Student: Zavaczki Péter - Tibor

**Group: 30433**

Table of Contents

1. Requirements Analysis 3

1.1 Assignment Specification 3

1.2 Functional Requirements 3

1.3 Non-functional Requirements 3

2. Use-Case Model 3

3. System Architectural Design 3

4. UML Sequence Diagrams 3

5. Class Design 3

6. Data Model 3

7. System Testing 3

8. Bibliography 3

1. Requirements Analysis

# Assignment Specification

This application is a client server architecture based application, expected to be used by a news agency’s writers and the readers.

# Functional Requirements

The application differentiates two types of users: regular users (readers), writers and admins (or power users). The readers do not need to log in, but the admins and writers both need a username and a password to log in to be able to access the features available to them.

The regular user can read the articles posted so far and don’t need to log in to do so.

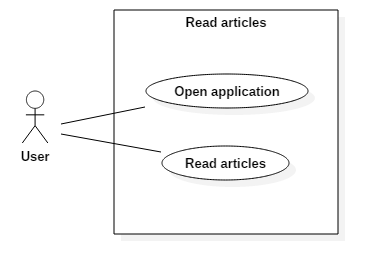
The writers can log in to be able to write an article, which then gets posted to the readers.

The admins can add writer accounts.

# Non-functional Requirements

* The application will be written in Java, with the UI written with JavaFX.
* The application will be based on client server architecture, the UI being on the client side, the data access layer on the server side, the model of the objects used in the application (Article, User) and the commands can be found on a common side.
* The data to be managed by the application will be stored in json files.

2. Use-Case Model



Use case: Read articles

Level: summary level

Primary actor: regular user

Main success scenario:

* The user starts the application
* The UI shows the currently available articles

Extensions:

* Add topics, filter by topic

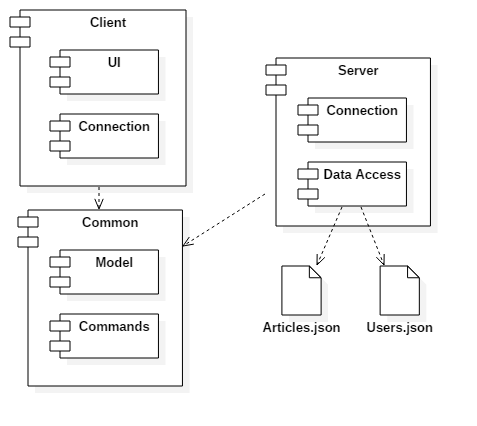
3. System Architectural Design

**3.1 Architectural Pattern Description**

The application will be designed and implemented using the client server architecture. The sub patterns are MVC, and three tier layered. The UI layer is present on the client side, the data access layer is present on the server side and the business layer is common, since it contains the model and the commands, such as serialization, deserialization. The client and the server are both entirely separate modules in the application, but they both depend on the common module.

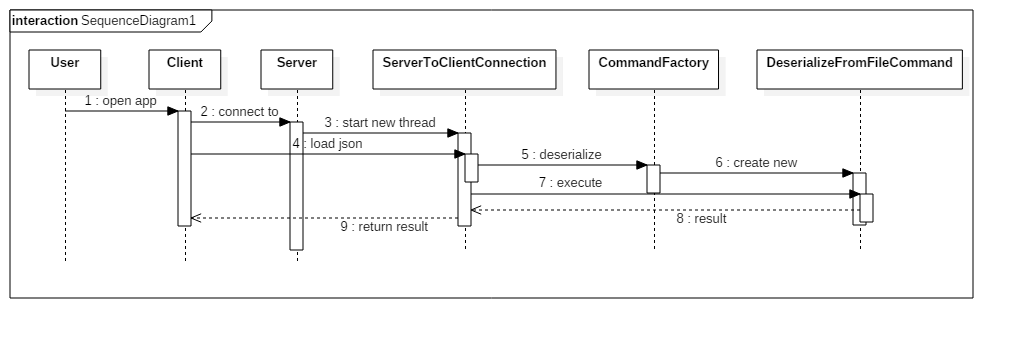
**3.2 Diagrams**

The application is built according to a client server layered architecture, this pattern separates the UI into the client, the DAO layer into the server and the business into the common layer.



Client and the server can run on a different device, but in our case they are both on localhost.

4. UML Sequence Diagrams



The above diagram is a sequence diagram representing the “Read articles” action.

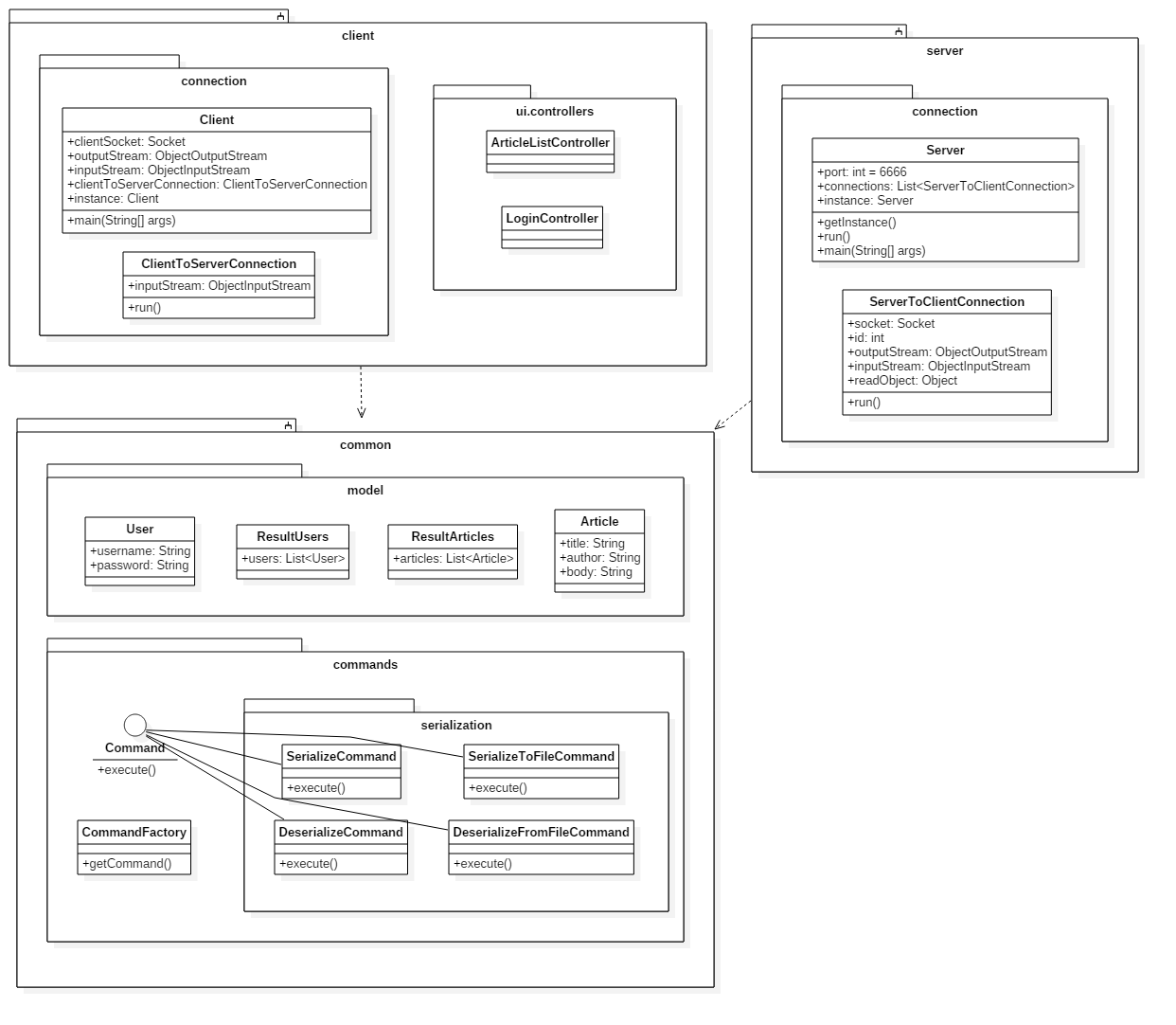
5. Class Design

**5.1 Design Patterns Description**

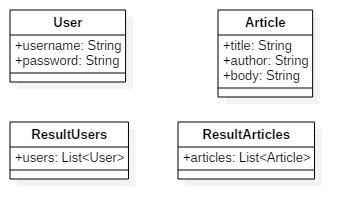
The factory pattern is used to create commands to be run on the server and return values.

The command pattern is used to execute commands from the client on the server, and on the client to process the server’s data.

**5.2 UML Class Diagram**

**

6. Data Model



7. System Testing

No tests are implemented at the creation of this document.

8. Bibliography

<https://github.com/buzea/SoftwareDesign2018>

<https://martinfowler.com/eaaCatalog/transactionScript.html>

<https://www.youtube.com/playlist?list=PL6gx4Cwl9DGBzfXLWLSYVy8EbTdpGbUIG>