Assignment 3

Student: Rednic Ana

**Group: 30432**

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 project consists of designing and implementing a client-server application for a news agency. The application has two types of users: the readers which can read articles, and the writers which can post, update, or delete articles.

# Functional Requirements

The system should allow a reader user to view a list of all available articles, to select and read one of them. The writer user should be able to view his own articles, to post an article, update or delete a selected article from the list.

# Non-functional Requirements

Performance: The system should have a minimal response time. Any operation should take less than 2 seconds.

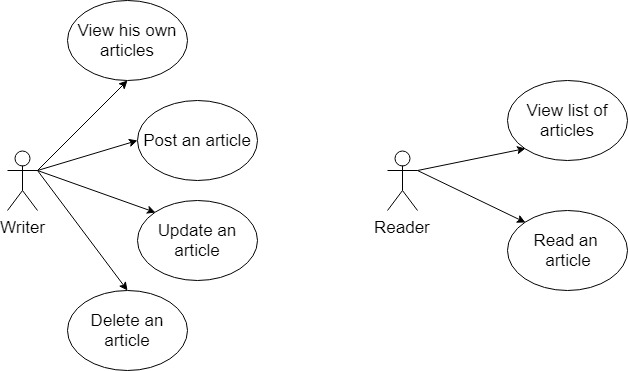
Testability: The application will be tested using JUnit tests.

Usability: The application is user-friendly. It has a graphical user interface and all possible operations for a reader or a writer are displayed.

Security: The system is secured with log-in for writers.

2. Use-Case Model

**Use case diagrams for Writer and Reader**



**Use case: Update an article**

Primary actor: Writer

Main success scenario: The writer logs in the application and a writer interface shows up. He sees a list of his articles, selects one of them, makes the desired modifications and clicks Update. The list should update in real time.

Extensions: The writer might forget the credentials for log-in.

3. System Architectural Design

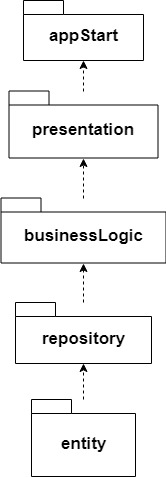
**3.1 Architectural Pattern Description**

Client/server architecture is a computing model in which the server hosts, delivers and manages most of the resources and services to be consumed by the client. This type of architecture has one or more client computers connected to a central server over a network or internet connection. This system shares computing resources.

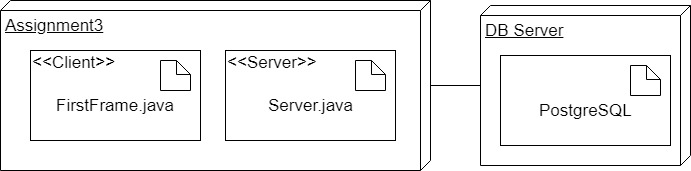
In this project, there is a server that receives a command, some information, and executes that command using the information. The client is a new FirstFrame, in which the user selects if he is a Reader, a Writer, or if he wants to register himself. In the case he wants to register, he input a username, a password, and a name for the new Writer to be registered and he clicks on Register. At this step, the server receives the information as a JSON String, parses it, creates a new Writer having that information and inserts it in the database. Finally, it sends a message to the Client that the Writer was registered successfully.

**3.2 Diagrams**

**Package Diagram**

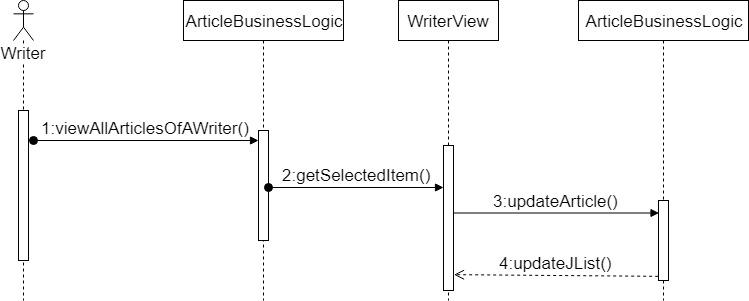
****

**Deployment Diagram**

****

4. UML Sequence Diagrams

**Sequence diagram for updating an article**

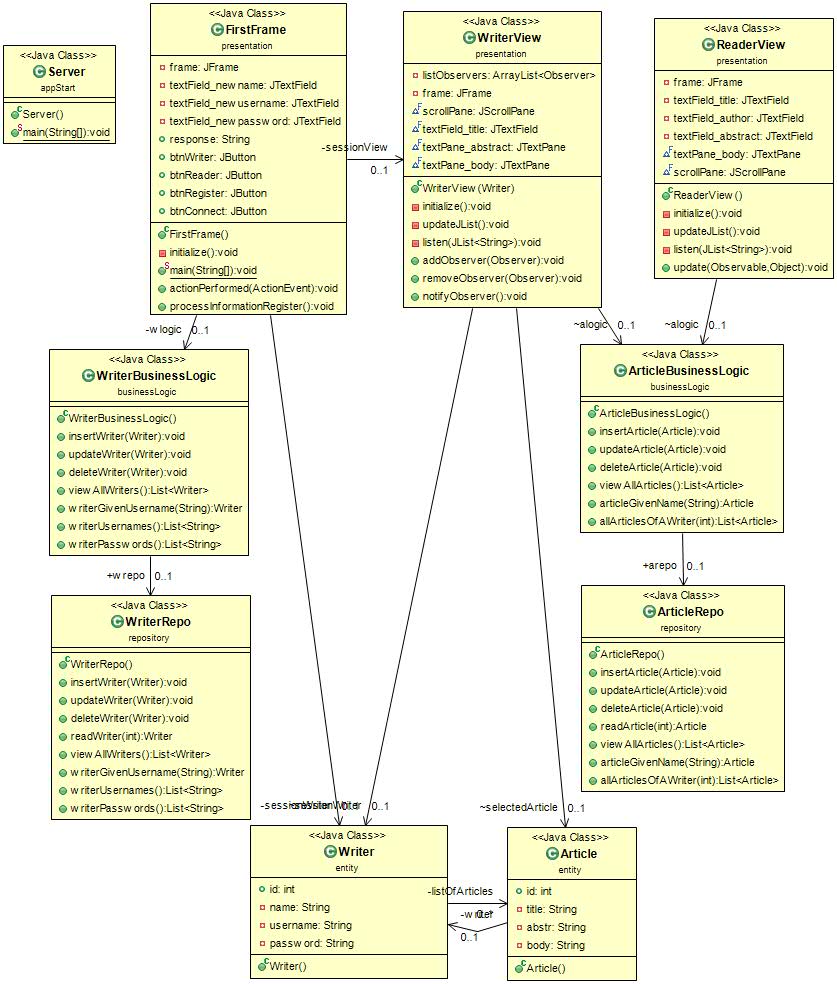


5. Class Design

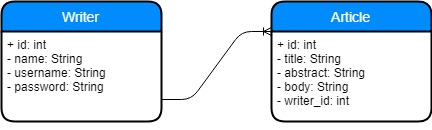
**5.1 Design Patterns Description**

The Observer pattern is used in this project because there is a one-to-many relationship between a Writer and his Articles. When a Writer posts, updates, or deletes an Article, the Readers must be notified about the change and the list of articles that they see must be updated. So, the WriterView extends the Observable class and ReaderView implements the Observer interface. When a Writer clicks on one of the buttons Post Article, Update Article, or Delete Article, the notifyObserver() method is called. This method updates the list of all articles in each ReaderView.

**5.2 UML Class Diagram**



6. Data Model

**

7. System Testing

Unit testing will be performed on each component of the system, validating it. The focus falls on the tests that impact the behavior of the system. This will be done using Dataflow testing.

In the Integration testing part, individual units will be combined and tested as a group. This will look forward to expose defects in the interfaces and in the interaction between integrated components.

The System testing part will verify if the whole application meets the specified requirements.

Validation testing will be performed to decide whether or not the application is ready to be made available to the end-users. The last three stages will be done using Black Box testing.

8. Bibliography

<https://www.techopedia.com/definition/438/clientserver-architecture>

<https://www.geeksforgeeks.org/introducing-threads-socket-programming-java/>

<https://dzone.com/articles/observer-pattern-java>

<https://l.facebook.com/l.php?u=https%3A%2F%2Fwww.geeksforgeeks.org%2Fintroducing-threads-socket-programming-java%2F&h=ATOT0WZk0SMXnL-m81V7agOxMrgcHXEAAIMt5sVu8MSb-N4fTOW851goX1V1VR9l34tbLd3DpVfGGEbCAmaK7OTYCG7cpushpbE4SE8HHMSUJjKsddTmYw>

<http://softwaretestingfundamentals.com/>

<https://www.tutorialspoint.com/software_testing_dictionary/data_flow_testing.htm>