Analysis and Design Document

Student:Ispas Ovidiu Mihai

**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

Use Java/C# API to design and implement a client-server application for a news agency. The application has three types of users: the readers, the writers and an administrator. The **readers** can view a list of articles, read an article and do not need to login in order the use the application.

The **writers** need to authenticate in order to create, update or delete articles. The **admin** is the only one who can create writer accounts, but cannot create new admin accounts. So the admin accounts are preset by the application developer and cannot be altered.

An article has the following components:

● Title

● Abstract

● Author

● Body

● List of related articles

When reading an article the user should be able to see the title and and the abstract of the related articles. By clicking on the title of the related article, he will be taken to a page that displays the full article.

The application must support multiple concurrent users. If a writer posts a new article, the

readers must see it in the list of articles in real time, without performing any refresh operation.

# Functional Requirements

* Client and administrator have the possibility to log in
* Clients can view the available articles, even without logging in
* Clients can open every article from the article list
* Clients can open every article from the related article list of every article
* Clients can view the updates in the list of articles in real time
* Writers (the clients that have to log in) can create update and delete articles
* Administrator can view the already created writers’ accounts
* Administrator can create, update and delete the writer account

# Non-functional Requirements

- The application have to be a client-server one

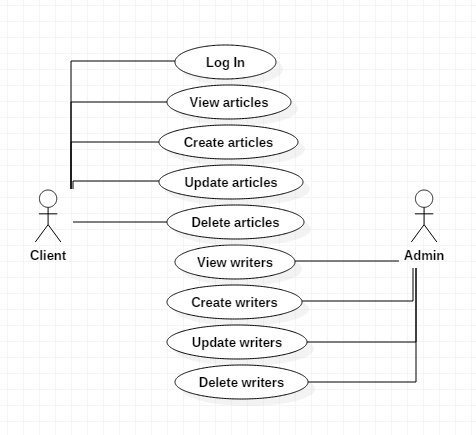
- The data will be sent between client and server by using json serialization

- Use Client-Server architectural design pattern

- All the inputs will be validated

- Persist the data into json files

2. Use-Case Model



Use case : Create a writer

Level : user-goal level

Primary actor : Administrator

Main success scenario :

1. Complete the log in fields with prerequisites

2. Click on Log in button

3. Complete all the fields necessary : Name, Age, Username and Password

4. Click on Insert button

Extensions :

User didn’t write the prerequisites right in the log in fields :

1. Complete the log in fields with prerequisites
2. Click on Log in button
3. Complete the log in fields with the right prerequisites
4. Click on Log in button

5. Complete all the fields necessary : Name, Age, Username and Password

6. Click on Insert button

User didn’t complete all the necessary fields for creating a player :

1. Complete the log in fields with prerequisites

2. Click on Log in button

3. Complete all the fields necessary : Name, Age, Username and Password

4. Click on Insert button

5. Click on the wrong created writer and select the Delete button

6. Complete all the fields necessary

7. Click on Insert button

3. System Architectural Design

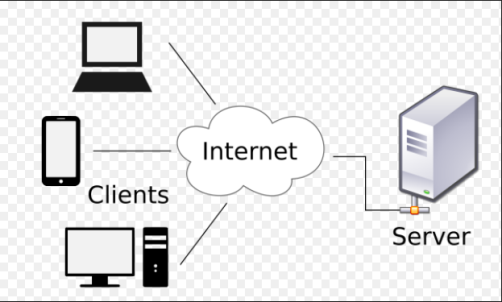
**3.1 Architectural Pattern Description**

The architectural design pattern used for this project is the Client-Server architecture which 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.

Client-Server architecture is also known as a networking computing model or client/server network because all the requests and services are delivered over a network.

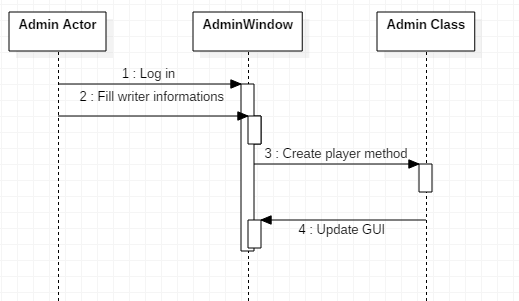
Another design pattern that I used is the observer design pattern which is a pattern I used in order to update the graphical user interface with the updated data whenever an action that requires it is done.

**3.2 Diagrams**



4. UML Sequence Diagrams

Scenario : Administrator creates a new writer account

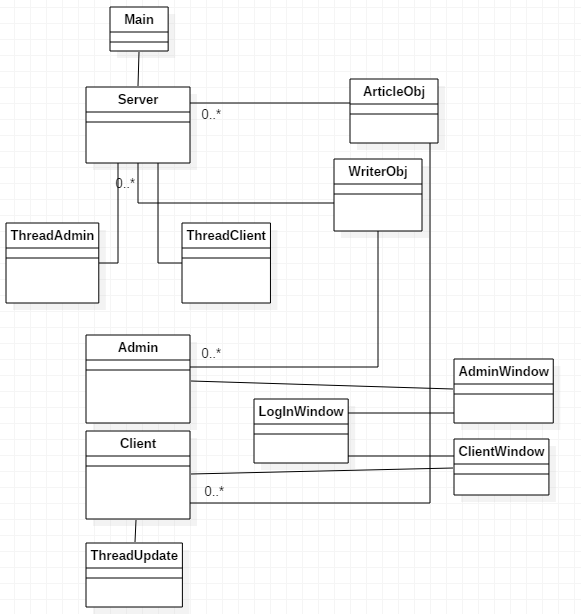


5. Class Design

**5.1 Design Patterns Description**

A design pattern I used for updating the GUI is the Observer design pattern . The observer pattern is a [software design pattern](https://en.wikipedia.org/wiki/Design_pattern_(computer_science)) in which an [object](https://en.wikipedia.org/wiki/Object_(computer_science)#Objects_in_object-oriented_programming), called the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their [methods](https://en.wikipedia.org/wiki/Method_(computer_science)).

* 1. **UML Class Diagram**



1. Data Model

Entities :

* ArticleObj – arguments : title, author, abstract, body, list of related articles
* WriterObj – arguments : name, age, username, password

The data which is sent back and forth between the server and the clients is saved into strings by using json serialization. The data persistence is done as well using json serialization into json files. Json serialization is the process of converting objects into json objects.

1. System Testing

8. Bibliography