Informatics Institute of Technology
Department of Computing

Bsc(Hons) Artificial Intelligence and Data Science

Module: CM2601 Object Oriented Development

Module Coordinator: Ms. Malsha Fernando

Coursework Report

RGU Student ID	:	2331420
IIT Student ID	:	20231903
Student Name	:	H.M Viruna Vidaswin

Table of Contents

Contents

List of Figures

1. Introduction

1.1 Problem Identification

The vast amount of news content spread across numerous platforms today poses a significant challenge for users trying to find articles that truly match their interests. This often leads to frustration and inefficiency as individuals sift through irrelevant material. Moreover, with user preferences being highly diverse and constantly evolving, there is a pressing need for a system that can dynamically adapt and provide personalized recommendations. Additionally, as news platforms are accessed by multiple users simultaneously, efficient processing mechanisms become crucial to ensure timely and accurate suggestions. These challenges underscore the necessity of an advanced solution that can effectively address information overload and enhance user satisfaction.

2.1 Solution Overview

The Personalized News Recommendation System is designed to offer a flexible and scalable solution for delivering relevant content and recommendations to users. It integrates a Large Language Model (LLM) named Ollama (Ollama, 2024) to analyze both user preferences and article content, providing personalized article suggestions with high accuracy and relevance.
The system categorizes articles into various topics, such as Business, Health, Science, Travel, and Technology, ensuring users receive content aligned with their interests The system achieves efficiency while maintaining the flexibility to scale for multiple simultaneous user requests.

3.1 Project Scope

The project delivers a functional and scalable recommendation system that supports multiple users and effectively processing data delivering the best reccomendations for the user using an LLM (Large Language Model).

Objectives included:

1. **User Account Management:** Enables users to register/Log in, and track their activity
2. **Article Categorizing:** Categorizes articles added into the system
3. **Recommendation Engine:** Suggests articles based on a comparison of the user's reading history and a dataset of available articles, prioritizing the top matches based on content similarity
4. **Concurrency:** Supports concurrent user requests, ensuring smooth and efficient processing even when multiple users are interacting with the system simultaneously.
5. **Exception Handling:** Ensures the system gracefully handles issues such as invalid inputs, Database failures, and other errors, providing robust error handling and feedback.

4.1 TimeLine

Week 1	Research and Requirements Gathering	Identify project scope, define use cases, descriptions for the system.
Week 2	Research and Requirements Gathering	Defining Class diagrams, Sequence diagrams
Week 3	User Account Management Development	Implement user registration, login, and tracking of user activity and database handling
Week 4	Article Management Development	Implementation of Article managements.
Week 5	LLM installation & Setup	Set up LLM integration for recommendation engine.
Week 6	Concurrency Implementation	Integrate Java concurrency features to handle multiple user requests efficiently
Week 7	Testing, Error handling and debugging	Implement robust error handling and resolve issues discovered during testing

2. Requirement Analysis

2.1 Stakeholder identification with Requirements

2.1.1 General Users

These are users who interact with the system to explore articles and receive recommendations.

Functional Requirements:

- Ability to create user accounts and log in.
- Browsing through available articles.
- Personalized recommendations based on user preferences using a recommendation engine.

Non-Functional Requirements:

- A simple and intuitive user interface.
- Support for multiple users accessing the system concurrently.
- Optimized performance to reduce delays in generating recommendations.
- Reliable error and exception handling mechanisms.

2.1.2 System Administrators

System administrators oversee system performance, manage content, and address any issues

Functional Requirements:

- Managing articles within the system.
- Organize and update categories.
- Oversee and maintain the database.

Non-Functional Requirements:

- An intuitive and easy-to-use administrative interface.
- Restricted access to specific features and functionalities.
- Efficient performance to ensure smooth article and category management
- Robust error handling and exception management.

3.1 Usecase Diagram

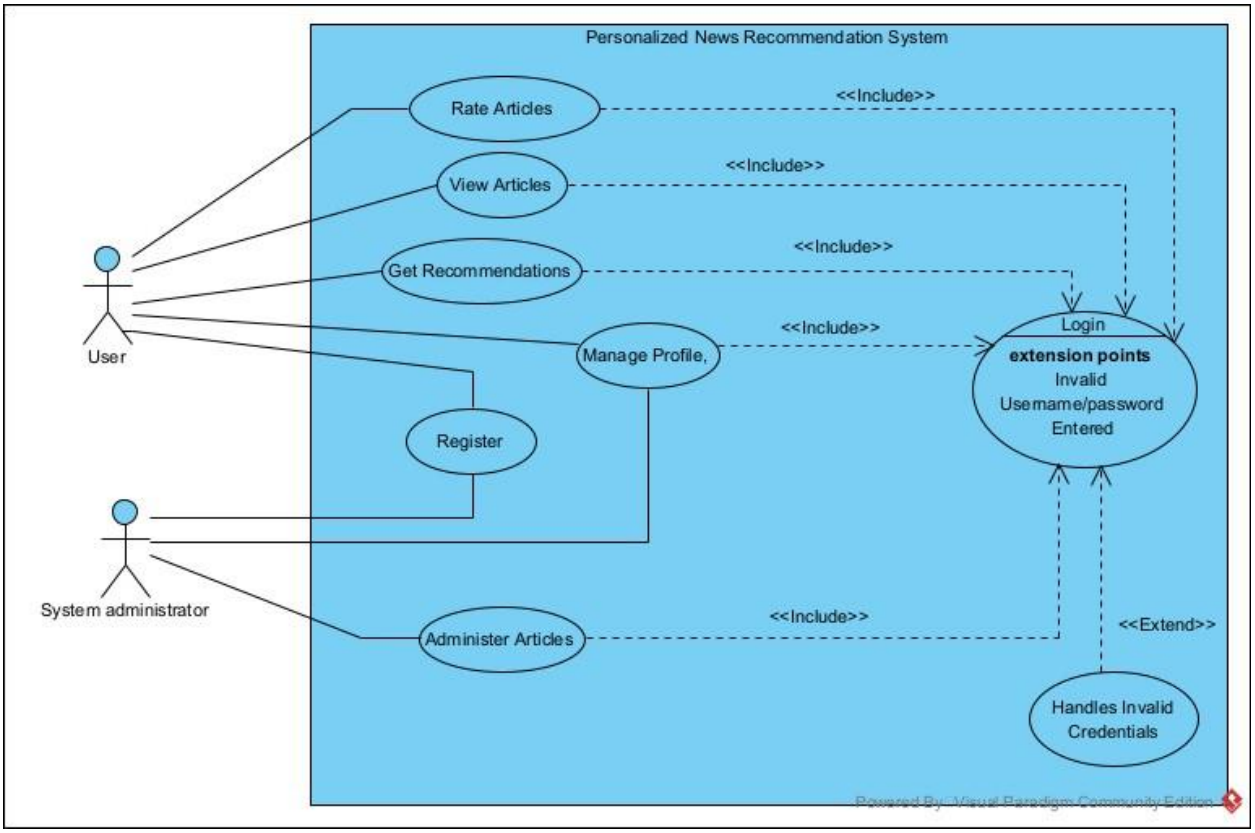


Figure 1 Use Case Diagram

4.1 Usecase Descriptions

4.1.1 Use Case: View Articles

Section: Main

System	Personalized News Recommendation System
Actor(s)	user
Purpose	Allows users to view available articles
Overview	
The use case begins when a user wants to view an article This use case allows user to rate articles This use case ends when the user stops viewing articles	
Preconditions	
The user must be registered and logged into the system Articles are available in the system	
Typical Course of Events	
Actor Action	**System Response**
1. User selects option to view articles	2. System prompts to enter category
3. User enters article category to view	4. System displays all articles relating to the category
	5. Prompts user to rate the article or exit
6. User rates or exits	7. Stores the user preferences
Post Condition	
Article is displayed	
Alternative Courses	
A1: Articles unavailable A2: User not logged in	
Related use cases	
Includes Login	

4.1.2 Use Case: Get Recommendations

Section: Main

System	Personalized News Recommendation System
Actor(s)	user
Purpose	Allows users to receive personalized article recommendations based on their reading history and preferences
Overview	
The use case begins when a user requests article recommendation The system analyzes the user's reading history and ratings to display recommended articles This use case ends when the recommendations are displayed to the user	
Preconditions	
The user must be registered and logged into the system Articles are available in the system and user has reading history or sufficient ratings	
Typical Course of Events	
Actor Action	**System Response**
1. User wants to get a recommendation	2. System retrieves the user's reading history and ratings.
	3. System generates most viable articles and Displays
Post Condition	
The user receives a list of personalized article recommendations	
Alternative Courses	
A1: Insufficient User preferences -User hasn't viewed or rated any articles A2: User not logged in	
Related usecases	
Includes login	

4.1.3 Use Case: Manage Profile

Section: Main

System	Personalized News Recommendation System
Actor(s)	User, System Administrator
Purpose	Allows actors to view and update user profile information, including personal details.
Overview	
The use case begins when actors choose to manage a user profile This use case enables actors to edit their profile or view Th is use case ends when the changes are saved or the actor decides to cancel the operation	
Preconditions	
The user must be registered and logged into the system	
Typical Course of Events	
Actor Action	**System Response**
1. User selects to manage profile	2. System displays the user's current profile information
3. User edits their profile information	4. System validates the input and prepares to save change
5. User confirms changes	6. System saves the updated profile
Post Condition	
The user’s profile is updated with the new information and displayed	
Alternative Courses	
A1: Invalid input data -system displays an error message and prompts for correction A2: User doesn’t make any changes and exists A3: User not logged in	
Related usecases	
Includes Login	

4.1.4 Use Case: Administer Articles

Section: Main

System	Personalized News Recommendation System
Actor(s)	System Administrator
Purpose	Allows the system administrator to manage the articles available in the system
Overview	
The use case begins when the system administrator wants to manage articles within the system This use case allows actor to add new articles, Update exsisting articles and delete articles This use case ends when the admin is done managing the articles	
Preconditions	
The user must be registered and logged into the system as an admin	
Typical Course of Events	
Actor Action	**System Response**
1. Administrator opts to manage articles	2. System prompts managing articles interface
3. Administrator selects an option (add a new, update existing or delete an article)	4. System does corresponding function and prompts to save the changes
5. Changes confirmed	6. Changes saved
Post Condition	
All changes are saved to data storage	
Alternative Courses	
A1: Cancels operation A2: Admin not logged in	
Related usecases	
Includes Login	

4.1.5 Use Case: Rate Articles

Section: Main

System	Personalized News Recommendation System
Actor(s)	User
Purpose	Allows the user to rate the article that they are currently viewing
Overview	
The use case begins when the user wants to rate the article This use case allows actor to like or dislike the article This use case ends when the user has rated or exited	
Preconditions	
The user must be registered and logged into the system	
Typical Course of Events	
Actor Action	**System Response**
1. User attempts to rate article	2. System prompts rating interface
3. User provides rating	4. System stores rating
Post Condition	
User rating successfully recorded	
Alternative Courses	
A1: User cancels rating A2: User attempts to rate the same article again A3: User not logged in	
Related usecases	
Includes Login	

4.1.6 Use Case: Register

Section: Main

System	Personalized News Recommendation System
Actor(s)	System Administrator, User
Purpose	Allows actors to register into the system
Overview	
The use case begins when the actor wants to register into the system This use case allows actor to register as an admin or normal user This use case ends when actor is successfully registered	
Preconditions	
Actors must create unique username	
Typical Course of Events	
Actor Action	**System Response**
1. Actors wants to register into the system	2. System prompts user details
3. Actor submits user details	4. System validates user details
	5. Stores the details
Post Condition	
Actor successfully registered into the system	
Alternative Courses	
A1: Invalid input data A2: Admin registration -Requires a unique code known to admins such that they can register	
Related usecases	

3. Activity Diagram

3.1 Register User/Admin

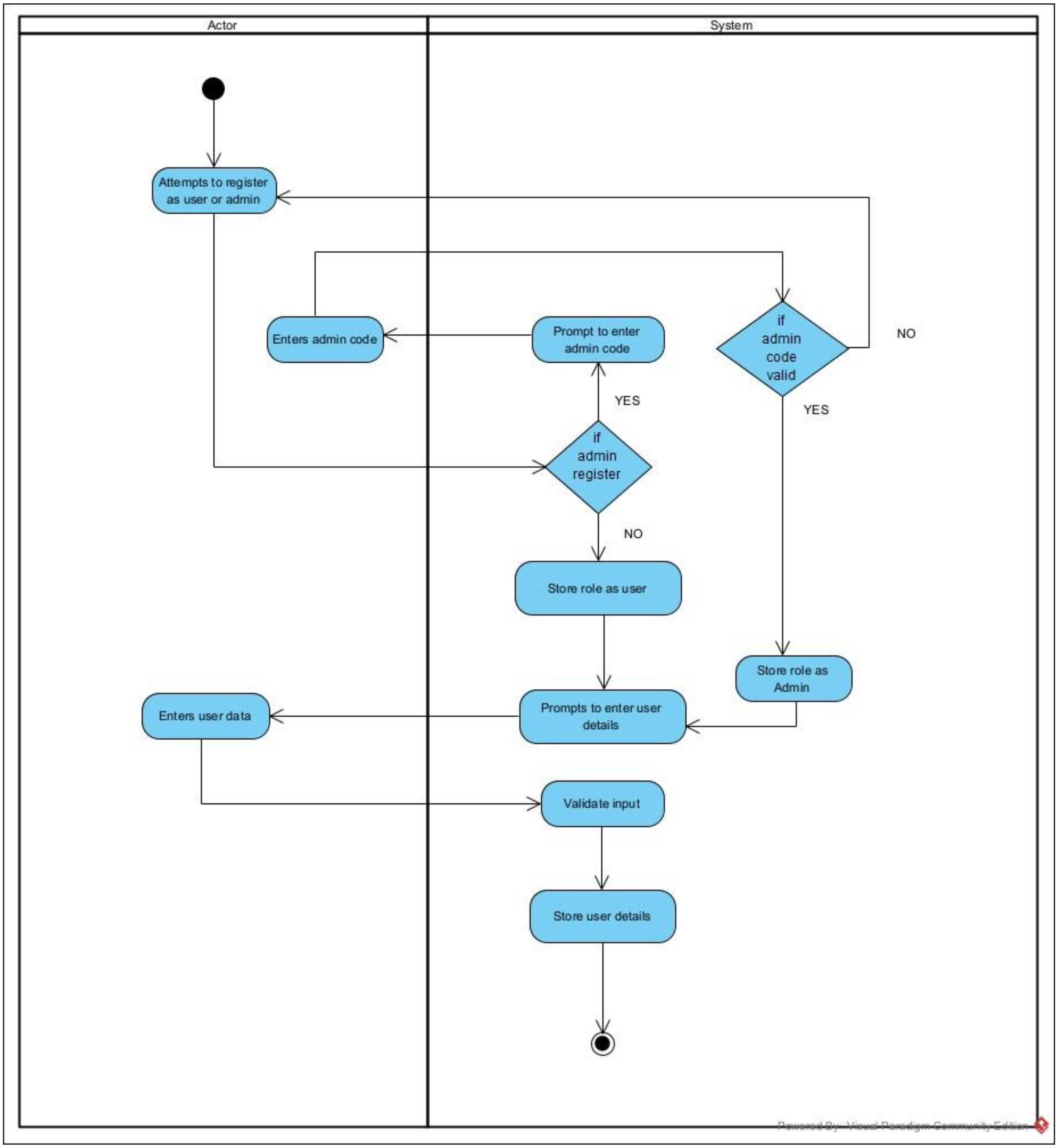


Figure 2 Register Activity

4.1 View Articles

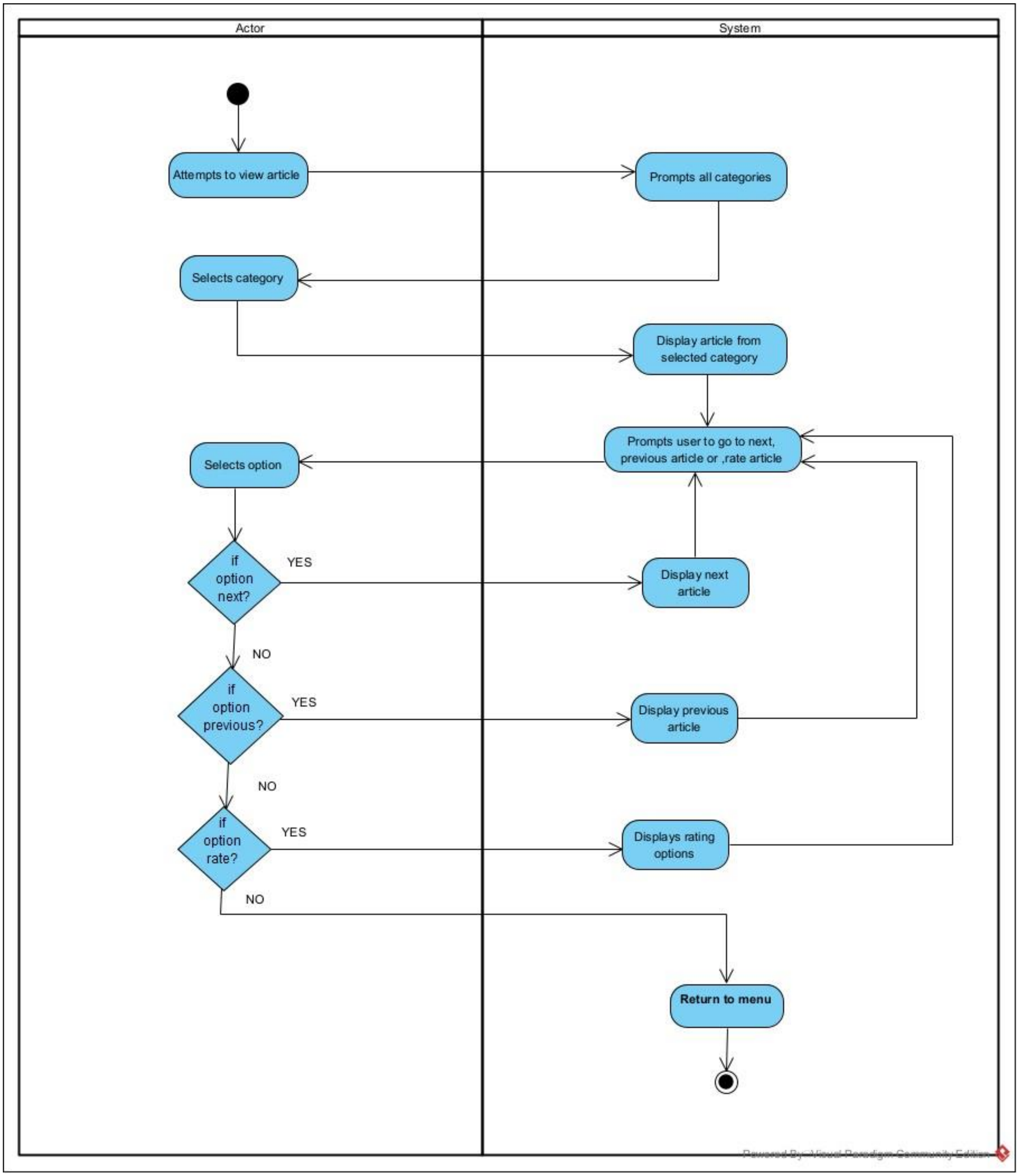


Figure 3 View Article Activity

5.1 Rate Articles

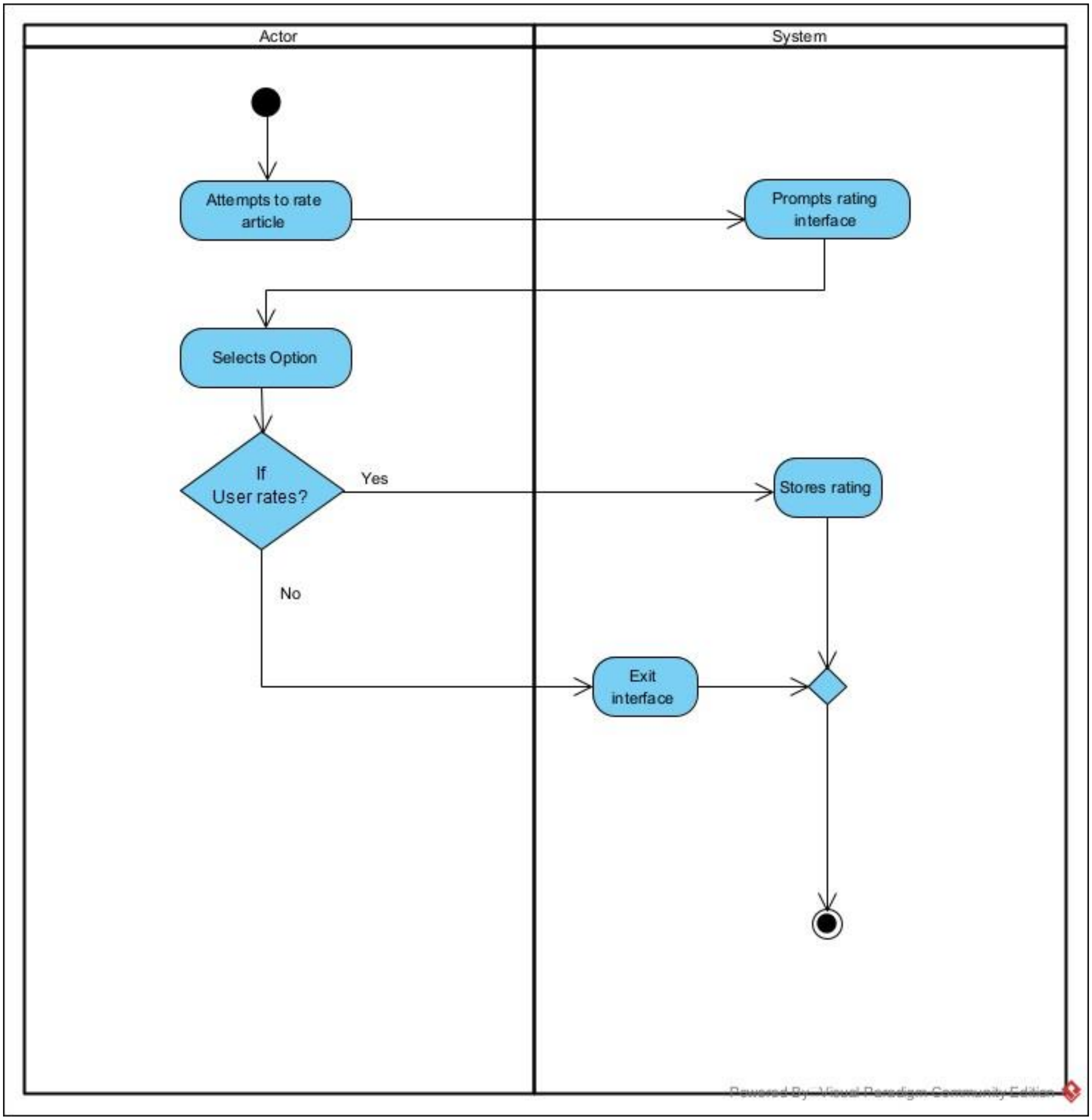


Figure 4 Rate Article Activity

6.1 User Manage Profile

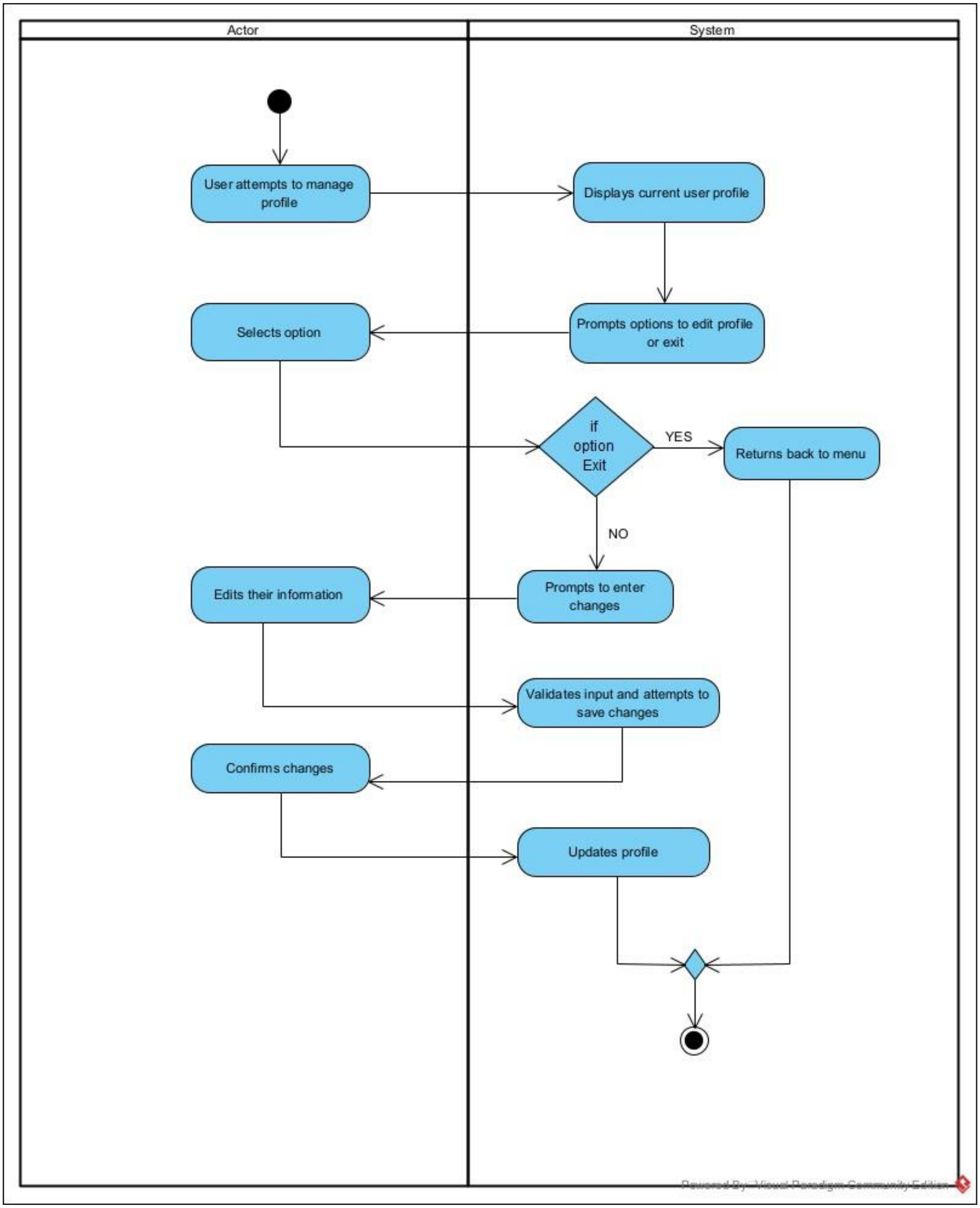

Actor
System
User attempts to manage profile
Displays current user profile
Prompts options to edit profile or exit
Selects option
if option Exit
YES
Returns back to menu
NO
Prompts to enter changes
Edits their information
Validates input and attempts to save changes
Confirms changes
Updates profile
Powered By: Visual Paradigm Community Edition

Figure 5 Manage Profile Activity

7.1 Get Recommendations

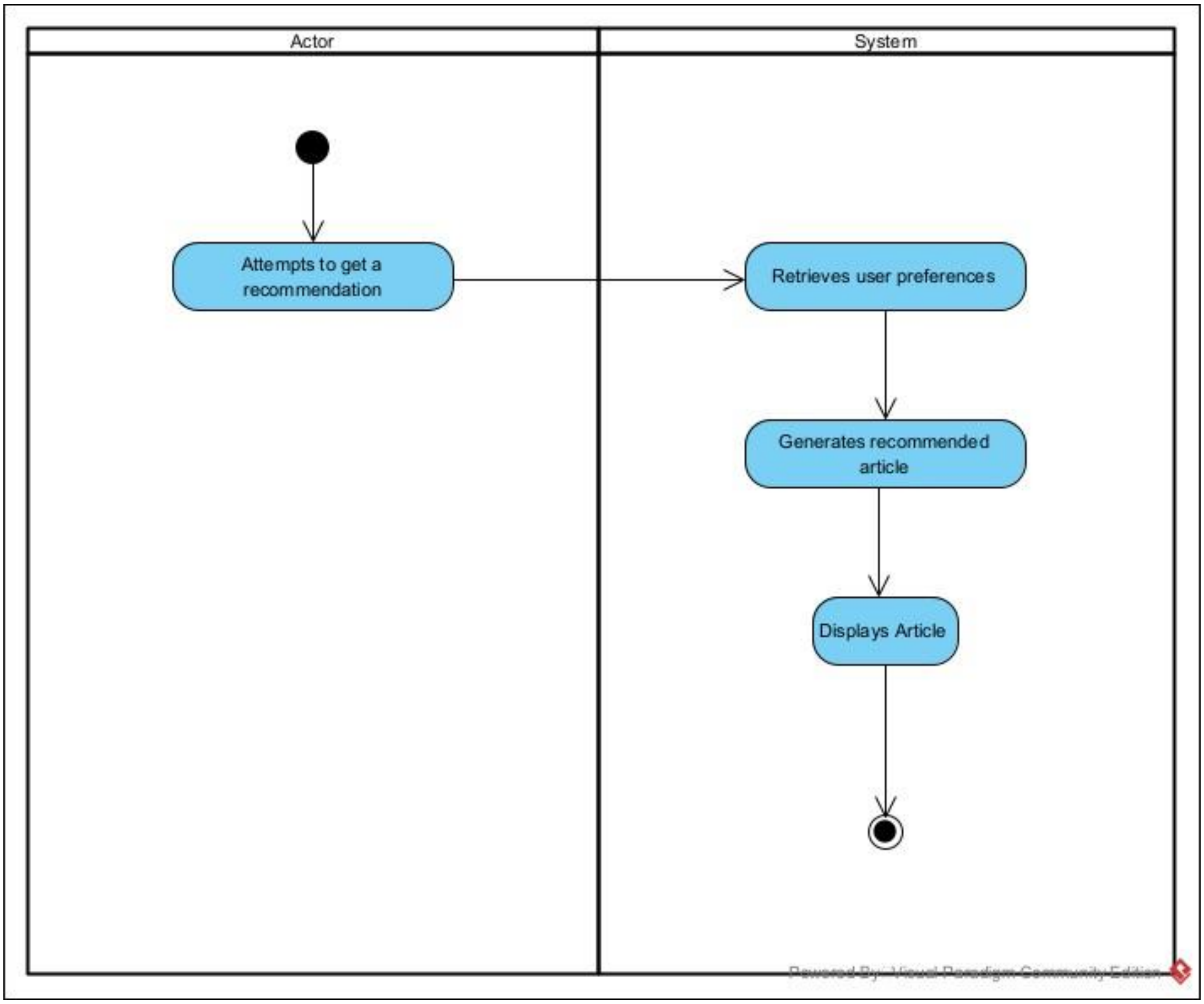

Actor
System
Attempts to get a recommendation
Retrieves user preferences
Generates recommended article
Displays Article
Powered By: Visual Paradigm Community Edition

Figure 6 Reccomendation Activity

8.1 Administer Articles

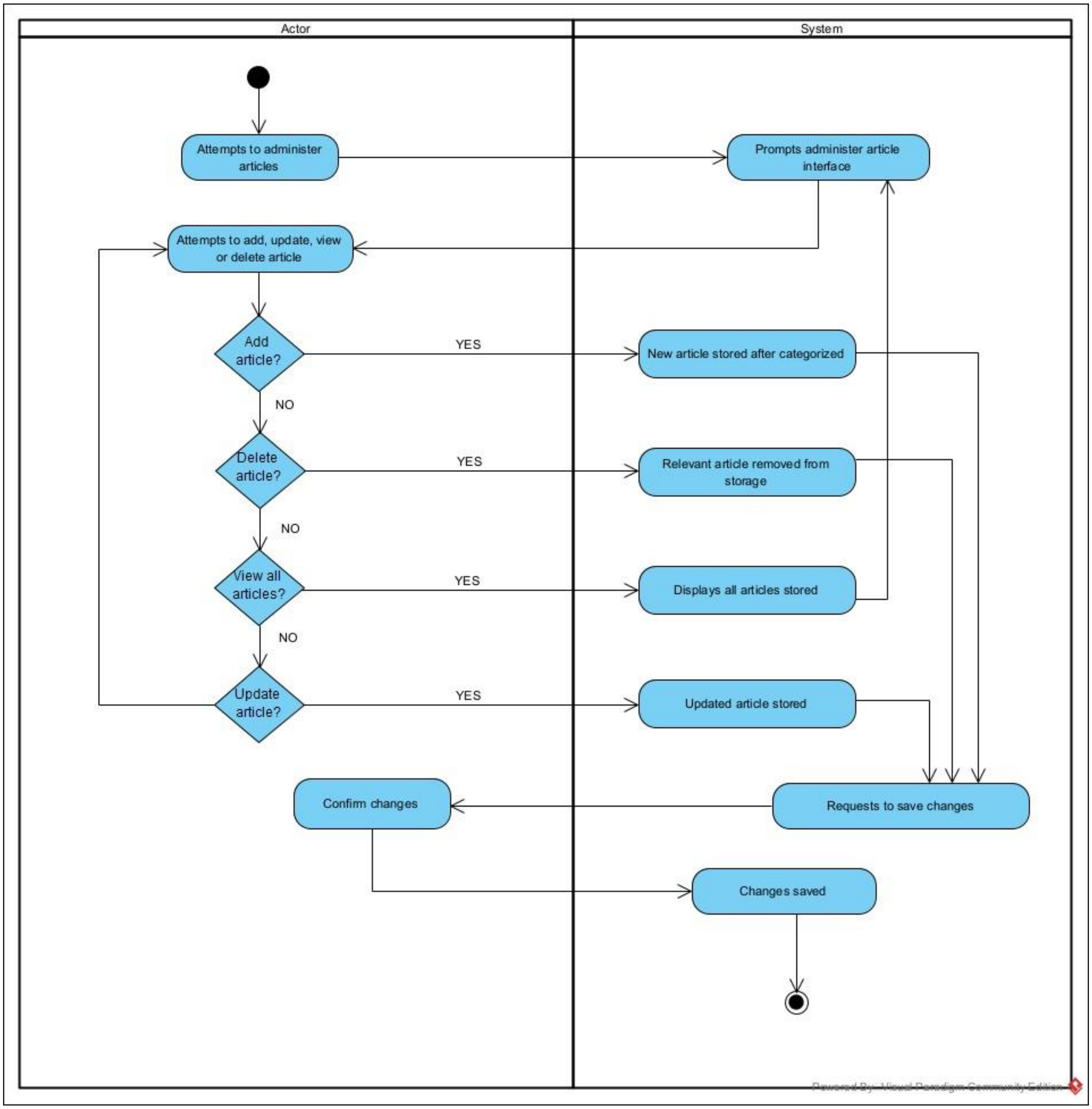

Actor
System
Attempts to administer articles
Prompts administer article interface
Attempts to add, update, view or delete article
Add article?
YES
New article stored after categorized
NO
Delete article?
YES
Relevant article removed from storage
NO
View all articles?
YES
Displays all articles stored
NO
Update article?
YES
Updated article stored
Confirm changes
Requests to save changes
Changes saved
Powered By: Visual Paradigm Community Edition

Figure 7 Administer Activity

4. Class Diagram

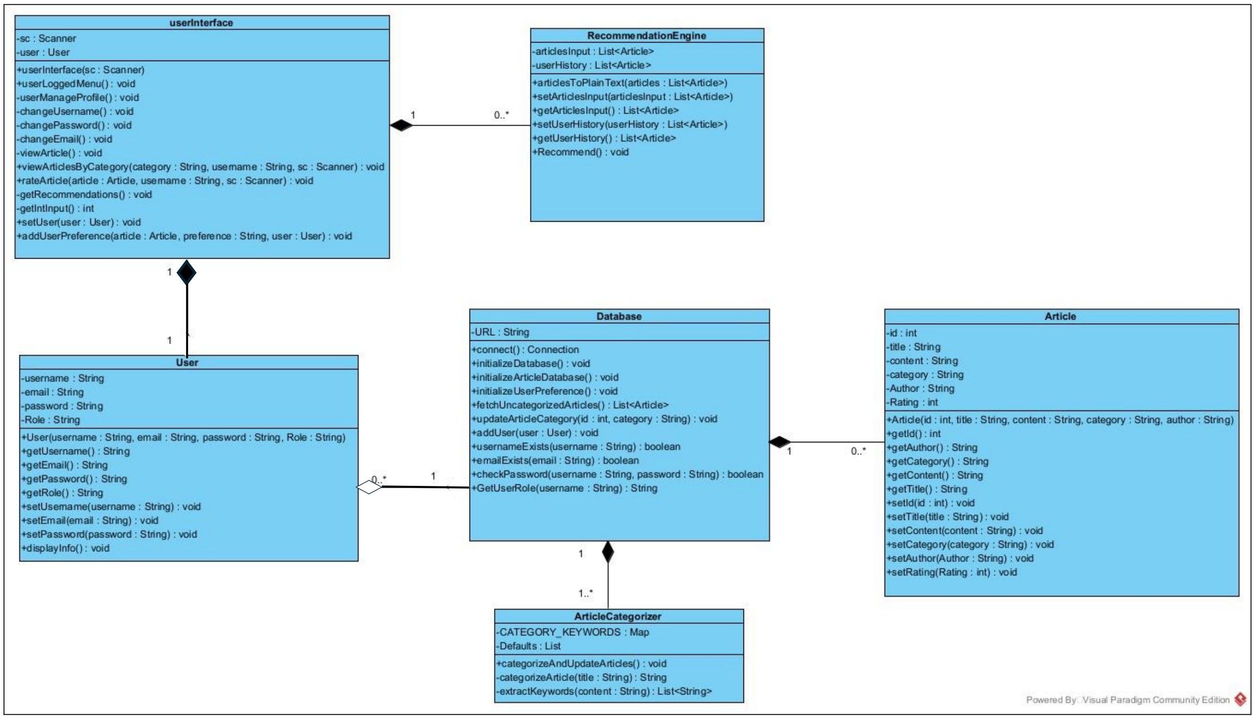

userInterface
-sc : Scanner
-user : User
+userInterface(sc : Scanner)
+userLoggedMenu() : void
-userManageProfile() : void
-changeUsername() : void
-changePassword() : void
-changeEmail() : void
-viewArticle() : void
+viewArticlesByCategory(category : String, username : String, sc : Scanner) : void
+rateArticle(article : Article, username : String, sc : Scanner) : void
-getRecommendations() : void
-getIntInput() : int
+setUser(user : User) : void
+addUserPreference(article : Article, preference : String, user : User) : void
1
0..*
RecommendationEngine
-articlesInput : List<Article>
-userHistory : List<Article>
+articlesToPlainText(articles : List<Article>)
+setArticlesInput(articlesInput : List<Article>)
+getArticlesInput() : List<Article>
+setUserHistory(userHistory : List<Article>)
+getUserHistory() : List<Article>
+Recommend() : void
1
1
User
-username : String
-email : String
-password : String
-Role : String
+User(username : String, email : String, password : String, Role : String)
+getUsername() : String
+getEmail() : String
+getPassword() : String
+getRole() : String
+setUsername(username : String) : void
+setEmail(email : String) : void
+setPassword(password : String) : void
+displayInfo() : void
0..*
1
Database
-URL : String
+connect() : Connection
+initializeDatabase() : void
+initializeArticleDatabase() : void
+initializeUserPreference() : void
+fetchUncategorizedArticles() : List<Article>
+updateArticleCategory(id : int, category : String) : void
+addUser(user : User) : void
+usernameExists(username : String) : boolean
+emailExists(email : String) : boolean
+checkPassword(username : String, password : String) : boolean
+GetUserRole(username : String) : String
1
0..*
Article
-id : int
-title : String
-content : String
-category : String
-Author : String
-Rating : int
+Article(id : int, title : String, content : String, category : String, author : String)
+getId() : int
+getAuthor() : String
+getCategory() : String
+getContent() : String
+getTitle() : String
+setId(id : int) : void
+setTitle(title : String) : void
+setContent(content : String) : void
+setCategory(category : String) : void
+setAuthor(Author : String) : void
+setRating(Rating : int) : void
1
1..*
ArticleCategorizer
-CATEGORY_KEYWORDS : Map
-Defaults : List
+categorizeAndUpdateArticles() : void
-categorizeArticle(title : String) : String
-extractKeywords(content : String) : List<String>
Powered By Visual Paradigm Community Edition

Figure 8 Class Diagram

5. Sequence Diagrams

5.1 Login

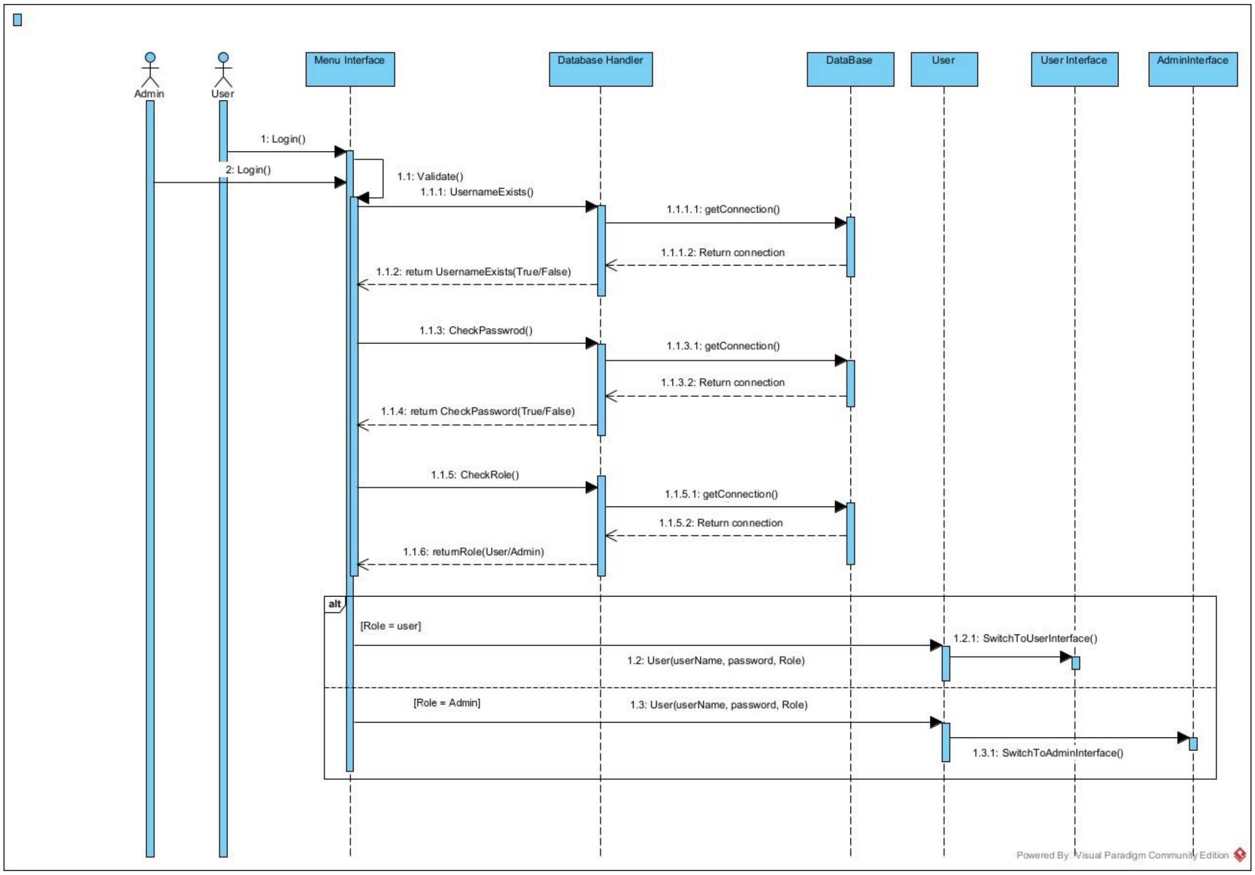

Admin
User
Menu Interface
Database Handler
DataBase
User
User Interface
AdminInterface
1: Login()
2: Login()
1.1: Validate()
1.1.1: UsernameExists()
1.1.1.1: getConnection()
1.1.1.2: Return connection
1.1.2: return UsernameExists(True/False)
1.1.3: CheckPasswrod()
1.1.3.1: getConnection()
1.1.3.2: Return connection
1.1.4: return CheckPassword(True/False)
1.1.5: CheckRole()
1.1.5.1: getConnection()
1.1.5.2: Return connection
1.1.6: returnRole(User/Admin)
alt
[Role = user]
1.2.1: SwitchToUserInterface()
1.2: User(userName, password, Role)
[Role = Admin]
1.3: User(userName, password, Role)
1.3.1: SwitchToAdminInterface()
Powered By: Visual Paradigm Community Edition

Figure 9 Login Sequence

5.2 Register

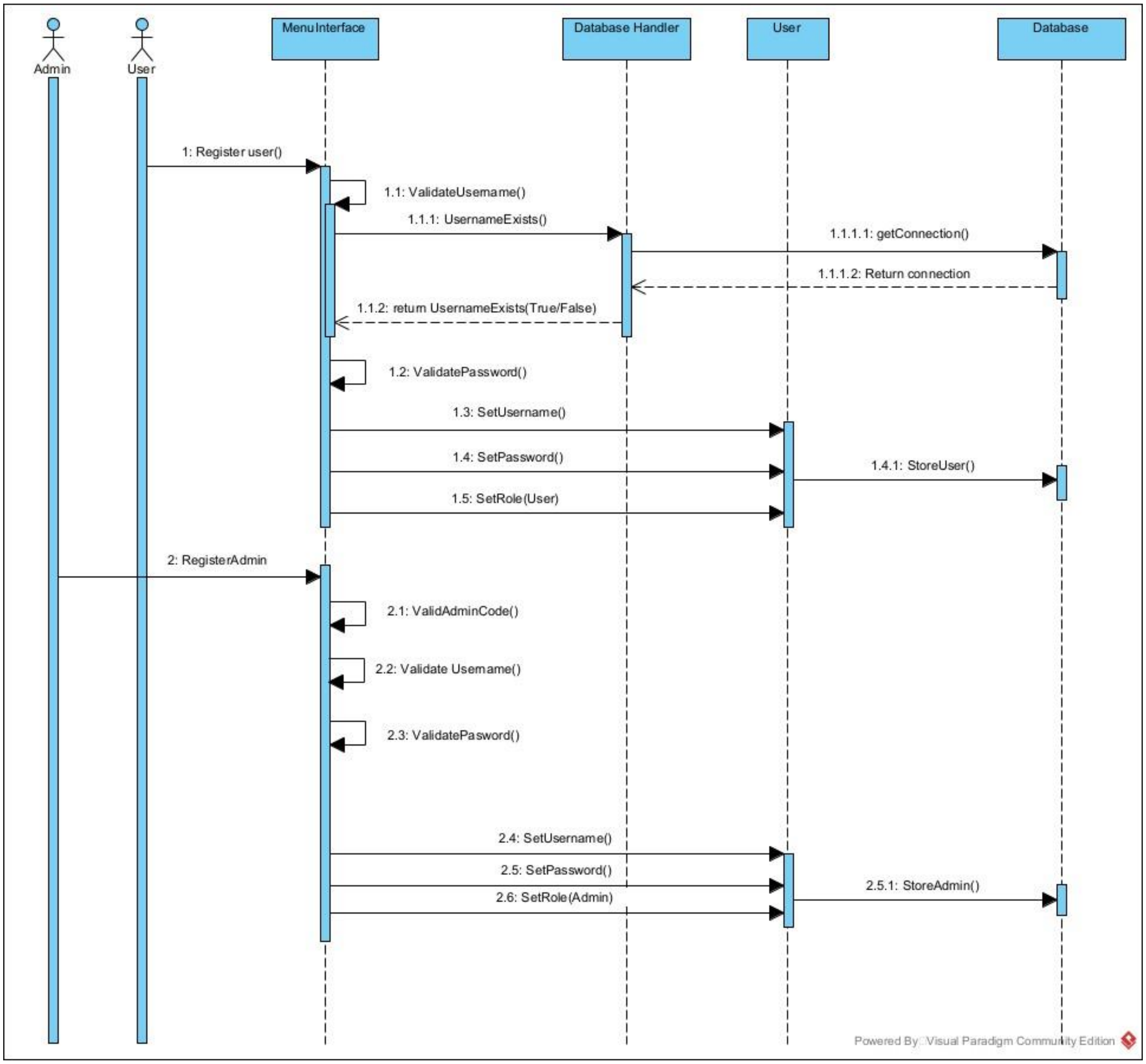


Figure 10 Register Sequence

5.3 User Interactions

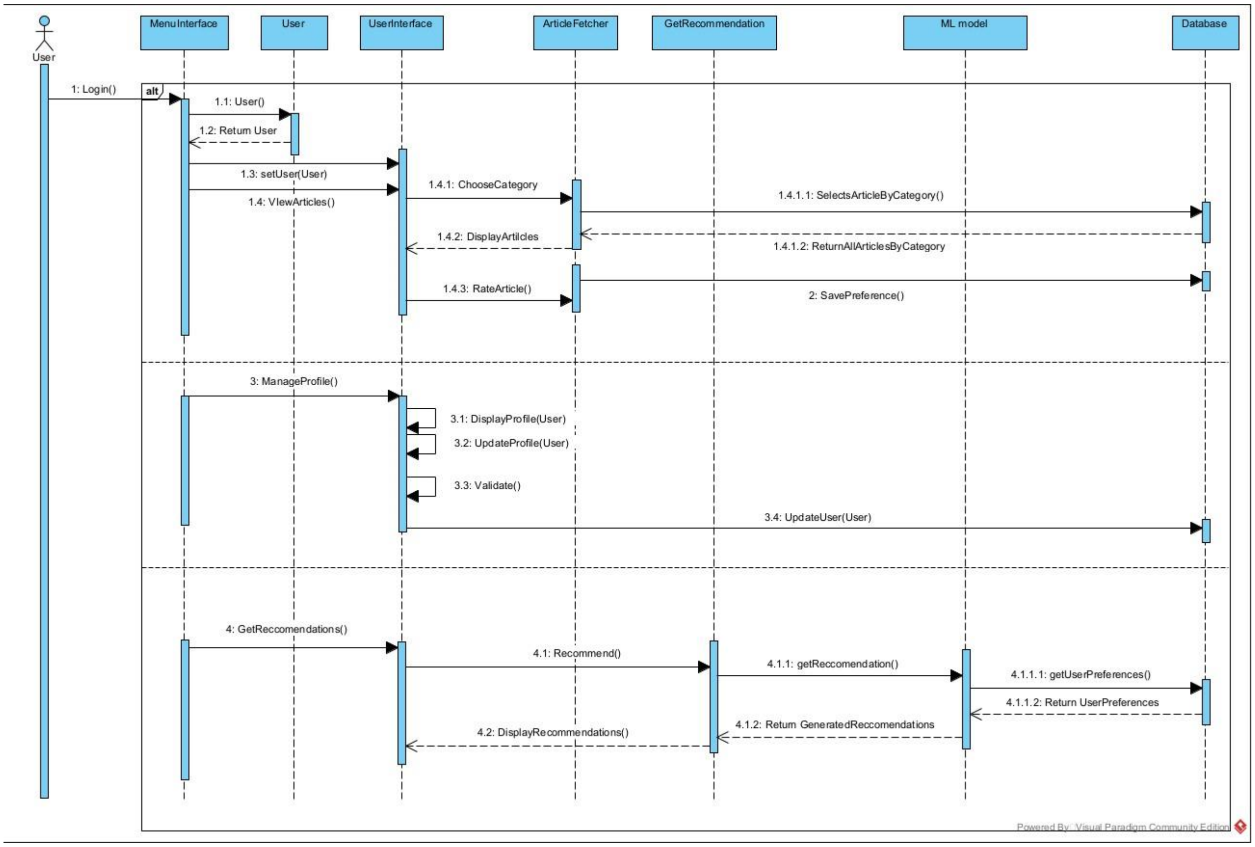


Figure 11 User interactions Sequence

5.4 Admin interactions

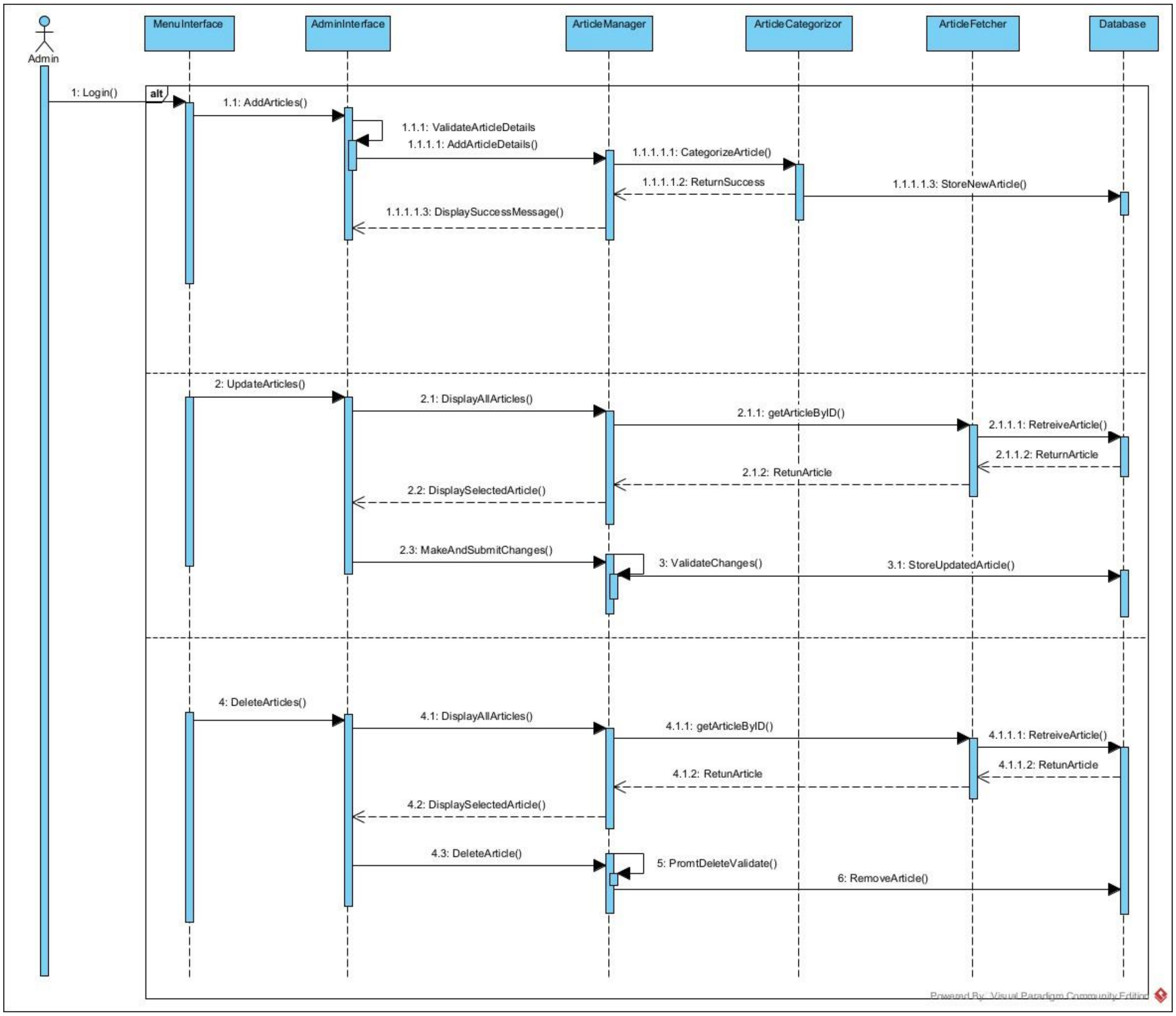
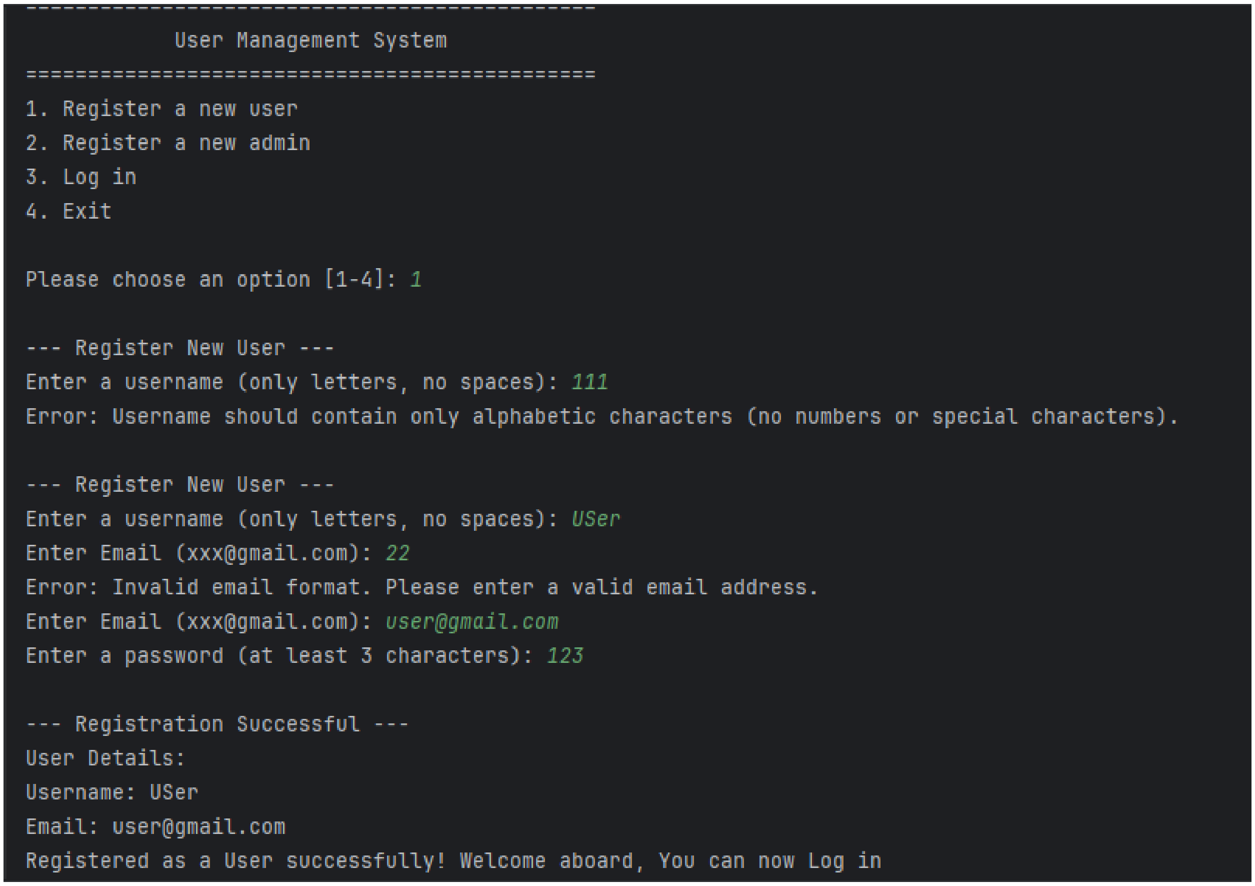

Admin
MenuInterface
AdminInterface
ArticleManager
ArticleCategorizor
ArticleFetcher
Database
1: Login()
alt
1.1: AddArticles()
1.1.1: ValidateArticleDetails
1.1.1.1: AddArticleDetails()
1.1.1.1.1: CategorizeArticle()
1.1.1.1.2: ReturnSuccess
1.1.1.1.3: StoreNewArticle()
1.1.1.1.3: DisplaySuccessMessage()
2: UpdateArticles()
2.1: DisplayAllArticles()
2.1.1: getArticleByID()
2.1.1.1: RetreiveArticle()
2.1.1.2: ReturnArticle
2.1.2: RetunArticle
2.2: DisplaySelectedArticle()
2.3: MakeAndSubmitChanges()
3: ValidateChanges()
3.1: StoreUpdatedArticle()
4: DeleteArticles()
4.1: DisplayAllArticles()
4.1.1: getArticleByID()
4.1.1.1: RetreiveArticle()
4.1.1.2: RetunArticle
4.1.2: RetunArticle
4.2: DisplaySelectedArticle()
4.3: DeleteArticle()
5: PromtDeleteValidate()
6: RemoveArticle()
Powered By: Visual Paradigm Community Edition

Figure 12 Admin Interactions Sequecne

6. Testing and Evaluation

6.1 Test Case 01: Registering User

User registering into the system where Username should be unique and then is validated, Email is taken which is validated and then password is taken, and after all fields have valid data, the user is then saved in the database

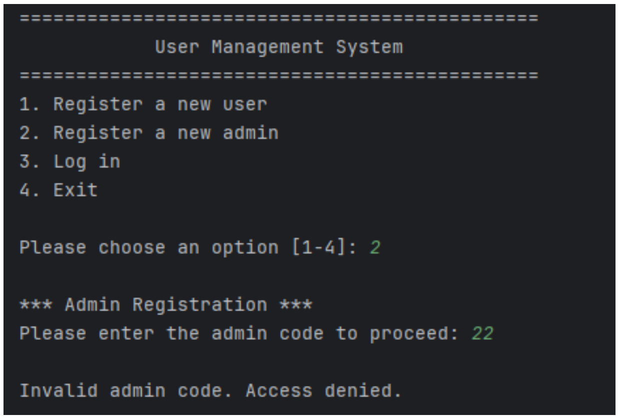


```
----------------------------------------------
            User Management System
==============================================
1. Register a new user
2. Register a new admin
3. Log in
4. Exit

Please choose an option [1-4]: 1

--- Register New User ---
Enter a username (only letters, no spaces): 111
Error: Username should contain only alphabetic characters (no numbers or special characters).

--- Register New User ---
Enter a username (only letters, no spaces): USer
Enter Email (xxx@gmail.com): 22
Error: Invalid email format. Please enter a valid email address.
Enter Email (xxx@gmail.com): user@gmail.com
Enter a password (at least 3 characters): 123

--- Registration Successful ---
User Details:
Username: USer
Email: user@gmail.com
Registered as a User successfully! Welcome aboard, You can now Log in
```

Figure 13 Test case 1.0

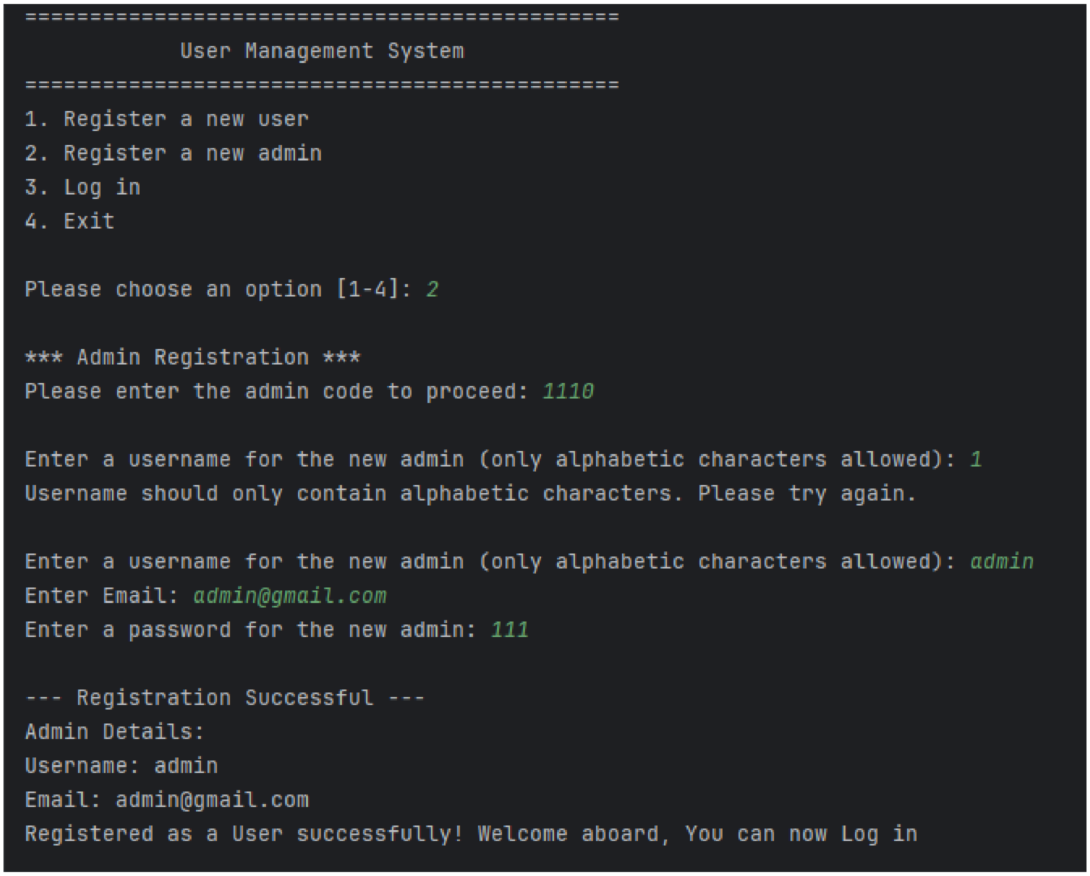
7.1 Test Case 02: Register Admin

Admin registration where an admin code (1110) is required to be registered in, where the username, Email and password is taken in which are validated beforehand and then updated on the database



```
==============================================
            User Management System
==============================================
1. Register a new user
2. Register a new admin
3. Log in
4. Exit

Please choose an option [1-4]: 2

*** Admin Registration ***
Please enter the admin code to proceed: 22

Invalid admin code. Access denied.
```

Figure 14 Test case 2.0

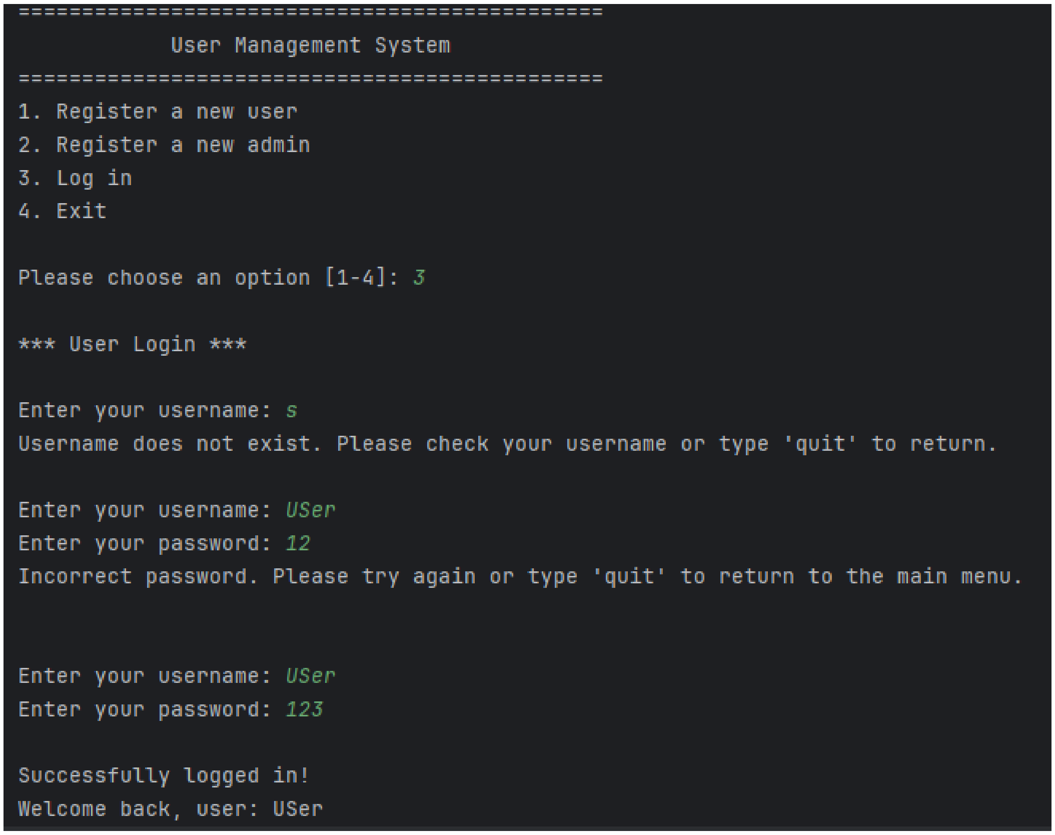


```
==============================================
            User Management System
==============================================
1. Register a new user
2. Register a new admin
3. Log in
4. Exit

Please choose an option [1-4]: 2

*** Admin Registration ***
Please enter the admin code to proceed: 1110

Enter a username for the new admin (only alphabetic characters allowed): 1
Username should only contain alphabetic characters. Please try again.

Enter a username for the new admin (only alphabetic characters allowed): admin
Enter Email: admin@gmail.com
Enter a password for the new admin: 111

--- Registration Successful ---
Admin Details:
Username: admin
Email: admin@gmail.com
Registered as a User successfully! Welcome aboard, You can now Log in
```

Figure 15 Test Case 2.1

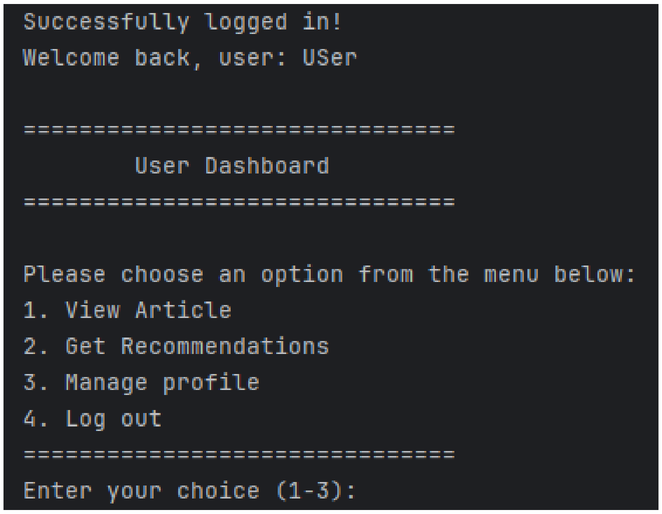
8.1 Test Case 03: Log in User

User login where it checks for the Username and the password where they are validated with the already existing user details and if successful it displays the userInterface where it has user functions, if failed it asks to reenter username and password or quit to return to menu



```
==============================================
            User Management System
==============================================
1. Register a new user
2. Register a new admin
3. Log in
4. Exit

Please choose an option [1-4]: 3

*** User Login ***

Enter your username: s
Username does not exist. Please check your username or type 'quit' to return.

Enter your username: USer
Enter your password: 12
Incorrect password. Please try again or type 'quit' to return to the main menu.


Enter your username: USer
Enter your password: 123

Successfully logged in!
Welcome back, user: USer
```

Figure 16 Test Case 3.0

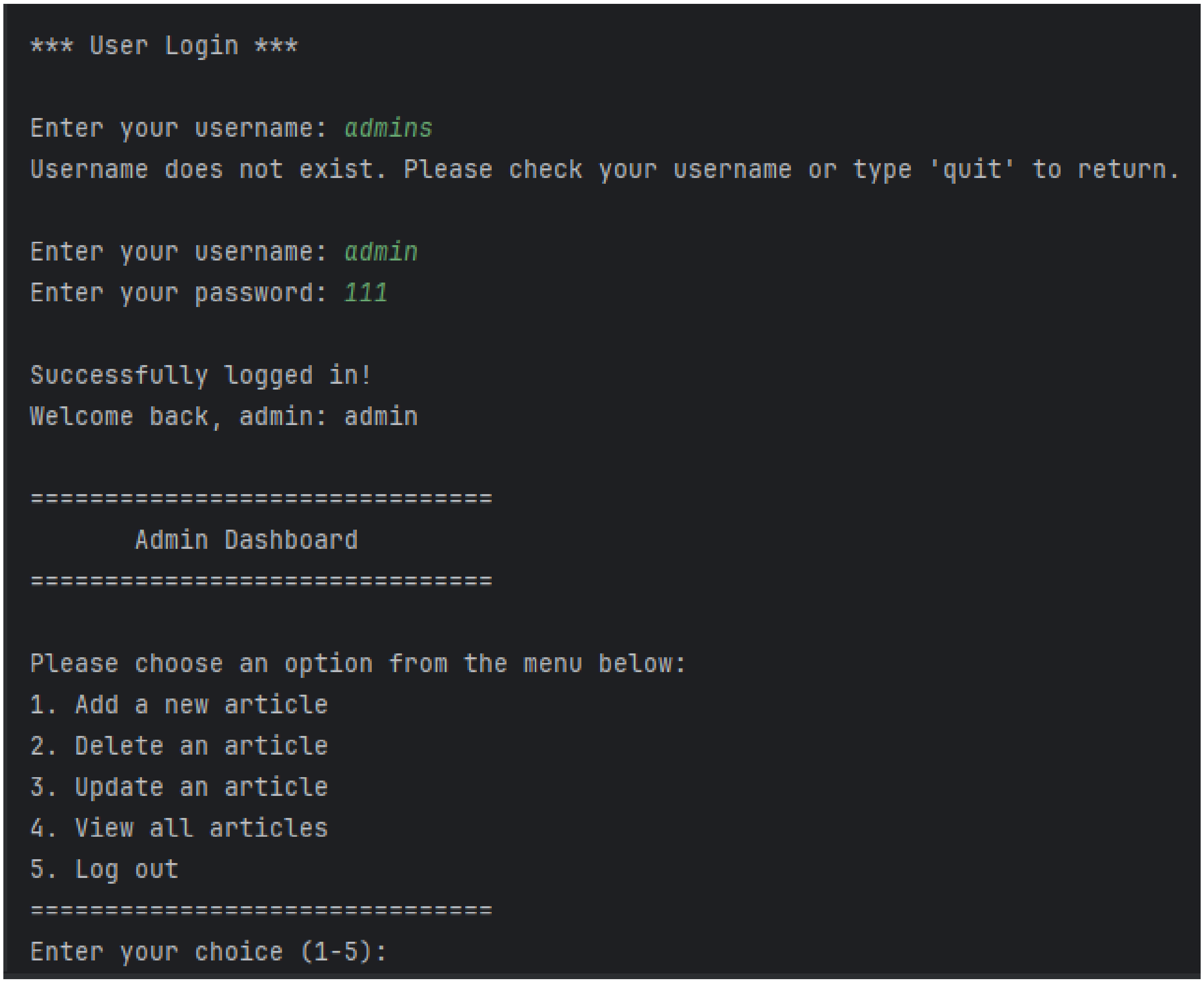


```
Successfully logged in!
Welcome back, user: USer

===============================
        User Dashboard
===============================

Please choose an option from the menu below:
1. View Article
2. Get Recommendations
3. Manage profile
4. Log out
===============================
Enter your choice (1-3):
```

Figure 17 Test Case 3.1

9.1 Test Case 04: Log in Admin

Admin login where both username and password are validated and if credentials matched it displays the adminInterface where admin has different options

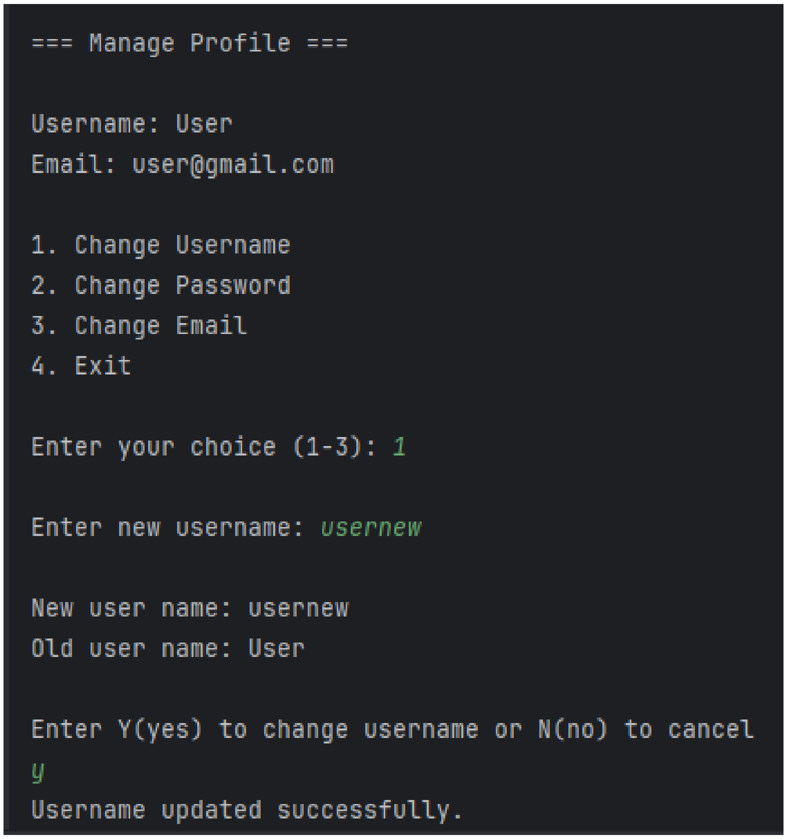


```
*** User Login ***

Enter your username: admins
Username does not exist. Please check your username or type 'quit' to return.

Enter your username: admin
Enter your password: 111

Successfully logged in!
Welcome back, admin: admin

===============================
       Admin Dashboard
===============================

Please choose an option from the menu below:
1. Add a new article
2. Delete an article
3. Update an article
4. View all articles
5. Log out
===============================
Enter your choice (1-5):
```

Figure 18 Test Case 4.0

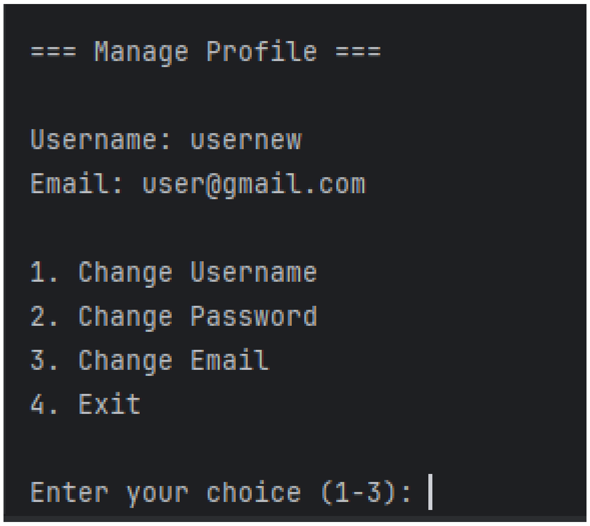
10.1 Test Case 05: Manage Profile

User updating their profile I.E changing their username, password and email address which are confirmed by the user and then changes are saved



```
=== Manage Profile ===

Username: User
Email: user@gmail.com

1. Change Username
2. Change Password
3. Change Email
4. Exit

Enter your choice (1-3): 1

Enter new username: usernew

New user name: usernew
Old user name: User

Enter Y(yes) to change username or N(no) to cancel
y
Username updated successfully.
```

Figure 19 Test Case 5.0

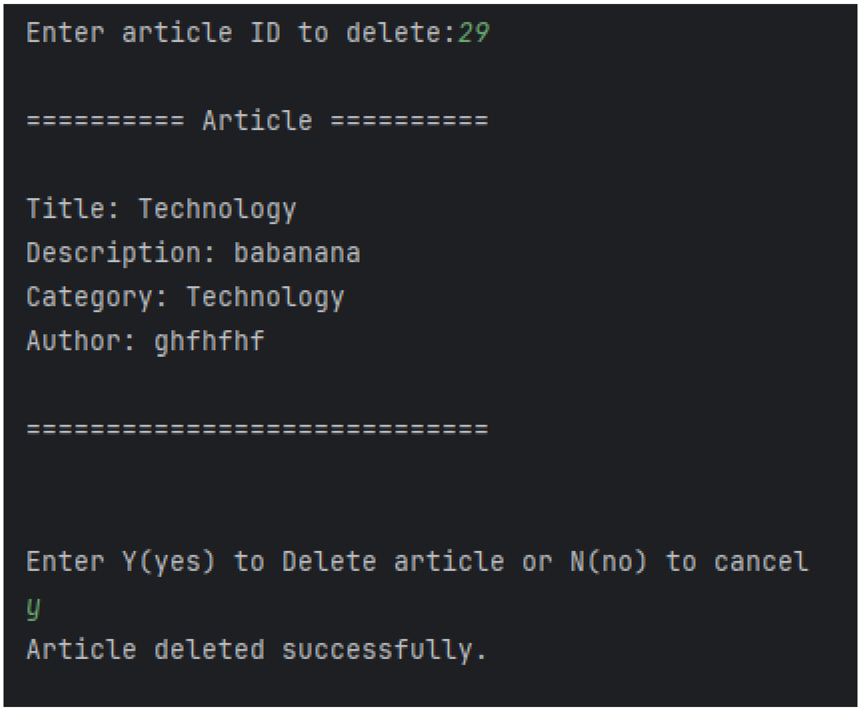


```
=== Manage Profile ===

Username: usernew
Email: user@gmail.com

1. Change Username
2. Change Password
3. Change Email
4. Exit

Enter your choice (1-3):
```

Figure 20 Test Case 5.1

11.1 Test Case 06: Deleting Article confirmation

All articles displayed first and then article ID is chosen to be deleted where confirmation is prompted

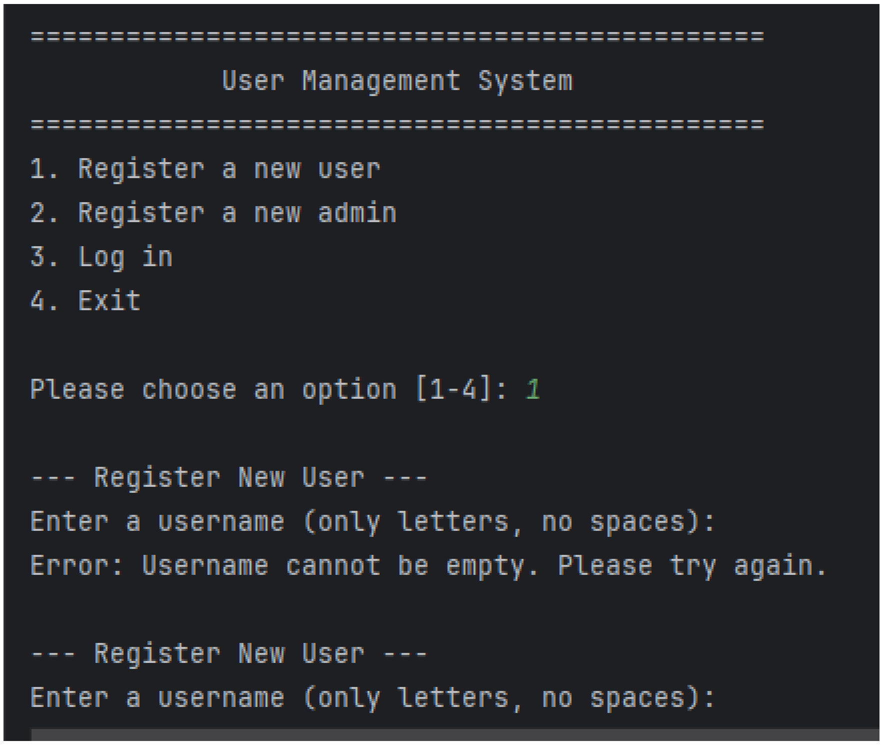


```
Enter article ID to delete:29

========== Article ==========

Title: Technology
Description: babanana
Category: Technology
Author: ghfhfhf

=============================


Enter Y(yes) to Delete article or N(no) to cancel
y
Article deleted successfully.
```

Figure 21 Test Case 6.0

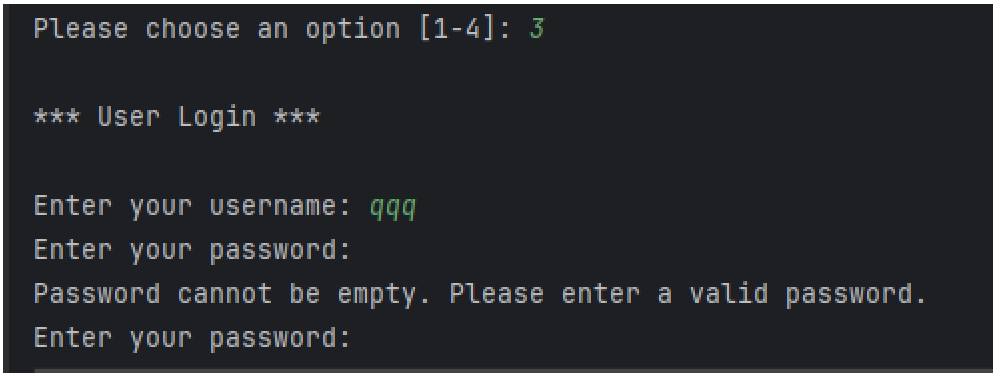
12.1 Test Case 07: Empty fields validation

Fields validated such that it cannot be left empty



```
==============================================
            User Management System
==============================================
1. Register a new user
2. Register a new admin
3. Log in
4. Exit

Please choose an option [1-4]: 1

--- Register New User ---
Enter a username (only letters, no spaces):
Error: Username cannot be empty. Please try again.

--- Register New User ---
Enter a username (only letters, no spaces):
```

Figure 22 Test Case 7.0

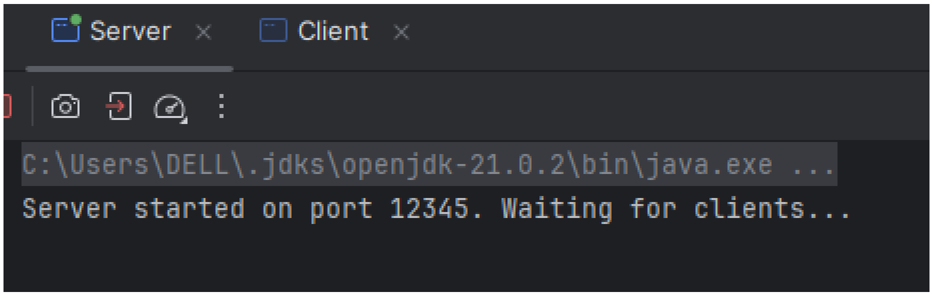


```
Please choose an option [1-4]: 3

*** User Login ***

Enter your username: qqq
Enter your password:
Password cannot be empty. Please enter a valid password.
Enter your password:
```

Figure 23 Test Case 7.1

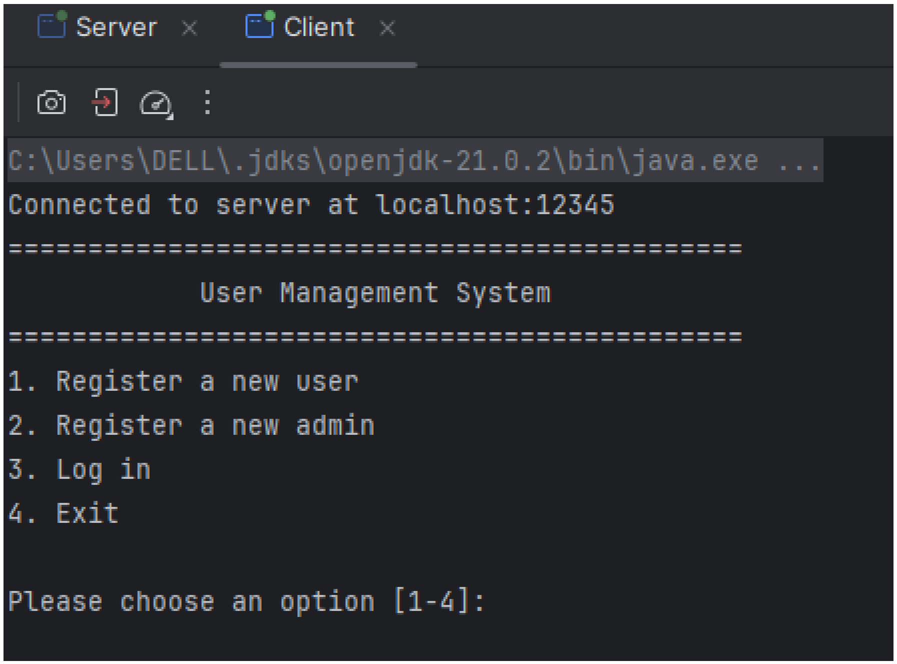
13.1 Test Case 08: Concurrency Testing

Server executed first and then clients are connected to the server



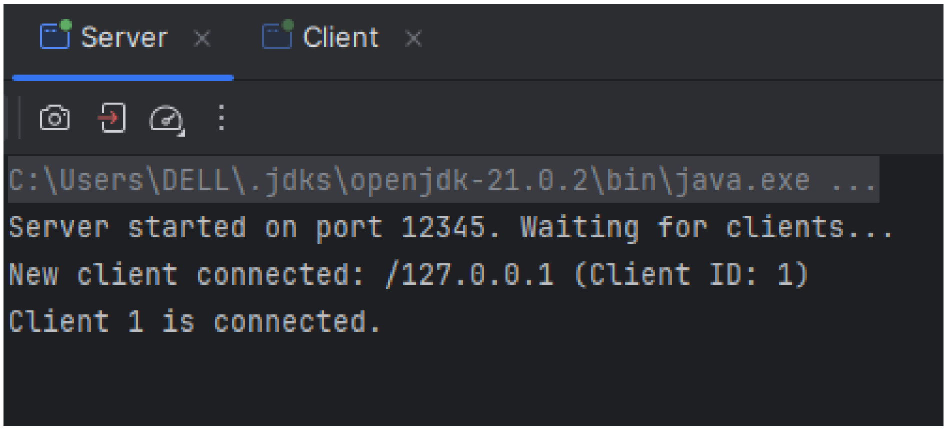
```
C:\Users\DELL\.jdks\openjdk-21.0.2\bin\java.exe ...
Server started on port 12345. Waiting for clients...
```

Figure 24 Server Connected 8.0



```
C:\Users\DELL\.jdks\openjdk-21.0.2\bin\java.exe ...
Connected to server at localhost:12345
==============================================
            User Management System
==============================================
1. Register a new user
2. Register a new admin
3. Log in
4. Exit

Please choose an option [1-4]:
```

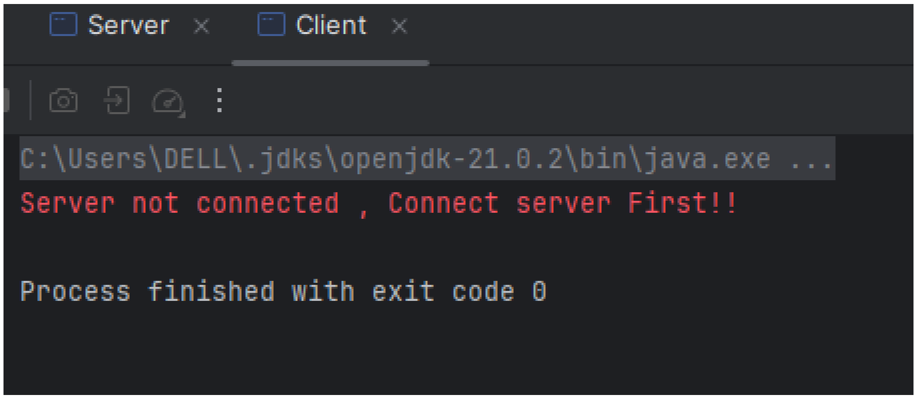
Figure 26 Client Connected 8.1



```
C:\Users\DELL\.jdks\openjdk-21.0.2\bin\java.exe ...
Server started on port 12345. Waiting for clients...
New client connected: /127.0.0.1 (Client ID: 1)
Client 1 is connected.
```

Figure 25 Client connected to server 8.2

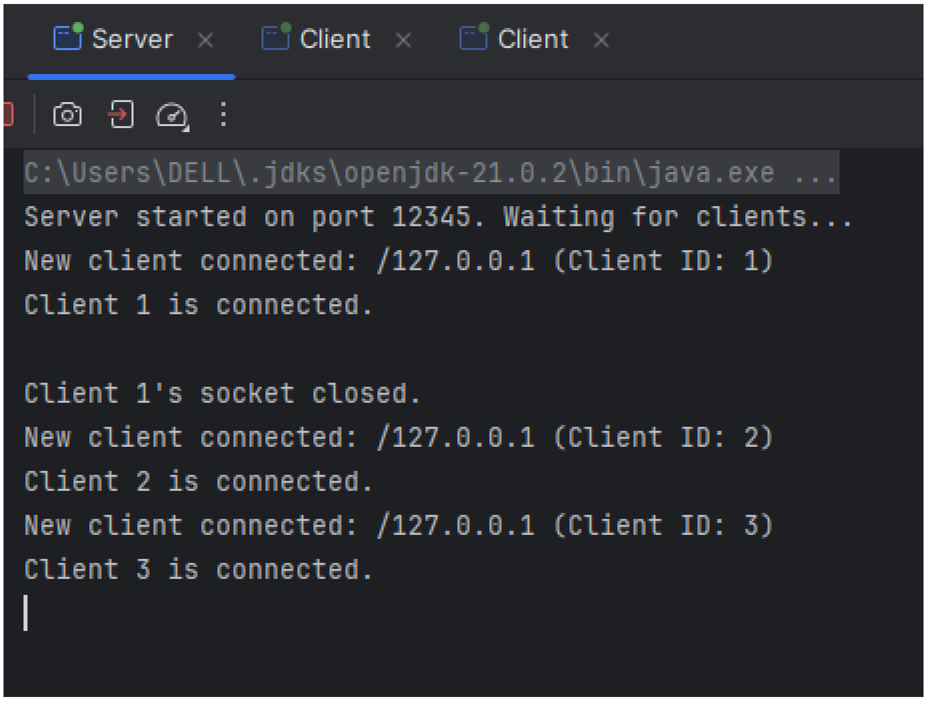
When client is run before running the server



```
C:\Users\DELL\.jdks\openjdk-21.0.2\bin\java.exe ...
Server not connected , Connect server First!!

Process finished with exit code 0
```

Figure 27 Client not connected 8.3

Two clients connected to the server



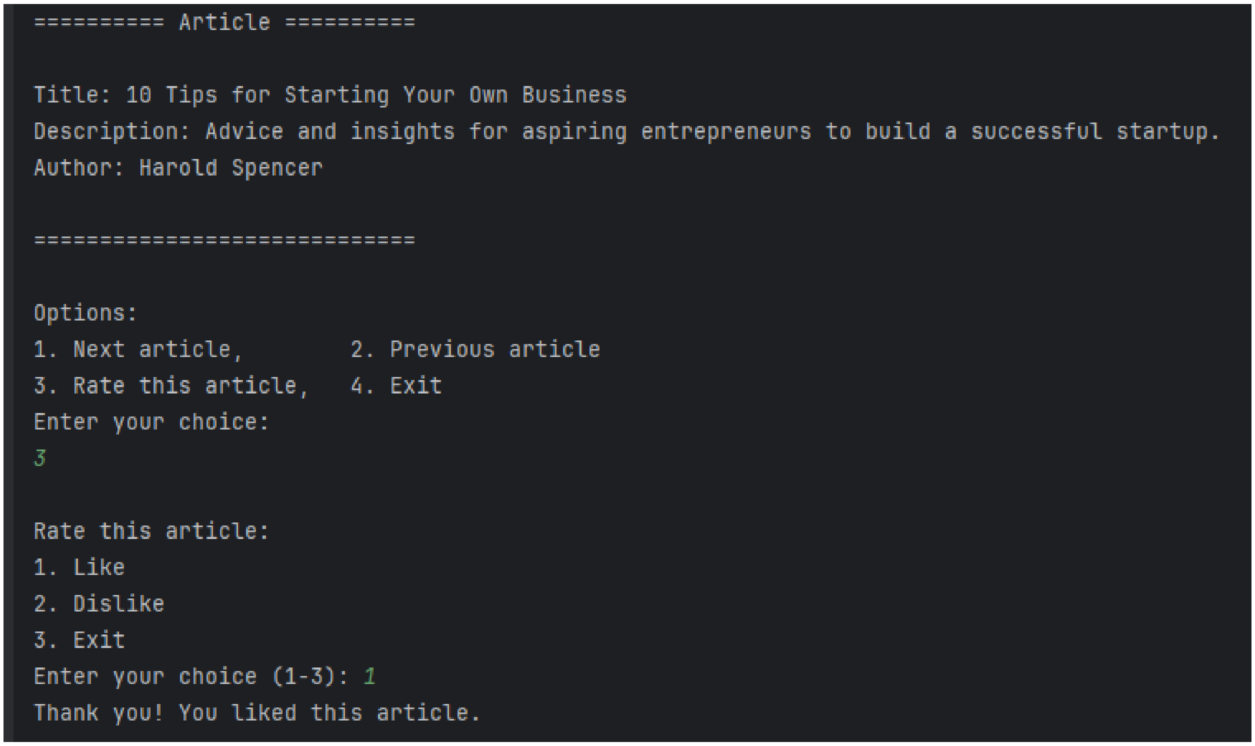
```
C:\Users\DELL\.jdks\openjdk-21.0.2\bin\java.exe ...
Server started on port 12345. Waiting for clients...
New client connected: /127.0.0.1 (Client ID: 1)
Client 1 is connected.

Client 1's socket closed.
New client connected: /127.0.0.1 (Client ID: 2)
Client 2 is connected.
New client connected: /127.0.0.1 (Client ID: 3)
Client 3 is connected.
```

Figure 28 Two clients Connected 8.4

14.1 Test Case 09: Recommendation Engine

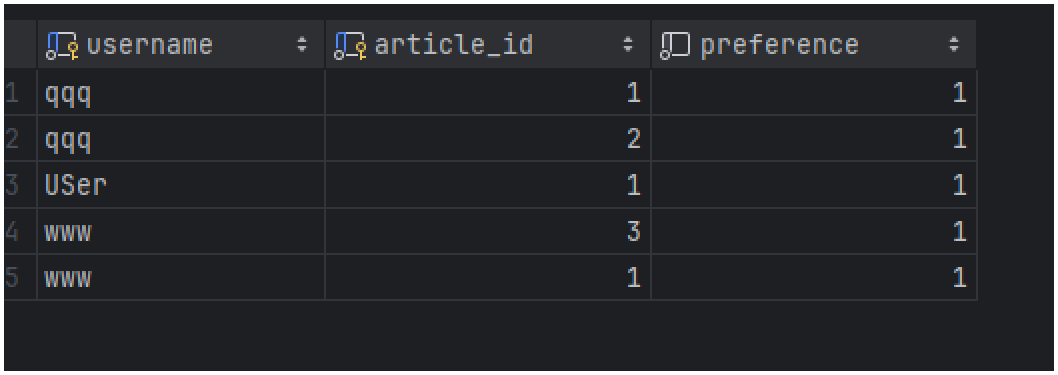
Recommendation Engine where user preferences are considered, and the recommendation is provided along with the similarity score using the LLM

1. User likes an artilce from the business category



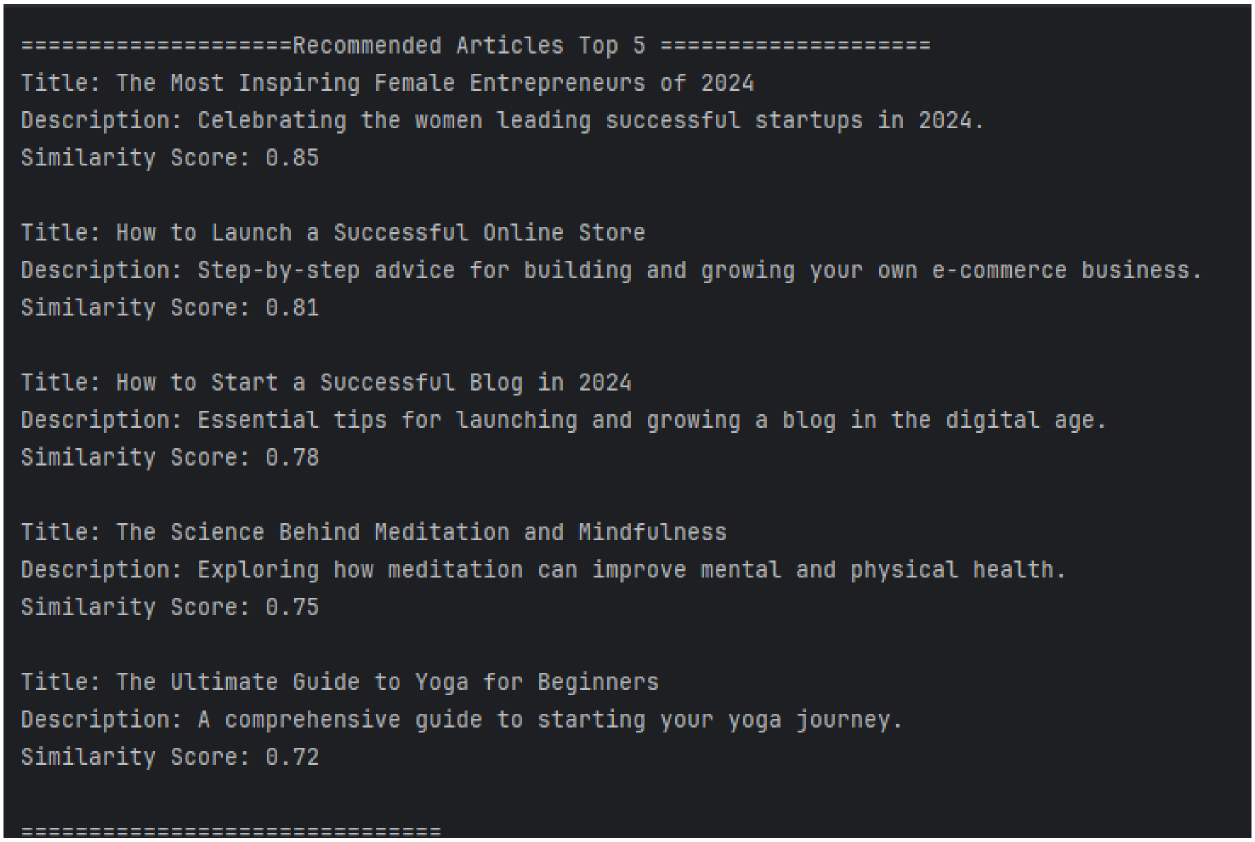
```
========== Article ==========

Title: 10 Tips for Starting Your Own Business
Description: Advice and insights for aspiring entrepreneurs to build a successful startup.
Author: Harold Spencer

=============================

Options:
1. Next article,        2. Previous article
3. Rate this article,   4. Exit
Enter your choice:
3

Rate this article:
1. Like
2. Dislike
3. Exit
Enter your choice (1-3): 1
Thank you! You liked this article.
```

2. User preferences stored in the database



username	article_id	preference
qqq	1	1
qqq	2	1
USer	1	1
www	3	1
www	1	1

3. Top 5 recommendations are provided based on user preferences



```
====================Recommended Articles Top 5 ====================
Title: The Most Inspiring Female Entrepreneurs of 2024
Description: Celebrating the women leading successful startups in 2024.
Similarity Score: 0.85

Title: How to Launch a Successful Online Store
Description: Step-by-step advice for building and growing your own e-commerce business.
Similarity Score: 0.81

Title: How to Start a Successful Blog in 2024
Description: Essential tips for launching and growing a blog in the digital age.
Similarity Score: 0.78

Title: The Science Behind Meditation and Mindfulness
Description: Exploring how meditation can improve mental and physical health.
Similarity Score: 0.75

Title: The Ultimate Guide to Yoga for Beginners
Description: A comprehensive guide to starting your yoga journey.
Similarity Score: 0.72

===============================
```

7. References

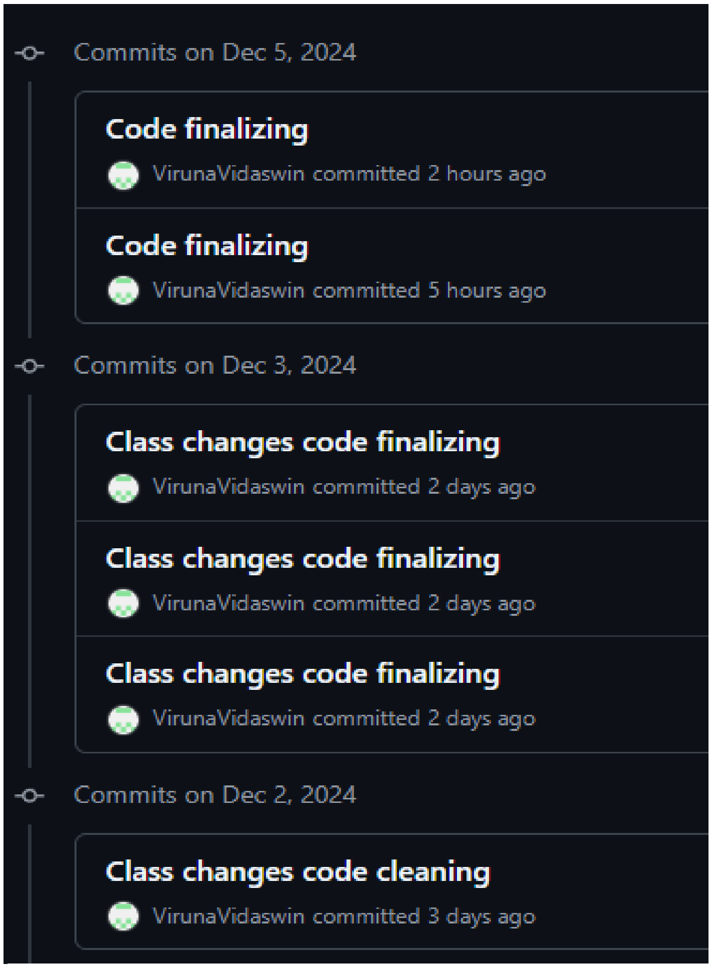
1. taketwo (2024). *GitHub - taketwo/llm-ollama: LLM plugin providing access to local Ollama models using HTTP API*. [online] GitHub. Available at: https://github.com/taketwo/llm-ollama [Accessed 4 Dec. 2024]

2. ollama.com. (n.d.). *Ollama*. [online] Available at: https://ollama.com/

3. careerfoundry.com. (2023). *NLP Algorithms: A Beginner's Guide for 2024*. [online] Available at: https://careerfoundry.com/en/blog/data-analytics/what-are-nlp-algorithms/

4. geeksforgeeks (2019). *Object Oriented Programming (OOPs) Concept in Java - GeeksforGeeks*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/

8. Appendix- Version Control



Commits on Dec 5, 2024

Code finalizing

VirunaVidaswin committed 2 hours ago

Code finalizing

VirunaVidaswin committed 5 hours ago

Commits on Dec 3, 2024

Class changes code finalizing

VirunaVidaswin committed 2 days ago

Class changes code finalizing

VirunaVidaswin committed 2 days ago

Class changes code finalizing

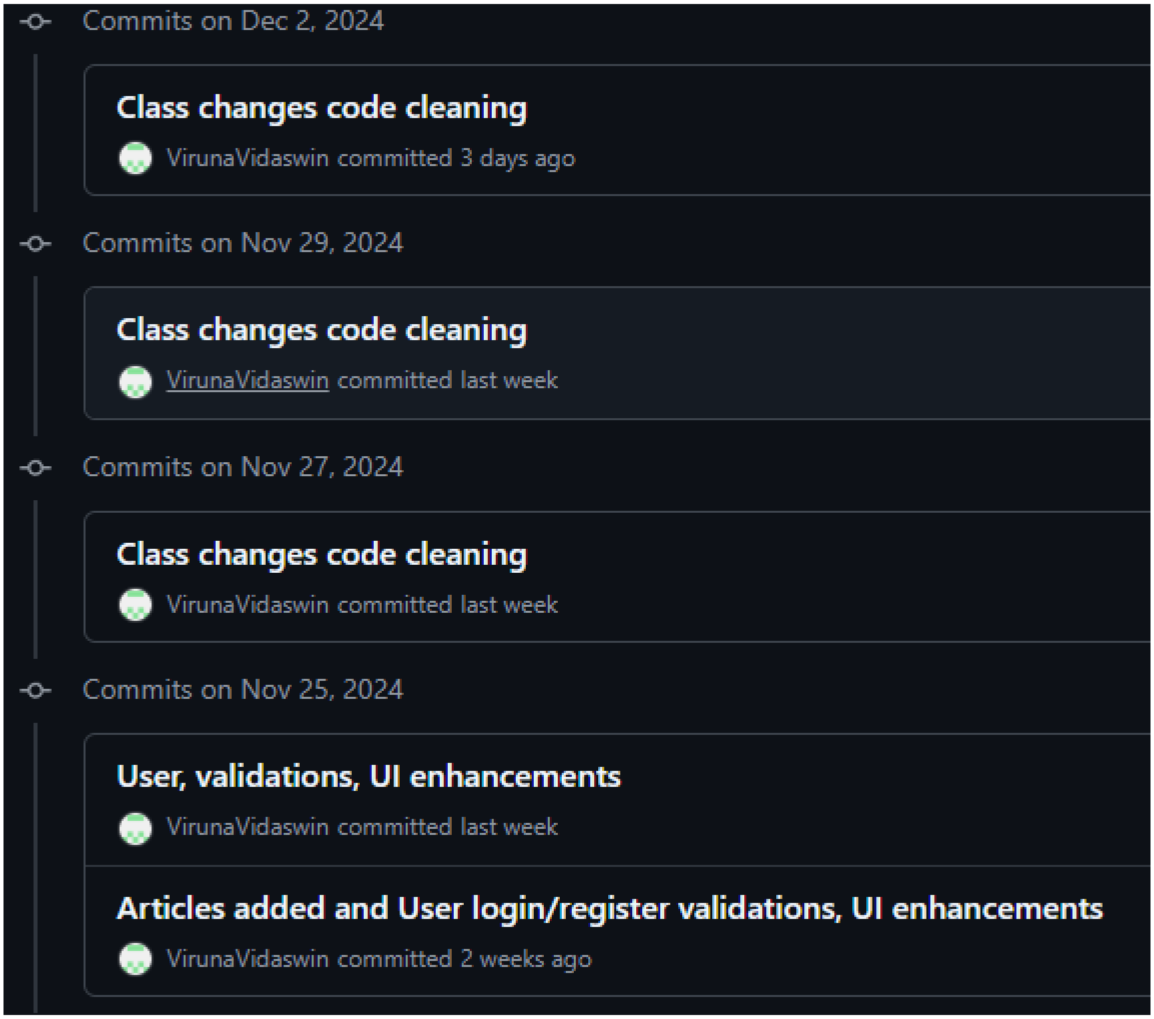
VirunaVidaswin committed 2 days ago

Commits on Dec 2, 2024

Class changes code cleaning

VirunaVidaswin committed 3 days ago

Figure 29 Version control 1.0



Commits on Dec 2, 2024

Class changes code cleaning
VirunaVidaswin committed 3 days ago

Commits on Nov 29, 2024

Class changes code cleaning
VirunaVidaswin committed last week

Commits on Nov 27, 2024

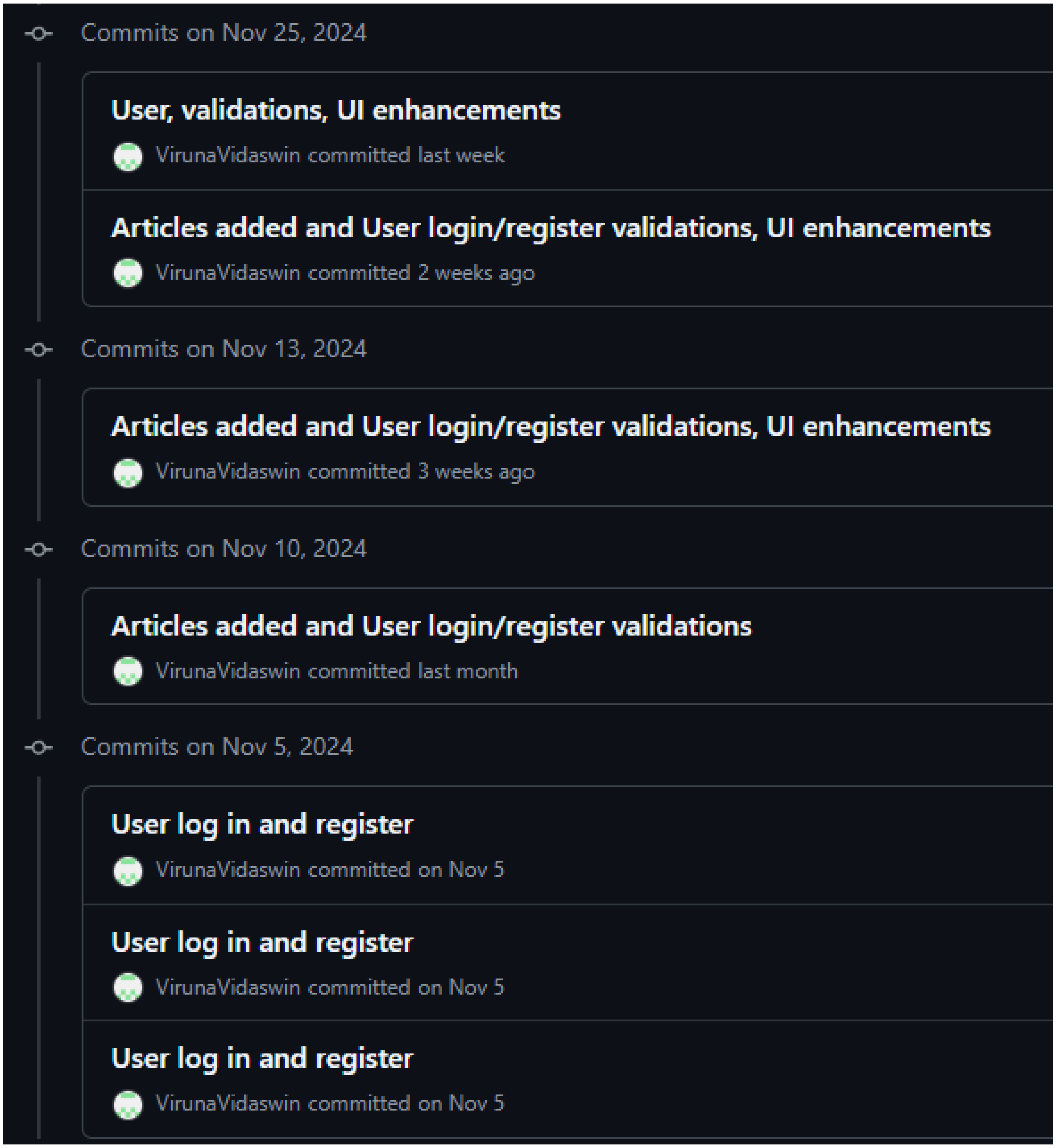
Class changes code cleaning
VirunaVidaswin committed last week

Commits on Nov 25, 2024

User, validations, UI enhancements
VirunaVidaswin committed last week

Articles added and User login/register validations, UI enhancements
VirunaVidaswin committed 2 weeks ago

Figure 30 Version control 1.1



Commits on Nov 25, 2024

User, validations, UI enhancements

Viruna Vidaswin committed last week

Articles added and User login/register validations, UI enhancements

Viruna Vidaswin committed 2 weeks ago

Commits on Nov 13, 2024

Articles added and User login/register validations, UI enhancements

Viruna Vidaswin committed 3 weeks ago

Commits on Nov 10, 2024

Articles added and User login/register validations

Viruna Vidaswin committed last month

Commits on Nov 5, 2024

User log in and register

Viruna Vidaswin committed on Nov 5

User log in and register

Viruna Vidaswin committed on Nov 5

User log in and register

Viruna Vidaswin committed on Nov 5

Figure 31 Version controlling 1.2