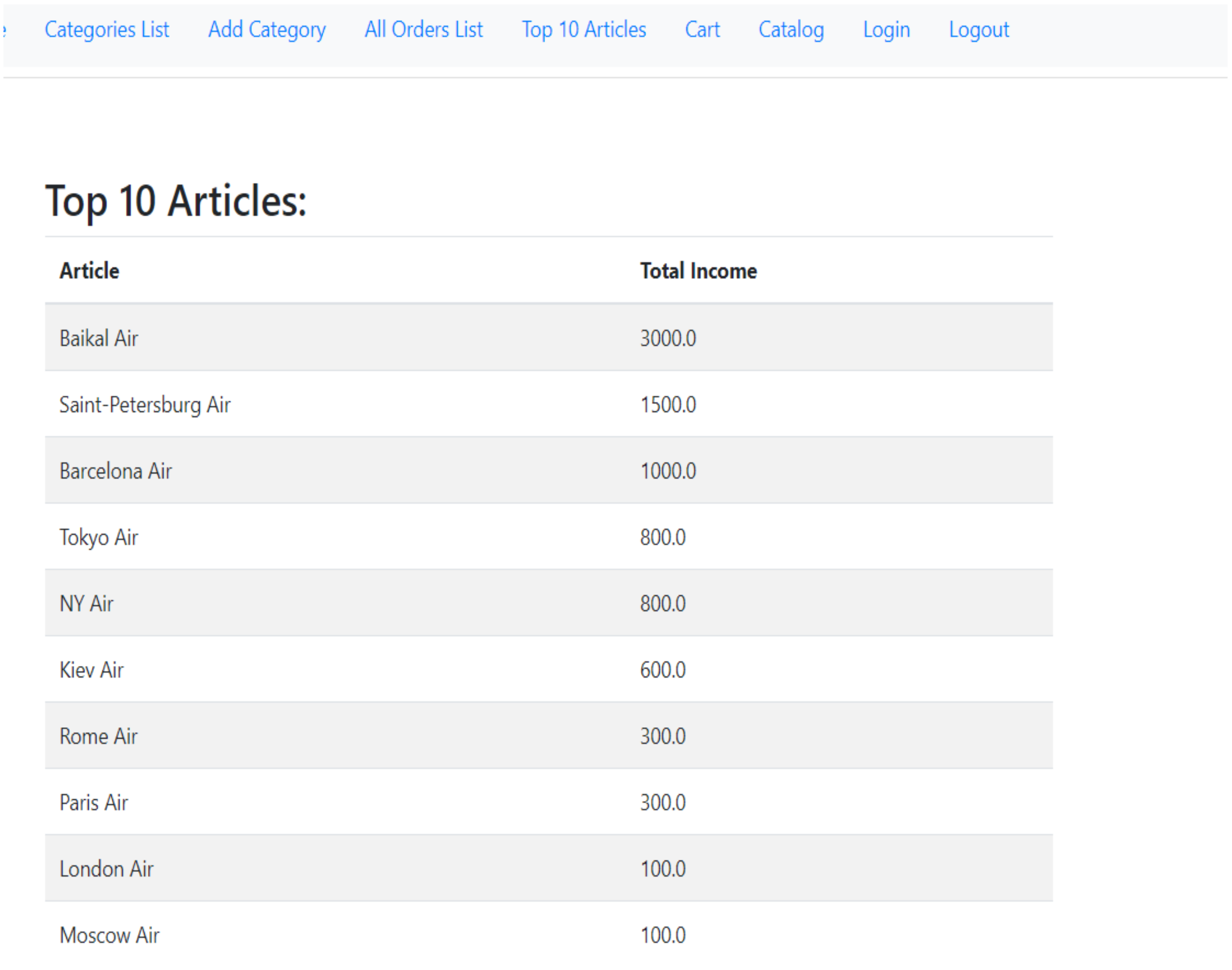
Picture 9 – Categories List



Total cost:3300.0

Article	Price	Quantity	Delete
Baikal Air	1000.0	2	Remove Article
Tokyo Air	800.0	1	Remove Article
Saint-Petersburg Air	500.0	1	Remove Article

Picture 10 – Cart



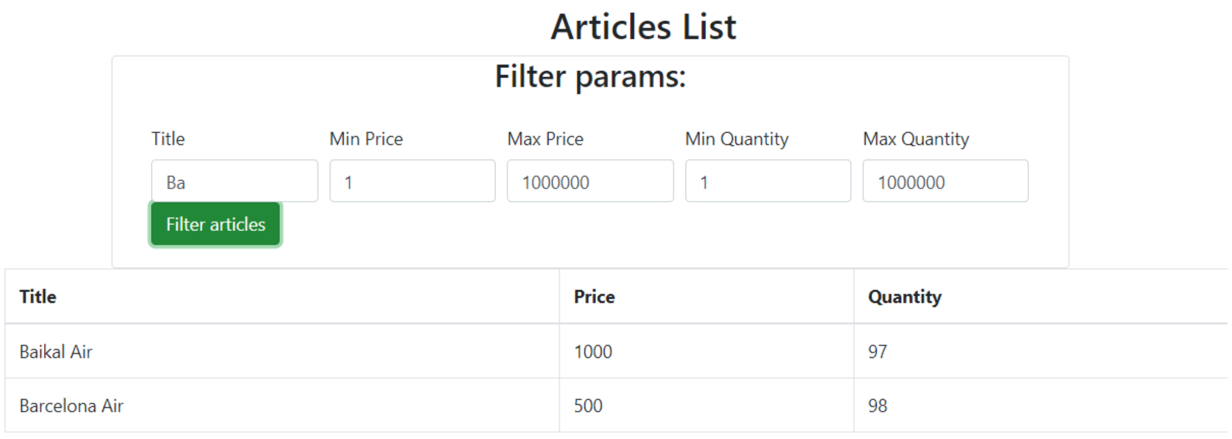
Top 10 Articles:

Article	Total Income
Baikal Air	3000.0
Saint-Petersburg Air	1500.0
Barcelona Air	1000.0
Tokyo Air	800.0
NY Air	800.0
Kiev Air	600.0
Rome Air	300.0
Paris Air	300.0
London Air	100.0
Moscow Air	100.0

Picture 11 – Top 10 articles

9. React

There is React SPA. User can filter articles by price, title and quantity.


Articles List

Filter params:

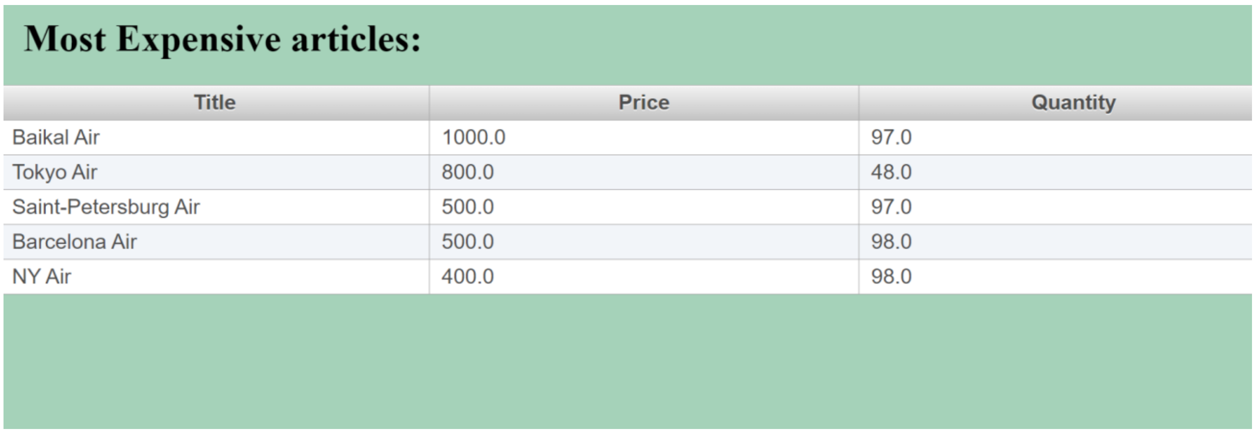
Title	Min Price	Max Price	Min Quantity	Max Quantity
Ba	1	1000000	1	1000000

Title	Price	Quantity
Baikal Air	1000	97
Barcelona Air	500	98

Picture 12 – React page

10. **Market showcase**

Showcase is a simple one-page application deployed on WildFly application server. It communicates with another application via rest requests. When the main application is uploaded, it sends to this application a message about that in topic on the server (using JMS). After that second app send REST request to take a data – 5 of the most expensive articles articles. When data receives, a message is pushed to the JSF side via websocket, after that data on this page is updating with AJAX. When second app could not connect to remote source, warning message appears.



Most Expensive articles:

Title	Price	Quantity
Baikal Air	1000.0	97.0
Tokyo Air	800.0	48.0
Saint-Petersburg Air	500.0	97.0
Barcelona Air	500.0	98.0
NY Air	400.0	98.0

Picture 13 - Showcase

11.Further impovements

1. Apply soft-deleting pattern when deleting data.
2. Refactor database to make it more flexible
3. Add chat for users to communicate with employers.