04.2016

Sergey Fedorov

Java school, T-SYSTEMS, 2016



MilkRoad

Yet another e-shop

Content

[1 Introduction 3](#_Toc447586372)

[1.1 Purpose 3](#_Toc447586373)

[1.2 Scope 3](#_Toc447586374)

[1.3 Definitions, Acronyms and Abbreviations 3](#_Toc447586375)

[2 Architecture overview 5](#_Toc447586376)

[2.1 Database schema 5](#_Toc447586377)

[2.2 Model 6](#_Toc447586378)

[2.3 Layers 6](#_Toc447586379)

[2.3.1 Presentation tier 7](#_Toc447586380)

[2.3.2 Business logic tier 8](#_Toc447586381)

[2.3.3 Data access tier 9](#_Toc447586382)

[2.3.4 Tiers interaction 10](#_Toc447586383)

[3 Use cases 11](#_Toc447586384)

[3.1 Actors 11](#_Toc447586385)

[3.2 List of use cases 11](#_Toc447586386)

[3.2.1 Admin use cases 11](#_Toc447586387)

[3.2.2 Seller use cases 11](#_Toc447586388)

[3.2.3 Customer use cases 11](#_Toc447586389)

[3.2.4 Anonymous use cases 12](#_Toc447586390)

[3.3 Use case diagrams 13](#_Toc447586391)

[3.3.1 Customer use cases 13](#_Toc447586392)

[4 User interface 14](#_Toc447586393)

[4.1 Main pages 14](#_Toc447586394)

[4.1.1 Welcome page 14](#_Toc447586395)

[4.1.2 Login page 14](#_Toc447586396)

[4.1.3 Registration page 15](#_Toc447586397)

[4.1.4 Catalog page 15](#_Toc447586398)

[4.1.5 Product page 16](#_Toc447586399)

[4.1.6 Cart page 16](#_Toc447586400)

[4.1.7 Checkout page 17](#_Toc447586401)

[4.1.8 Profile page 17](#_Toc447586402)

[4.1.9 Order management page 18](#_Toc447586403)

[4.1.10 Category management page 18](#_Toc447586404)

[4.1.11 Add/Edit product page 19](#_Toc447586405)

[4.1.12 Statistics page 19](#_Toc447586406)

[5 MilkRoad connector 20](#_Toc447586407)

[6 Technologies and frameworks 20](#_Toc447586408)

[6.1 Data storage 20](#_Toc447586409)

[6.2 Core 20](#_Toc447586410)

[6.3 Presentation 20](#_Toc447586411)

[6.3.1 Server side 20](#_Toc447586412)

[6.3.2 Client side 20](#_Toc447586413)

[6.4 Testing 20](#_Toc447586414)

[6.5 MilkRoad connector 21](#_Toc447586415)

[6.6 Logging 21](#_Toc447586416)

[7 Testing 22](#_Toc447586417)

[7.1 User service tests 23](#_Toc447586418)

[7.2 Address service tests 23](#_Toc447586419)

[7.3 Catalog service tests 23](#_Toc447586420)

[7.4 Order service tests 23](#_Toc447586421)

[7.5 Statistics service tests 23](#_Toc447586422)

[8 Future release plan 24](#_Toc447586423)

[8.1 Features 24](#_Toc447586424)

[8.2 UI 24](#_Toc447586425)

# Introduction

MilkRoad is a system of software components that models routine activities of online shop.

## Purpose

The main purpose of this project is implementation almost real marketplace where customers and sellers can do appropriate cases such as searching goods and make orders.

## Scope

MilkRoad provides a base level of functionality to show feasibility for large scale production use.

## Definitions, Acronyms and Abbreviations

**Entity** – **JPA Entity** is user defined class which instances can be stored in a database.

**DTO** – **Data transfer object** is just data container which is used to transport data between layers and tiers. It mainly contains attributes. You can even use public attributes without getters and setters. Data transfer objects do not contain any business logic.

**DAO** – **Data access object** is an object that provides an abstract interface to some type of database or other persistence mechanism. By mapping application calls to the persistence layer, DAO provide some specific data operations without exposing details of the database. This isolation supports the single responsibility principle. It separates what data accesses the application needs, in terms of domain-specific objects and data types (the public interface of the DAO), from how these needs can be satisfied with a specific DBMS, database schema, etc. (the implementation of the DAO)

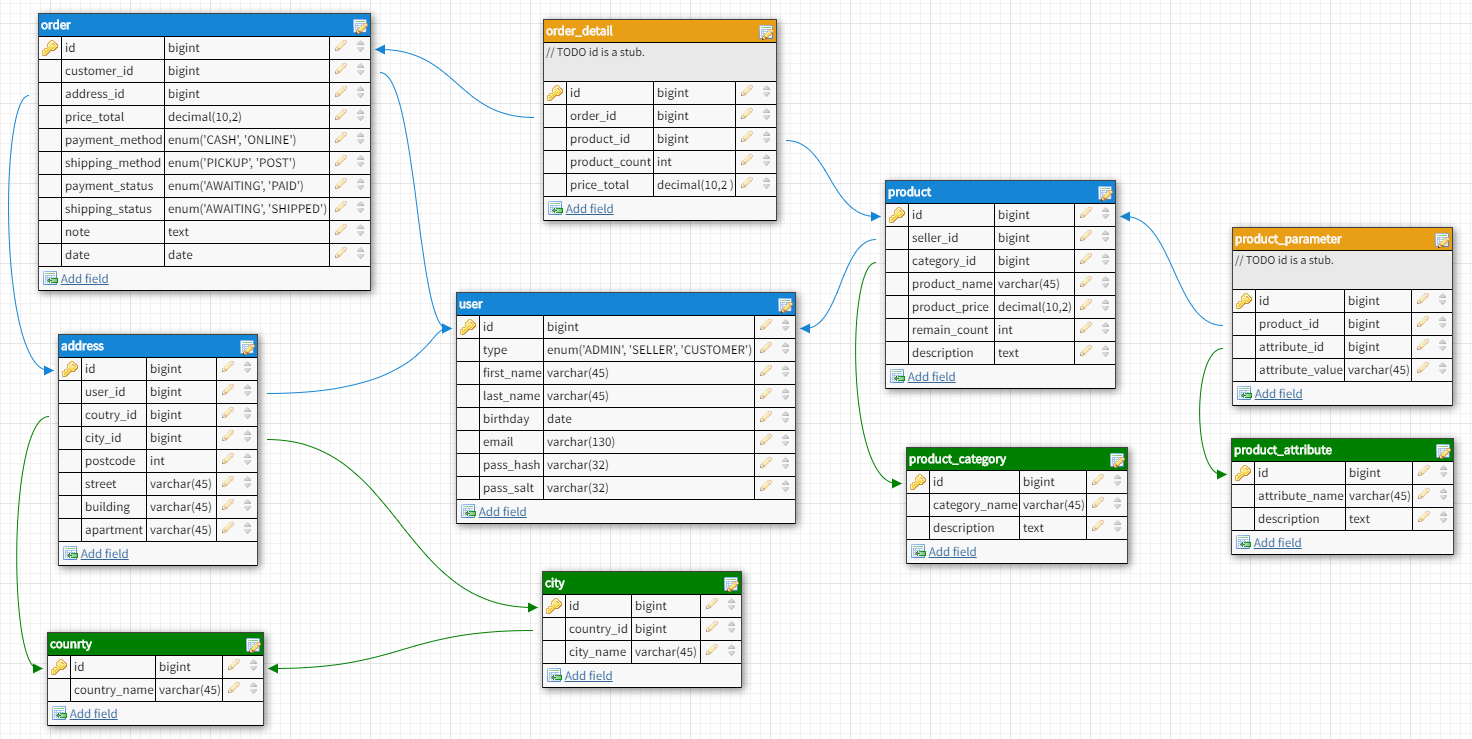
**UI** – The **user interface**, in the industrial design field of human–machine interaction, is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process.

**Header** – The **header** of a webpage typically includes the company or organization's logo, as well as the main navigation bar. This section, which resides at the top of each webpage, is often part of a template and therefore is the same across all pages within a website or section of a website. HTML 5 even includes a <header> tag that developers can use to specify the header section of each webpage.

**Footer** - In website design, a **page footer** is called a webpage footer which is the bottom section of a website. They can be designed using HTML and CSS. Website footers have text that can show some of the details about the website.

# Architecture overview

## Database schema



## Model

**User** – User is a base entity which keeps information about the person. Fields are: role, name, surname, date of birth, email address and pass.

**Address** – Address is a base entity which contains details about certain location. Fields are: user, country, city, postcode, street, building and apartment.

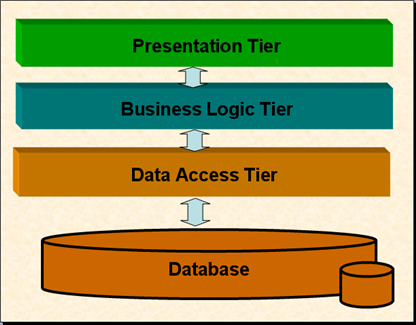
**Product** – Product is a base entity that keeps info about goods. Fields are: seller, category, name, description, price and remain count.

**Order** – Order is a base entity which provides information about bought out products. Fields are: customer, total price, payment and shipping methods and status, checkout date, customer note and address, if shipping method is post.

**Cart** – Cart (basket case) stores info about customer selection. Fields are: list of product article and it’s count.

## Layers

MilkRoad application has a three-tier architecture. Communication between presentation and business logic layers is done by DTO. Communication between business logic and data access layers is done by Entities.



### Presentation tier

The presentation tier displays information related to such services as browsing merchandise, purchasing and shopping cart contents. It communicates with service layer by which it puts out the results to the client browser and all other tiers in the network.

#### Auth controller

Authentication controller dispatches login/logout related requests. Returns login page.

#### Register controller

Register controller dispatches register requests. It invokes appropriate user service methods. Returns registration page.

#### Profile controller

Profile controller dispatches edit profile, add/edit address requests. It communicates with user, address and order services. Returns profile page with appropriate information.

#### Catalog controller

Catalog controller dispatches product catalog browsing requests. It takes the data from catalog service. Returns catalog or single product page.

#### Cart controller

Cart controller dispatches cart operations requests such as add or remove product from basket case. It fetches product info from catalog service. Returns cart page.

#### Checkout controller

Checkout controller dispatches checkout request. It invokes create order method from order service. Returns checkout page.

#### Management controller

Management controller dispatches administrative requests such as add/edit category, attribute and product requests, update order and view statistics requests. It communicates with order, catalog and statistics services. Returns appropriate view or json data for ajax requests.

### Business logic tier

The business logic tier controls application functionality by performing detailed processing.

#### User service

User service provides interface for user management. For example, there’re methods allow to add, get or edit user details to/from MilkRoad information system.

#### Address service

Address service provides interface for address management. There’re two methods allow to add and edit address information.

#### Catalog service

Catalog service provides interface for categories, attributes and products survey and management.

#### Order service

Order service provides interface for order survey and management. There’re methods allow to get, add or update order details.

#### Statistics service

Statistics service provides information about top customers and products, about total cash by period.

### Data access tier

The data access tier includes the data persistence mechanisms and exposes the data. This layer provides an API to the business logic tier that exposes methods of managing the stored data without exposing or creating dependencies on the data storage mechanisms.

#### Generic DAO

Generic DAO provides methods allow to persist, merge (update), remove, get by primary key or get all entities from/to database. All other DAOs extend it.

#### User DAO

User DAO has additional API allows to get user by email and get top customers.

#### Address DAO

Address DAO has additional method which returns list of addresses of certain user.

#### Product DAO

Product DAO provides API allows to get products by certain category, search product by name and get top products.

#### Category DAO

Category DAO has additional method which returns category by name.

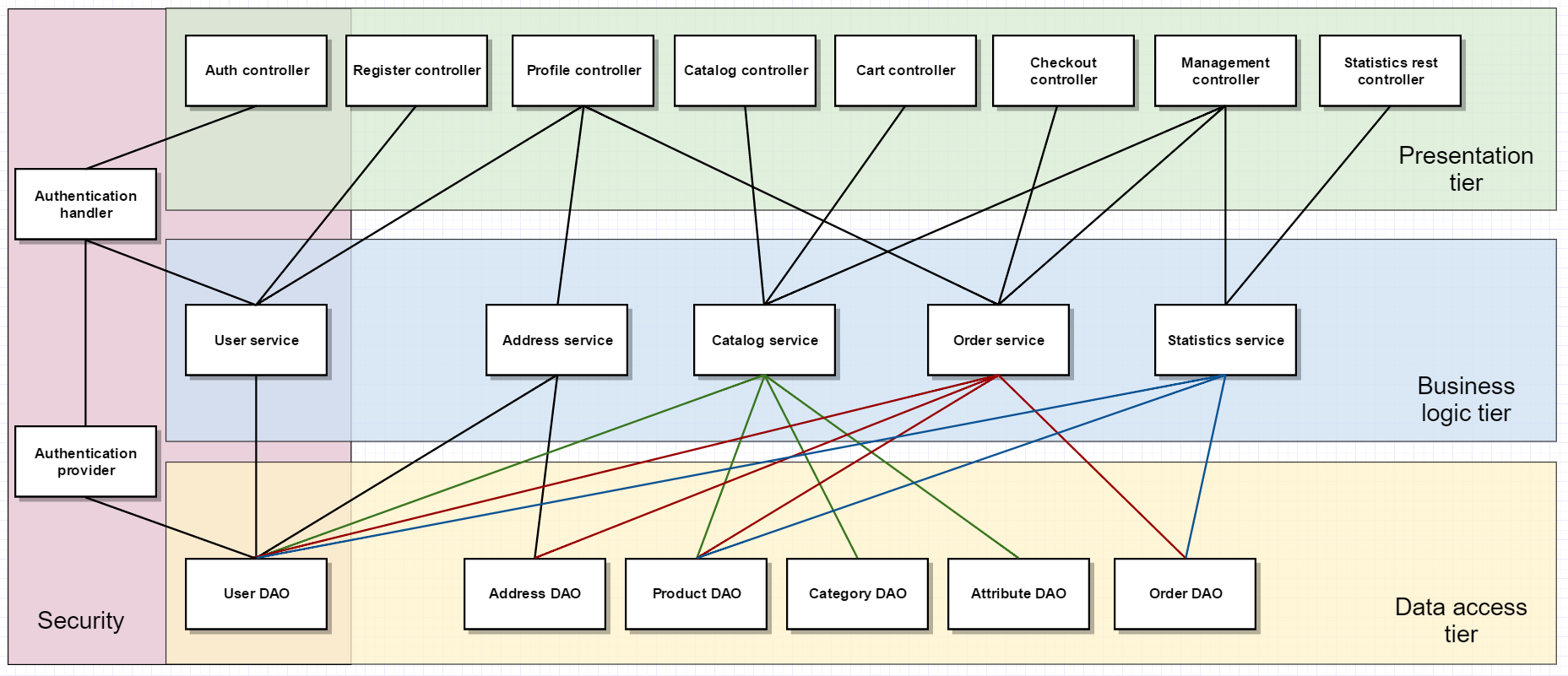
#### Attribute DAO

Attribute DAO has extra method that returns attribute by name.

#### Order DAO

Order DAO provides API allows to get total cash by period.

### Tiers interaction



# Use cases

## Actors

Actors have different authority rules.

**ADMIN** – The admin is a user who does administrative actions such as editing product catalog or updating order details.

**SELLER** – Same authority as admin. (Current release)

**CUSTOMER** – The customer is a user who can browse product catalog, add products to cart and checkout.

**ANONYMOUS** – The anonymous is unauthorized user.

## List of use cases

### Admin use cases

* Edit profile
* View product catalog
* Search product
* View product details
* View product list
* Add new product to catalog
* Edit product details
* Add new product category
* Edit category details
* Add new product attribute
* View order list
* Edit order details
* Logout

### Seller use cases

Same as admin use cases (Current release)

### Customer use cases

* Edit profile
* View product catalog
* Search product
* View product details
* Add product to cart
* View cart page
* Remove product from cart
* Checkout
* Logout

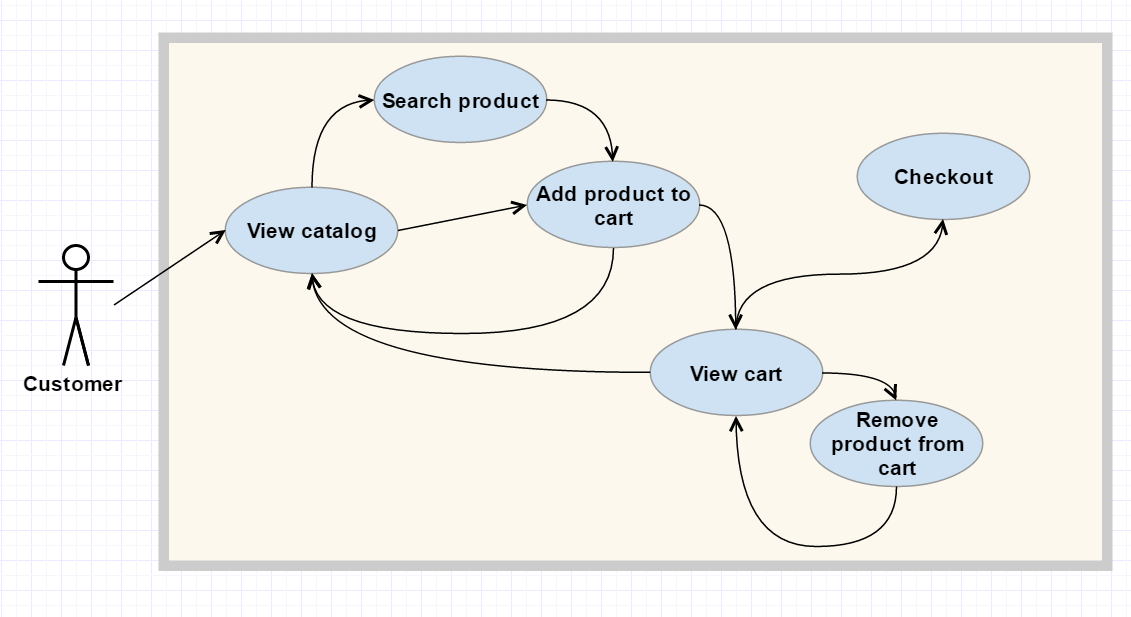
### Anonymous use cases

* Login
* View product catalog
* Search product
* View product details
* Add product to cart
* View cart page
* Remove product from cart

## Use case diagrams

### Customer use cases

#### Checkout diagram



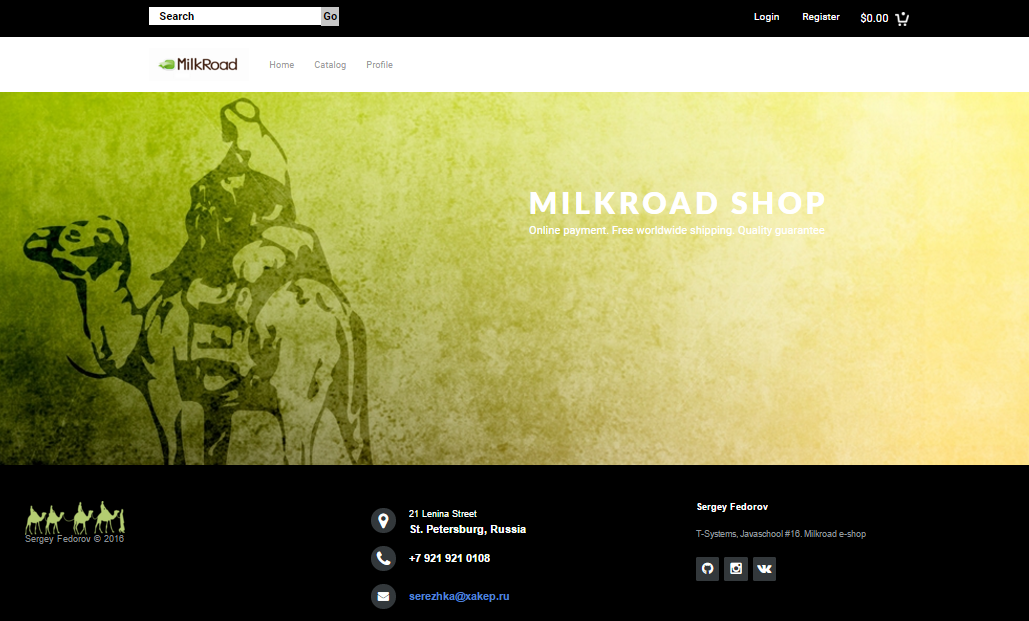
# User interface

MilkRoad has a trendy and responsive UI supported by all major browsers.

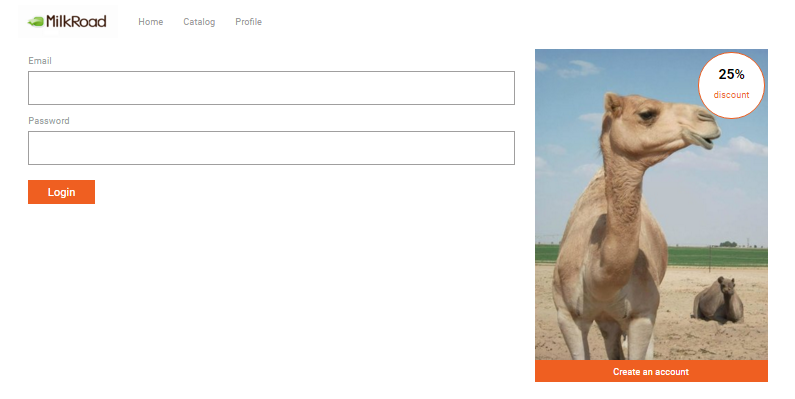
## Main pages

Page header and footer are same for each web page. So that they are presented only in welcome page section.

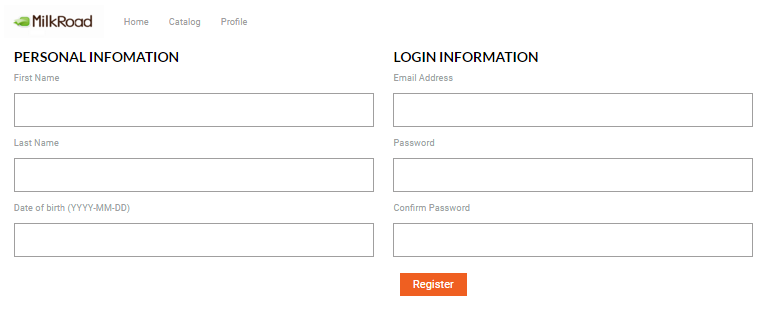
### Welcome page



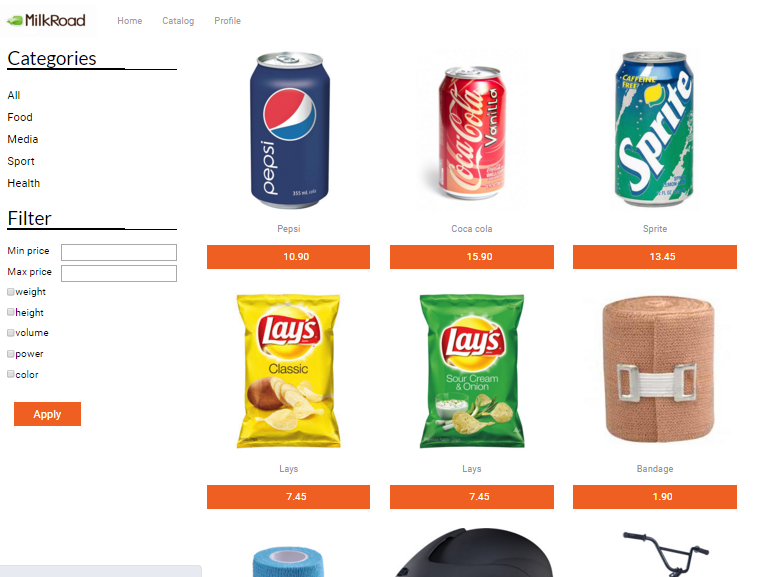
### Login page



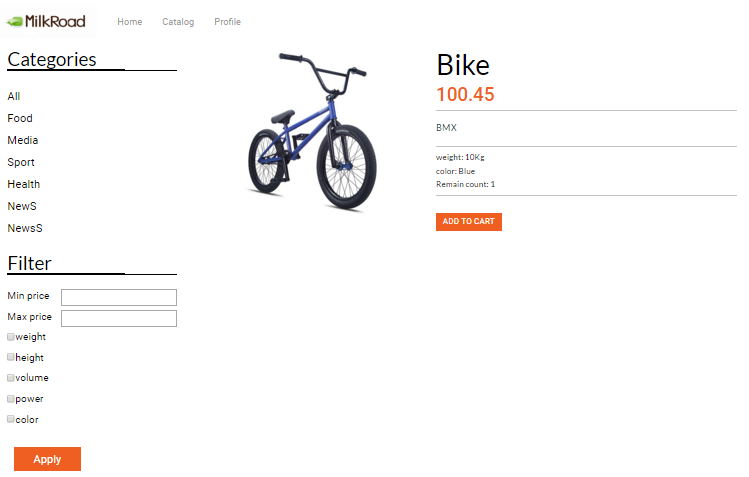
### Registration page



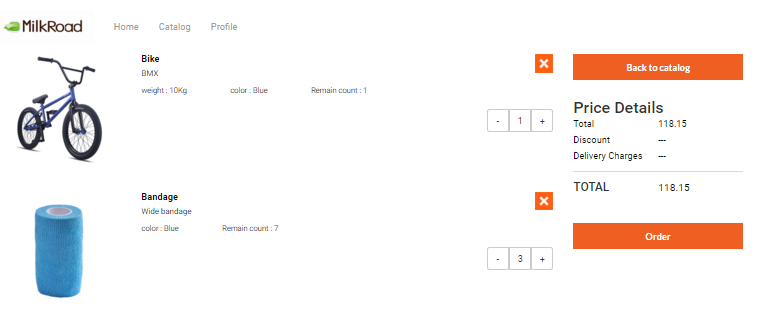
### Catalog page



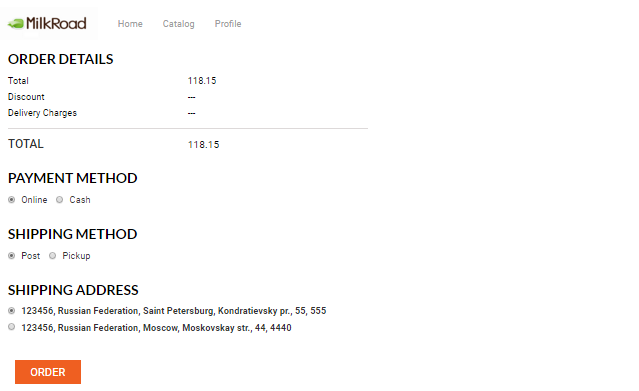
### Product page



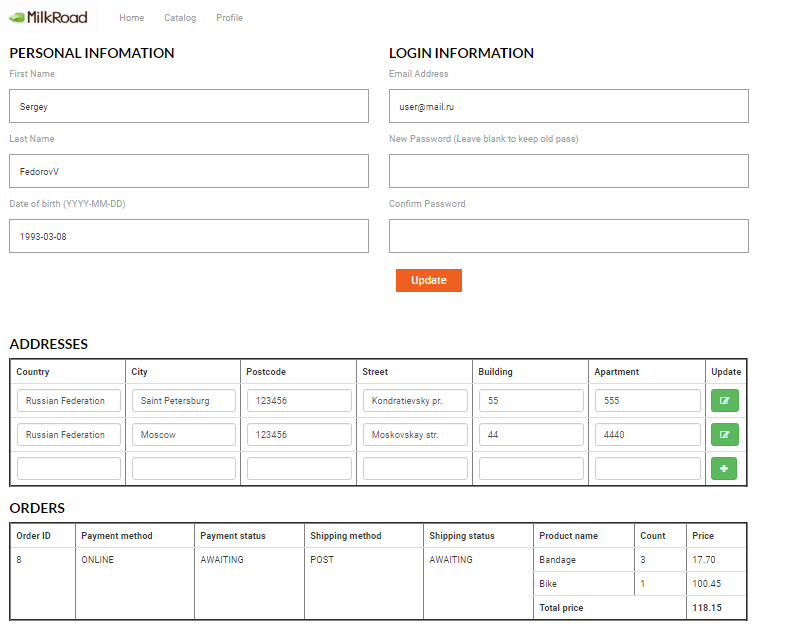
### Cart page



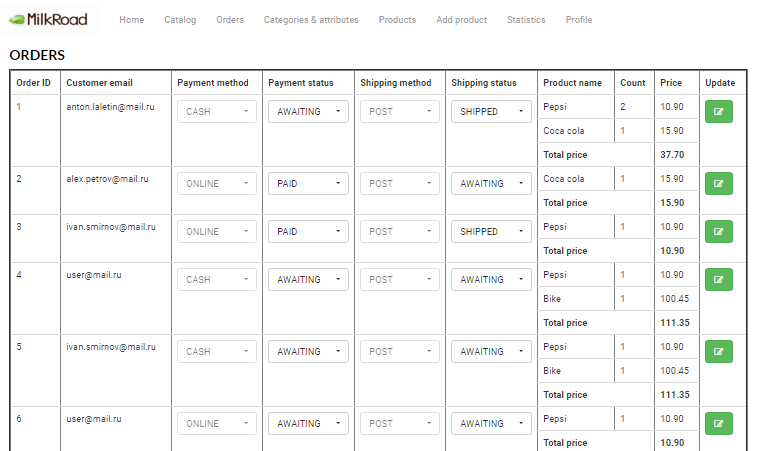
### Checkout page



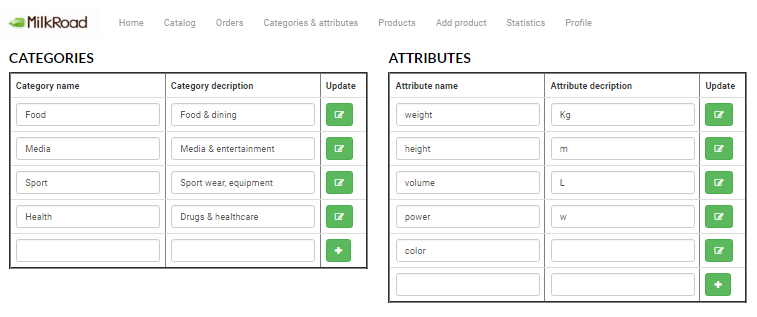
### Profile page



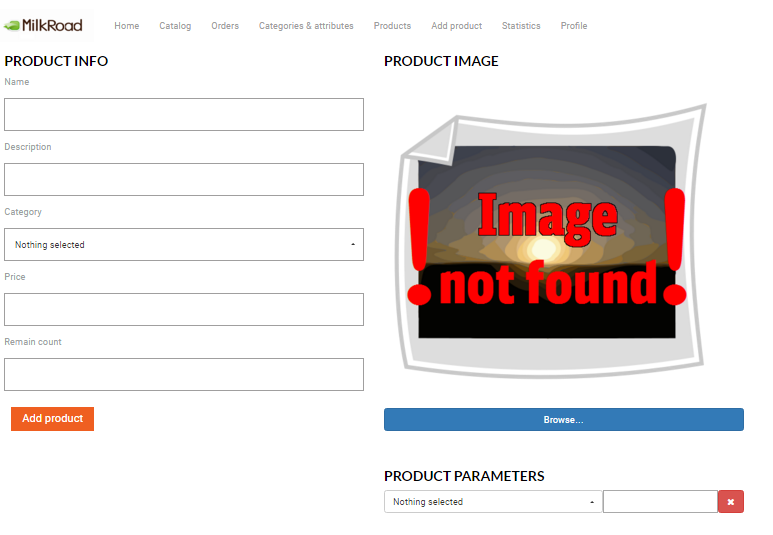
### Order management page



### Category management page



### Add/Edit product page



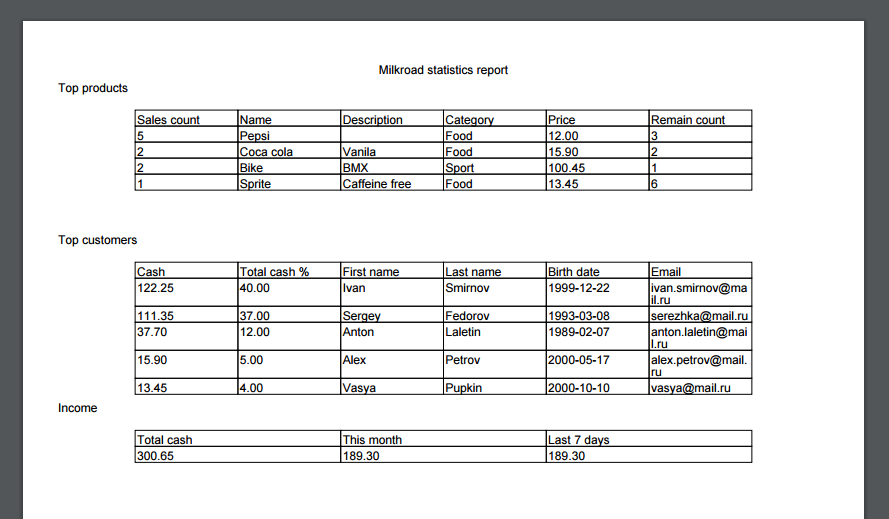
### Statistics page



# MilkRoad connector

MilkRoad connector is a separate application which connects to REST services of main app and generates PDF document with gathered statistics.

## Statistics report



# Technologies and frameworks

MilkRoad is a maven-based web application written in Java language. It uses few other libraries and frameworks.

## Data storage

MilkRoad uses MySQL open-source relational database for entities storage.

* MySQL 5.7
* MySQL connector java 5.1.38
* Hibernate 5.0.7
* Hibernate JPA 2.1

## Core

* Java 1.8
* Spring framework 4.2.5

## Presentation

### Server side

* Spring security 4.0.4
* JSP 2.1
* JSTL 1.2
* Javax validation 1.1.0
* Hibernate validator 5.2.4
* Jackson 2.7.2
* Apache commons fileupload 1.2

### Client side

* Bootstrap 3.3.6
* JQuery 2.2.0
* FontAwesome 4.5.0

## Testing

* JUnit 4.12
* PowerMock 1.6.4
* Maven surefire 2.19.1
* JaCoCo 0.7.6
* Sonar JaCoCo listeners 3.12

## MilkRoad connector

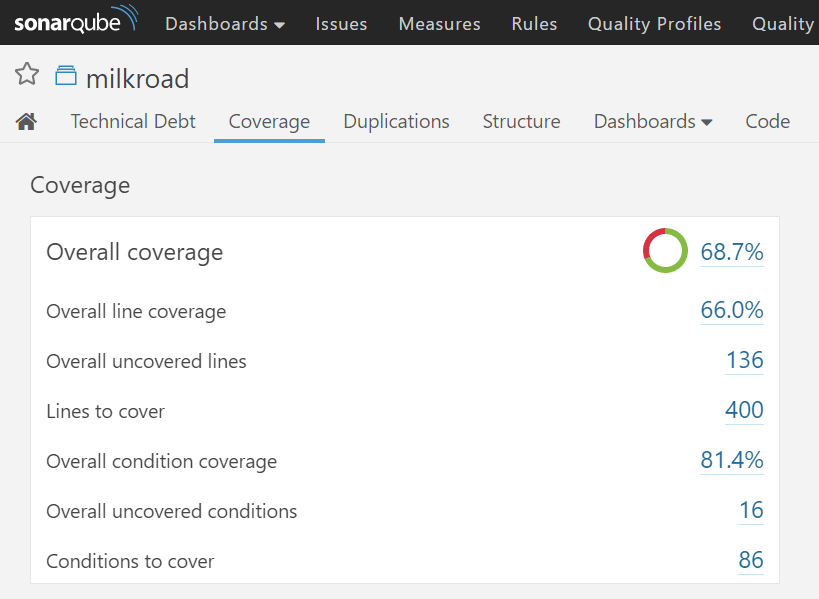
* EJB 3.2 spec
* JSF 2.2 spec
* iText PDF 5.5.9
* RESTEasy 3.0.12

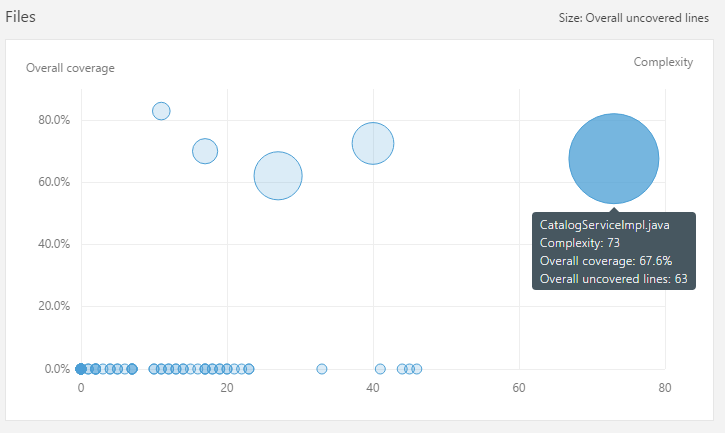
## Logging

* Log4j

# Testing

There are JUnit tests for each business logic service class with overall coverage up to 68.7%. Statistics provided by JaCoCo and Sonarqube:





## User service tests

* Complexity: 40
* Overall coverage: 72.5%
* Overall uncovered lines: 25

## Address service tests

* Complexity: 11
* Overall coverage: 82.9%
* Overall uncovered lines: 6

## Catalog service tests

* Complexity: 73
* Overall coverage: 67.6%
* Overall uncovered lines: 63

## Order service tests

* Complexity: 27
* Overall coverage: 62.1%
* Overall uncovered lines: 30

## Statistics service tests

* Complexity: 17
* Overall coverage: 70.0%
* Overall uncovered lines: 12

# Future release plan

## Features

* Product information data mining from other online shops
* PayPal support

## UI

* Popup login form