<Pharmacy>
Analysis and Design Document

Student: Plesa Gabriel
Group: 30235

Table of Contents

1. Requirements Analysis

1.1 Assignment Specification

Use Java/C# API to design and implement an application for the employees of a pharmacy (chemists). The application should have two types of users (a regular user represented by the chemist employee and an administrator user) which have to provide a username and a password in order to use the application.

1.2 Functional Requirements

The regular user can perform the following operations:

- Search medication by name, ingredients, manufacturer.
- Sell medication.

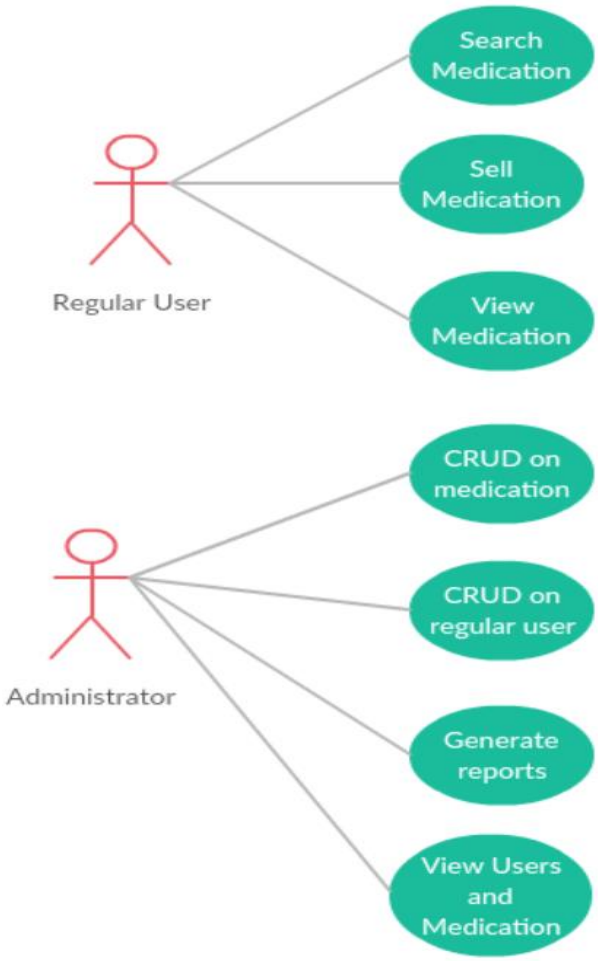
The administrator can perform the following operations:

- CRUD on medication (medication information: by name, ingredients, manufacturer, quantity and price).
- CRUD on regular users' information.
- Generate two types of reports files, one in pdf format and one in csv format, with the medication out of stock.

1.3 Non-functional Requirements

1. The information about users, medication and selling will be stored in multiple XML files. Use the Model View Controller in designing the application. Use the Factory Method design pattern for generating the reports.
2. All the inputs of the application will be validated against invalid data before submitting the data and saving it.

2. Use-Case Model



Use case: <Sell medication>
Level: <user-goal level >
Primary actor: <User>
Main success scenario: <The user successfully logins and then he enters the Request ID from a client, if the client has enough money the request is processed and deleted>
Extensions: <the user wrongly introduce his username or password, an error message is shown, after that he finally login>

3. System Architectural Design

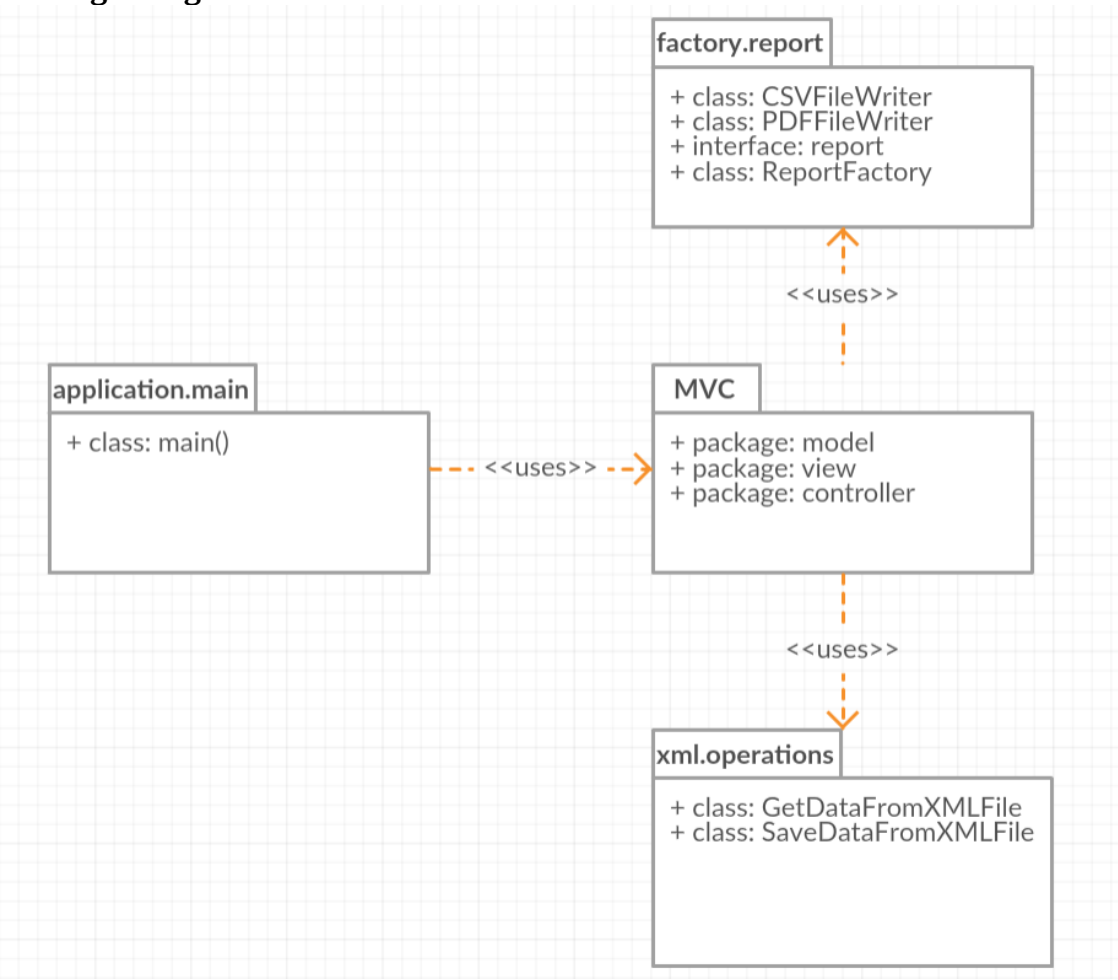
3.1 Architectural Pattern Description

Model view controller (MVC) pattern is used to separate application's concerns.

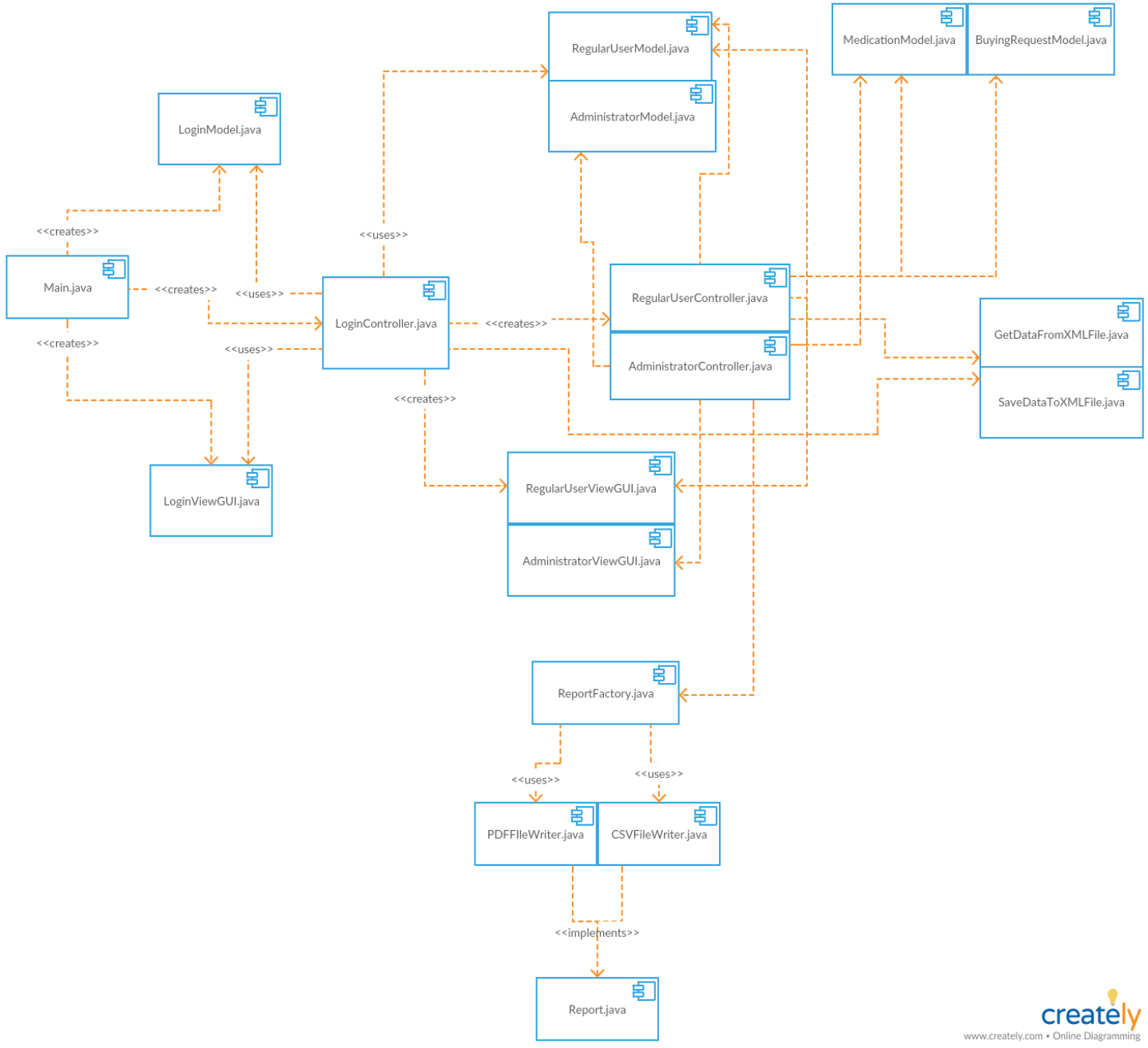
- **Model** - Model represents an object or JAVA POJO carrying data. It can also have logic to update controller if its data changes.
- **View** - View represents the visualization of the data that model contains.
- **Controller** - Controller acts on both model and view. It controls the data flow into model object and updates the view whenever data changes. It keeps view and model separate.

3.2 Diagrams

Package Diagram

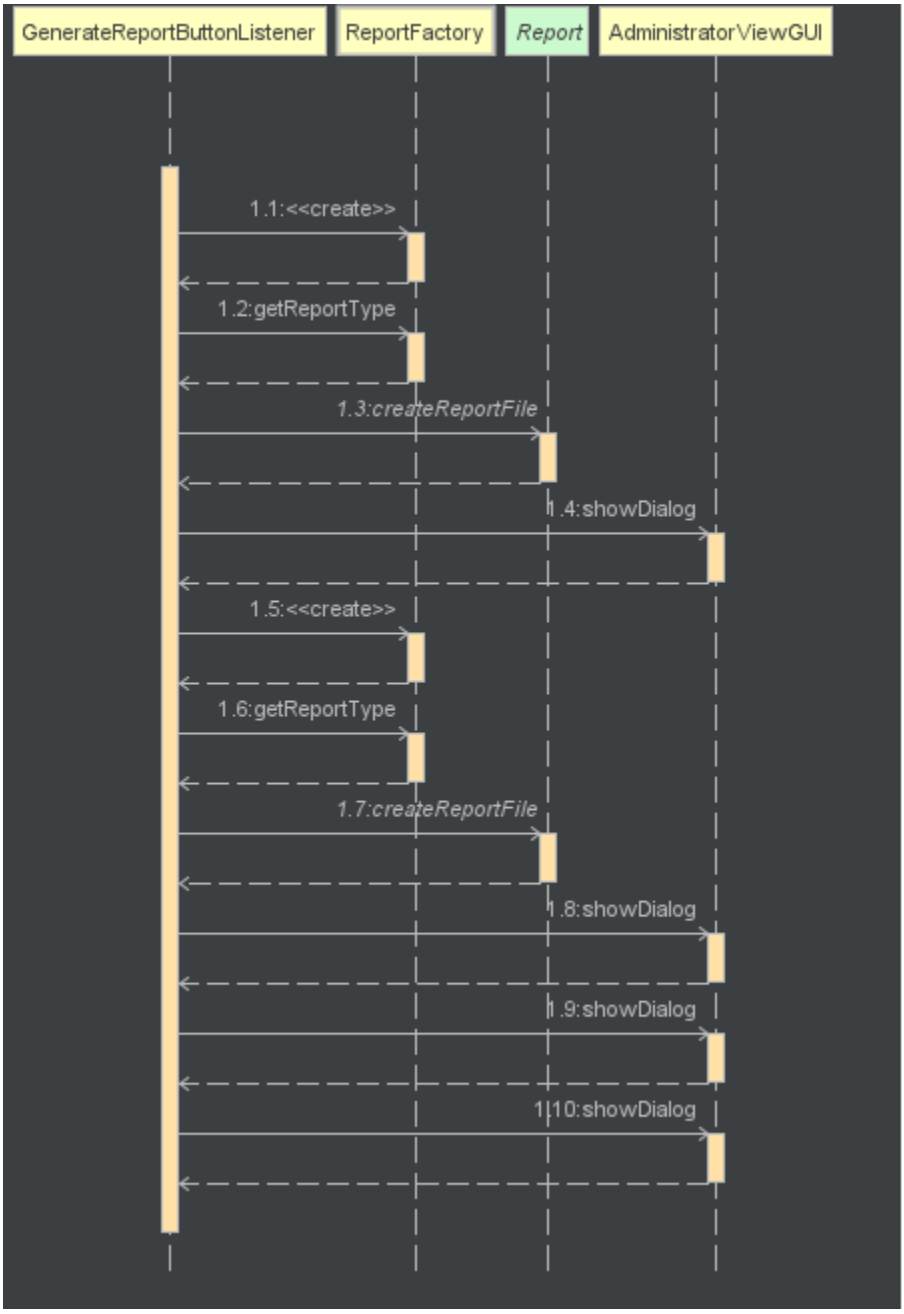


Component diagram



4. UML Sequence Diagrams

Sequence diagram for generating a report with Report Factory pattern



5. Class Design

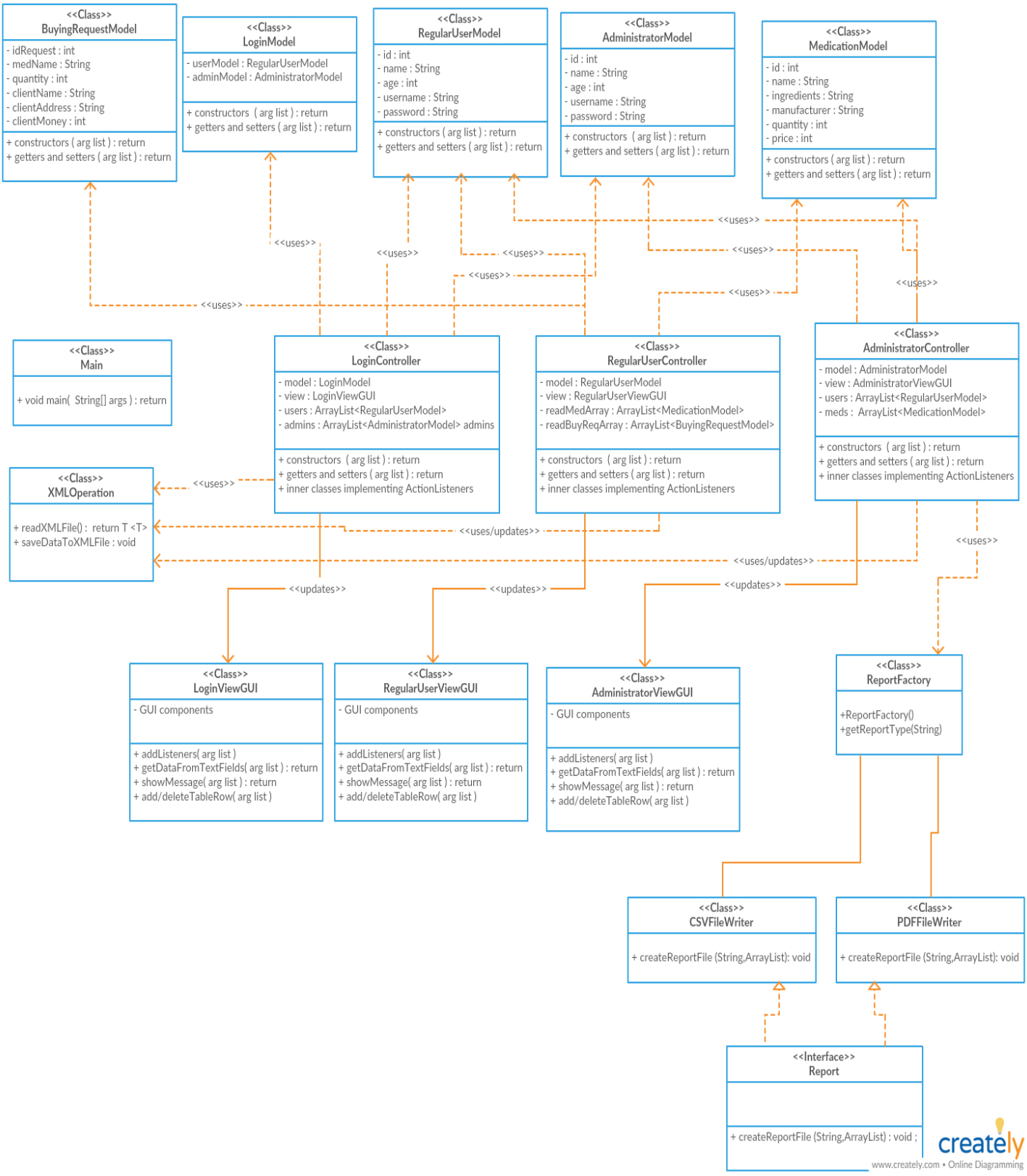
5.1 Design Patterns Description

Model–view–controller (**MVC**) is an architectural pattern commonly used for developing user interfaces that divides an application into three interconnected parts. This is done to separate internal representations of information from the ways information is presented to and accepted from the user. The MVC design pattern decouples these major components allowing for efficient code reuse and parallel development.

Traditionally used for desktop graphical user interfaces (GUIs), this architecture has become popular for designing web applications and even mobile, desktop and other clients. Popular programming languages like Java, C#, Ruby, PHP and others have popular MVC frameworks that are currently being used in web application development straight out of the box.

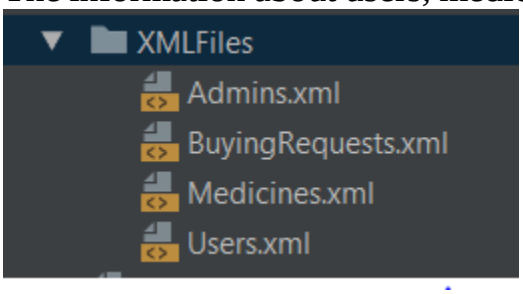
Factory pattern is one of the most used design patterns in Java. This type of design pattern comes under creational pattern as this pattern provides one of the best ways to create an object. In Factory pattern, we create object without exposing the creation logic to the client and refer to newly created object using a common interface.

5.2 UML Class Diagram

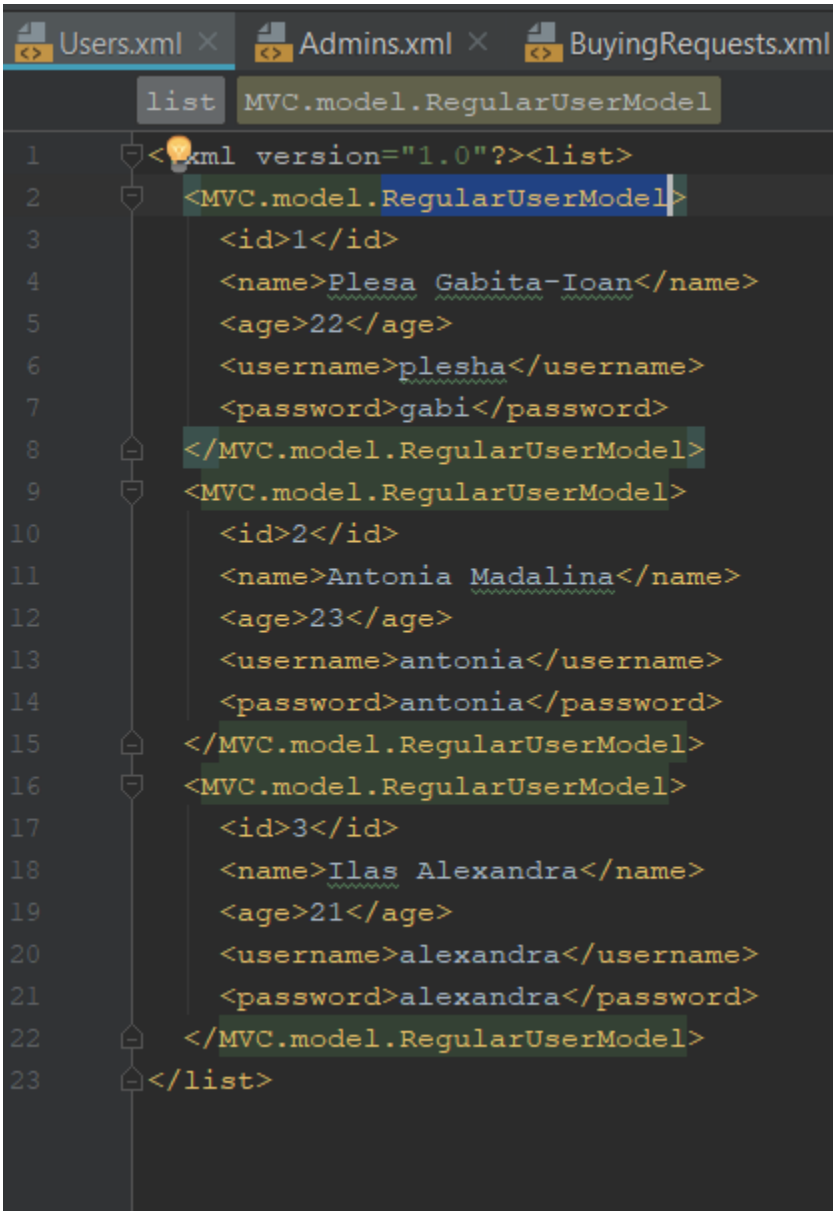


6. Data Model

The information about users, medication and selling is stored in multiple XML files.



A file with XML data looks like this:



7. System Testing

For the main operations the system supports tests: insert, delete, update, view, etc. If something is going wrong the application send an error message to inform the user. The information it is tested and validated before it is inserted into a XML file.

8. Bibliography

[1] https://github.com/kittyrad/laborator1
[2] https://creately.com/
[3] https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller