COMSATS University Islamabad (CUI)

Department of Computer Science

Assignment-03

CLO-3

Software Design Description
(SDS DOCUMENT)

for

ROOMIEFIND

Version 1.0

Submitted By:

Zeeshan Qureshi FA24-BSE-078

Muhim Akhtar FA24-BSE-080

BSE-2-B

Supervisor

Dr.Tehseen Riaz Abbasi

Submission Date: (22-05-2024)

Original Version 1.0

Bachelor of Science in Software engineering (2024-2028)

Table Contents

1. Introduction 1
1.1 Scope 1
1.2 Modules 1
2. Design methodology and Software Process Model 3
3. System overview 3
4. Design Models 4
4.1 Architectural Design 4
4.2 Use Case(s) Diagram: 5
Figure 3.1.1: UC Diagram for user authentication Module (user) 5
Figure 3.5.1: UC Diagram for Secure Chat & Agreement Templates Module (user) 13
Figure 3.8.1: UC Diagram for Admin Dashboard & Analytic Templates Module (User) 19
5. Design Models 20
5.1 Activity Diagrams 20
5.2 Sequence Diagrams 36
5.3 Class Diagram 52
6. Data design 53
6.1 Data dictionary 53
7. Algorithm & Implementation 54
8. Human Interface Design : 59
User interface: 59
7.1. Screen objects and actions 63
9. Conclusion 64
10. References: 65
11. Plaragism Report 66

1. Introduction

Briefly explain scope of the project covered till now including modules.

1.1 Scope

Roomie Find is an easy-to-use application designed for the users to find compatible roommates to make their living experience better and seamless. Our application filters out your potential roommates the basis of four main parameters:
1.Lifestyle
2.Budget
3.Location
4.Hobbies.
Our goal is to make the users experience living according to their preferred standards and diminish personality clashes among the roommates and give our esteemed users harmony. To sum up, the platform aims to rationalize the roommate matching process, making it tranquil for individuals to find suitable shared living arrangements. Admins will oversee the system, ensuring quality and comfortable experience.

1.2 Modules

Roomie Find will have the following modules:

Module 1: User Authentication
This module is used to authenticate the user.

UC-1*:* User Registration
UC-2*:* User Login
UC-3*:* Password Reset
UC-4: Two-Factor Authentication
UC-5*:* Session Management
UC-6: Email Verification
UC-7: CAPTCHA Integration
UC-8: User Role Management

Module 2: Roommate Compatibility Assessment
This module helps in matchmaking of roommates based on the entered data.
UC-1*:* Personality Questionnaire
UC-2*:* Lifestyle Preferences Input
*UC-**3**:* Daily Schedule Input
UC-4: Compatibility Score Calculation
UC-5*:* Profile Matching Based on Scores
UC-6: Match Feedback Collection
UC-7: *Incompatibility Alerts*
UC-8: *Compatibility History*

Module 3: Property and Room Listings
In this module, the user can upload pictures of their rooms and apartments.

UC-1: Post Property Listing
UC-2*:* Edit Property Listing
UC-3*:* Search Listings
UC-4: Filter Listings

***UC-5**:* View Property Details
***UC-6**:* Save Favorite Listings
***UC-7:** Contact Property Owner*
***UC-8:** Report Listing*

Module 4: AI-based Matching Algorithm

In this module an AI-based Matching will happen.

***UC-1**:* Process Compatibility Inputs
***UC-2**:* Generate Match Scores
UC-3: Suggest Top Matches
UC-4: Recalculate on Profile Update
UC-5: Filter Incompatible Profiles
***UC-6**:* AI Feedback Loop
***UC-7:** Update Match Algorithm*
***UC-8:** Log Match Outcomes*

Module 5: Review & Rating System
This module is used Reviews and rating system so that feedback can be used to improve the application.

***UC-1**:* Report submission
UC-2: Rating in terms of stars
***UC-3**:* Editing a review
UC-4: Reporting false language or misleading reviews
UC-5: View uploaded reviews
UC-6: Filtering reviews
UC-7: Delete the Review
***UC-8:** Send notification of reviews*

Module 6: Notification & Alert System
In this module user and admin receives the notifications about their Agreements.

UC-1: Send Notification
***UC-2**:* Set Notification Preferences
UC-3: View Notification History
***UC-4**:* Dismiss Notification
UC-5: Customize Notification Sound
***UC-6:** Mute Notifications*
***UC-7:** Configure Notification Types*
***UC-8:** Receive System Alerts*

Module 7: Admin Dashboard & Analytics
In this module admin can see all the activities taking place. This will help with the smooth running of the application.

UC-1: View User Activity Analytics
UC-2: View Financial Analytics
UC-3: Filter Analytics Data
UC-4: Download Analytics Report
UC-5: View Admin Activities
***UC-6:** Edit Admin Settings*
***UC-7:** Monitor System Health*

UC-8: **Generate System Performance Report**

Module 8: Secure Chat & Agreement Templates
This module is used for the secure chat between the differed
UC-1: Initiate Secure Chat
UC-2: Send Encrypted Message
UC-3: Receive Secure Message
UC-4: Terminate Chat Session
UC-5: Upload Agreement Template
UC-6: Share Agreement in Chat
UC-7: *Accept Agreement*
UC-8: *View Agreement History*

2. Design methodology and Software Process Model

We will use **incremental process model** for development of our application because we don't have a lot of staff involved development and we will build the application by dividing it and implementing it in increments. This Software process methodology will easily accommodate changes in the system easier without introducing bugs in the system.
The design methodology used for the project will be **Object Oriented Programming.** The application will be divided in classes and each class having their attributes and functionality. This strategy fits very nicely with the development of a system such as RoomieFind, in which different entities can be modelled as objects with unique properties and behaviors, including patients, healthcare providers, and imaging data. It encourages readability, maintainability, and code organizational —all of which are essential for a complicated system like an application for Matching Roomie. The design process makes sure that every class is a logical, self-contained entity by using OOP principles, which adds to the RoomieFind application's overall modularity and versatility.

3. System overview

RoomieFind is a web-based platform designed to help young and students professionals find suitable rental accommodation and compatible roommates. It matches individuals based on preferences, location budget and lifestyle habits.

Objectives:
- It connects users that are seeking roommates with similar preferences.
- Provides the list for rental properties and shared accommodation.
- Facilitate seamless communication between roommates and landlords.

Core Features:
1. **User Profiles**
   - Personal detail about user
   - His preferences
   - His location and Budget
2. **Roommate Matching**
   - Filter based matching system with the AI-Algorithm
   - Compatibility Scoring
3. **Property listings**
   - Listings of the apartments and room that are available
   - Details about property that include location, rent photos and amenities
4. **Search and Filters**

- You can search by budget availability location and preferences
- You can see the best matches by applying filters

5. **Communication and Chat**
    - Private and secure chat
    - In the app you can message to connect
6. **Reviews and Ratings**
    - Users can leave reviews for the properties and roommates
    - Build transparency and Trust

Target Audience

- University and school students
- Professional that work
- People that are relocating to a new city

Technology Stack

- Frontend: html/CSS
- Backend: java
- Database: MongoDB

4. Design Models

4.1 Architectural Design

3-Tier Architecture design will be used in RoomieFind.

The three-tier architecture consists of 3 independent layers involving Client layer, Business layer and Data layer. The business layer is the application server or the backend of the program whose function is to convert the queries presented by the client layer to the machine code. This reduces the load on database server.

A three-phase Architecture is a well-designed software program that organizes applications into three logical and physical computer components: a launch phase, or an easy-to-use interface; application phase, in which data is processed; and the data section, where the data associated with the application is stored and managed. The great advantage of three-phase design is that because each phase operates on its own infrastructure, each phase can be developed simultaneously by a separate development team and can be updated or modified as needed without compromising other components. The main advantages of a 3-tier architecture design are rapid development, advanced scaling, reliability, and safety.

Reasons for using 3-tier architecture are:

- Better scalability
- Better Security
- Better Reliability

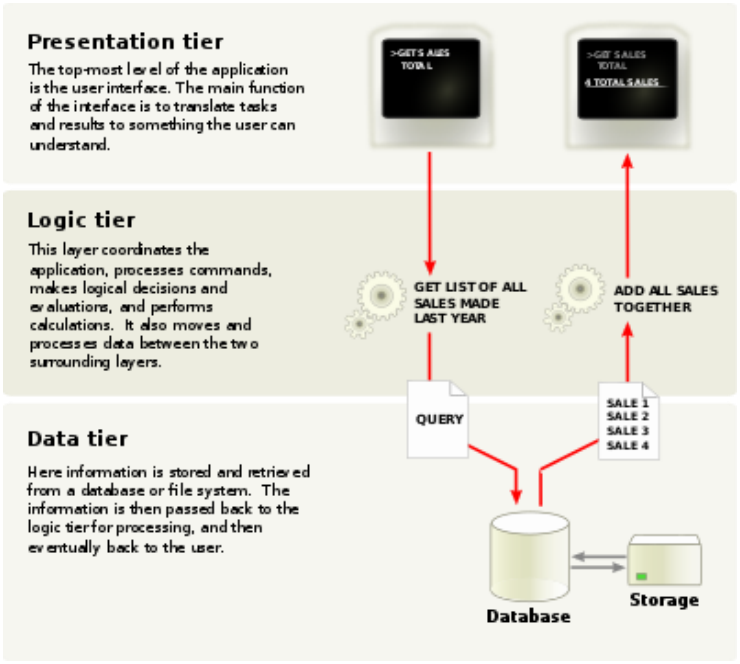


Figure 3.1: 3-tier Architecture Diagram

4.2 Use Case(s) Diagram:

The following are the usecase diagrams of all the modules:

Module 1: User Authentication

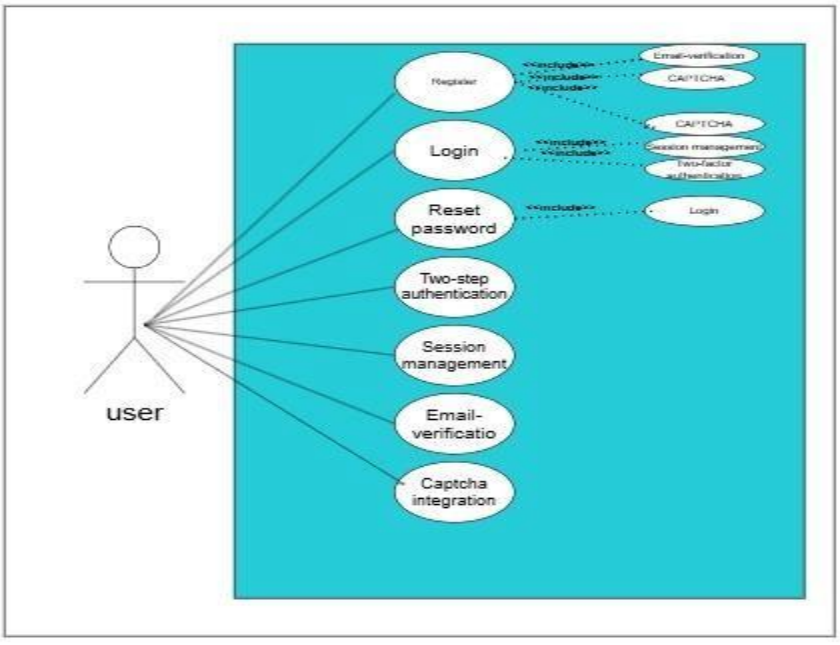


Figure 3.1.1: UC Diagram for user authentication Module (user)

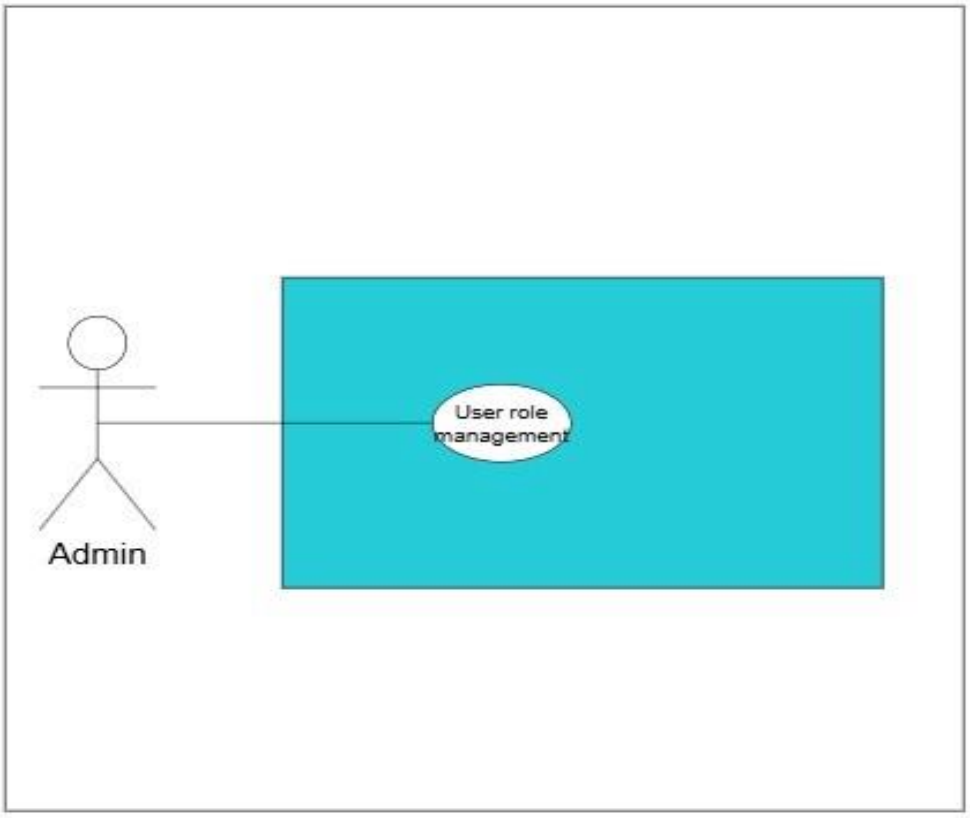


Figure 3.1.2: UC Diagram for user authentication Module (Admin)

Module 2: Roommate Compatibility Assessment

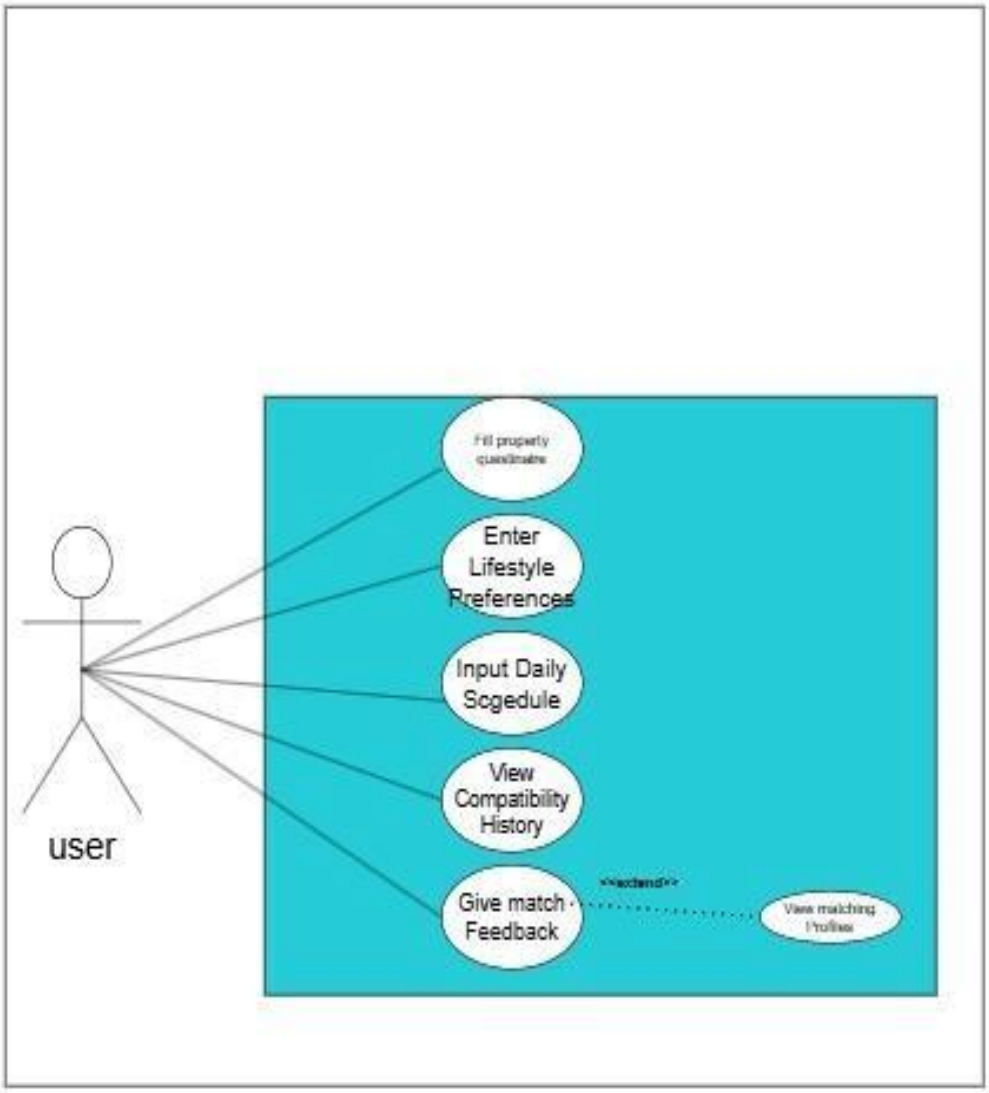


Figure 3.2.1: UC Diagram for Roommate Compatibility Module(user)

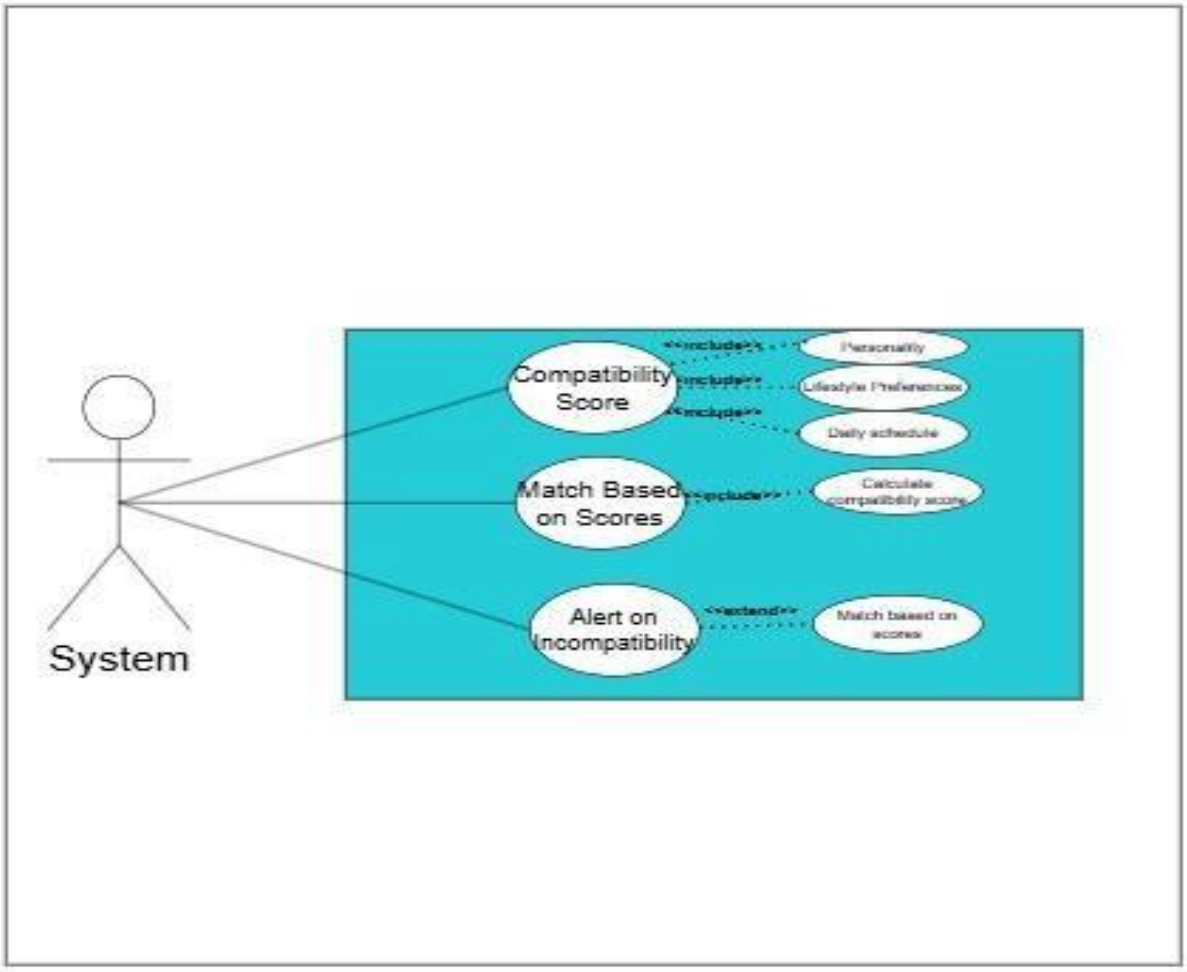
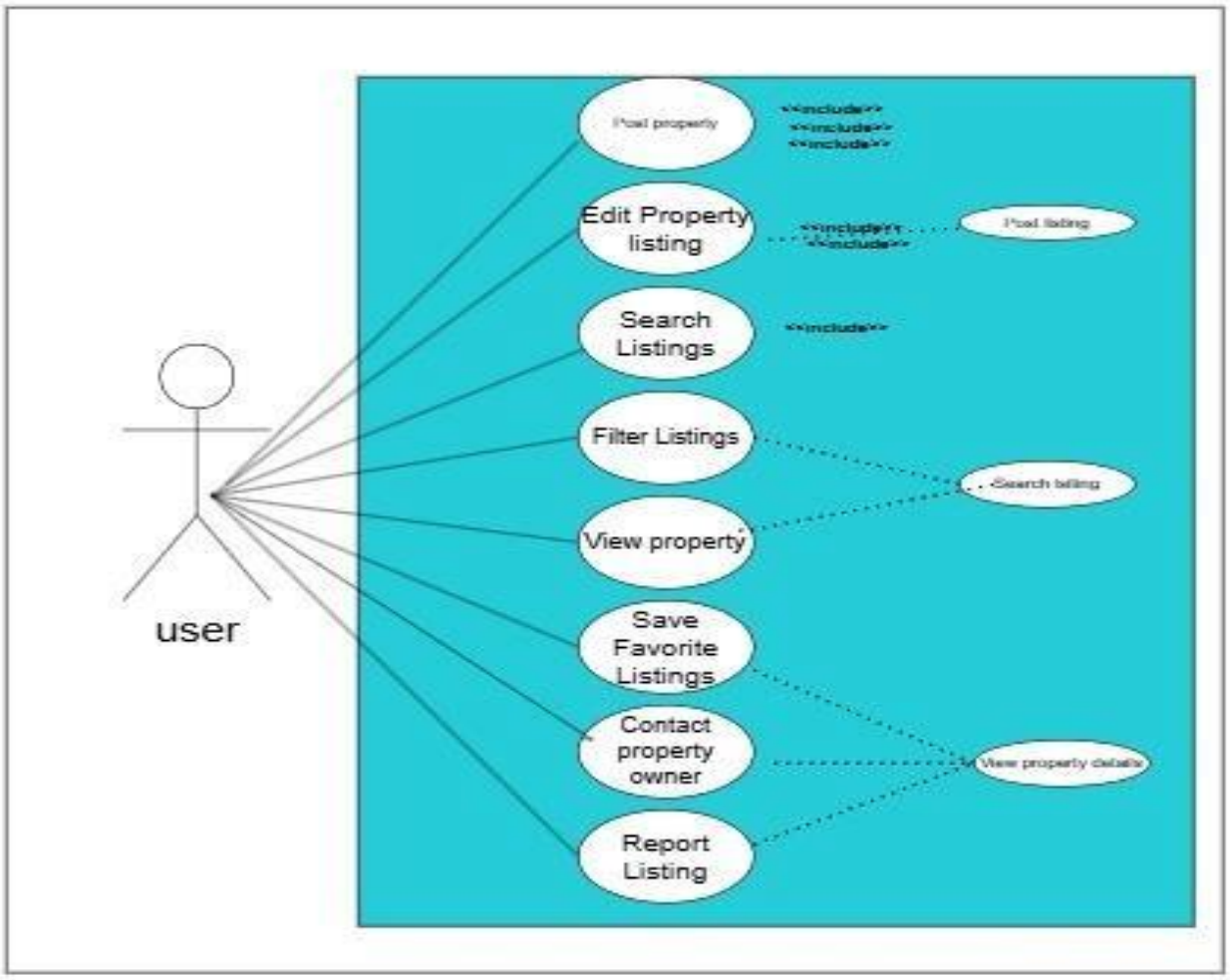


Figure 3.2.2: UC Diagram for Roommate Compatibility Module (System)

Module 3: Property & Room Listings



3.2.1: uc diagram for property and Room Listings (User)

Module 4: AI-based Matching Algorithm

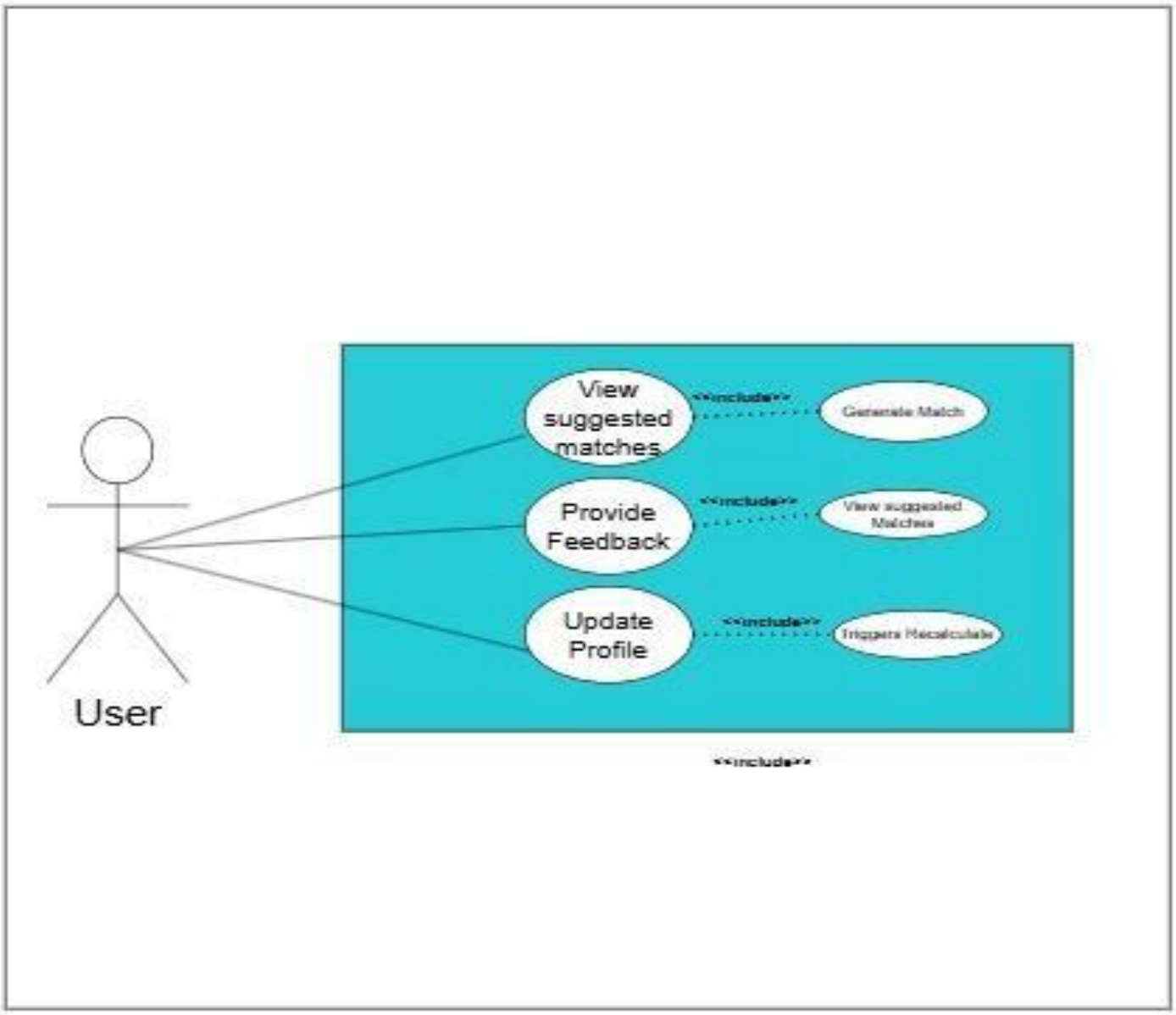


Figure 3.4.1: UC Diagram for AI-based Matching Algorithm Module (user)

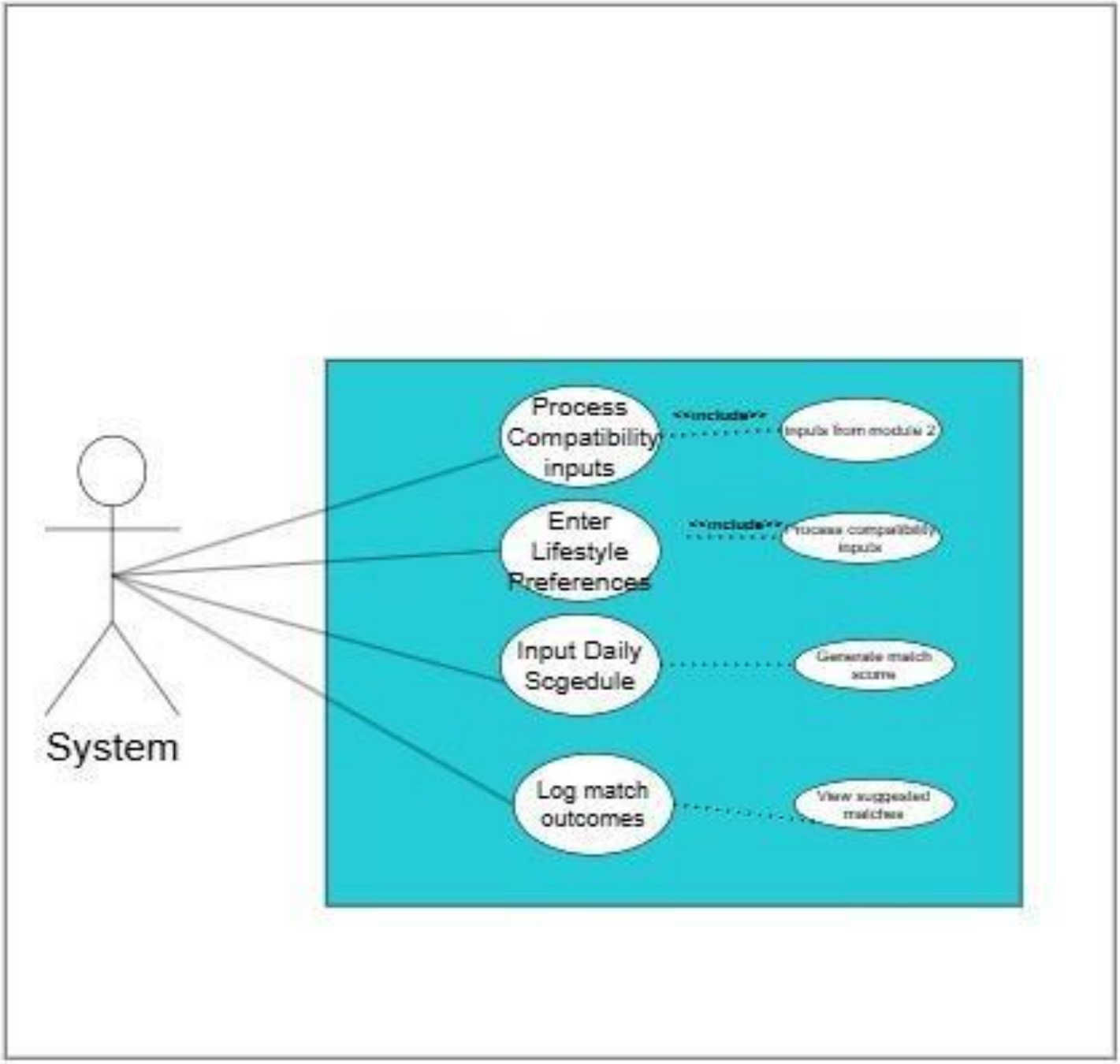


Figure 3.4.2: UC Diagram for AI-based Matching Algorithm Module (system)

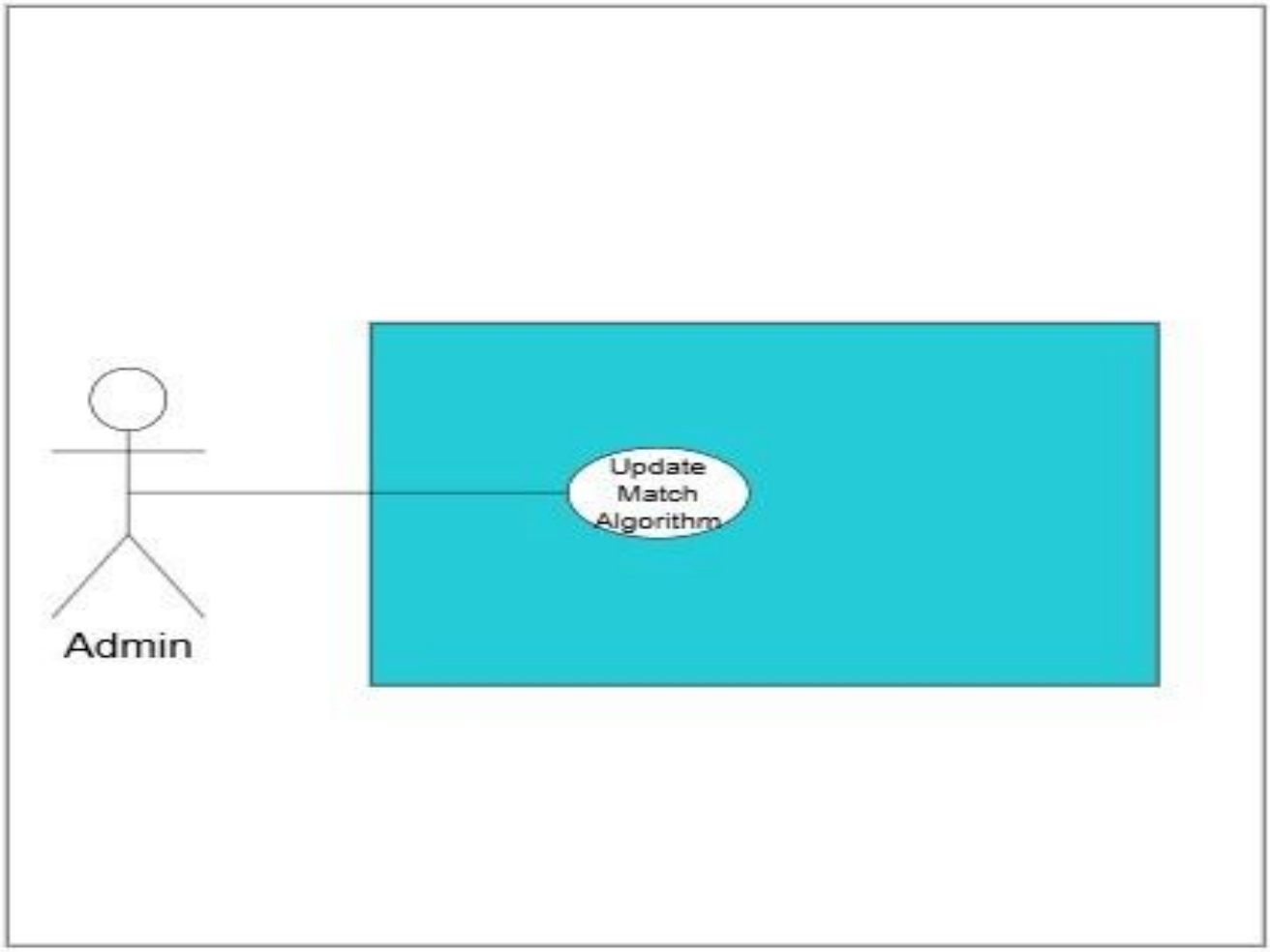


figure 3.4.3: UC Diagram for AI-based Matching Algorithm Module (Admin)

Module 5: Secure Chat & Agreement Templates

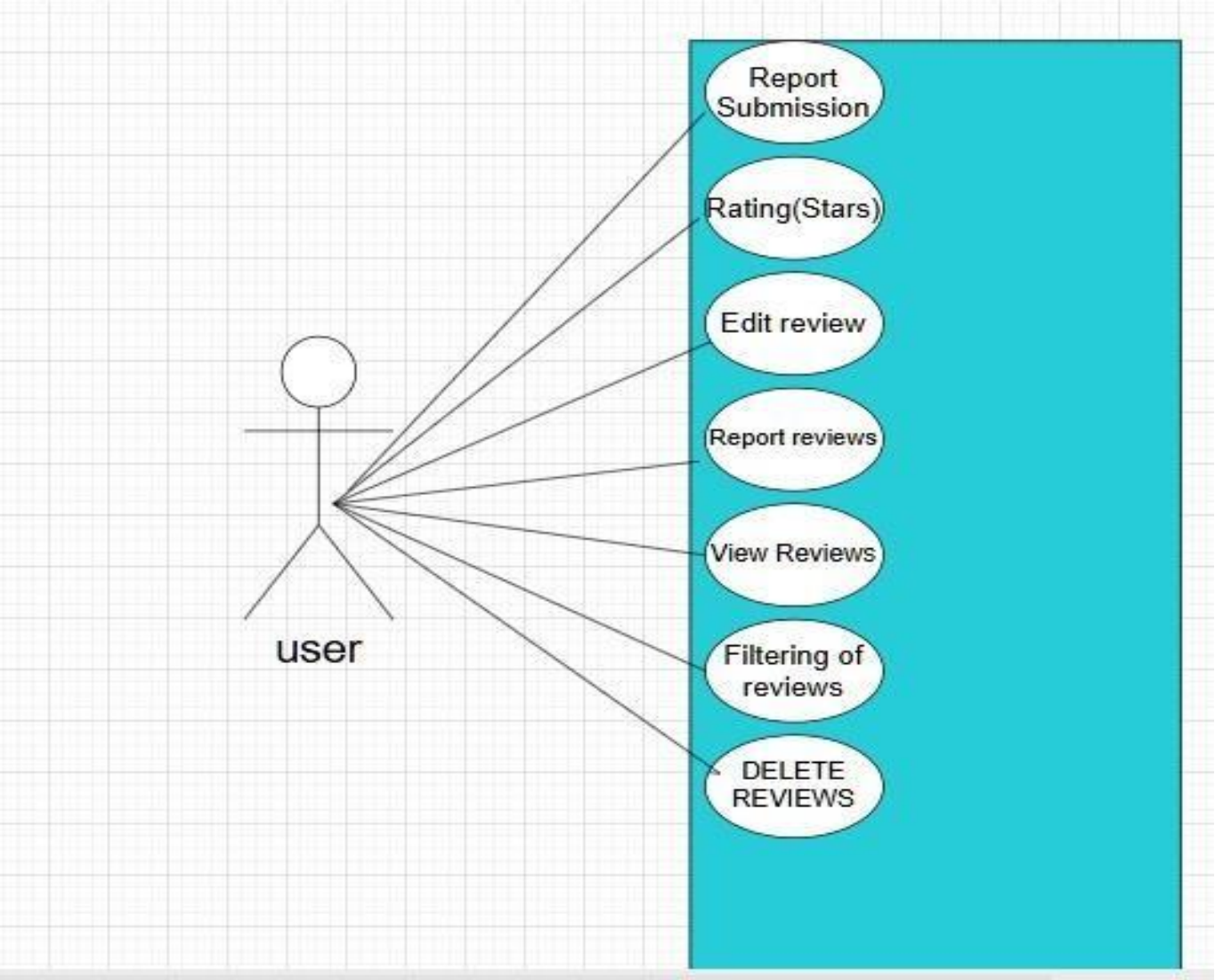


Figure 3.5.1: UC Diagram for Secure Chat & Agreement Templates Module (user)

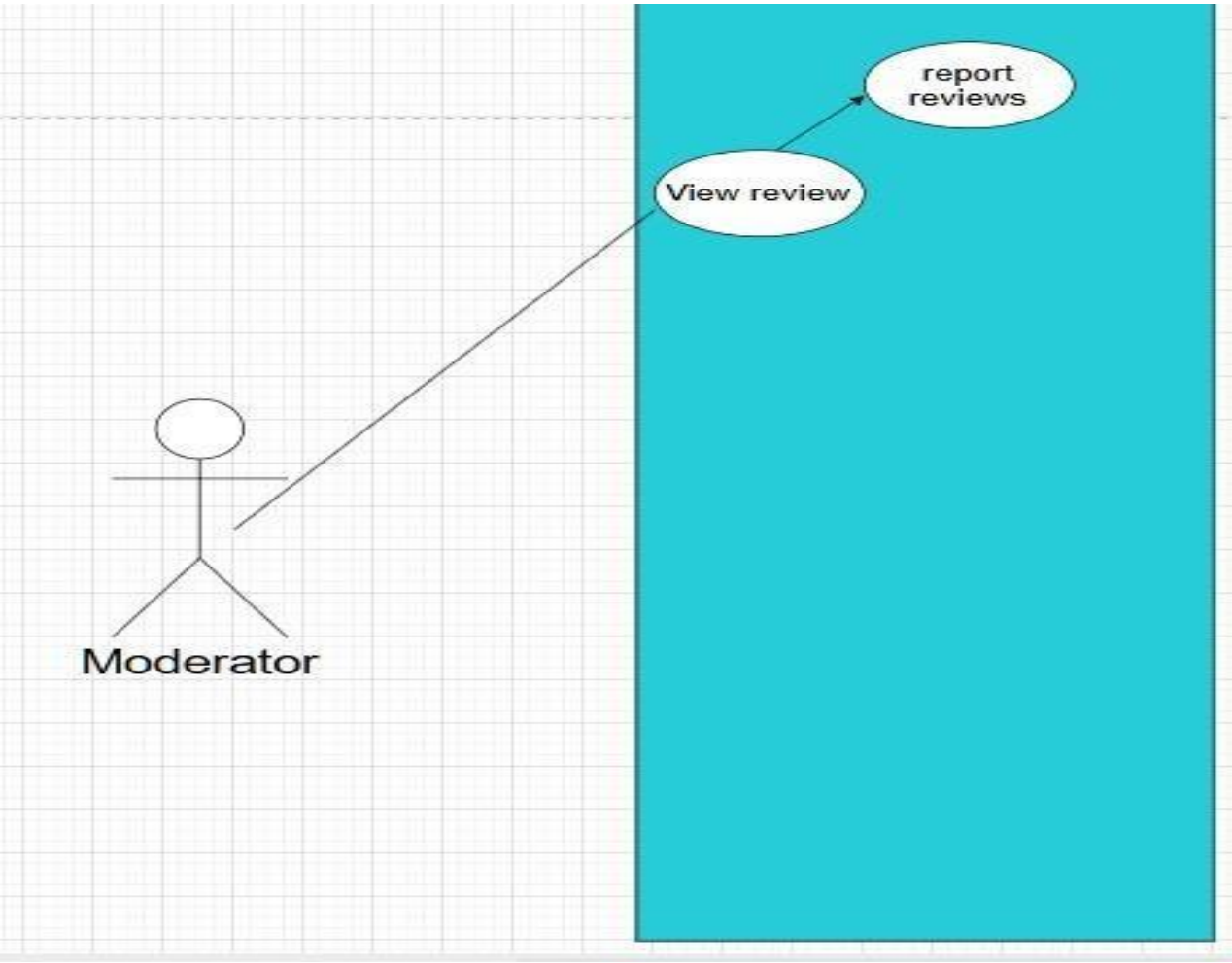


Figure 3.5.2: UC Diagram for Secure Chat & Agreement Templates Module (moderator)

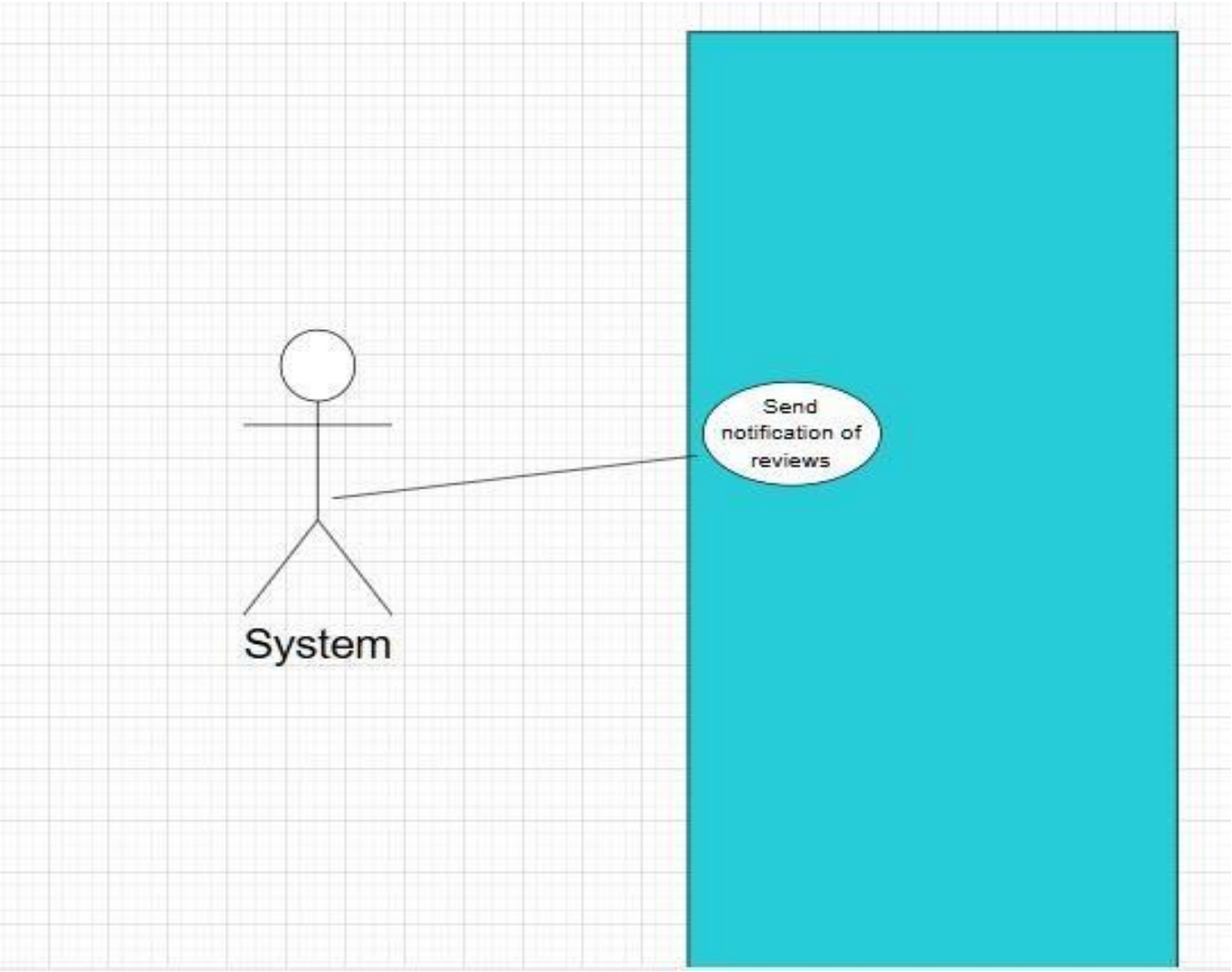


Figure 3.5.3: UC Diagram for Secure Chat & Agreement Templates Module (system)

Module 6: Review & Rating System

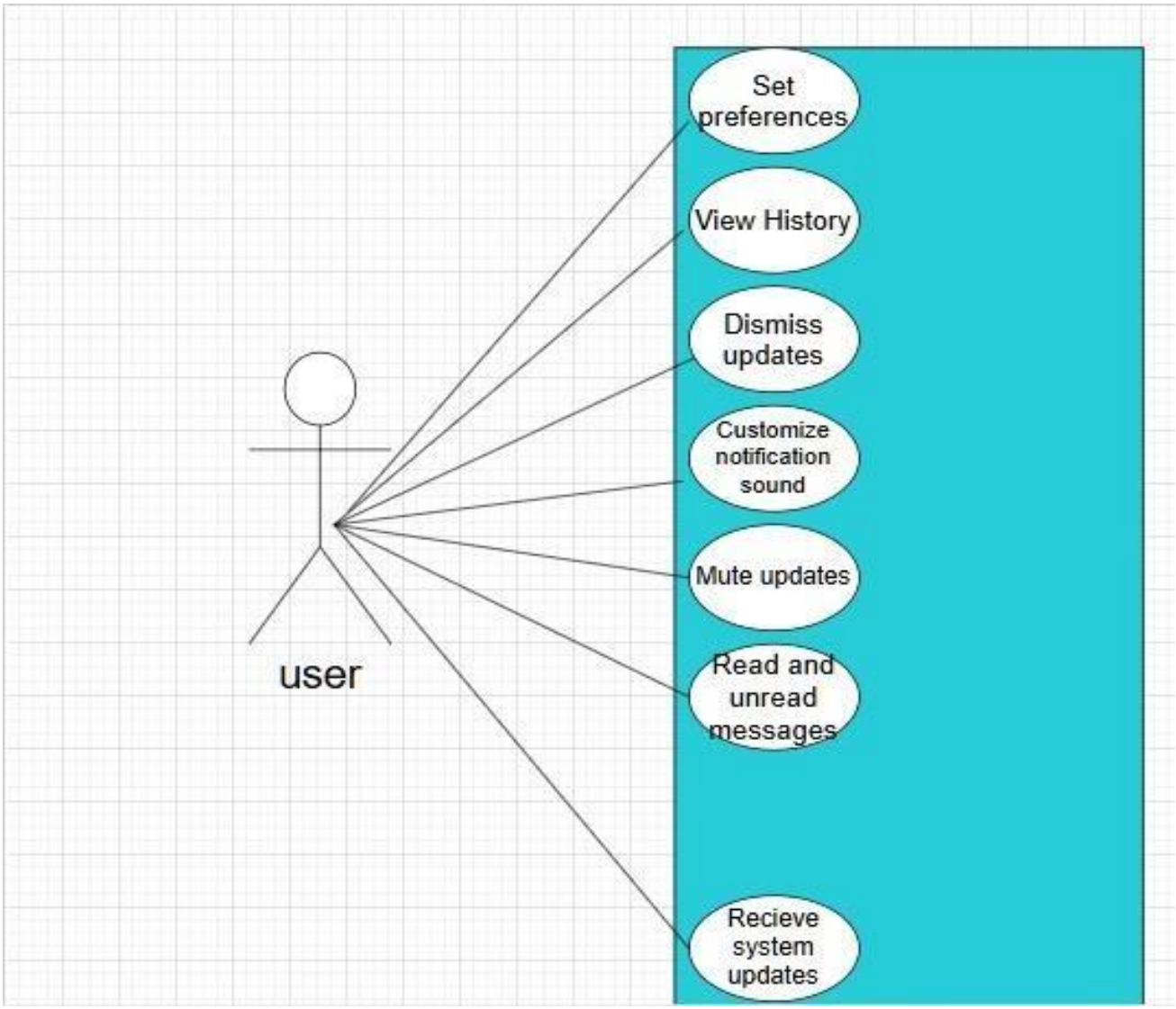


Figure 3.6.1: UC Diagram for Secure Review & Rating System Module (user)

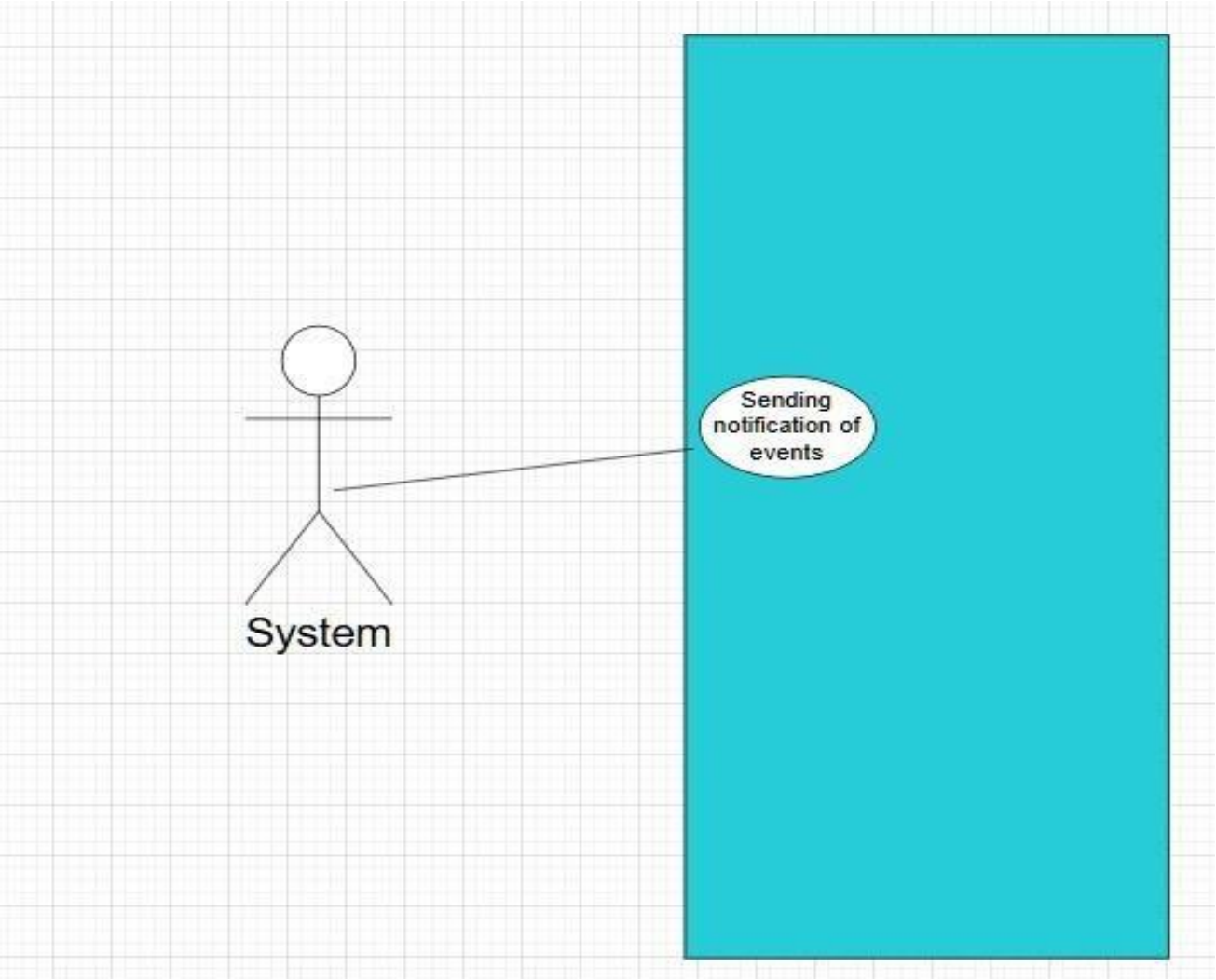


Figure 3.6.2: UC Diagram for Secure Review & Rating System Module (system)

Module 7: Notification & Alert System

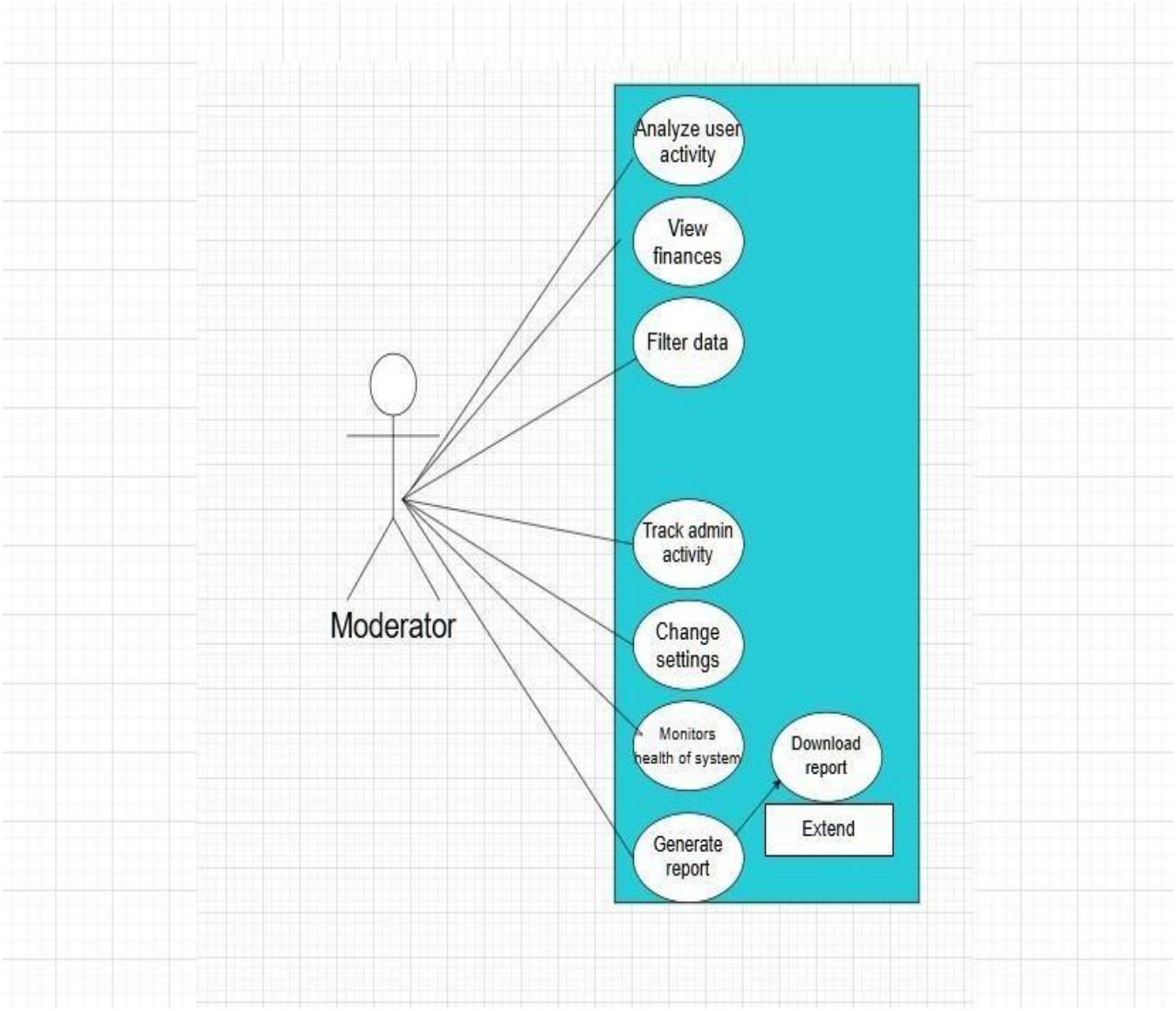


Figure 3.7.1: UC Diagram for Notification & Alert System Templates Module (moderator)

Module 8: Admin Dashboard & Analytic

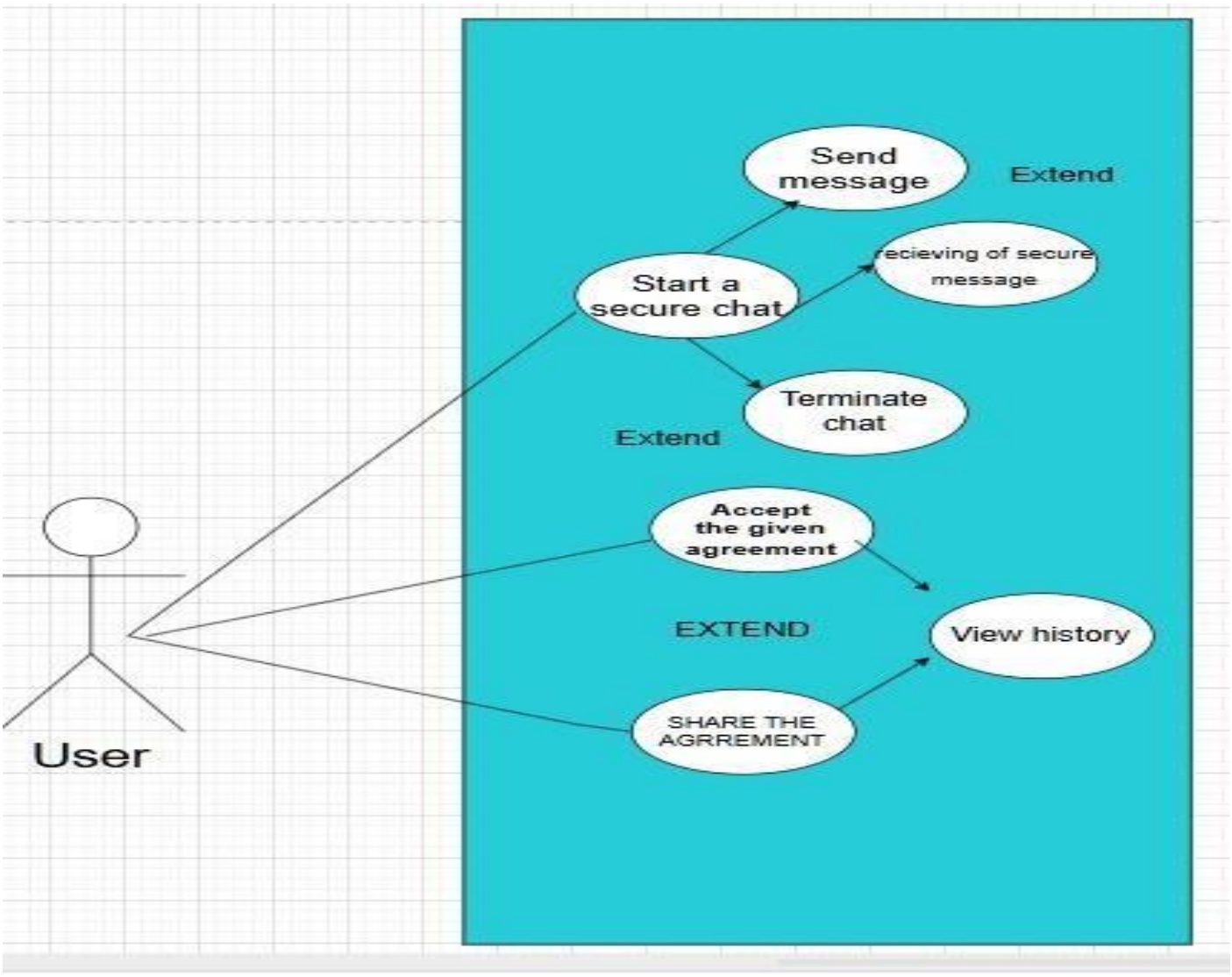
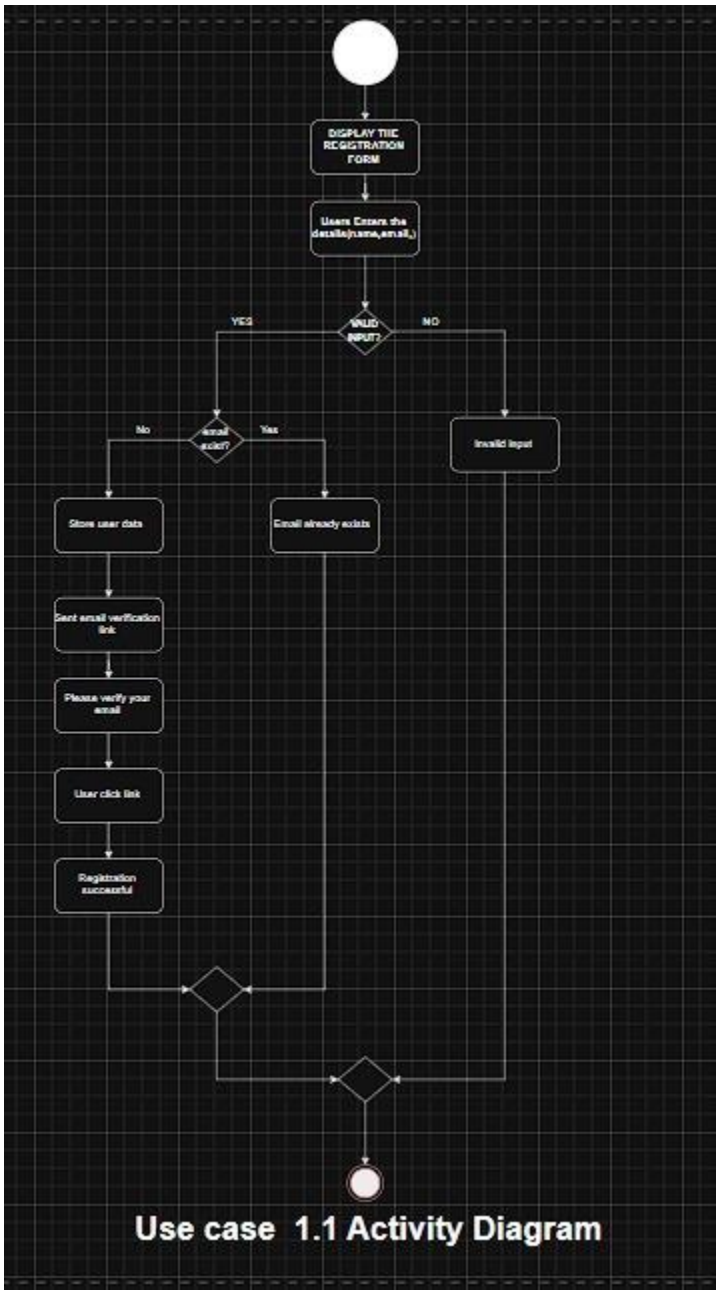


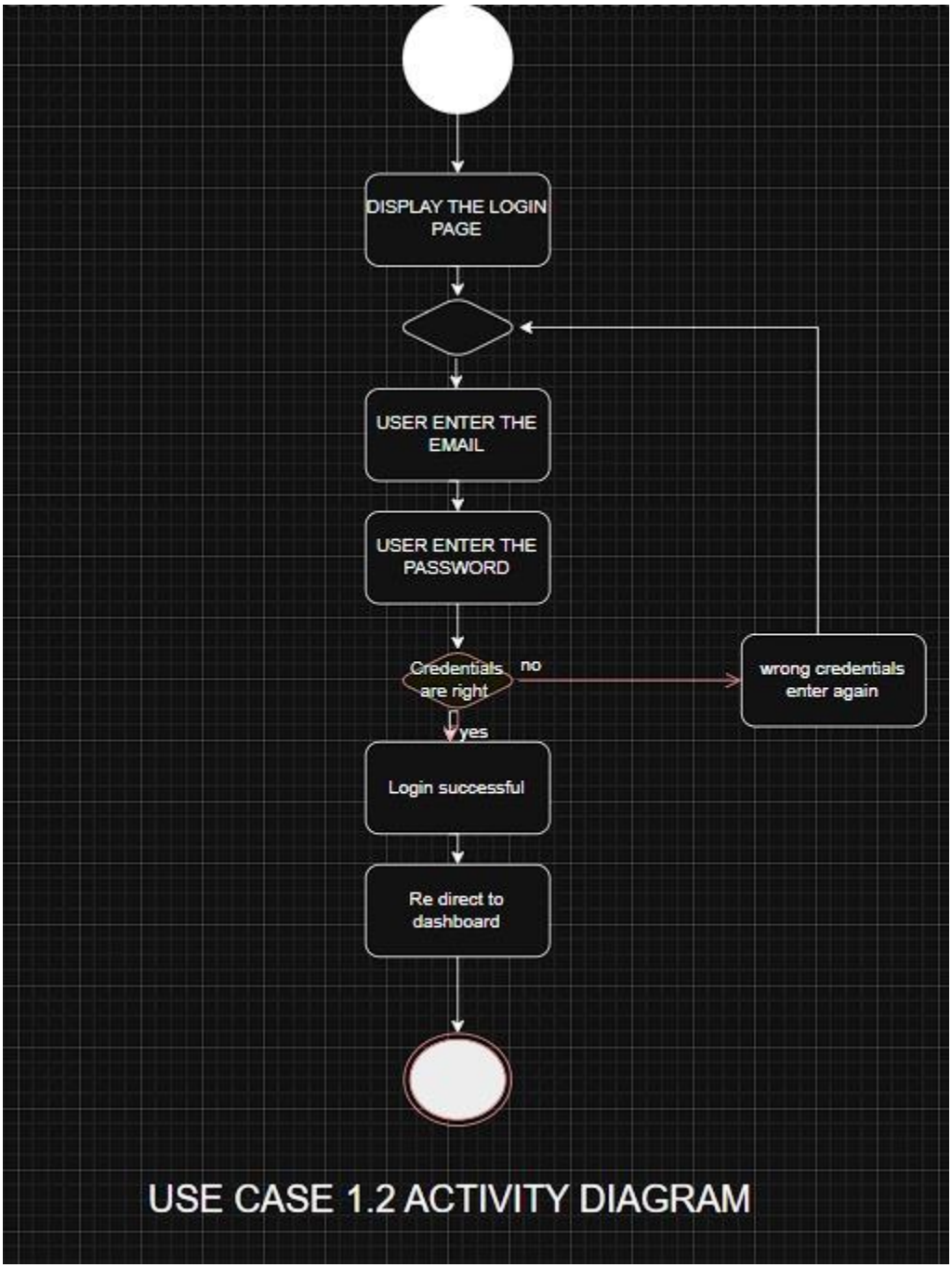
Figure 3.8.1: UC Diagram for Admin Dashboard & Analytic Templates Module (User)

5. Design Models

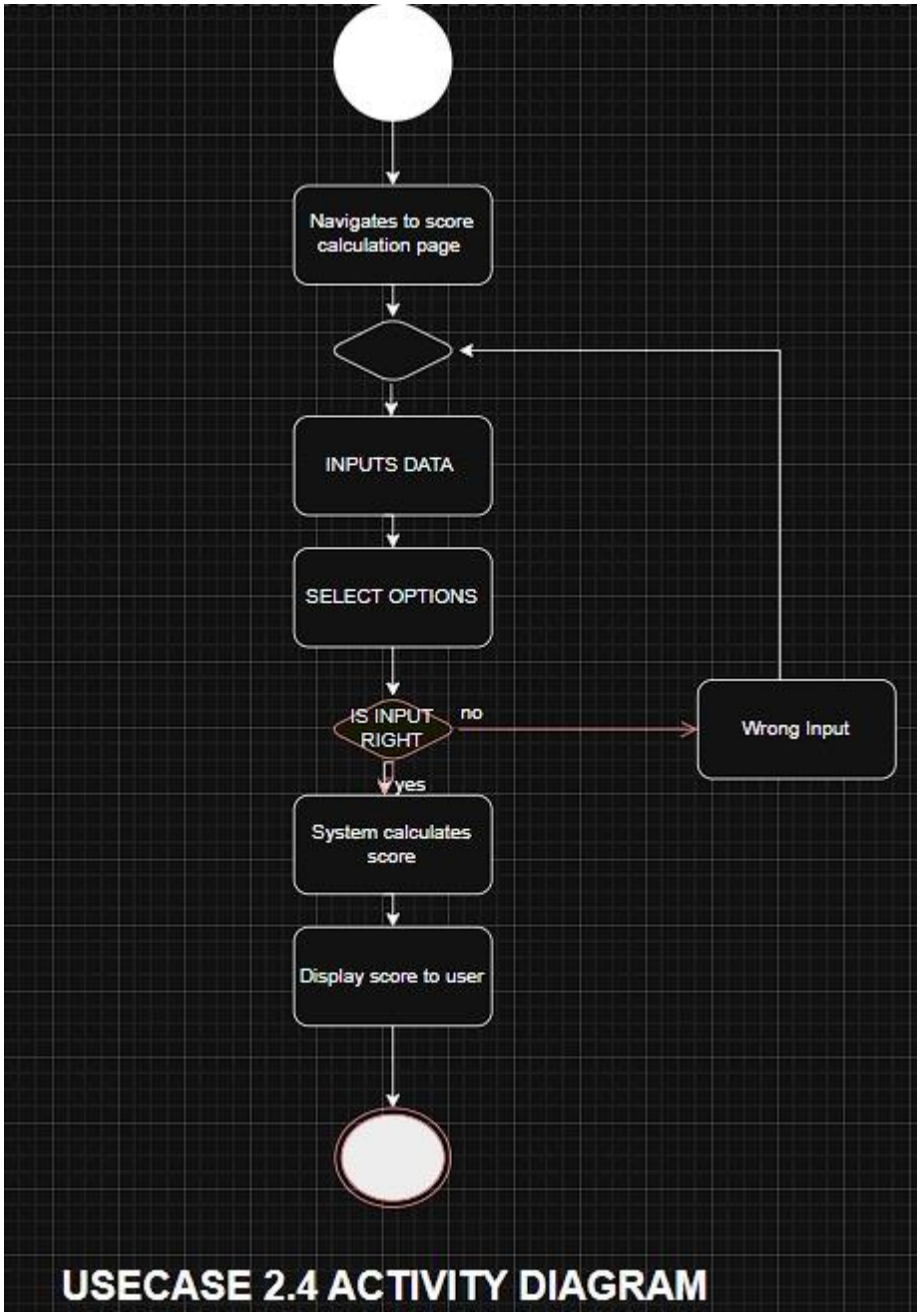
5.1 Activity Diagrams



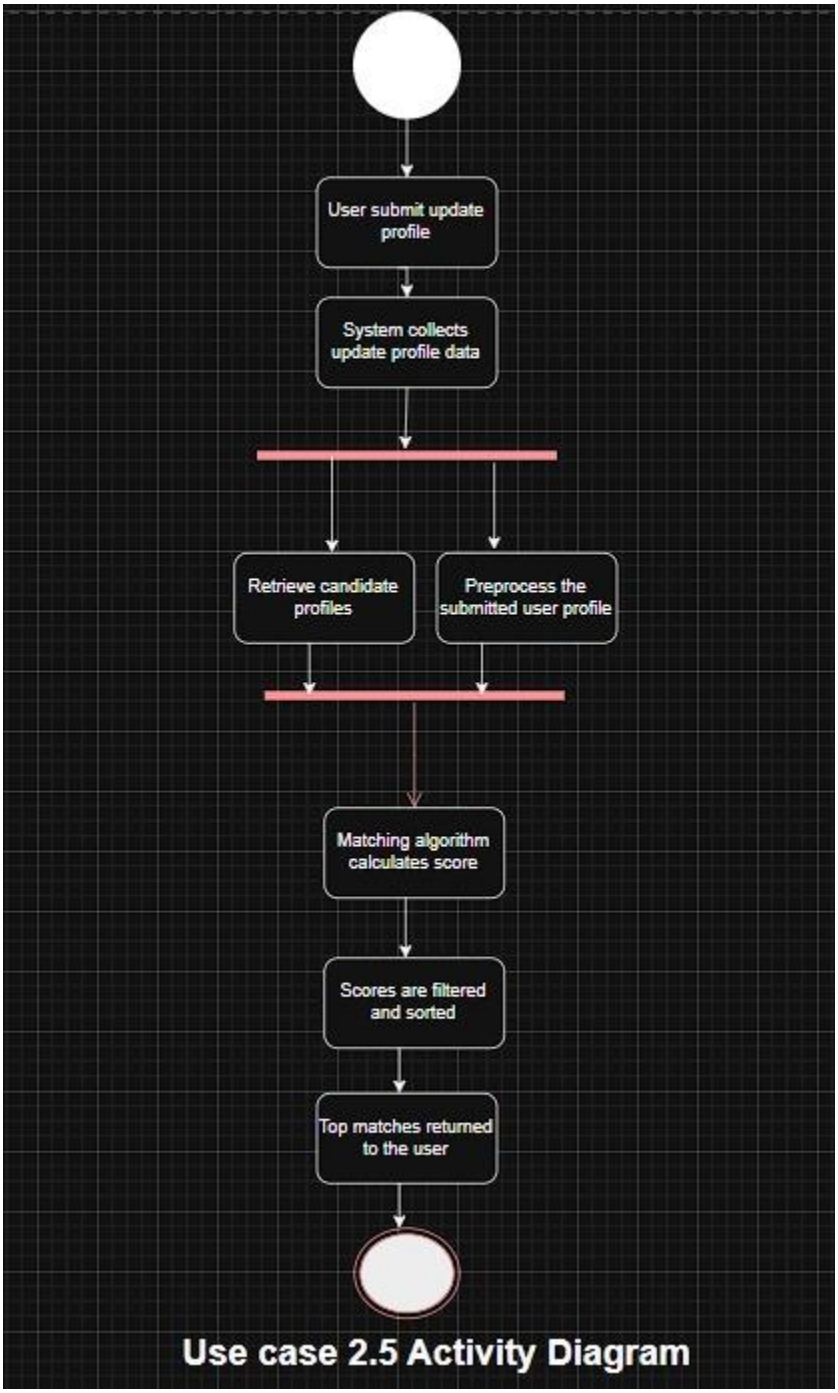
M1-UC-1.1-User Registration



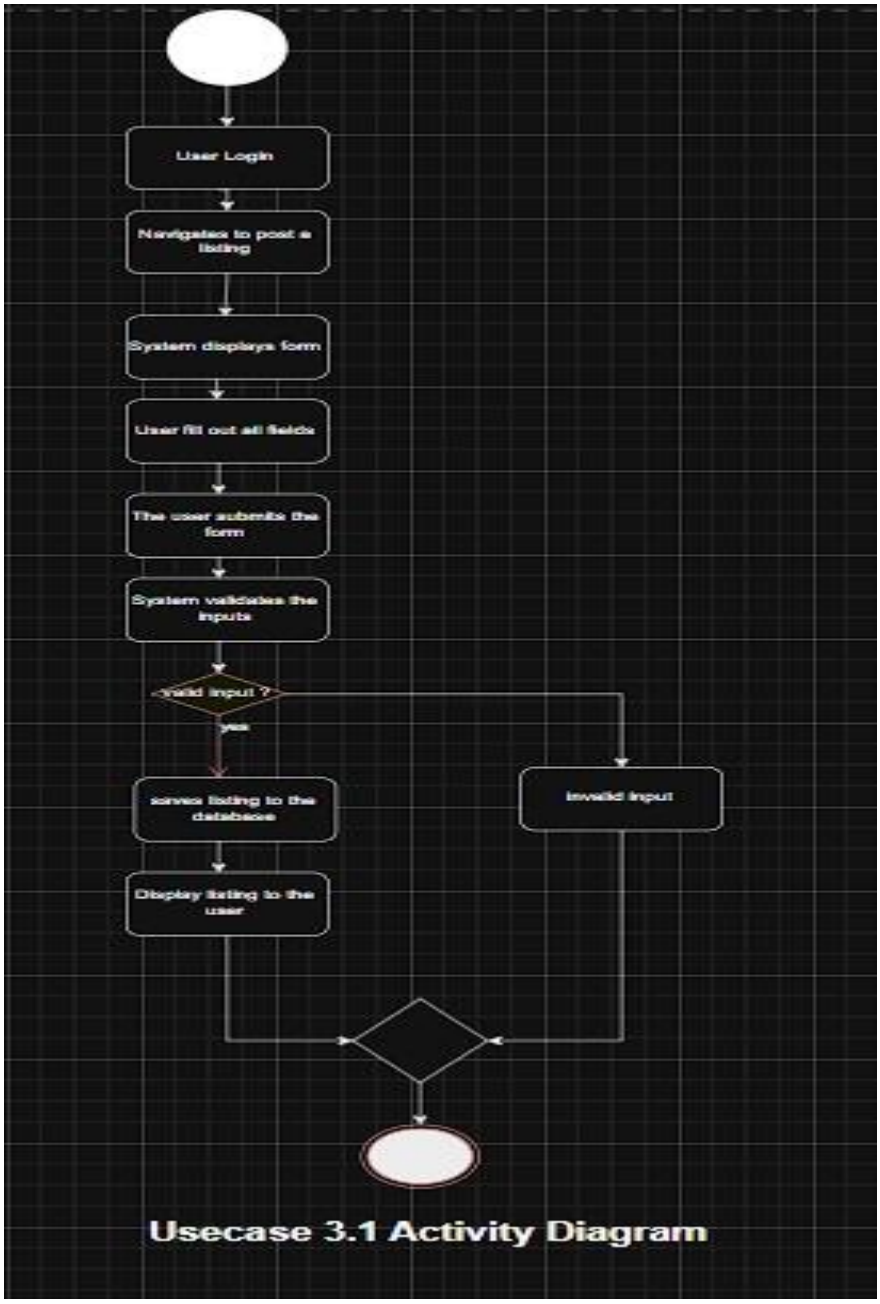
M1-UC-1.2-User Login



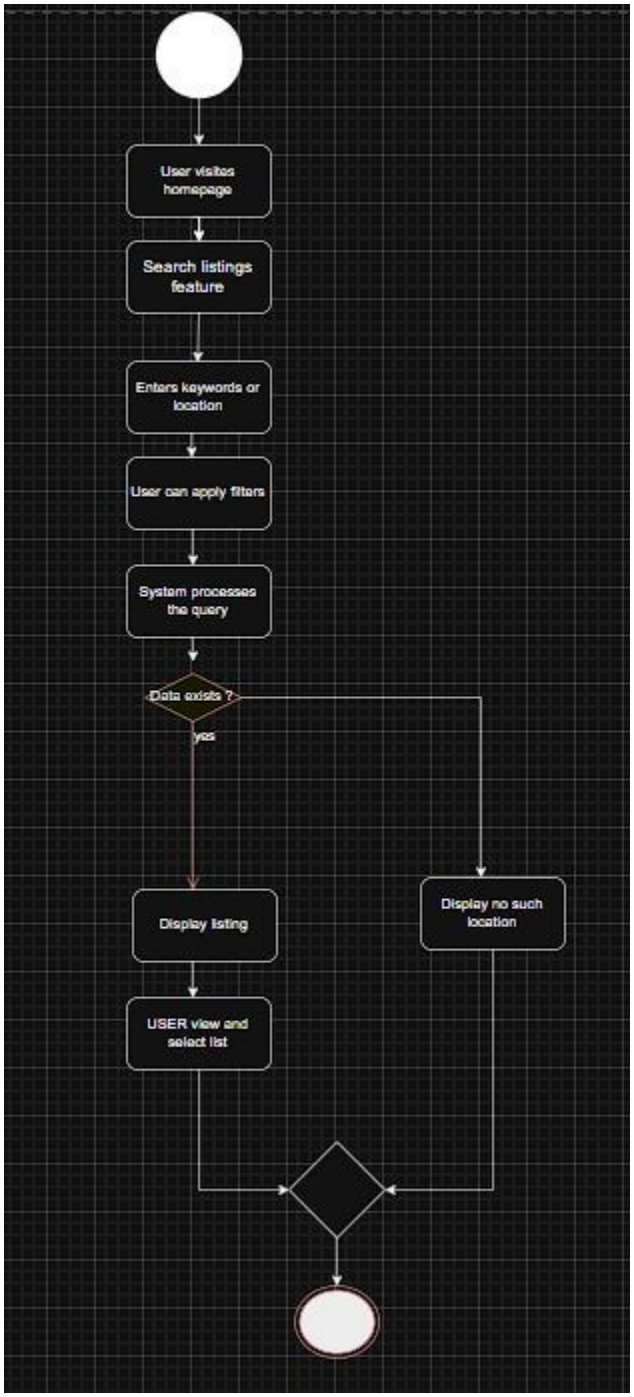
M2-UC-2.4 Compatibility Score calculation



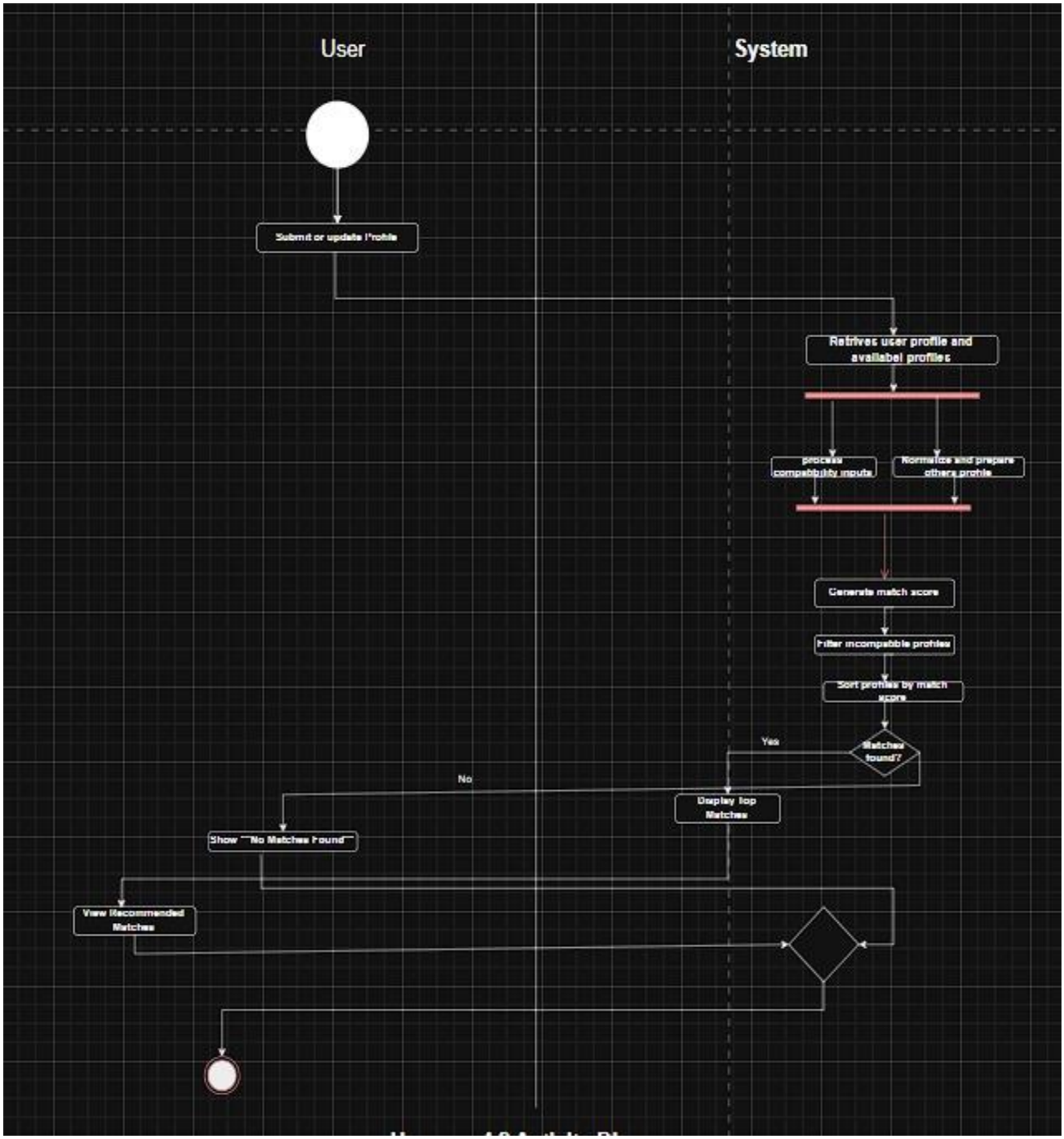
M2-UC-2.5- Profile Matching Based on Scores



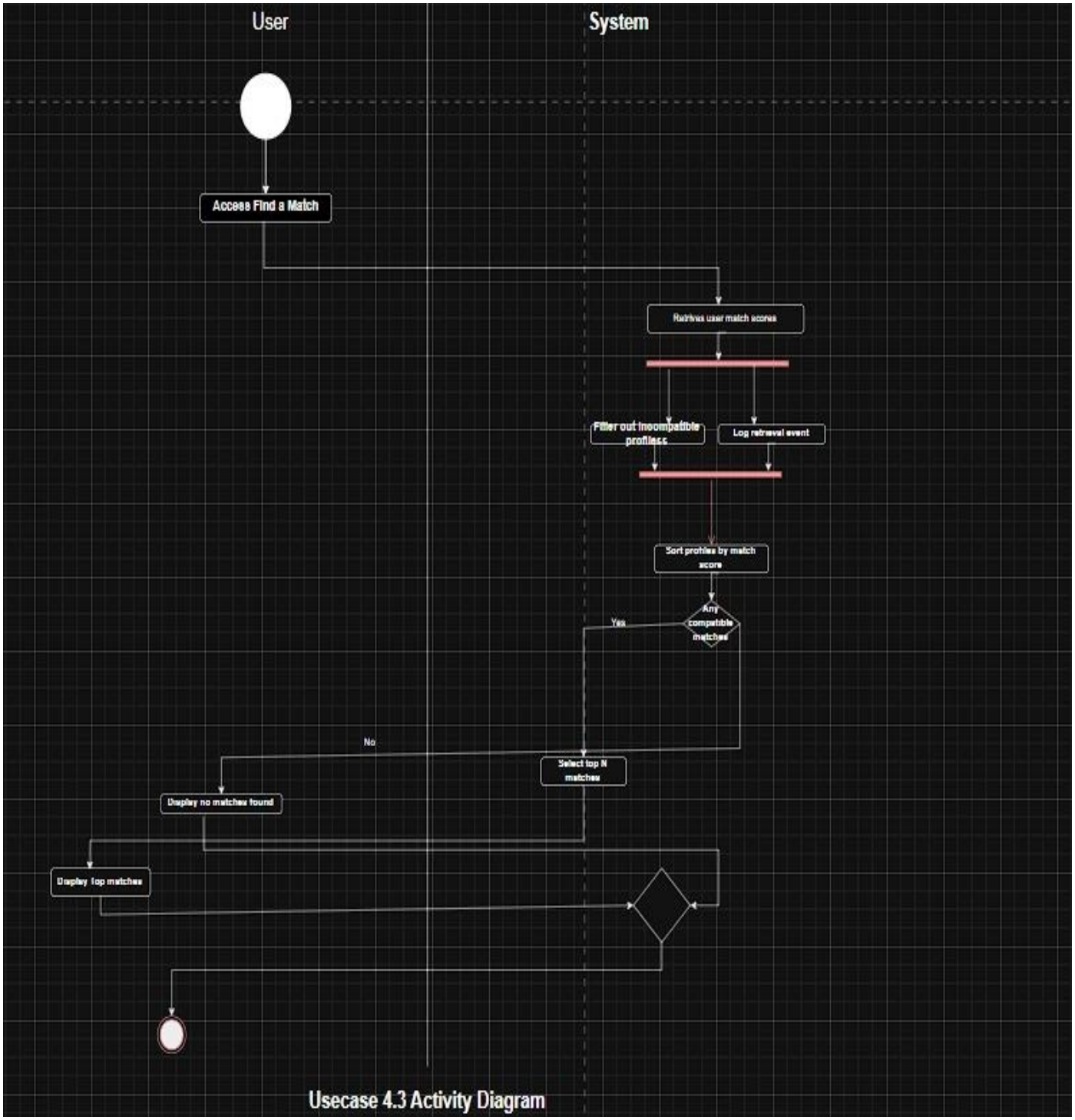
M3-UC-3.1-Post property listing



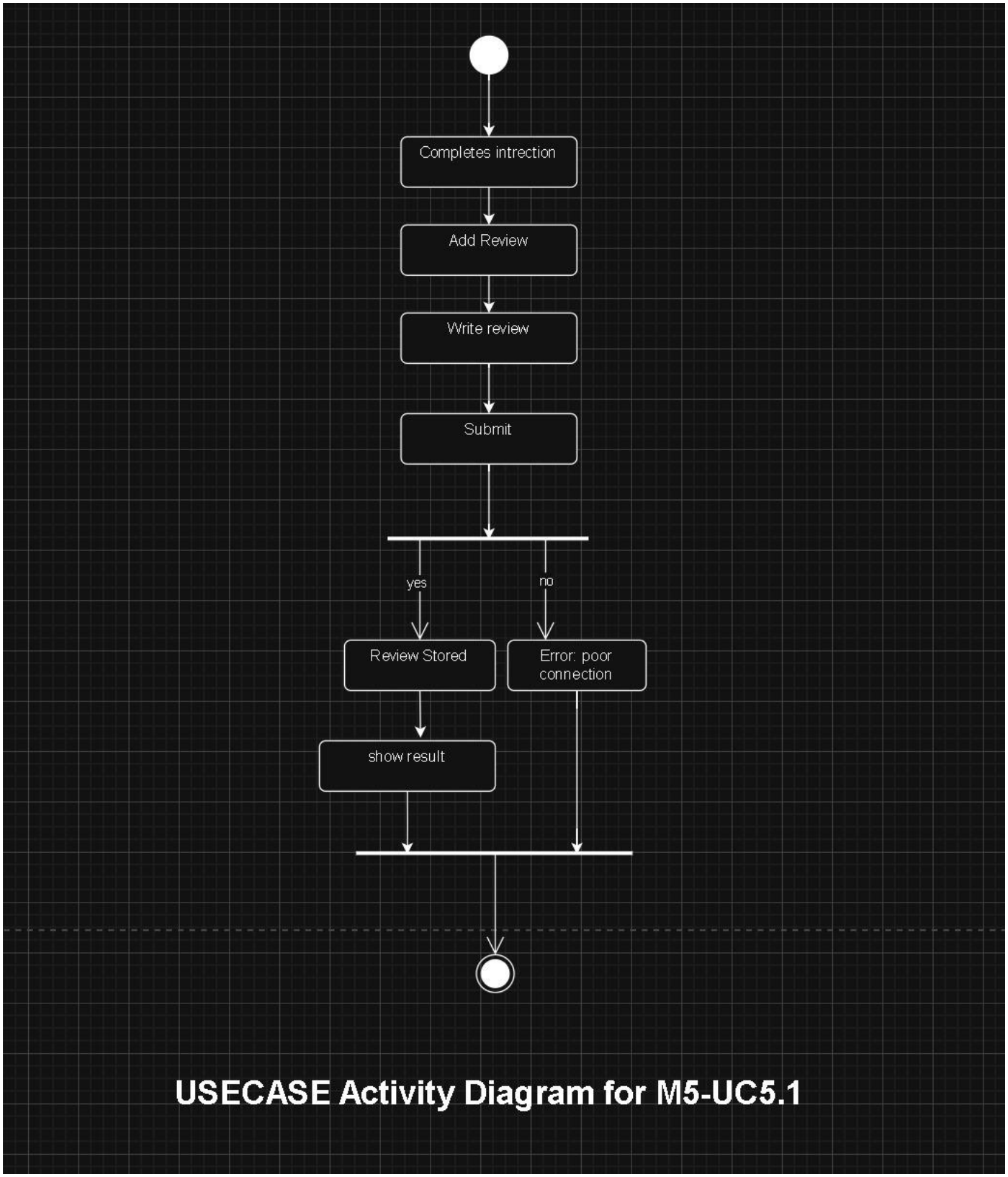
M3-UC-3.3-Search Property Listing



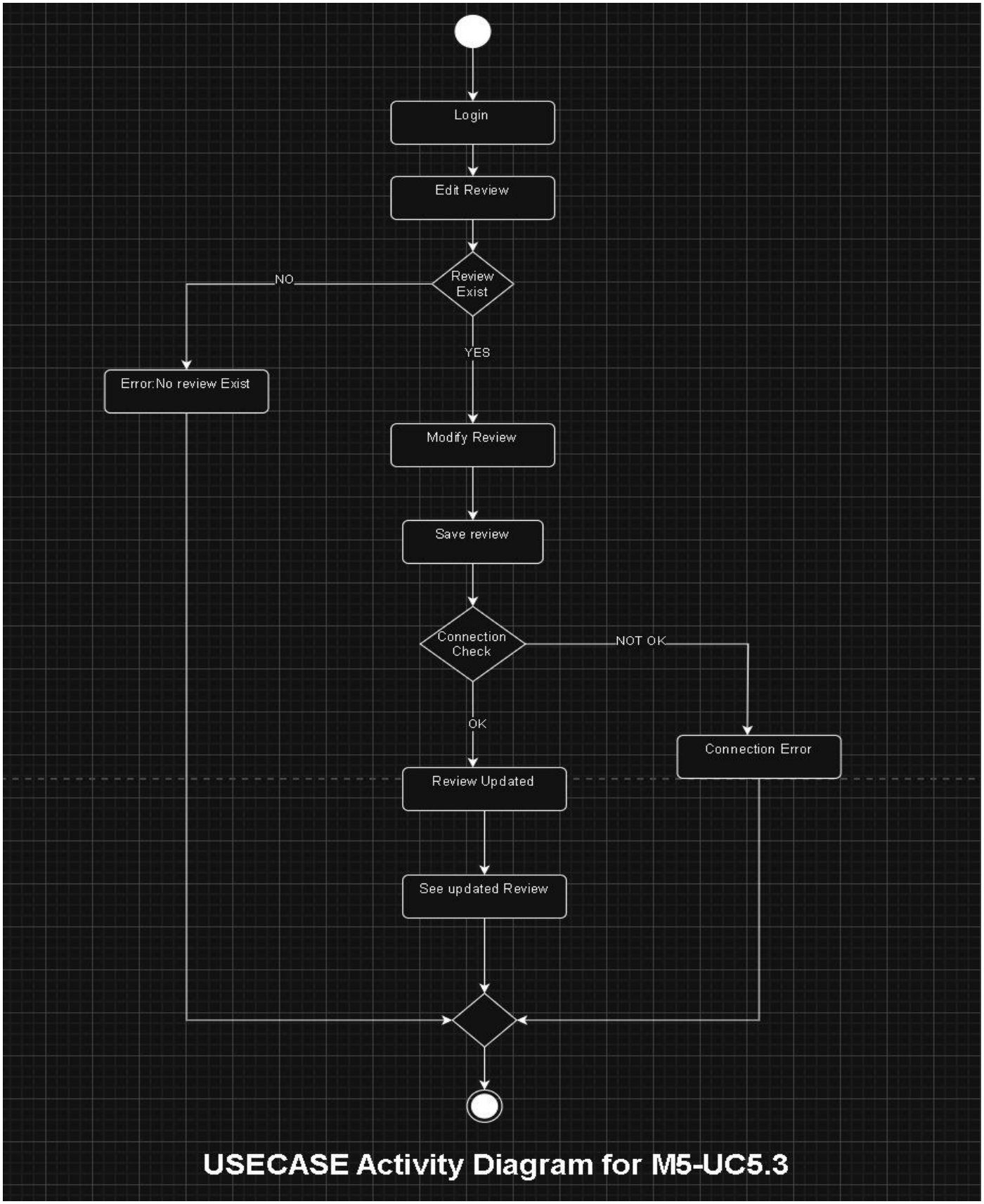
M4-UC-4.2- Generate Match Scores



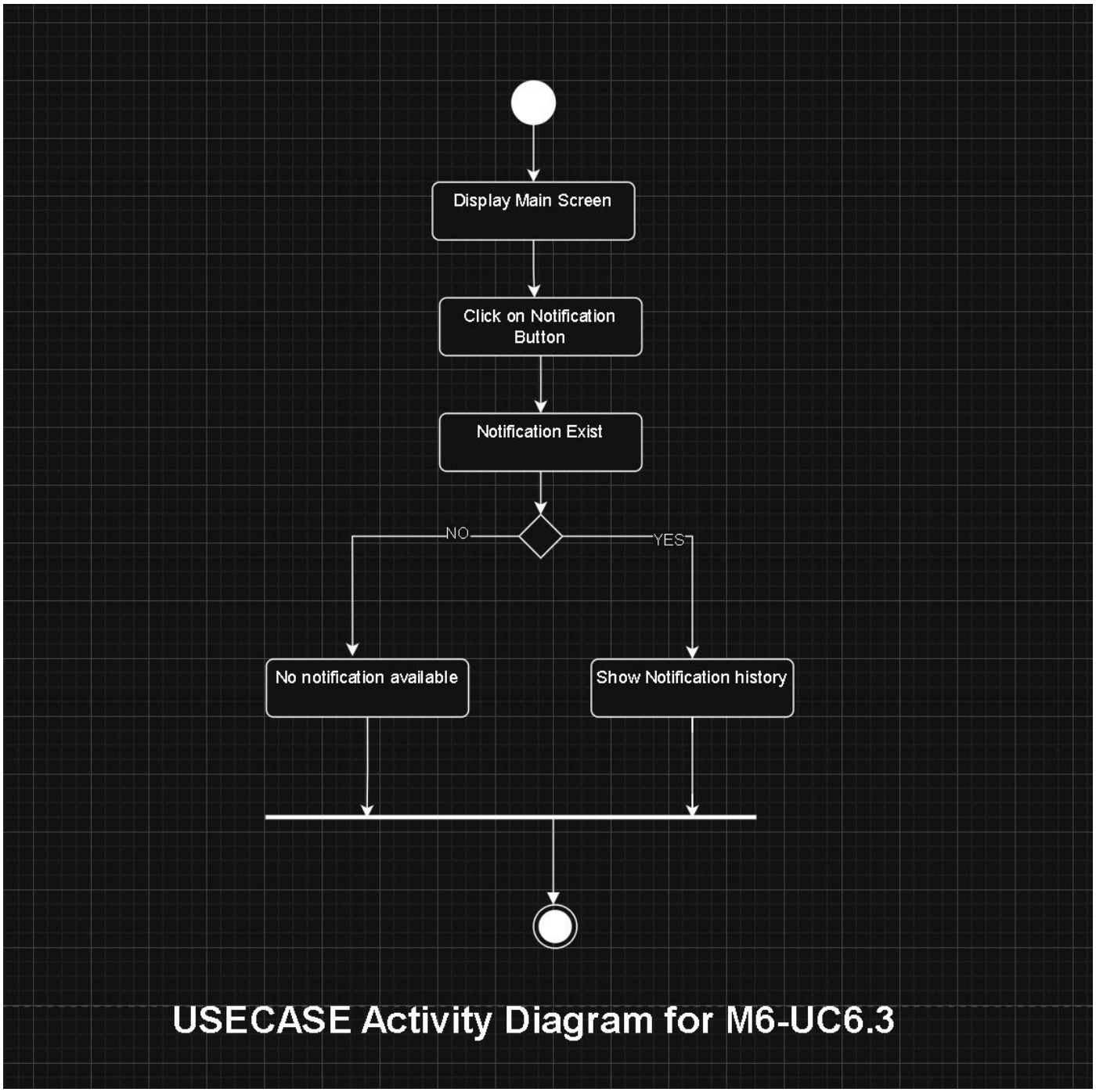
M4-UC-4.3-Suggest top matches



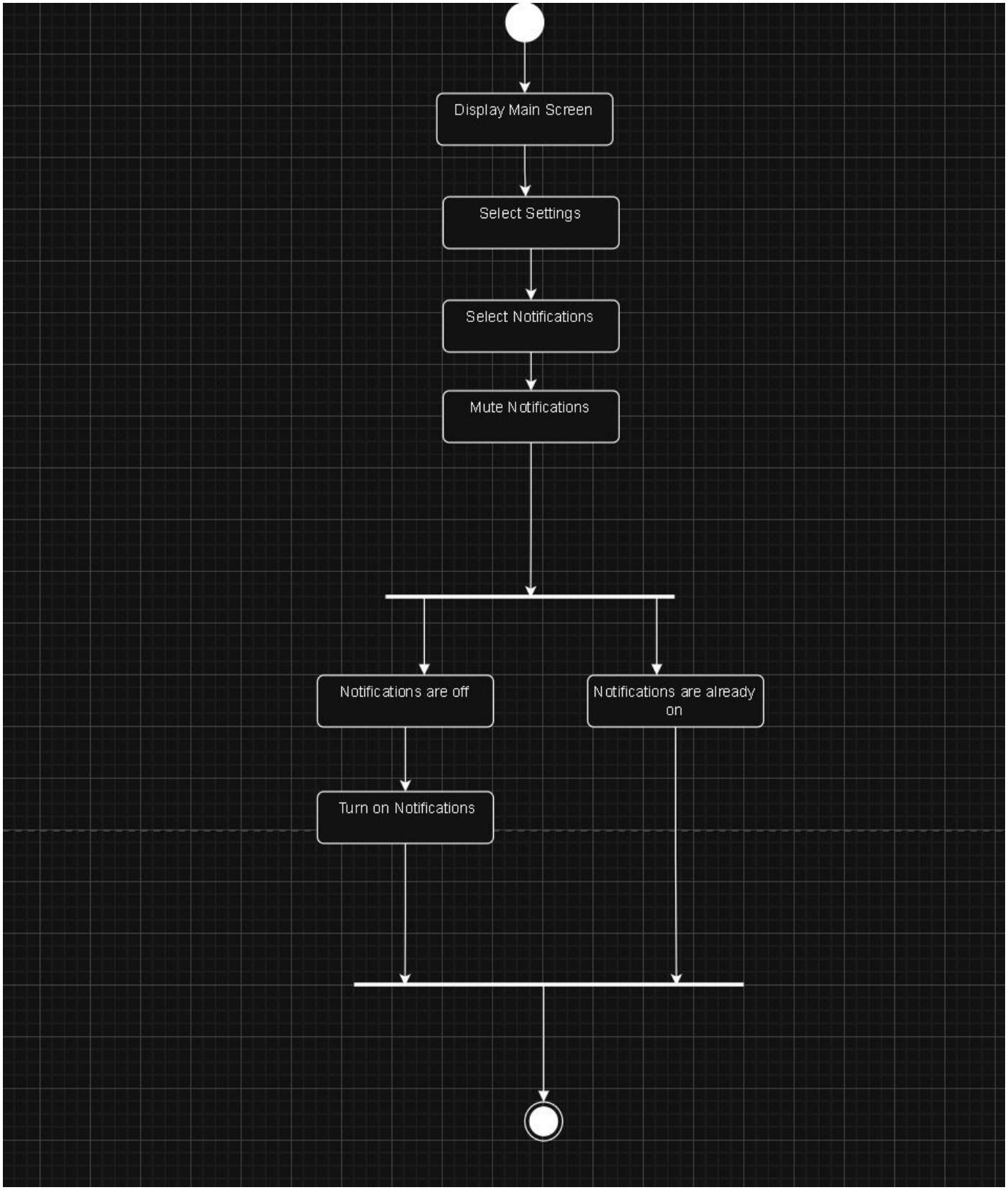
M5-UC-5.1- Report submission



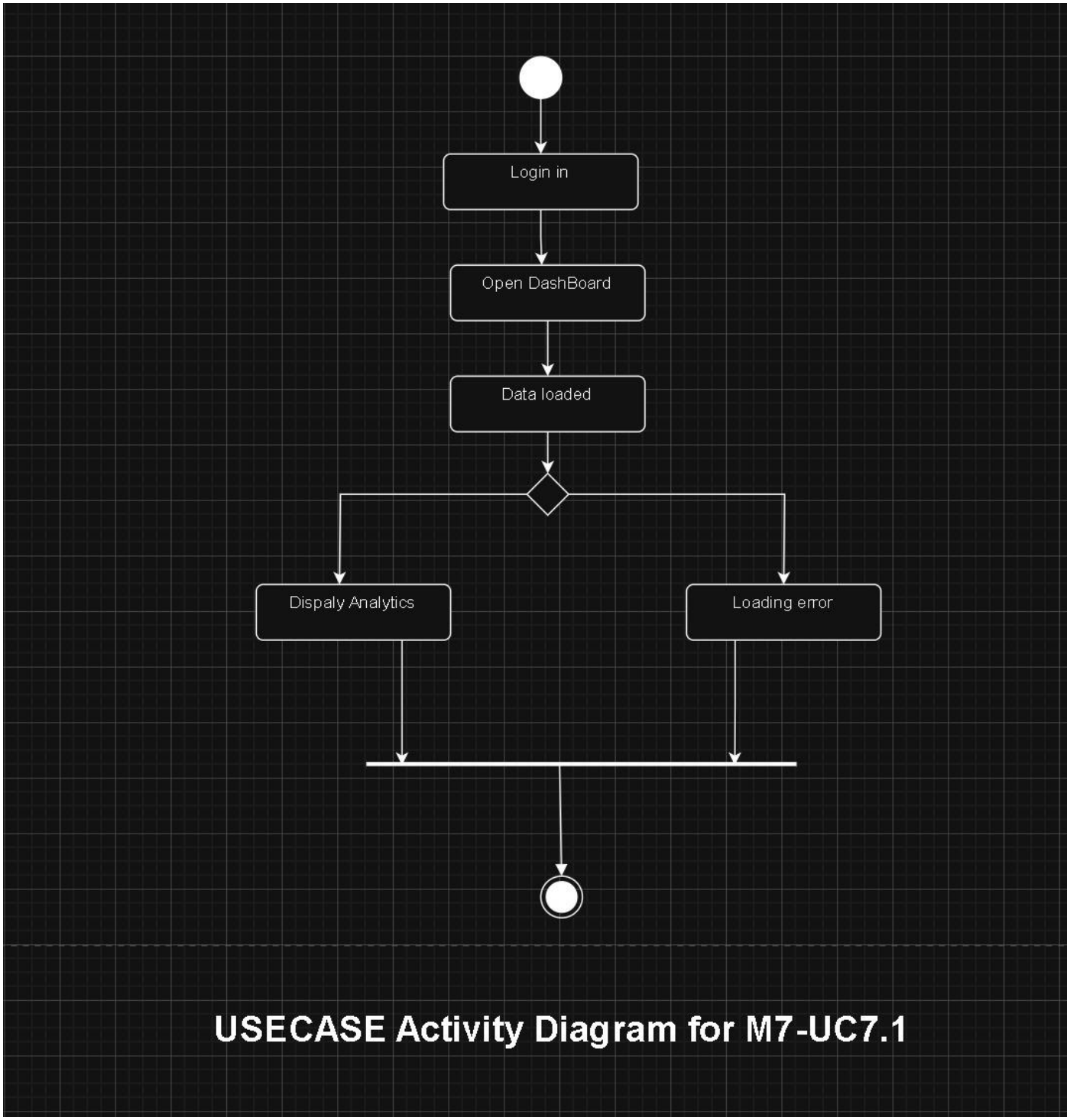
M5-UC-5.3-Editing a Review



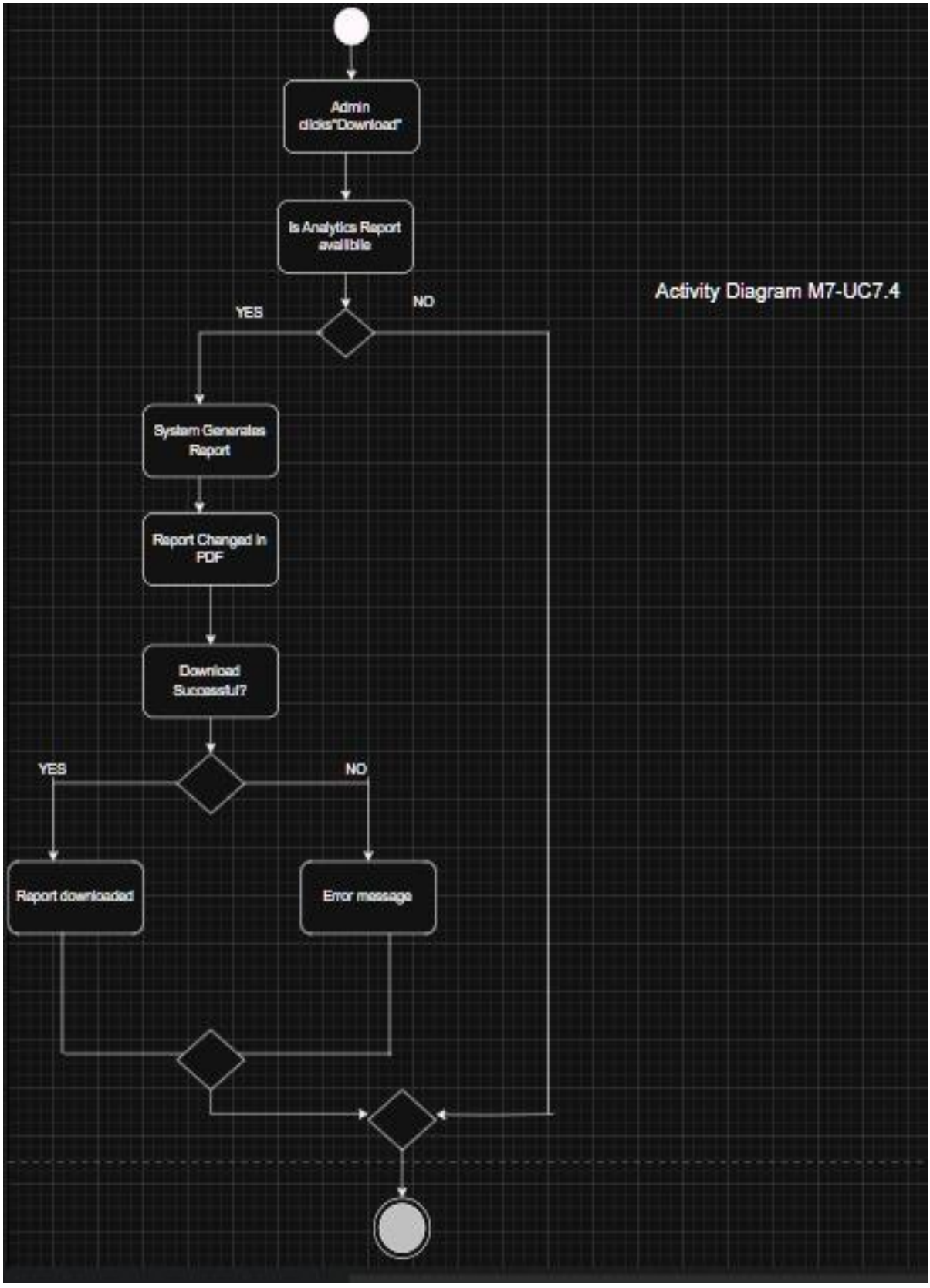
M6-UC-6.3- View Notification History



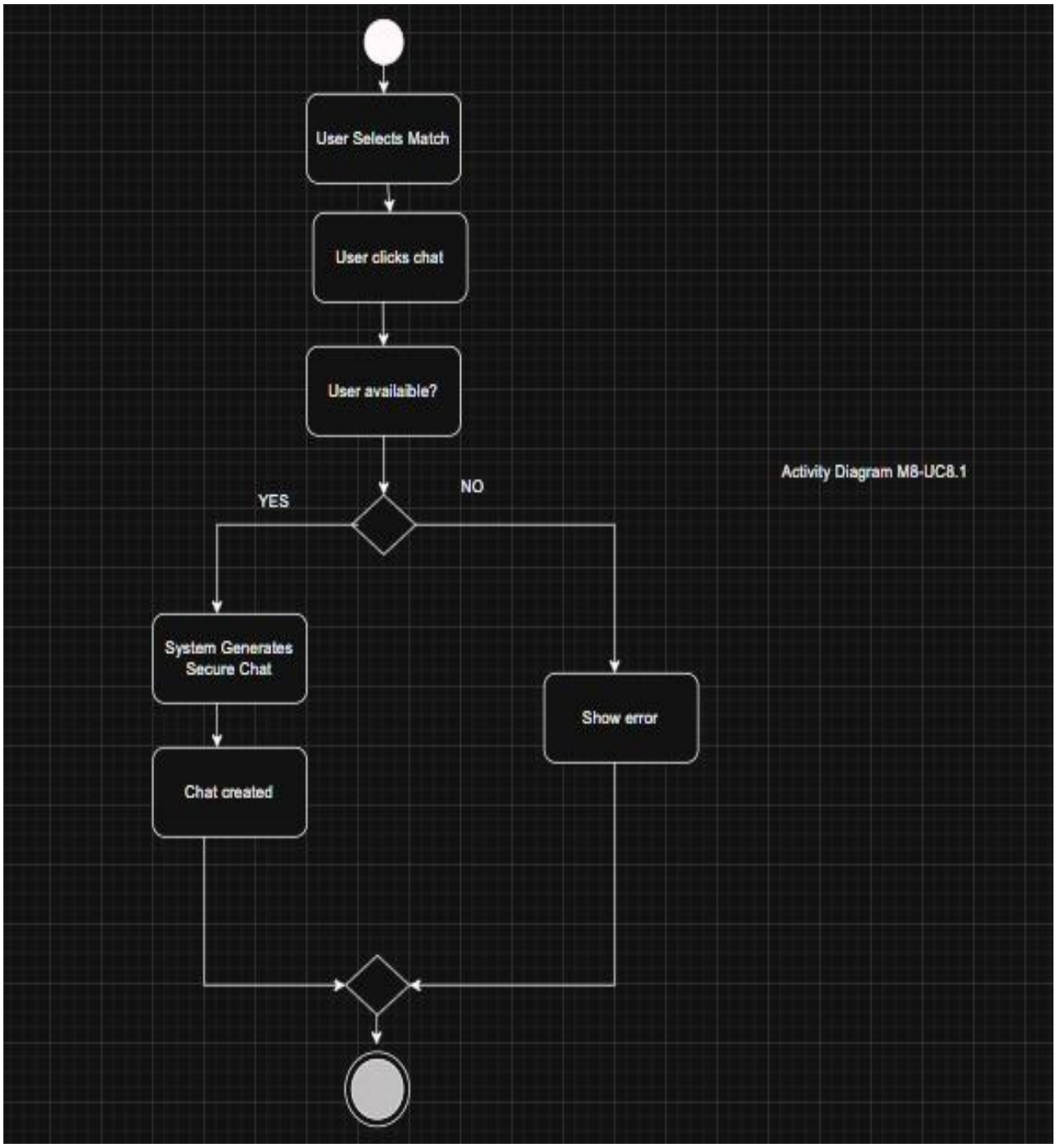
M6-UC-6.6- ***Mute Notifications***



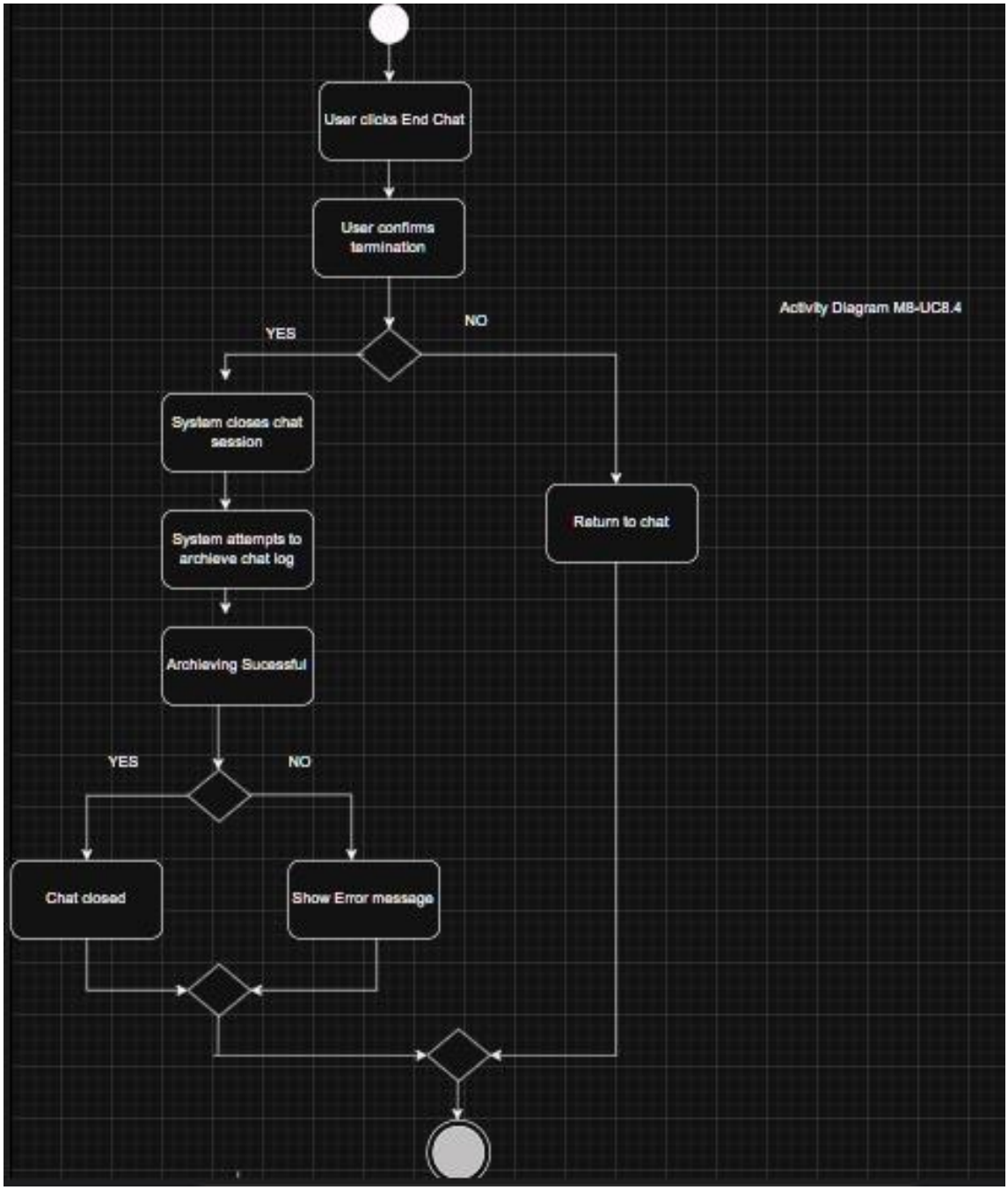
M7-UC-7.1- View User Activity Analytics



M7-UC-7.4- Download Analytics Report



M8-UC-8.1- Initiate Secure Chat



M8-UC-8.4- Terminate Chat Session

-

5.2 Sequence Diagrams

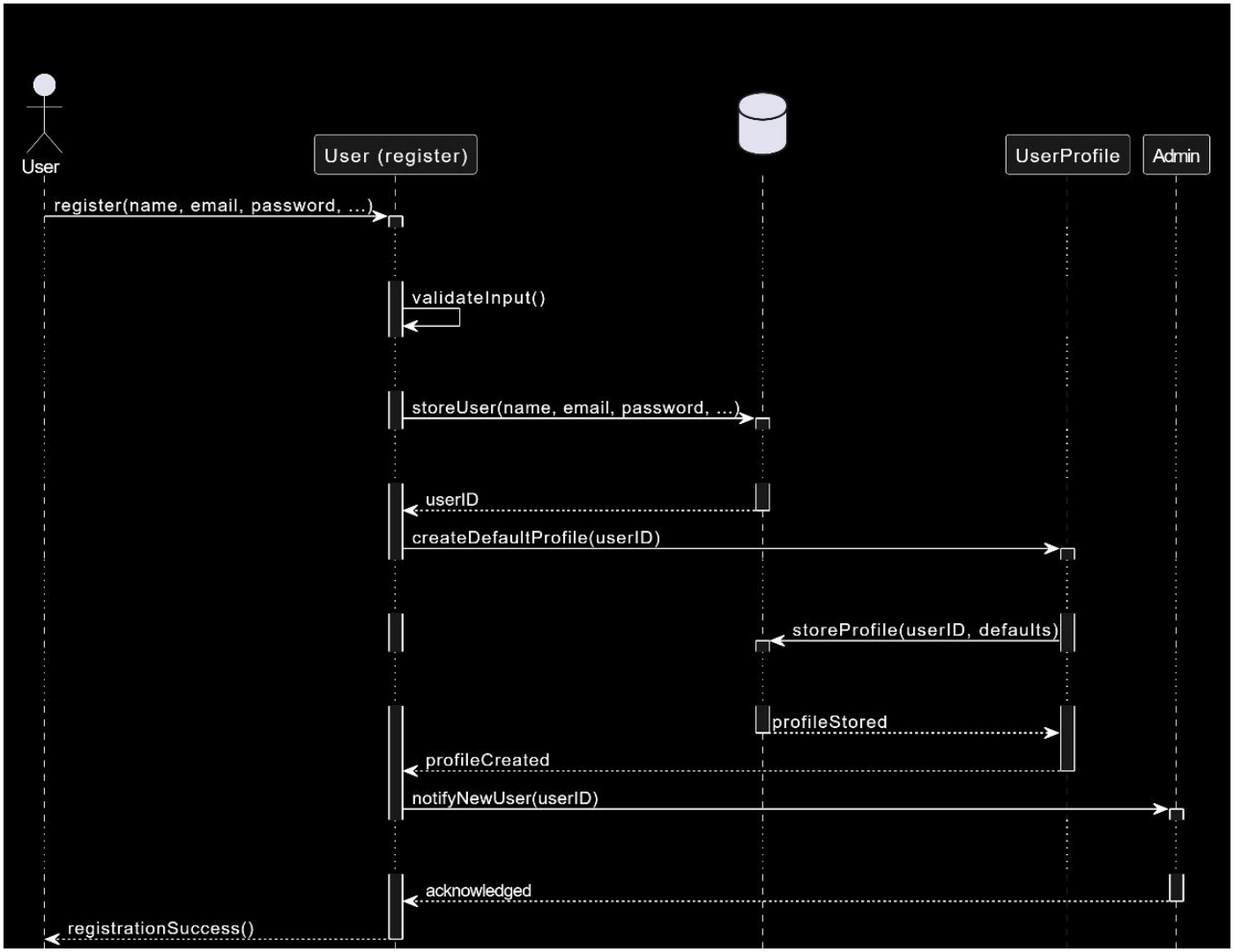


Figure 4.2.1: Sequence Diagram (User Registration)

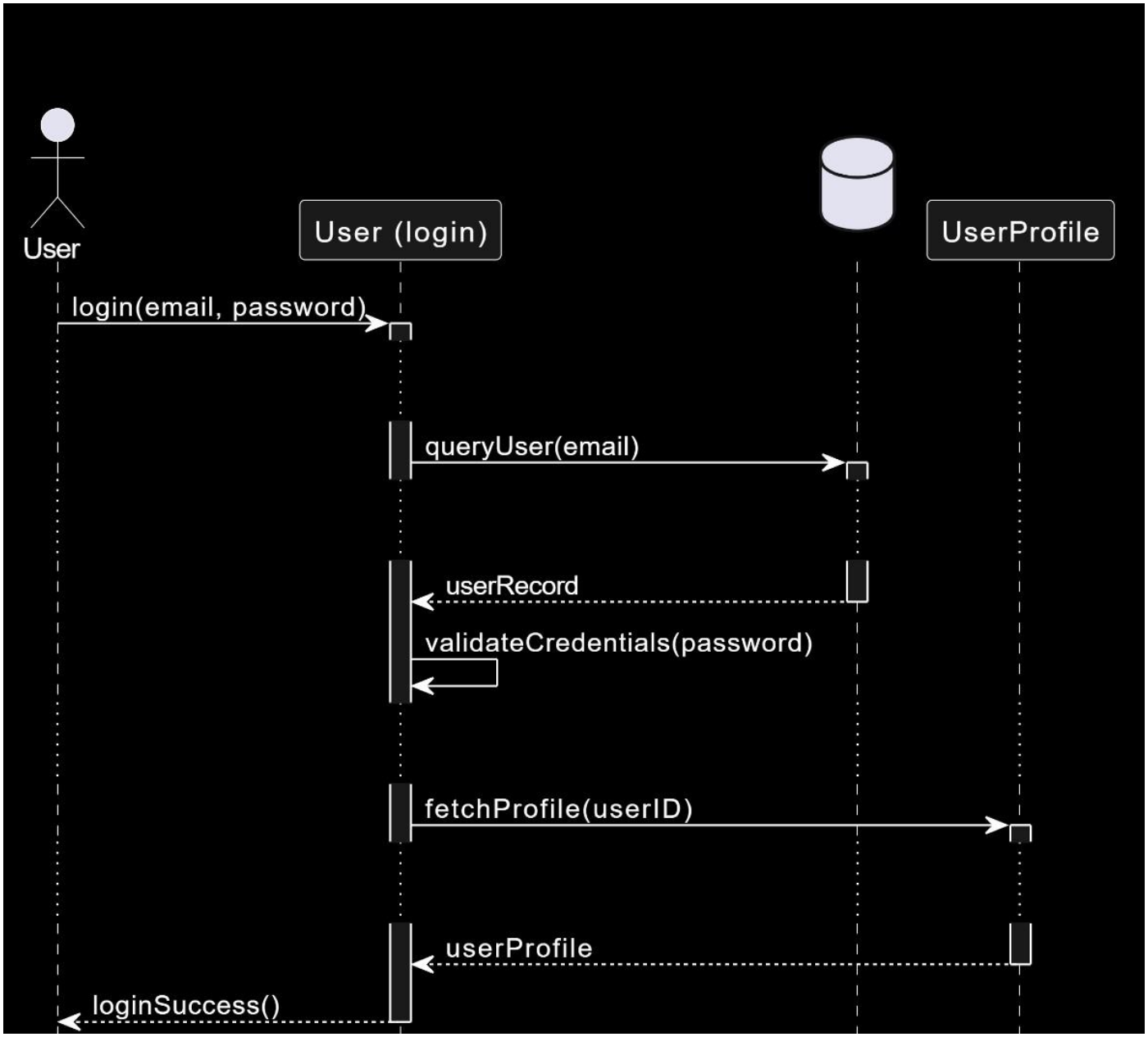


Figure 4.2.2: Sequence Diagram (user Login)

Figure 4.2.3: Sequence Diagram (Compatibility score Calculation)

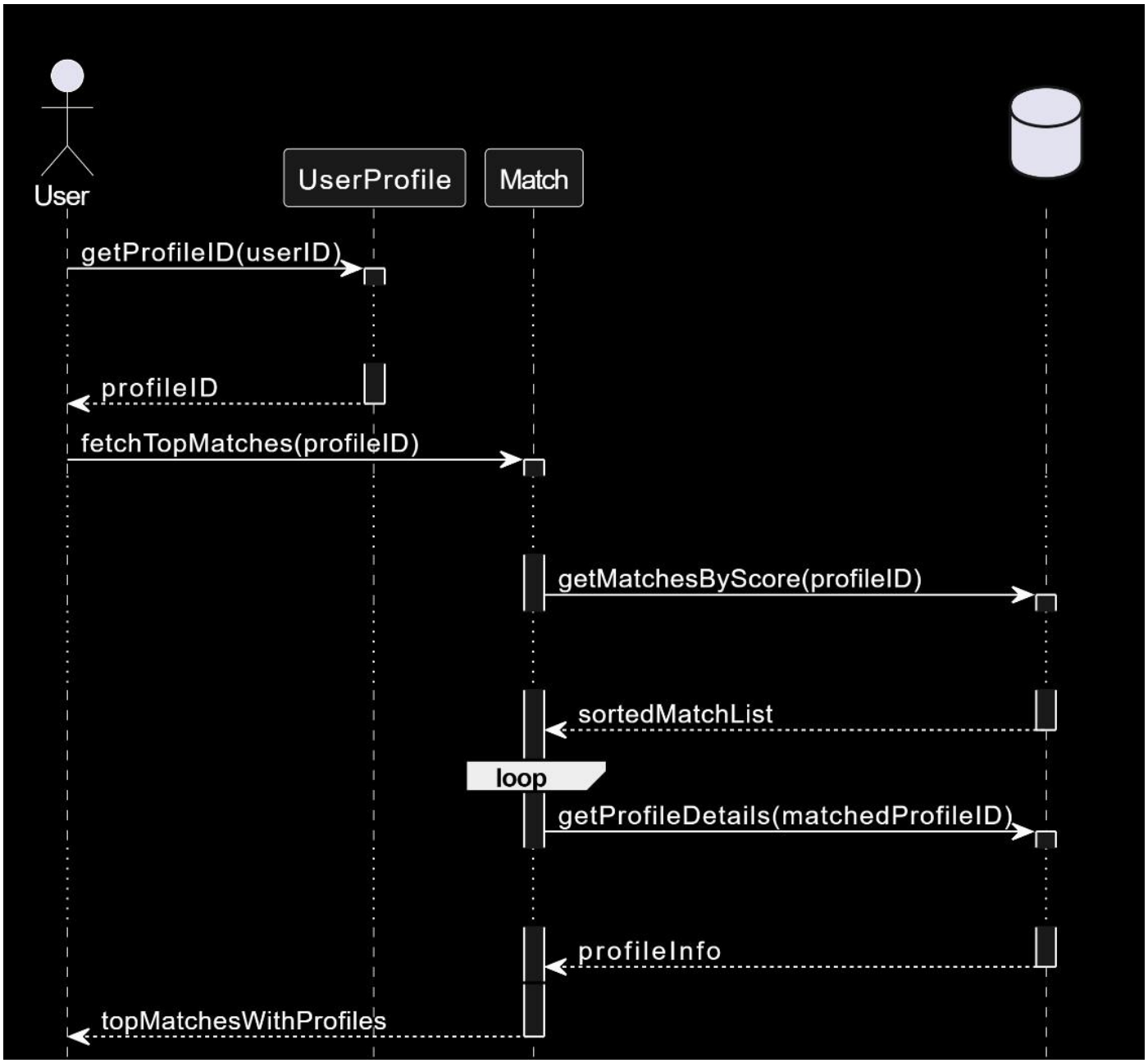


Figure 4.2.4: Sequence Diagram (Profile Matching Based on Scores**)**

Figure 4.2.5: Sequence Diagram (Post Property Listing)

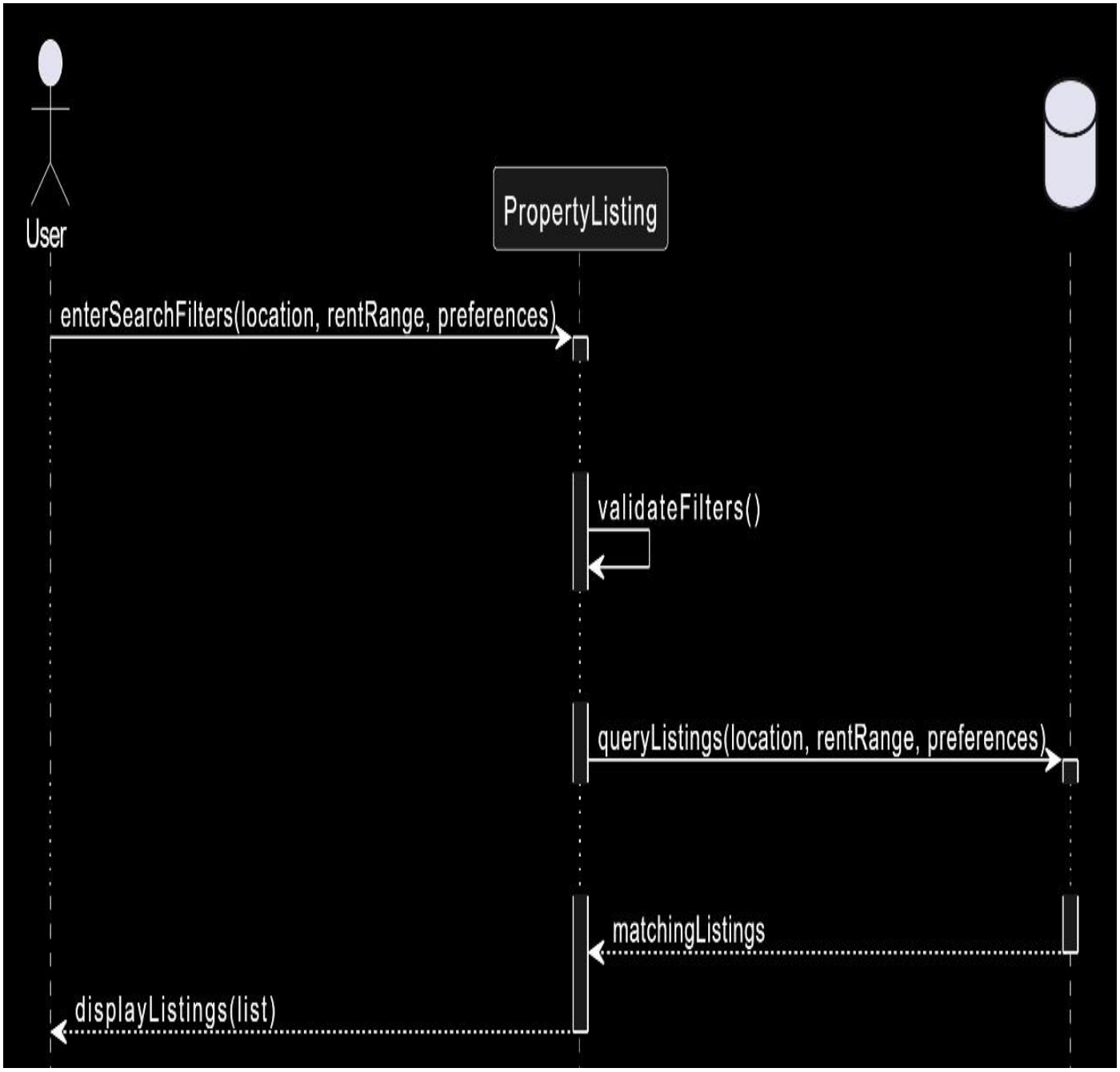


Figure 4.2.6: Sequence Diagram (Search Listings)

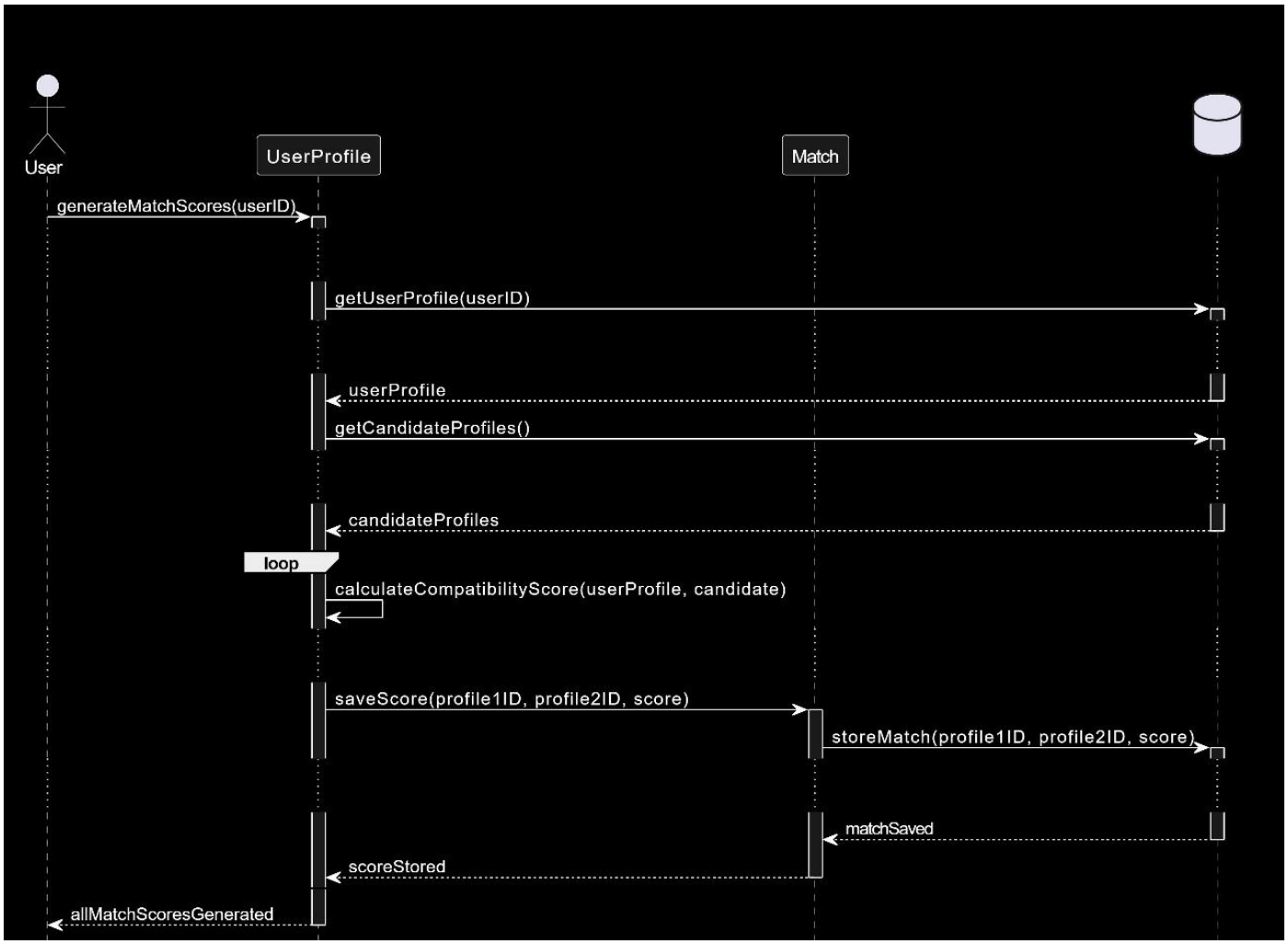


Figure 4.2.7: Sequence Diagram (Generate Match Scores)

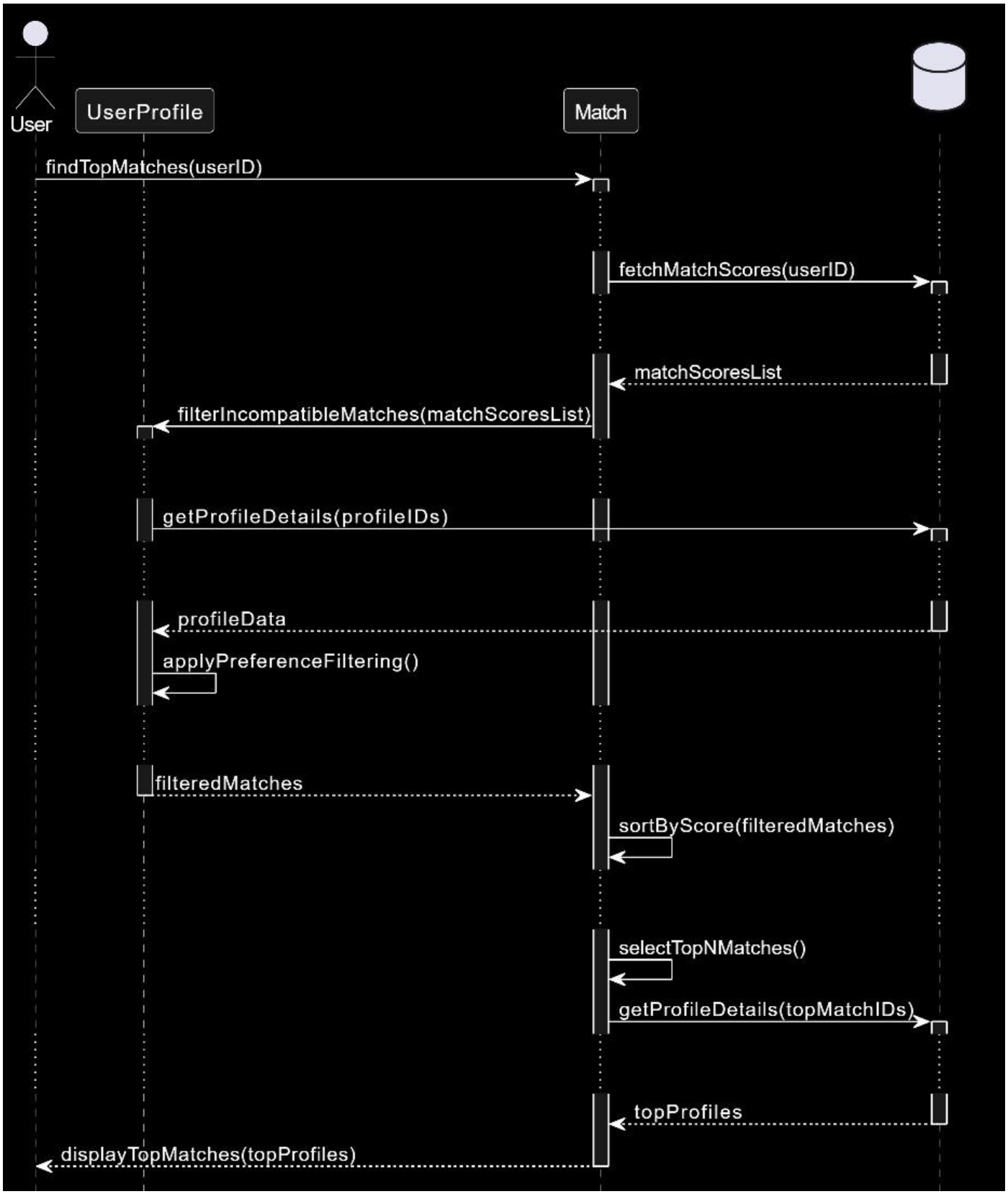


Figure 4.2.8: Sequence Diagram (Suggest Top Matches)

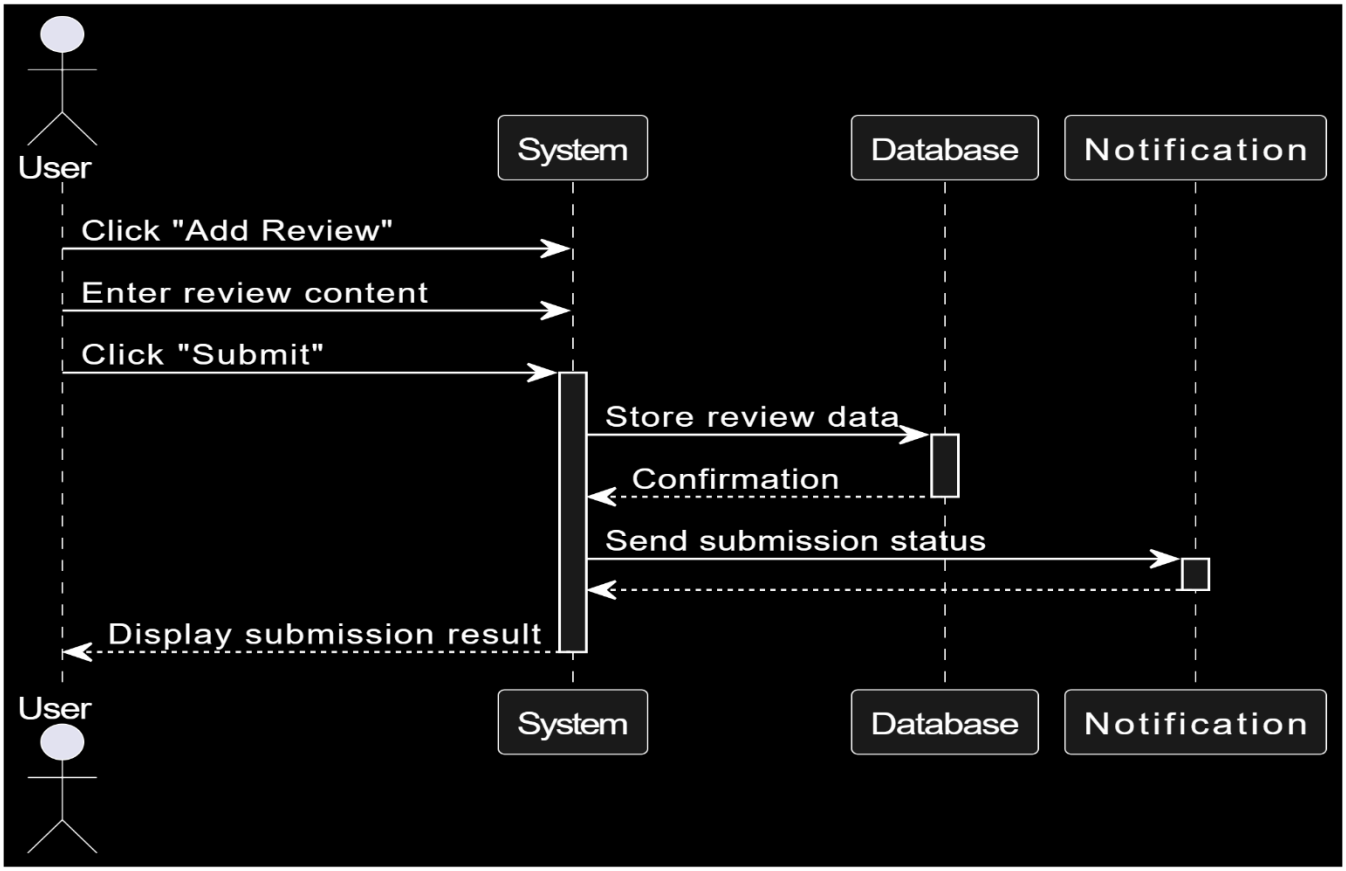


Figure 4.2.9: Sequence Diagram (Report Submission)

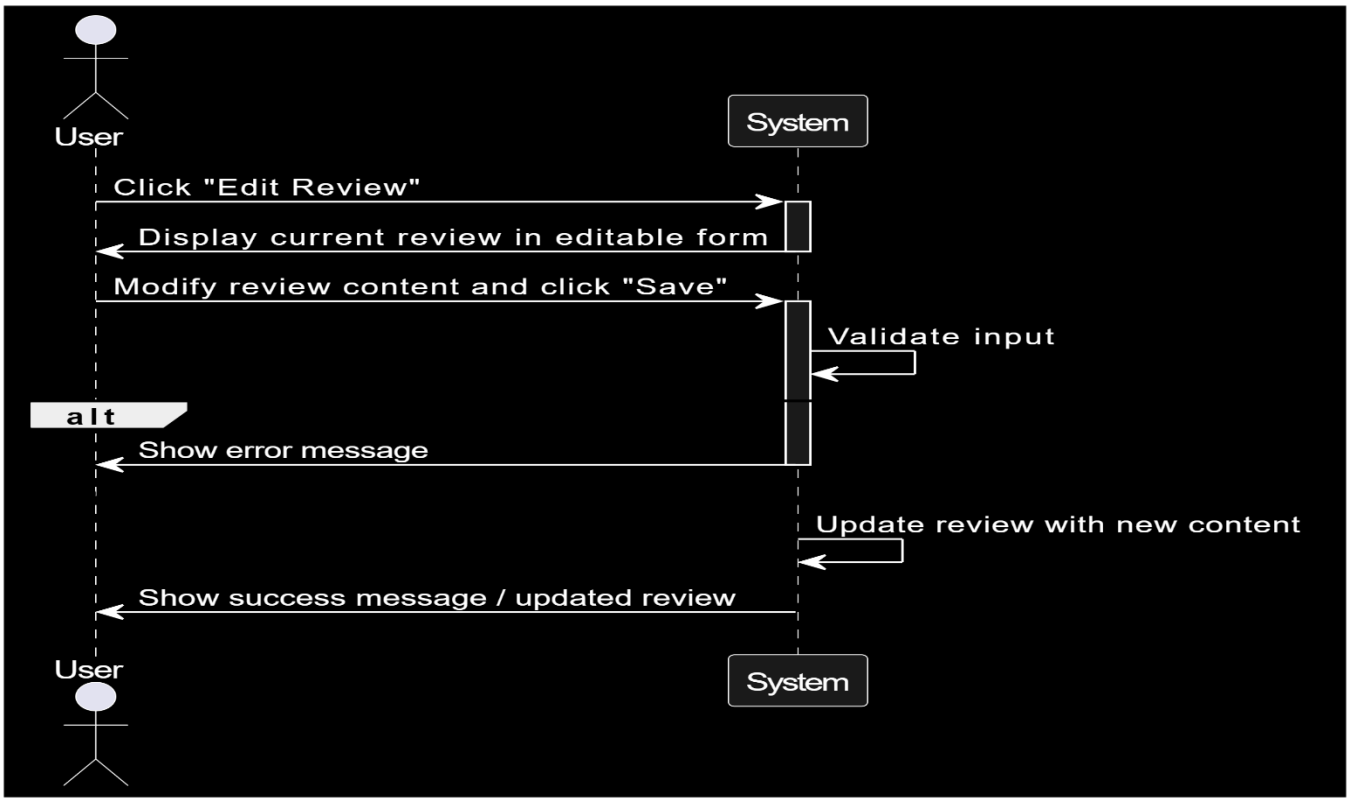


Figure 4.2.10: Sequence Diagram (Editing a Review)

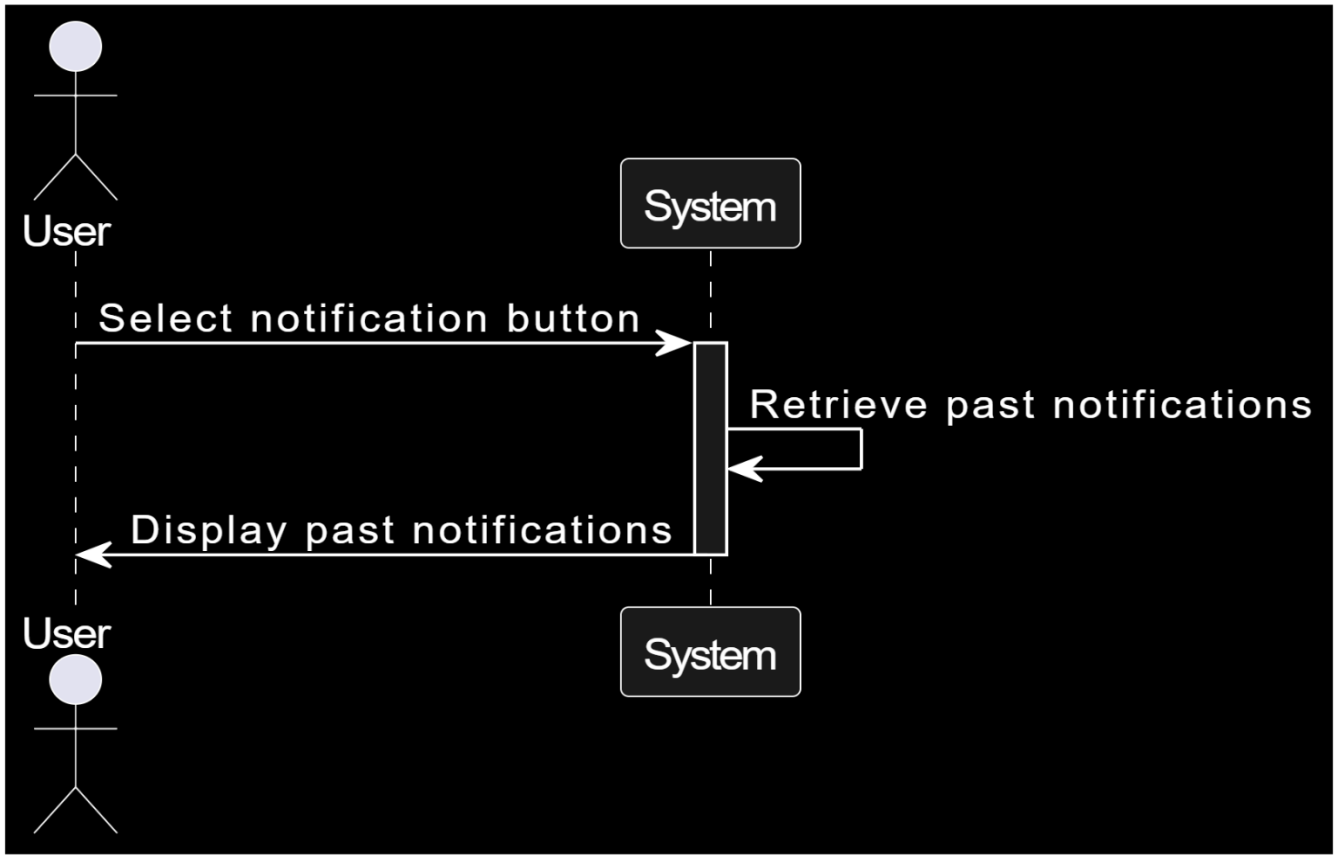


Figure 4.2.11: Sequence Diagram (View History)

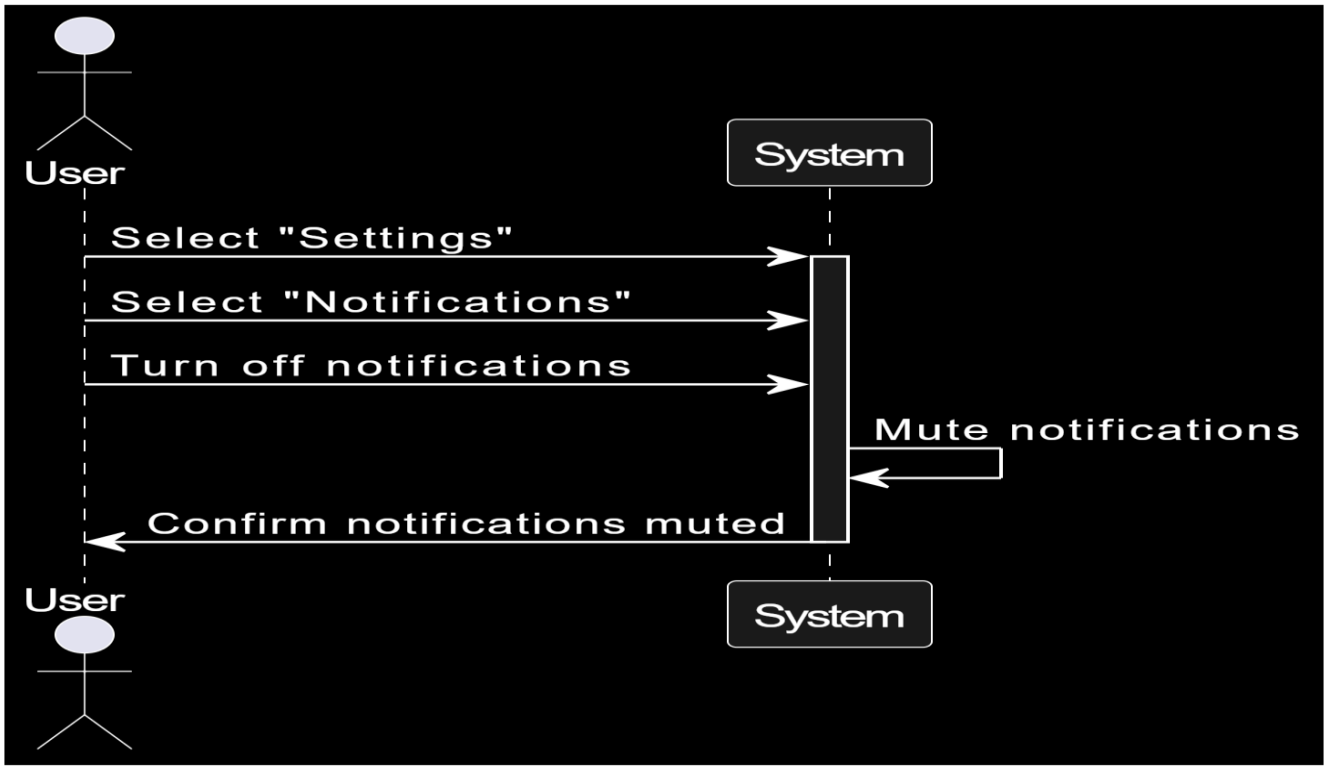


Figure 4.2.12: Sequence Diagram (Mute the Updates)

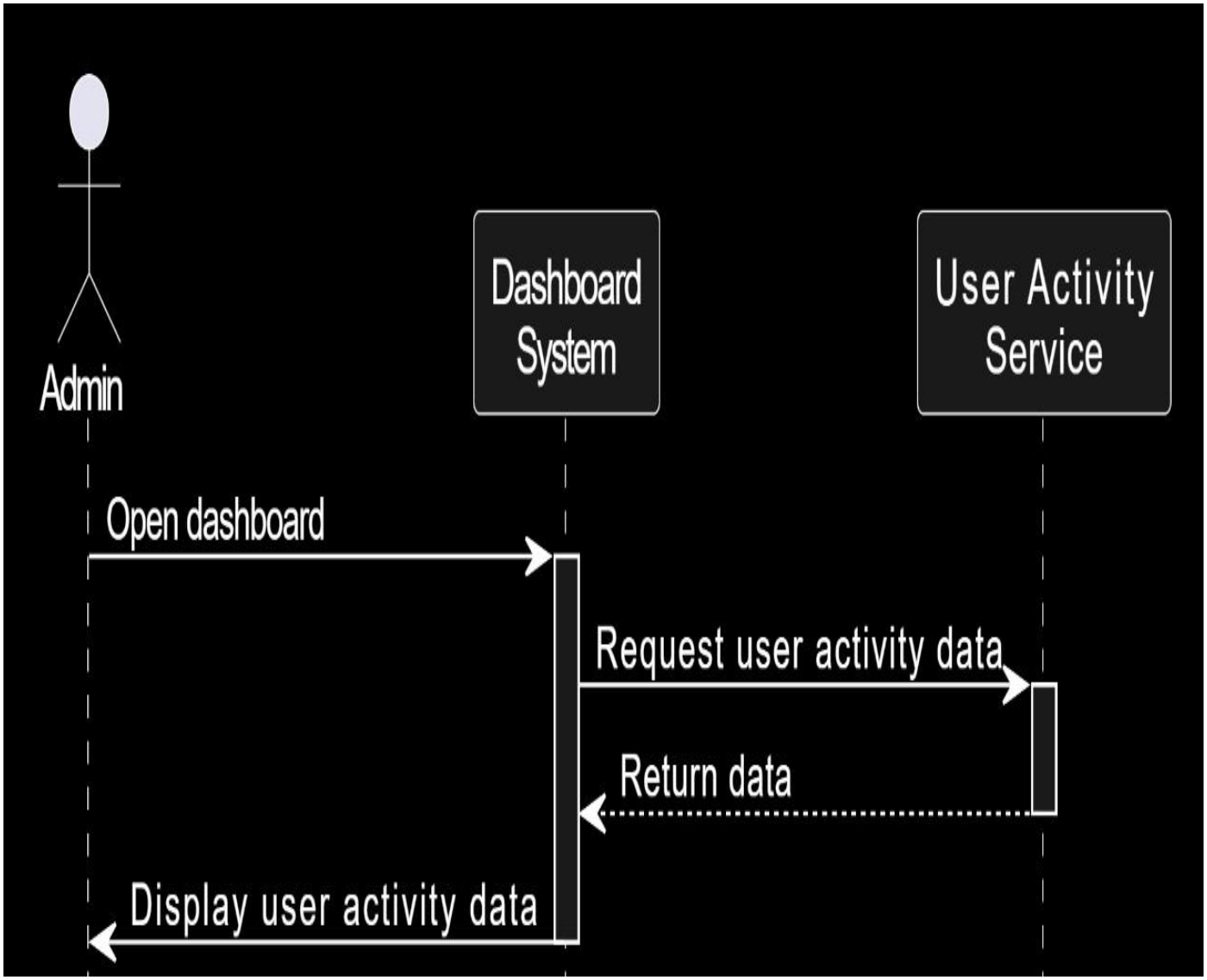


Figure 4.2.13: Sequence Diagram (Analyze User activities)

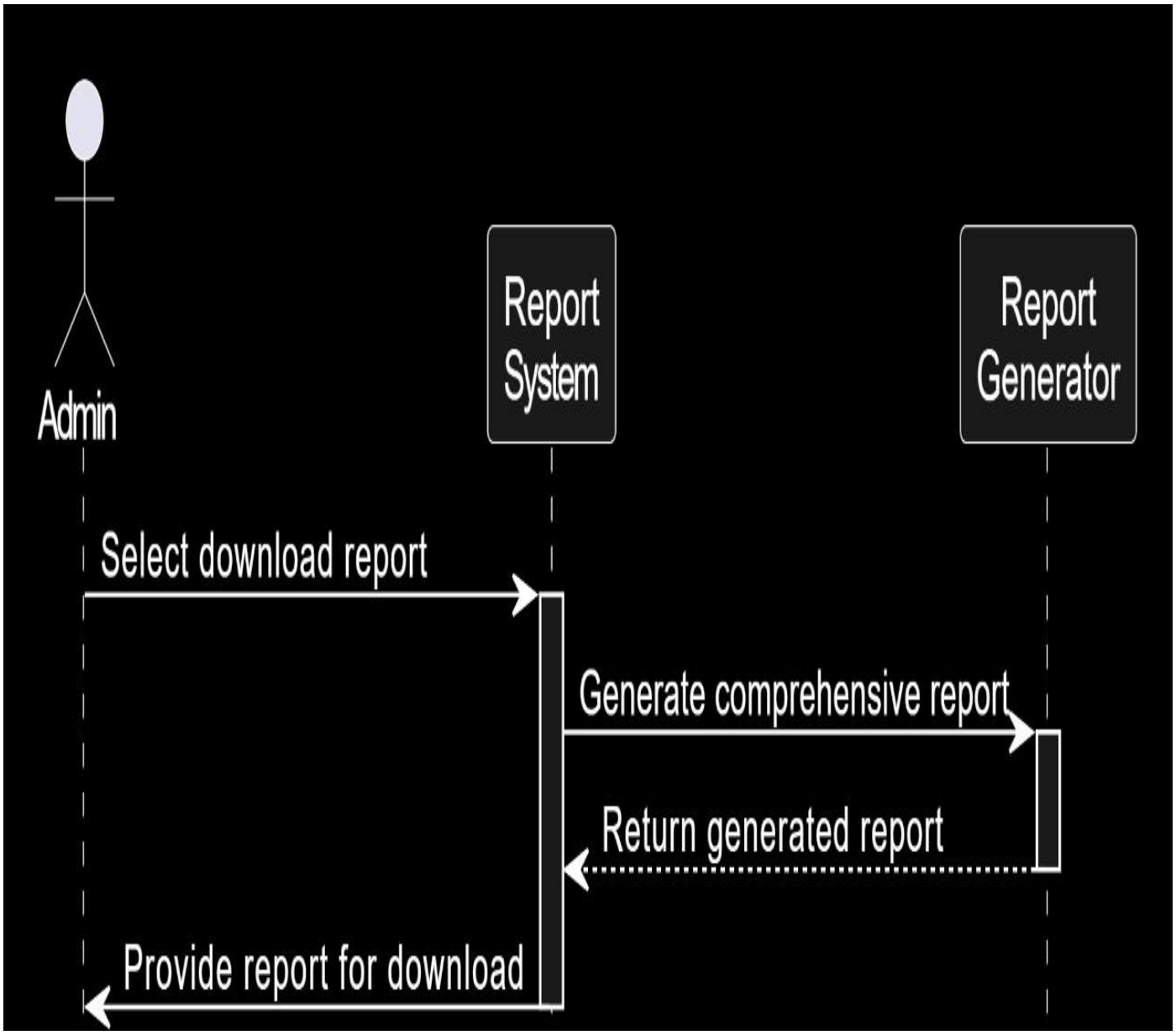


Figure 4.2.14: Sequence Diagram (Download Report)

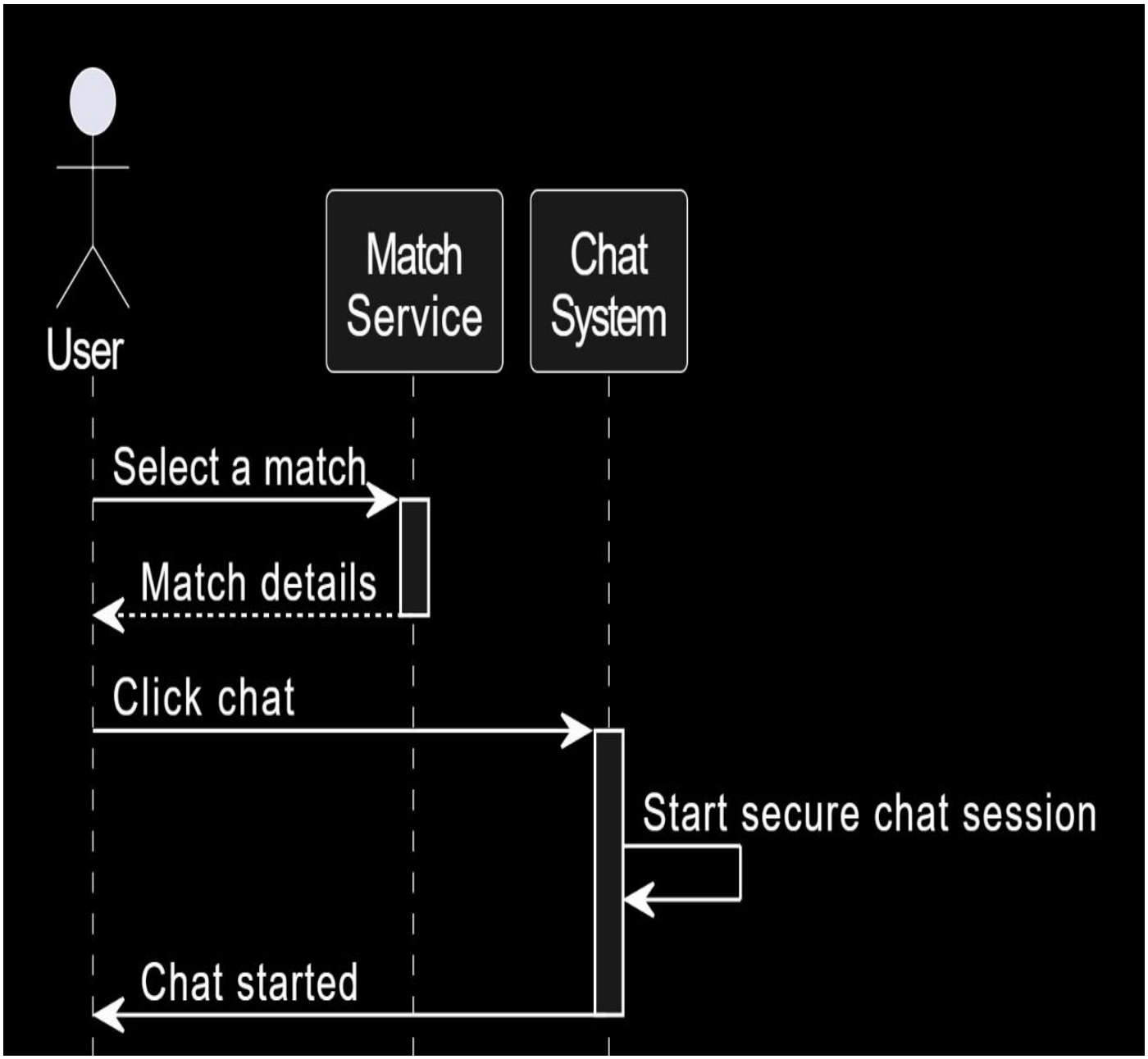


Figure 4.2.15: Sequence Diagram (Start a secure chat)

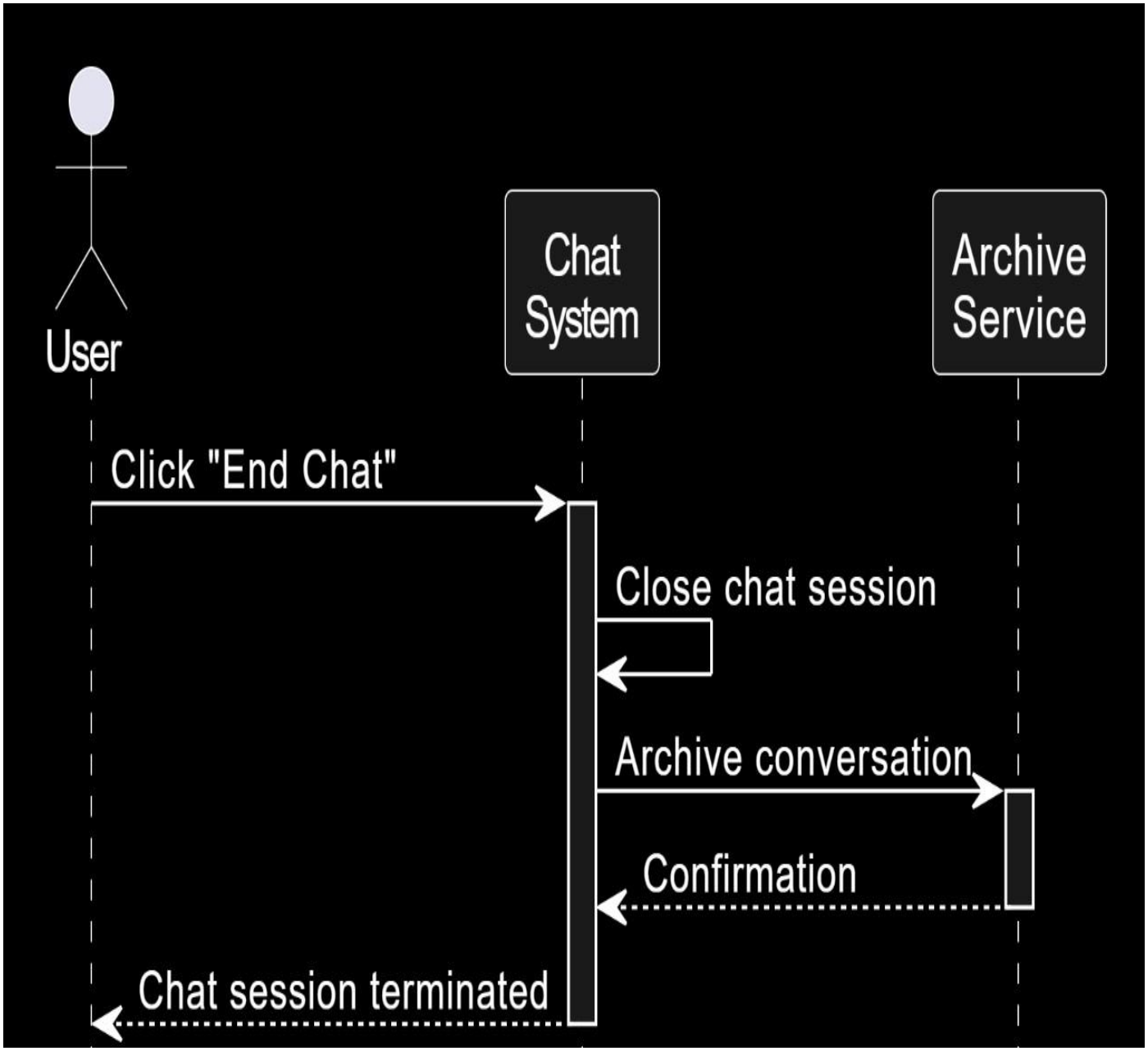


Figure 4.2.16: Sequence Diagram (Terminate Chat Session)

5.3 Class Diagram

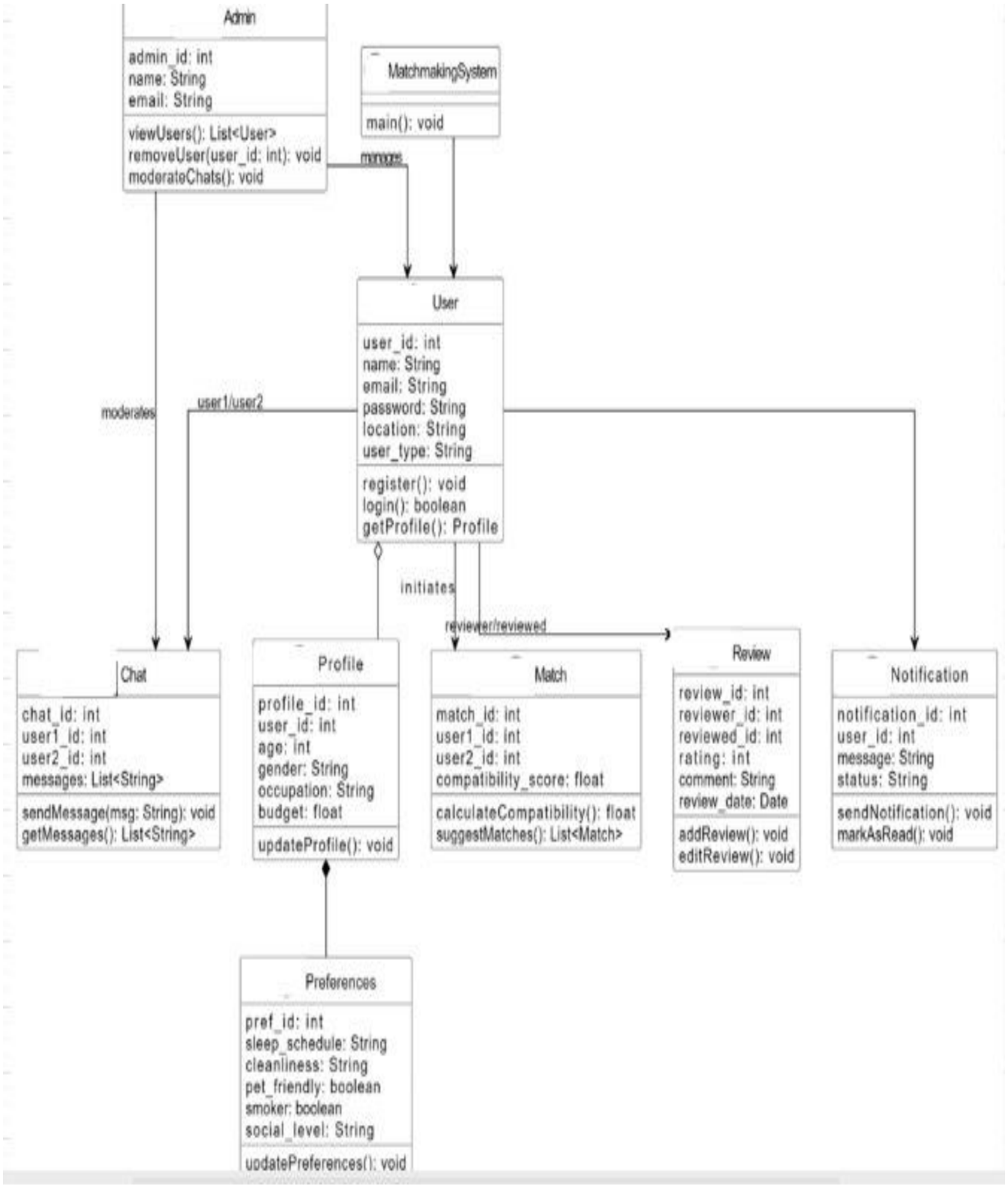


Figure 4.3.1: Class Diagram

6. Data design

Following is the Entity relation diagram:

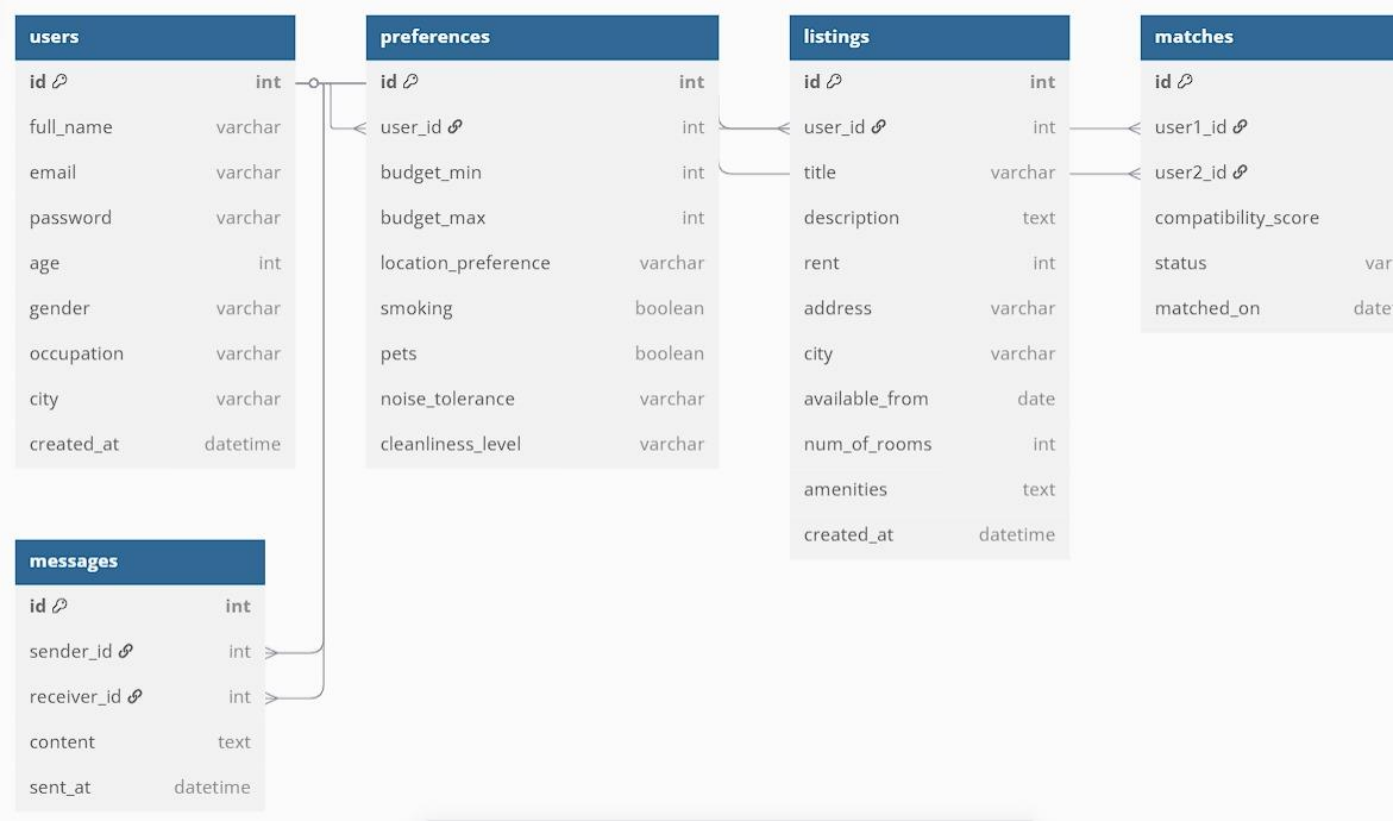


Figure 5.1: Entity Relationship Diagram, Logical Model

6.1 Data dictionary

Entity	Type	Description	Attributes/Methods/Parameters
User	Object	Represents a Registered user of a Roomie Find.	- UserID (Primary Key) – First Name – LastName - Email - PasswordHash - PhoneNumber - Gender - CreatedAt - DateofBirth - LastLogin
Profile	Object	Extended user profile with the roommate preferences	- ProfileID (Primary Key) – UserID(FK) - occupation - Budget - PreferredLocation - LifestylePreferences- Bio - ProfilePicture
Room	Object	Represents a room or apartment available for sharing.	- RoomID (Primary Key) - Title - Description- Address - City - Rent- AvailableFrom - Furnished - Amenities – PostedBy (FK) -

Room Image	Object	Stores images of the listed rooms.	-ImageID (Primary Key) - RoomID (FK)- ImageURL- UploadedAt-
Match	Object	Represents a match between two users based on preferences.	- MatchID (Primary Key) - User1ID (FK) - User2ID (FK) - MatchScore- Status - CreatedAt-
Message	Object	Stores chat messages exchanged between matched users.	- MessageID (Primary Key) -SenderID(FK) - ReceiverID (FK) -Content-Timestamp - IsRead
Review	Object	Review/Rating given by one user to another after stay experience.	- ReviewID (Primary Key) - ReviewerID(FK) - ReviewedUserID(FK) -Rating-Comment - CreatedAt
Report	Object	Complaints or reports about users or listings.	- ReportID (Primary Key) -ReporterID(FK) - ReportedUserID (FK, nullable) - ReportedRoomID (FK, nullable) -Reason-Status - CreatedAt
Notification	Object	In-app or push notifications sent to users.	- NotificationID (Primary Key) -UserID(FK)-Message- TypeIsRead - Timestamp
Subscription	Object	Stores premium feature access and subscription status.	- SubscriptionID (Primary Key) -UserID(FK)-PlanType-StartDate -EndDate - IsActive
Admin	object	Represents admin users who manage the platform.	-AdminID(Primary Key) -Name -Email-Role - CreatedAt
Feedback	Object	Feedback submitted by users about the platform.	- FeedbackID (Primary Key) - UserID(FK)-Message-Rating - SubmittedAt

7. Algorithm & Implementation

1. User Registration & Login

Registration:

- Check if the user's email already exists in the database
- If it does, show an error: "Email already registered"
- If it doesn't, hash the password for security
- Store the name, email, and hashed password in the database
- Show a message: "Registration successful"

Login:

- Search for the email in the database
- If the user is not found, return: "User not found"
- If found, hash the input password and compare it to the stored hashed password
- If they match, return: "Login successful"
- Otherwise, return: "Incorrect password"

2. Roommate Matching AI model

1. **Collect user data** (profile attributes)
2. **Preprocess**: Convert text to numbers, normalize values
3. **Train ML model** (KNN or Decision Tree) on past user match data
4. **Predict compatibility score** between users
5. **Sort users by score** and return top 5 matches

3. Search with Filters

- Loop through all users in the database
- Check if user matches each filter:
  - Gender (if specified)
  - Location (if specified)
  - Budget range (minimum to maximum)
- If all filters match, add the user to the filtered list
- Return the filtered list of potential roommates

4.Secure Chat (Encryption and Decryption)

Encrypting Messages:

1. Accept the plain message and secret key
2. Use an AES algorithm to create a cipher with the key
3. Encrypt the message using the cipher
4. Return the encrypted version

Decrypting Messages:

1. Accept the encrypted message and secret key
2. Use the same AES cipher with the key
3. Decrypt the message
4. Return the original plain message

5.Agreement Sharing

- Take agreement data from the sender
- Save agreement data along with sender and receiver IDs
- Notify the receiver about the agreement
- Wait for receiving confirmation
- Once confirmed, update the agreement status to "Confirmed"
- Notify the sender

6.Review and Rating

Submitting a Review:

1. Collect rating and comment from one user about another
2. Create a new review record
3. Save it in the database
4. Return a confirmation message

Calculating Average Rating:

1. Fetch all reviews for a specific user
2. Loop through them, add up all the ratings
3. Divide the total by the number of ratings
4. Return the average rating

7. Notification System

- **Sending Notifications:**
- Create a new notification with the user ID and message
- Mark it as "Unread"
- Store it in the database
- **Viewing Notifications:**
- Fetch all notifications where user ID matches and status is "Unread"
- Return the list

8. Admin Dashboard Analytics

1. Count total number of registered users
2. Count number of confirmed agreements (successful matches)
3. Count number of active chat sessions
4. Calculate average rating across users

5. Return this data in a report format

Pseudocode:

```
MODULE RoomieFindSystem

FUNCTION registerUser(name, email, password):
    IF emailExists(email):
        RETURN "Email already registered"
    ELSE:
        hashedPassword = hash(password)
        saveToDatabase(name, email, hashedPassword)
        RETURN "Registration successful"

FUNCTION loginUser(email, password):
    user = getUserByEmail(email)
    IF user == NULL:
        RETURN "User not found"
    ELSE:
        IF hash(password) == user.hashedPassword:
            RETURN "Login successful"
        ELSE:
            RETURN "Incorrect password"

FUNCTION matchCompatibleUsers(currentUserProfile):
    allUsers = getAllUsers()
    preprocessedUsers = preprocess(allUsers + currentUserProfile)
    model = loadTrainedModel()
    scores = []

    FOR user IN preprocessedUsers:
        IF user != currentUserProfile:
            score = model.predictCompatibility(currentUserProfile, user)
            scores.append((user, score))

    sortedMatches = sortDescending(scores)
    RETURN topN(sortedMatches, 5)
```

.

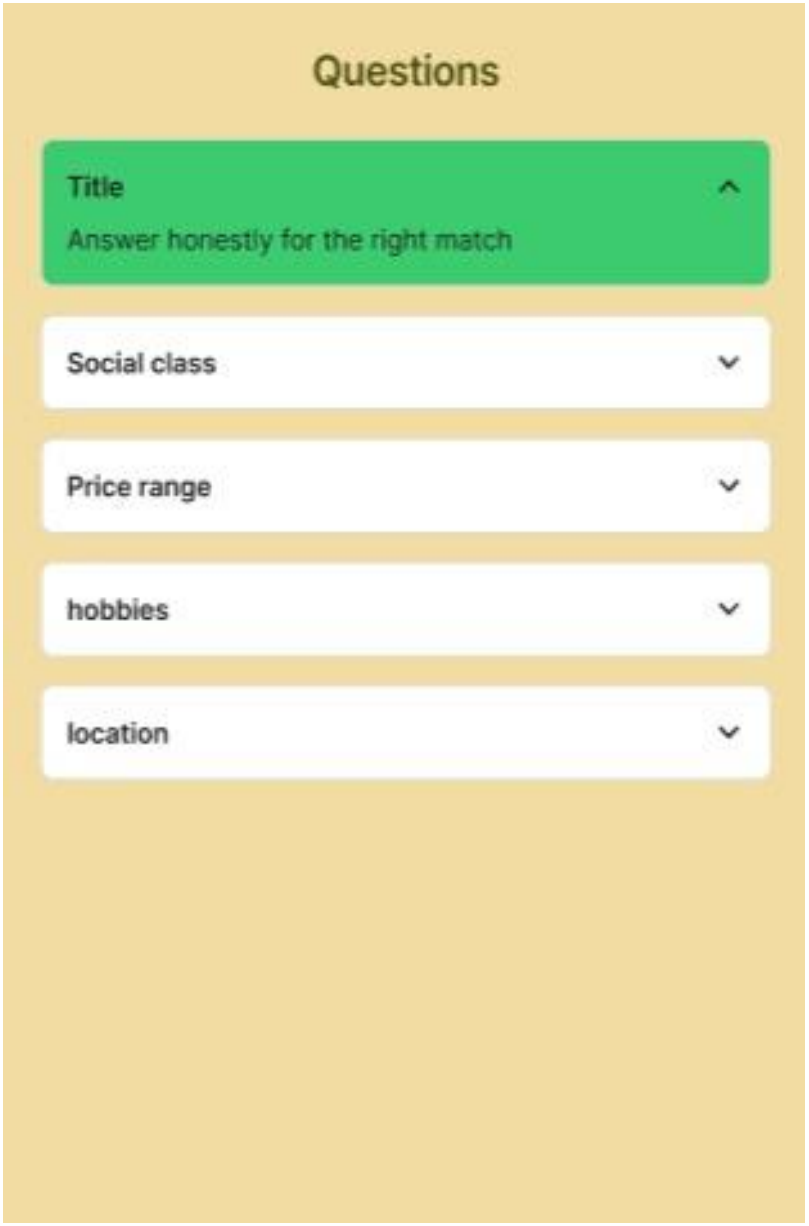
```
FUNCTION searchUsers(filters):
    matchedUsers = []
    FOR user IN getAllUsers():
        IF matchFilters(user, filters):
            matchedUsers.append(user)
    RETURN matchedUsers

FUNCTION encryptMessage(message, key):
    cipher = createCipher(key)
    encrypted = cipher.encrypt(message)
    RETURN encrypted

FUNCTION decryptMessage(encryptedMessage, key):
    cipher = createCipher(key)
    decrypted = cipher.decrypt(encryptedMessage)
    RETURN decrypted

FUNCTION shareAgreement(senderID, receiverID, agreementData):
    saveAgreement(senderID, receiverID, agreementData)
    notifyUser(receiverID, "New agreement received")
    WAIT for confirmation from receiver
    updateAgreementStatus(agreementID, "Confirmed")
    notifyUser(senderID, "Agreement confirmed")

FUNCTION submitReview(fromUserID, toUserID, rating, comment):
    review = createReview(fromUserID, toUserID, rating, comment)
    saveReview(review)
    RETURN "Review submitted"

FUNCTION calculateAverageRating(userID):
    reviews = getReviews(userID)
    total = 0
```

```
    reviews = getReviews(userID)
    total = 0
    count = length(reviews)
    FOR review IN reviews:
        total += review.rating
    IF count > 0:
        RETURN total / count
    ELSE:
        RETURN "No ratings"

FUNCTION sendNotification(userID, message):
    notification = createNotification(userID, message, "Unread")
    saveNotification(notification)

FUNCTION viewNotifications(userID):
    notifications = getUnreadNotifications(userID)
    RETURN notifications

FUNCTION getAdminDashboardStats():
    totalUsers = countUsers()
    totalAgreements = countConfirmedAgreements()
    activeChats = countActiveChats()
    avgRating = calculateGlobalAverageRating()
    RETURN {
        "users": totalUsers,
        "agreements": totalAgreements,
        "chats": activeChats,
        "averageRating": avgRating
    }

END MODULE
```

8. Human Interface Design :

Below are the mockups that represent the system and give us a rough idea of the major components of the software 'ROOMIE FIND'.

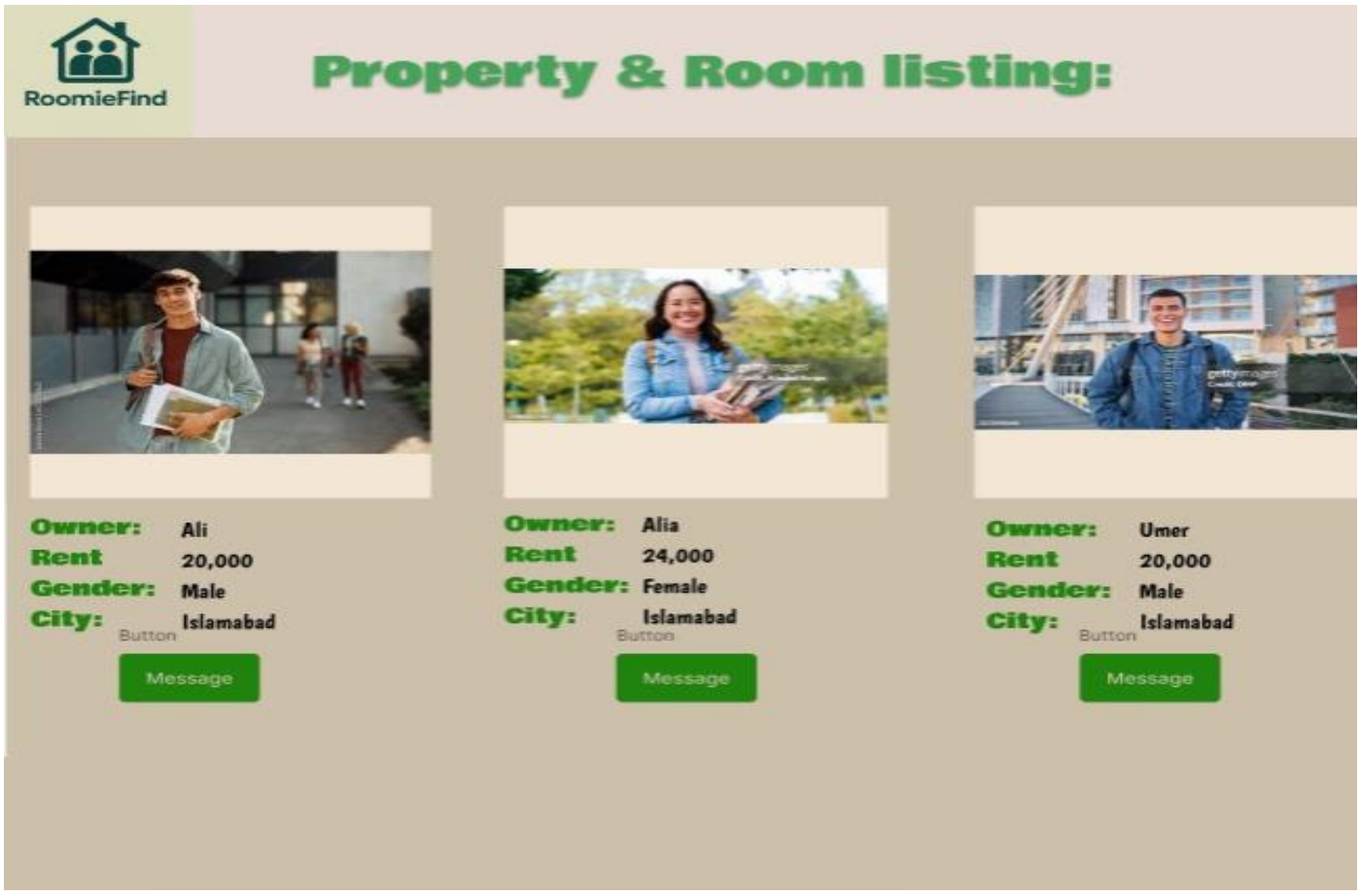
User interface:



Compatibility Quiz

Matchmaking

Review and rating



Roommate listing

Chat Module

7.1.Screen objects and actions

1. **User Dashboard:**
    a. Objects: Profile summary, Roommate matches, Notifications, Quick actions
    b. Actions: View match suggestions, update profile, navigate to chat or listings, review notifications
2. **Room Listing Screen:**
    a. Objects: Room cards, Filter options, Sort options, Map view
    b. Actions: Browse rooms, filter by budget/location, sort by latest or price, view rooms on map.
3. **User Profile Setup Screen:**
    a. Objects: Personal details form, Preference sliders, Upload profile picture
    b. Actions: Enter personal info, set preferences (budget, habits, lifestyle), upload images.
4. **Room Details Screen:**
    a. Objects: Image carousel, Room description, Amenities list, Contact/Apply buttons
    b. Actions: View room details, check amenities, contact lister, send application/request.
5. **Room Posting Screen:**
    a. Objects: Room form fields, Image upload buttons, Rent and availability settings
    b. Actions: Post a new room listing, upload images, enter availability and pricing.
6. **Matchmaking Results Screen:**
    a. Objects: Match cards, Match percentage, View profile buttons
    b. Actions: Review potential roommate matches, open user profiles, send connection requests.
7. **Chat/Message Center:**
    a. Objects: Chat list, Message box, User info panel, Media sharing option

b. Actions: Send messages, view chat history, share media, check roommate info.

8. **Search and Filter Screen:**
    a. Objects: Search bar, Filter toggles (budget, gender, location), Reset button
    b. Actions: Perform room/roommate search, apply filters, clear search.
9. **Appointment Scheduling Screen (Room Viewing):**
    a. Objects: Calendar, Available time slots, Schedule button
    b. Actions: Book a visit for a room, select date/time, confirm appointment.
10. **User Settings Screen:**
    a. Objects: Profile settings, Notification preferences, Privacy controls
    b. Actions: Update personal settings, manage app notifications, control profile visibility.
11. **Review & Rating Screen:**
    a. Objects: Rating stars, Comment box, Submit button
    b. Actions: Rate previous roommates or hosts, leave a review.
12. **Notifications Center:**
    a. Objects: Notification list, Mark as read, Delete options
    b. Actions: View new updates, mark notifications as read, delete unwanted ones.
13. **Payment & Subscription Screen:**
    a. Objects: Subscription plans, Payment methods, Billing history
    b. Actions: Choose/upgrade plan, enter payment info, view past invoices.
14. **Report & Feedback Screen:**
    a. Objects: Report form, Feedback input, Category selector
    b. Actions: Submit feedback on listings/users, report inappropriate behavior.
15. **Admin Dashboard (for platform managers):**
    a. Objects: User statistics, Flagged content, Admin tools
    b. Actions: Review system usage, handle reports, manage platform content.
16. **Roommate Compatibility Questionnaire:**
    a. Objects: Multiple-choice questions, Sliders, Save responses button
    b. Actions: Fill compatibility quiz, save preferences for better matching.
17. **Favorites Screen:**
    a. Objects: Saved room/roommate cards, Remove from favorites
    b. Actions: View bookmarked rooms or profiles, remove from list.
18. **Location-based Search Screen:**
    a. Objects: Interactive map, Nearby filters, Pin drop info cards
    b. Actions: Search rooms or roommates near a location, tap pins to view details.
19. **Shared Living Agreement Screen:**
    a. Objects: Agreement form, Rules checklist, Digital signature fields
    b. Actions: Draft/shared agreements, set living rules, sign digitally.

9. Conclusion

Roomie Find is a smart platform for rental matching designed to simplify the problem of compatible roommates and suitable accommodation. By utilizing different filters, a user can easily find its compatible roommates based on their location behavior and habits. Throughout the project, we followed structured development methodology, applied Object-Oriented Design (OOD) principles, and used modern tools like Figma to ensure user-friendly, efficient and scalable solutions. This project also gives us the experience to explore key software engineering concepts including APIs, UI/UX design, matchmaking algorithms, and database integration. So overall Roomie Find stands as practically and socially valuable application.

10. References:

1.MongoDB, Inc. “MongoDB Documentation.” Internet:
https://www.mongodb.com/docs/.
2.:Express JS Team. “Express – Node.js Web Application Framework. “Internet:
https://expressjs.com/, 2024
3. Figma Inc. “Figma: The Collaborative Interface Design Tool.” Internet:
https://www.figma.com/, 2024
4.K. Tandon. “Roommate Matching Algorithm.” Internet:
https://medium.com/@kunaltandon.official/roommate-matching-algorithm-e9dc7495cf19, Oct. 12, 2022.

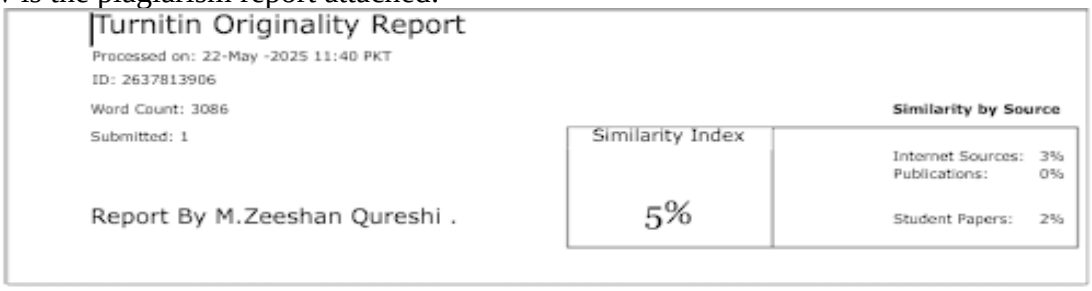
BOOKS:

5 .K. Schwalbe. *Information Technology Project Management*. Boston, MA: CengageLearning,2015, pp. 45–67**.**
Covers scope planning, WBS, and stakeholders’ analysis in detail.

6. C. W. Dawson. *Projects in Computing and Information Systems: A Student’s Guide*. Harlow, UK: Pearson Education, 2015, pp. 91–130.
   Excellent for beginners learning how to define scope, goals, and deliverables.

11. Plaragism Report

Below is the plagiarism report attached:

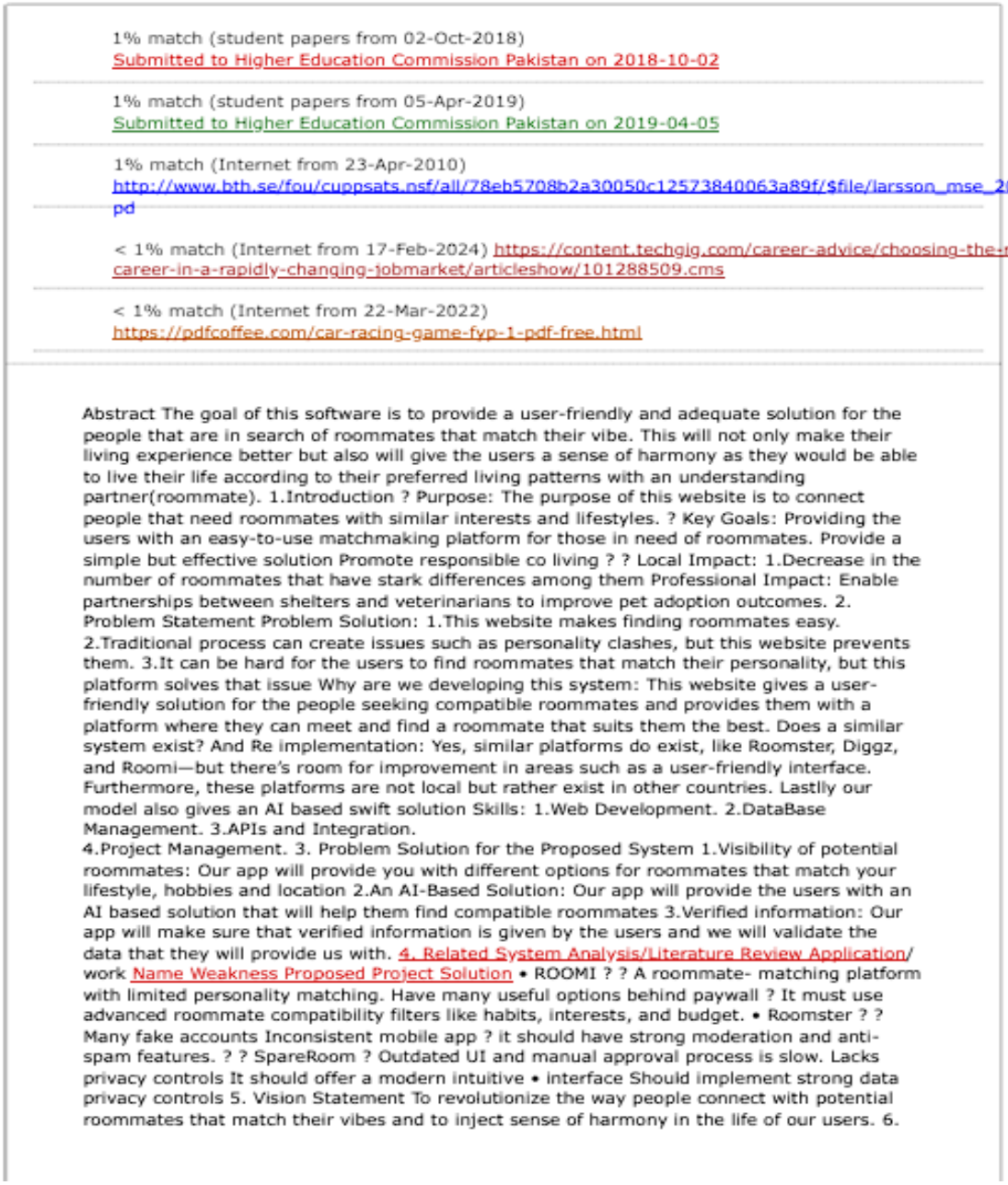


Turnitin Originality Report

Processed on: 22-May -2025 11:40 PKT
ID: 2637813906
Word Count: 3086
Submitted: 1

Report By M.Zeeshan Qureshi .

Similarity Index	Similarity by Source
5%	Internet Sources: 3% Publications: 0% Student Papers: 2%



1% match (student papers from 02-Oct-2018)
Submitted to Higher Education Commission Pakistan on 2018-10-02

1% match (student papers from 05-Apr-2019)
Submitted to Higher Education Commission Pakistan on 2019-04-05

1% match (Internet from 23-Apr-2010)
http://www.bth.se/fou/cuppsats.nsf/all/78eb5708b2a30050c12573840063a89f/$file/larsson_mse_2007_23pd

< 1% match (Internet from 17-Feb-2024) https://content.techgig.com/career-advice/choosing-the-right-career-in-a-rapidly-changing-jobmarket/articleshow/101288509.cms

< 1% match (Internet from 22-Mar-2022)
https://pdfcoffee.com/car-racing-game-fyp-1-pdf-free.html

Abstract The goal of this software is to provide a user-friendly and adequate solution for the people that are in search of roommates that match their vibe. This will not only make their living experience better but also will give the users a sense of harmony as they would be able to live their life according to their preferred living patterns with an understanding partner(roommate). 1.Introduction ? Purpose: The purpose of this website is to connect people that need roommates with similar interests and lifestyles. ? Key Goals: Providing the users with an easy-to-use matchmaking platform for those in need of roommates. Provide a simple but effective solution Promote responsible co living ? ? Local Impact: 1.Decrease in the number of roommates that have stark differences among them Professional Impact: Enable partnerships between shelters and veterinarians to improve pet adoption outcomes. 2. Problem Statement Problem Solution: 1.This website makes finding roommates easy. 2.Traditional process can create issues such as personality clashes, but this website prevents them. 3.It can be hard for the users to find roommates that match their personality, but this platform solves that issue Why are we developing this system: This website gives a user-friendly solution for the people seeking compatible roommates and provides them with a platform where they can meet and find a roommate that suits them the best. Does a similar system exist? And Re implementation: Yes, similar platforms do exist, like Roomster, Diggz, and Roomi—but there's room for improvement in areas such as a user-friendly interface. Furthermore, these platforms are not local but rather exist in other countries. Lastlly our model also gives an AI based swift solution Skills: 1.Web Development. 2.DataBase Management. 3.APIs and Integration.
4.Project Management. 3. Problem Solution for the Proposed System 1.Visibility of potential roommates: Our app will provide you with different options for roommates that match your lifestyle, hobbies and location 2.An AI-Based Solution: Our app will provide the users with an AI based solution that will help them find compatible roommates 3.Verified information: Our app will make sure that verified information is given by the users and we will validate the data that they will provide us with. 4. Related System Analysis/Literature Review Application/ work Name Weakness Proposed Project Solution • ROOMI ? ? A roommate- matching platform with limited personality matching. Have many useful options behind paywall ? It must use advanced roommate compatibility filters like habits, interests, and budget. • Roomster ? ? Many fake accounts Inconsistent mobile app ? it should have strong moderation and anti-spam features. ? ? SpareRoom ? Outdated UI and manual approval process is slow. Lacks privacy controls It should offer a modern intuitive • interface Should implement strong data privacy controls 5. Vision Statement To revolutionize the way people connect with potential roommates that match their vibes and to inject sense of harmony in the life of our users. 6.