Blogging Application

Student: Andreea Ionutas

**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

The application has 2 types of users: the readers and the writers. 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. So the writer accounts are preset by the application developer and cannot be altered.

# Functional Requirements

* The application must be client-server.
* Use the Observer design pattern for updating the list of articles in real time
* For sending data from the client to the server use JSON serialization.
* When writing an article, show a list that supports multi-select for choosing the related articles.

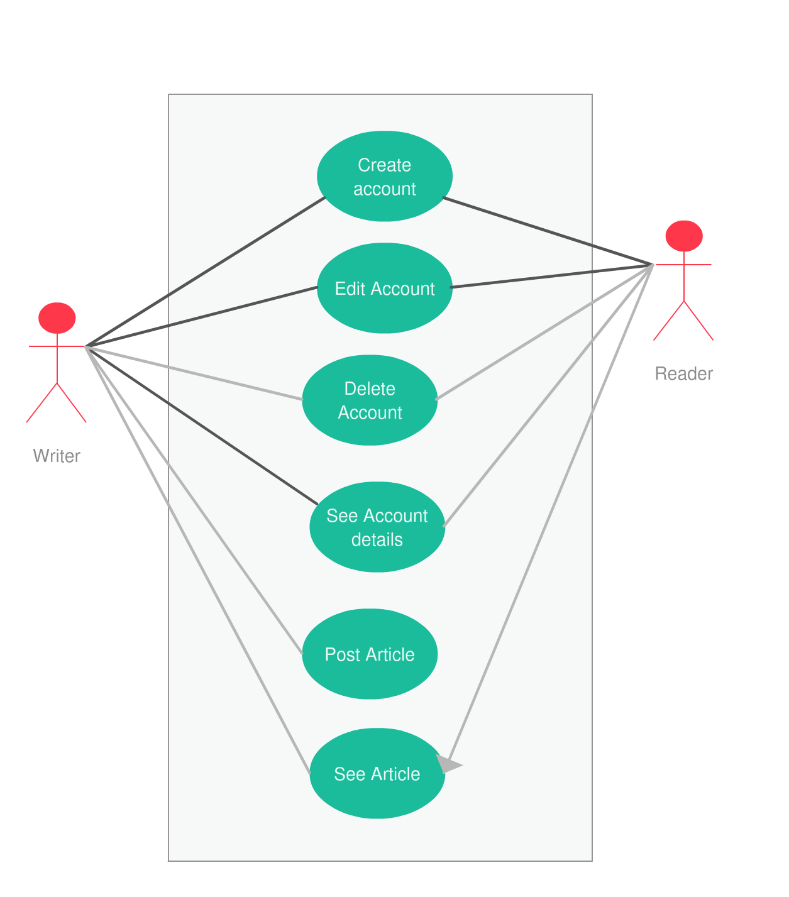
# 1.3 Non-functional Requirements

• Adaptability: the system should be able to adapt itself fast and efficiently to any type of changes.

• Stability: most of the objects will be stable over time and will not need changes

• Reusability: the system can be used in various platforms/contexts.

2. Use-Case Model



***Use case:*** Write Article

*Level:* User Goal Level

*Primary actor:* Writer

*Main success scenario:*

* Log in into application
* Write new article
* Click on post article
* The article and notify the subscribers

*Extensions:* The subscribers were not notified about the new article.

3. System Architectural Design

**3.1 Architectural Pattern Description**

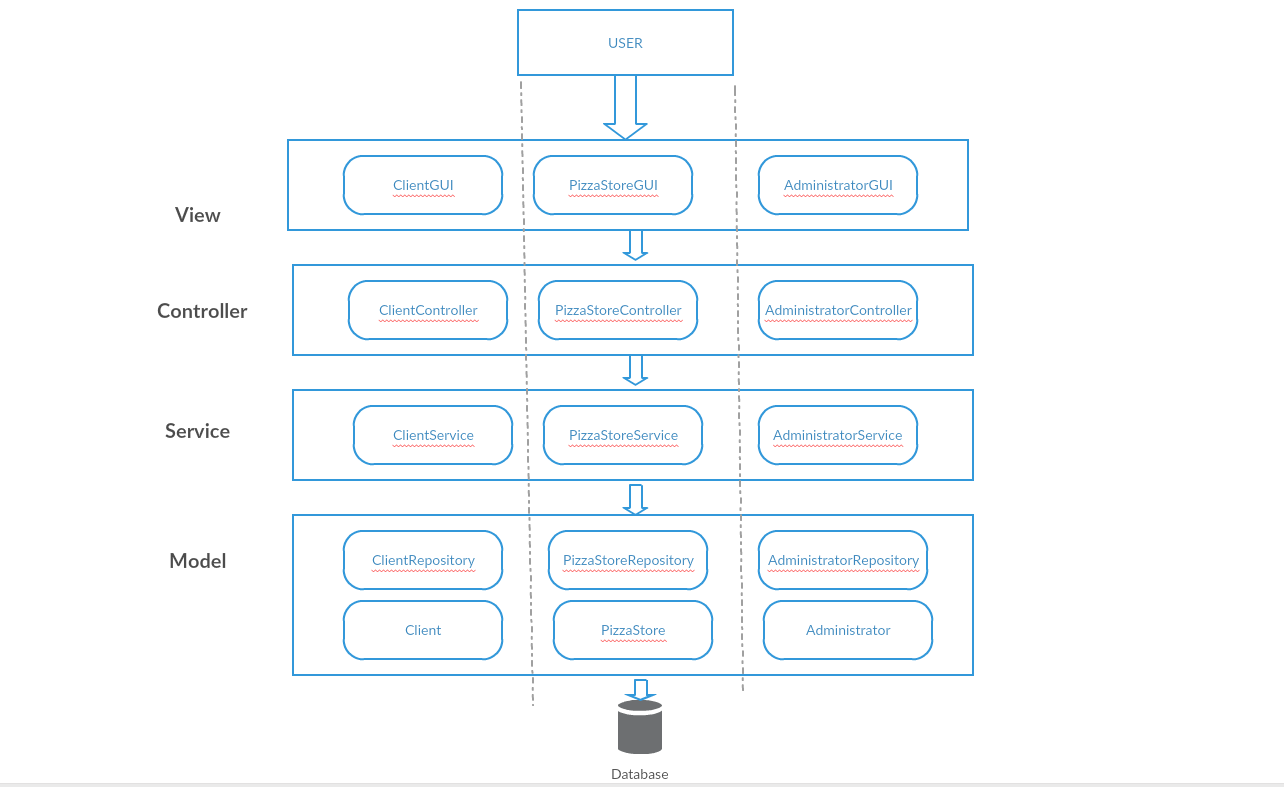
MVC Pattern stands for Model-View-Controller Pattern. This pattern is used to separate application's concerns.

• **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**



Reader GUI

Article

Writer

Reader

ArticleRepo

WriterRepopo

ReaderRepo

ArticleService

ReaderController

WriterController

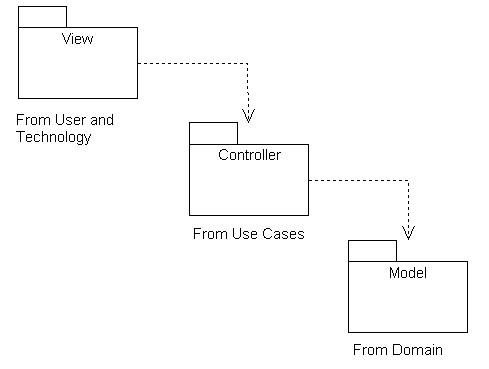
WriterService

ReaderService

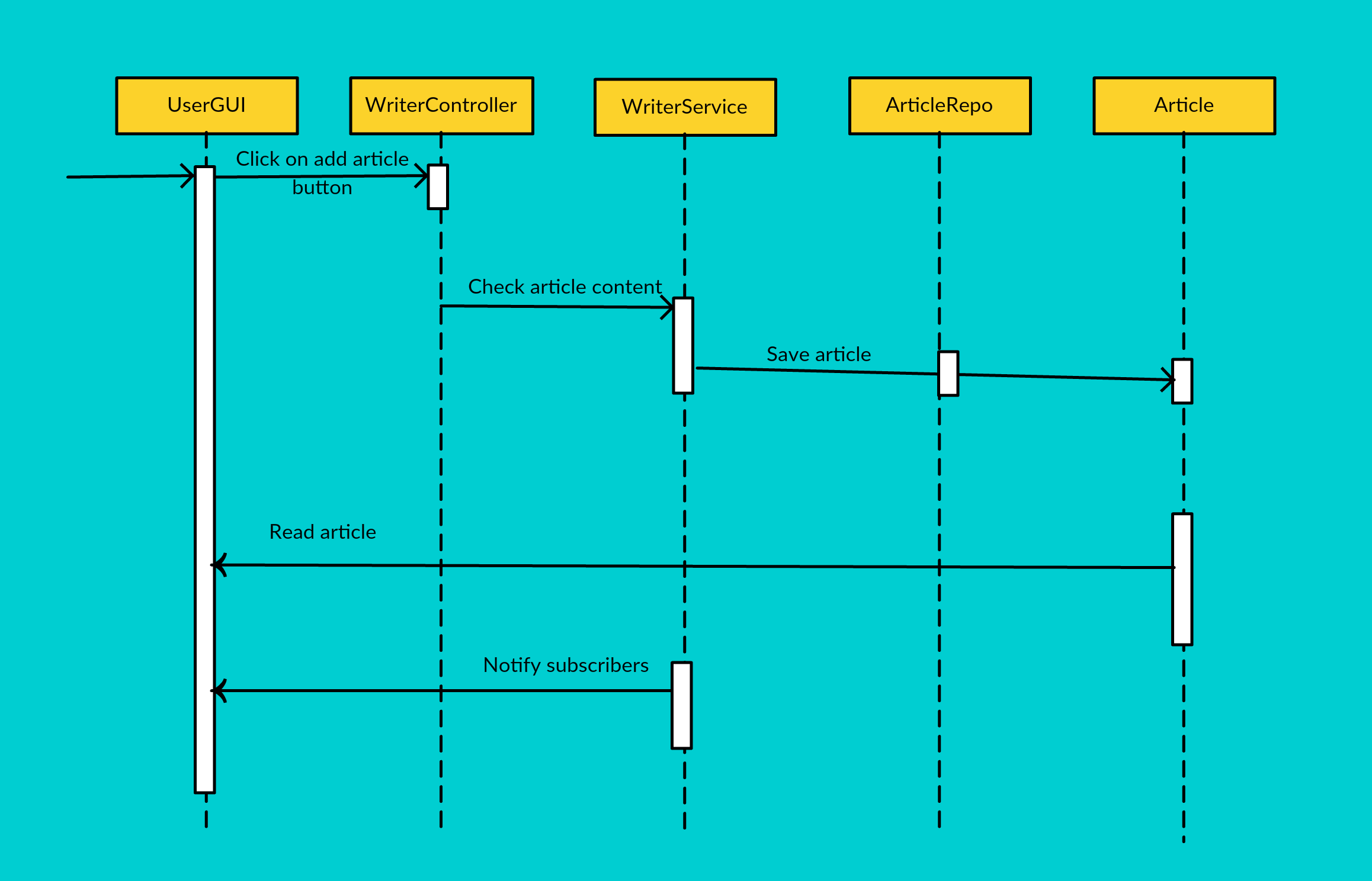
ArticleController

Article GUI

Writer GUI



4. UML Sequence Diagrams

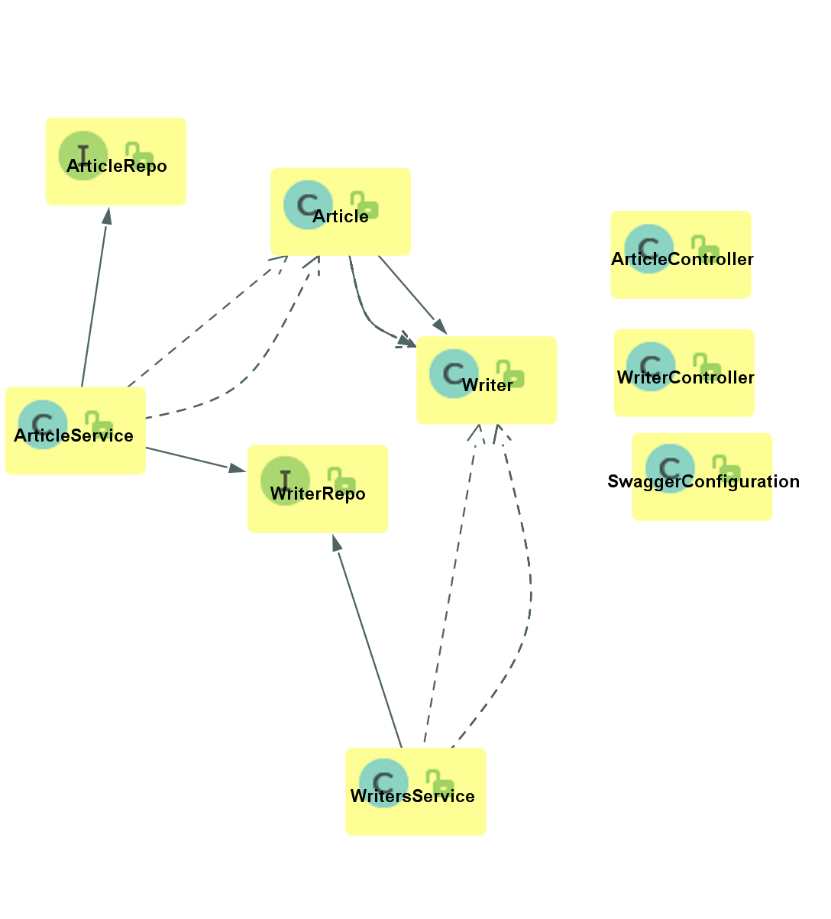
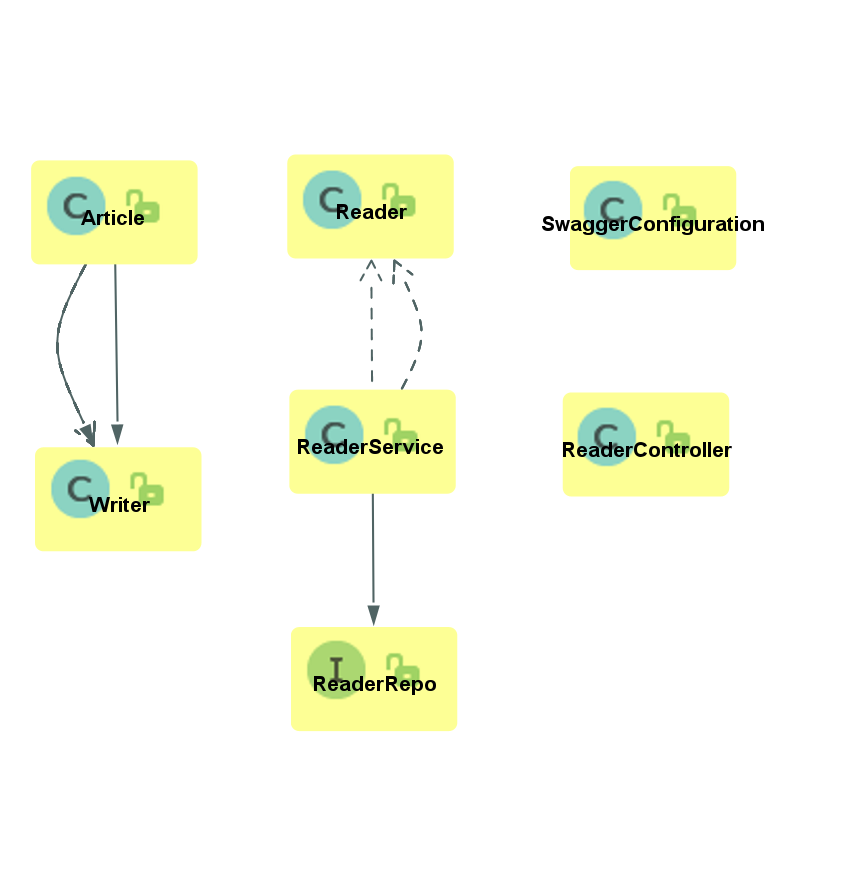


5. Class Design

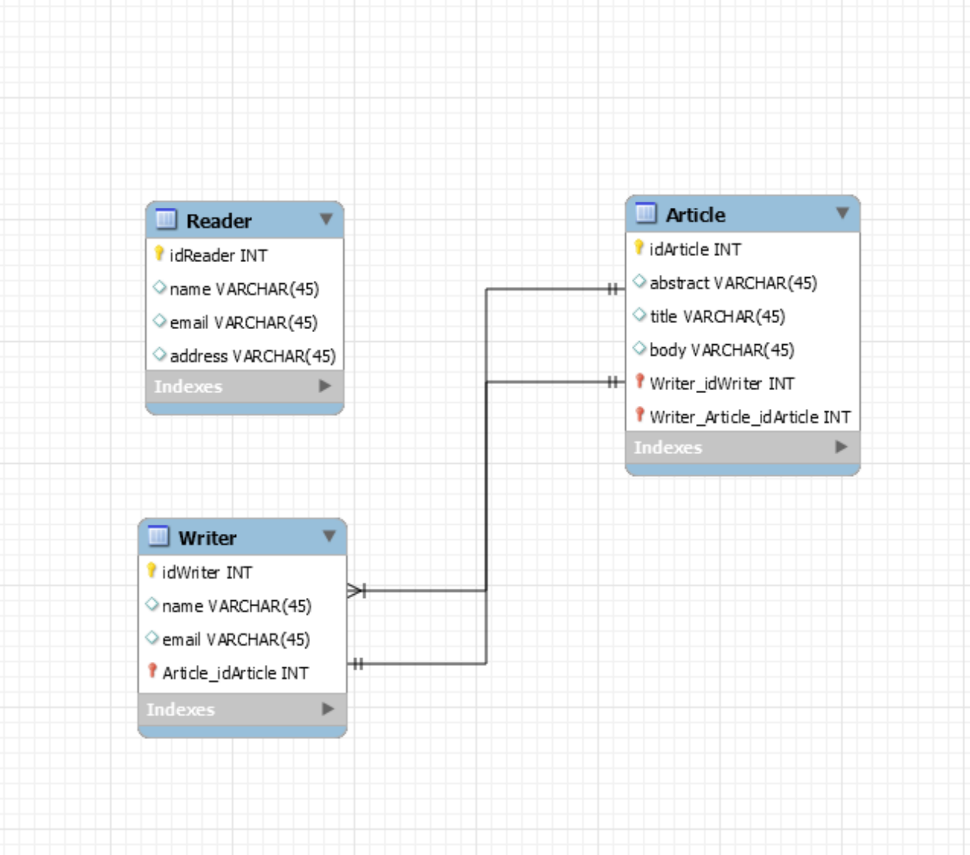
**5.1 Design Patterns Description**

Observer pattern is used when there is one-to-many relationship between objects such as if one object is modified, its dependent objects are to be notified automatically. Observer pattern falls under behavioral pattern category. Observer pattern uses three actor classes. Subject, Observer and Client. Subject is an object having methods to attach and detach observers to a client object. We have created an abstract class Observer and a concrete class Subject that is extending class Observer.

**5.2 UML Class Diagram**



6. Data Model



7. System Testing

The system will be tested mainly with unit tests. Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. For the unit testing, we will use Junit4 together with Mockito.

8. Bibliography

https://www.tutorialspoint.com/design\_pattern/observer\_pattern.htm