1-Functional requirements:

1- The system has to allow users to create an account with details name, password, email and location (country, city).
2- The system must ask the user to login to use all features.
3- - The system enables user to know accurate prayer times according to his location(country, city) known when user is logging.
4- The system provides library includes Quran audio with different sheikhs.
5- The system must notify the user when it’s time for pray.
6- The has to system recommends halal restaurants.
7- The system provides nearby mosques according to user location.
8- The system must provide Qibla compass according to user location.
9- The system must provide Islamic Hijri calendar.
10- The system includes zakat calculator.
11- The system provides daily inspiration content.
12- The admin manages daily inspiration content.
13- The system provides Daily Duaa.
14- The admin manages daily duaa.
15- The system provides fasting tracker.
16- The system provides prayer tracker.
17- The system displays popular versus as daily routine.
18- The admin manages popular versus.
19- The system must have tasbih way for worship.
20- The system should store the number of finished khatma within a month.
21- The system should include a Todo list to obligate the user doing good things.
22- The system must memorize the user about his progress in Quran.
23- The system must notify the user to remember the prophet.
24- The system must stores all user progress in the system.
25- The system recommends Islamic book for the user.
26- The system provides Hajj journey.
27- The system should include greeting messages to introduce our Islam.
28- The system shall include Quran tafseer.
29- The system should provide way to donate for organizations which can help Muslims around the world.
30- The system should include shahada.
31- The system must include all asma Al-Husna.
32- The system must provide way to watch makka live stream.
33- The system must include hajj-umrah guide and its resources.
34- The system must include morning and evening prayers.
35- The system must provide way for the user to ask questions.
36- The system must include way to make meetings to help users in their prayers.
37- The system must provide resource to watch islamic videos.
38- The system must guide the user if he want to reserve ticket for saudiArabia.
39- The system must include way to read holy quran.

2- Non functional requirements:

1- performance

- the system must respond to each user request in at most 3 seconds. The application doesn’t depend on the internet except on some operations.
- The system should be flexible.

2- Serviceability

- The errors in the system could be handle easily according to database dimensional model and APIs used in the system

3- Data integrity

- The system contains accuracy data that can’t be change because it’s knowledgeable data about Islam and has data can be change according to user information.

4- Interface

- The system should provide a friendly interface that serve all user ages starting from 7 years old.

5- look and feel

- The system shall be painted in comfortable colors for the eyes.

6- security

- The system should check user authorization before access any system service.

- every user has his own information and his own progress data.

7- usability

- The system shall have a user-friendly interface.

-The system shall be easy to use by different users.

8- Availability

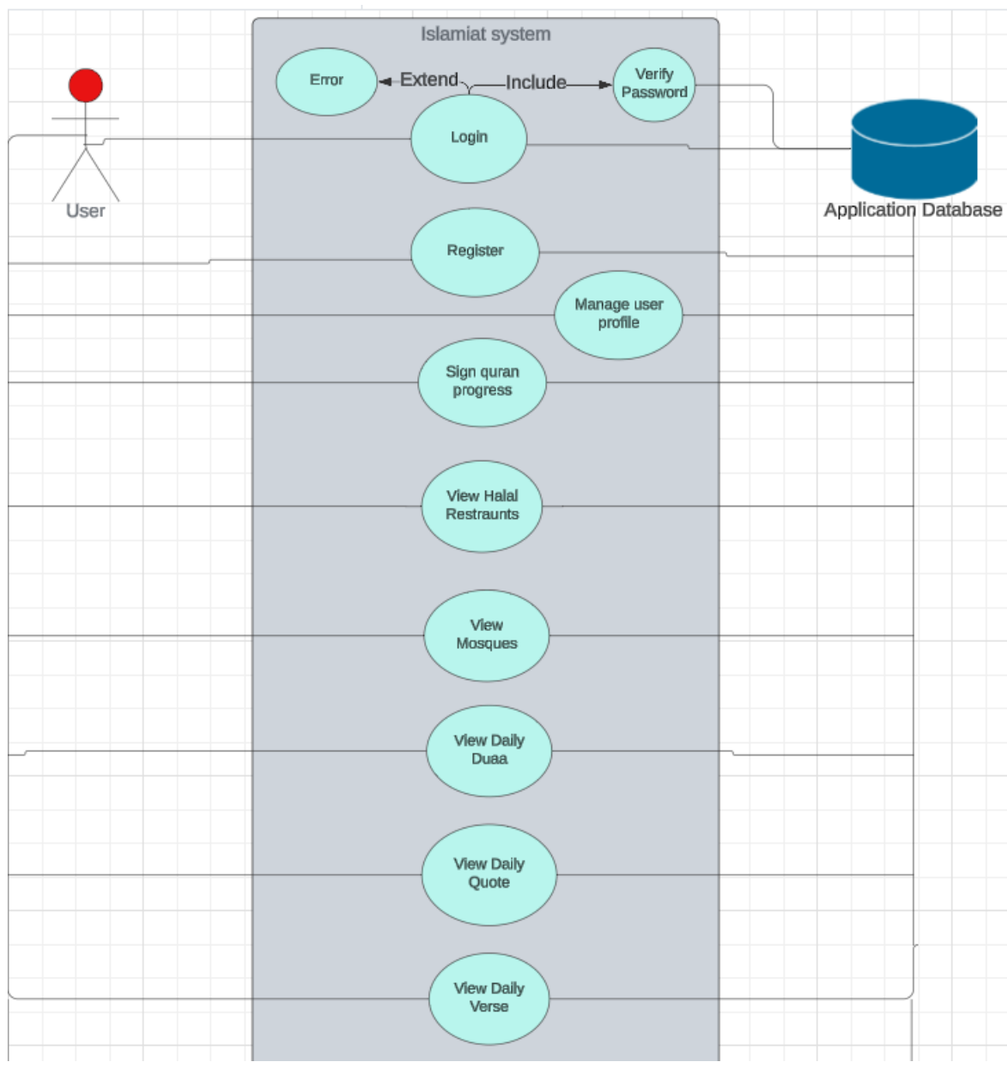
-The system works 24 hours.

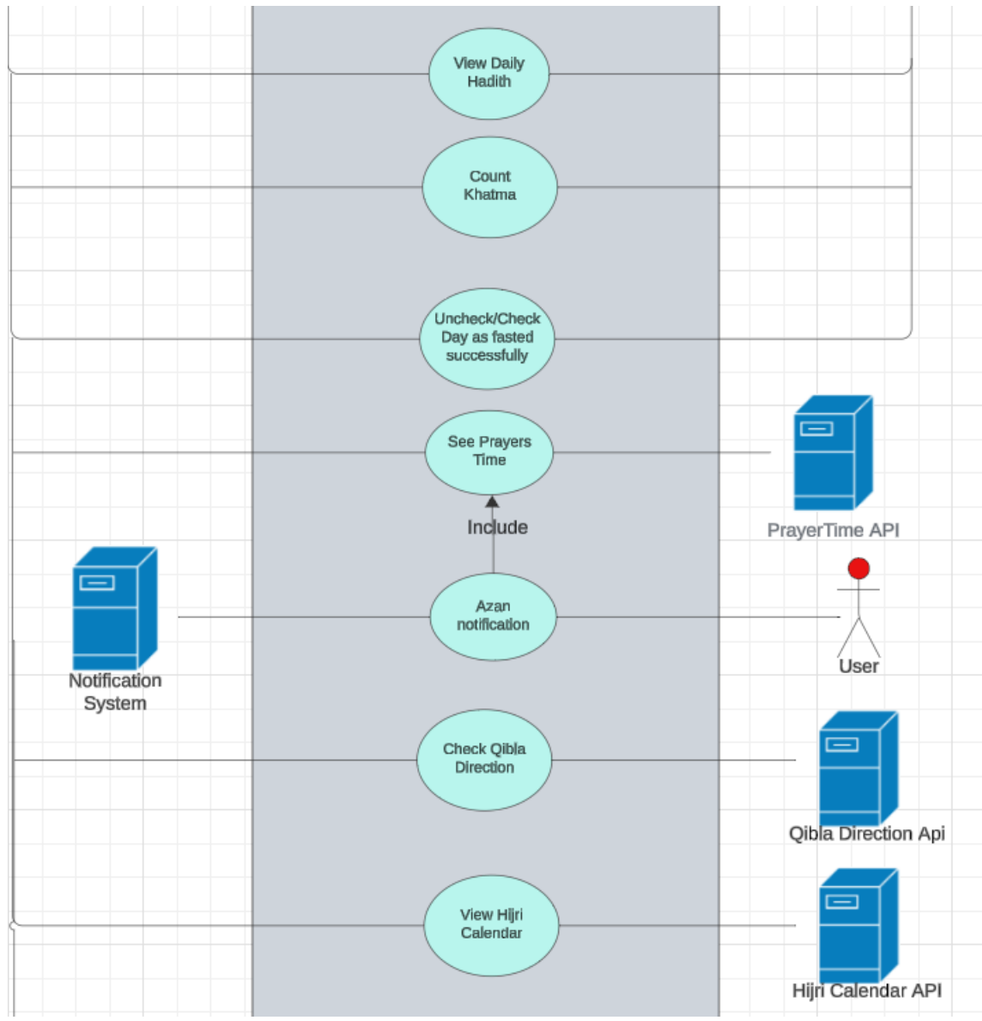
9- Reliability

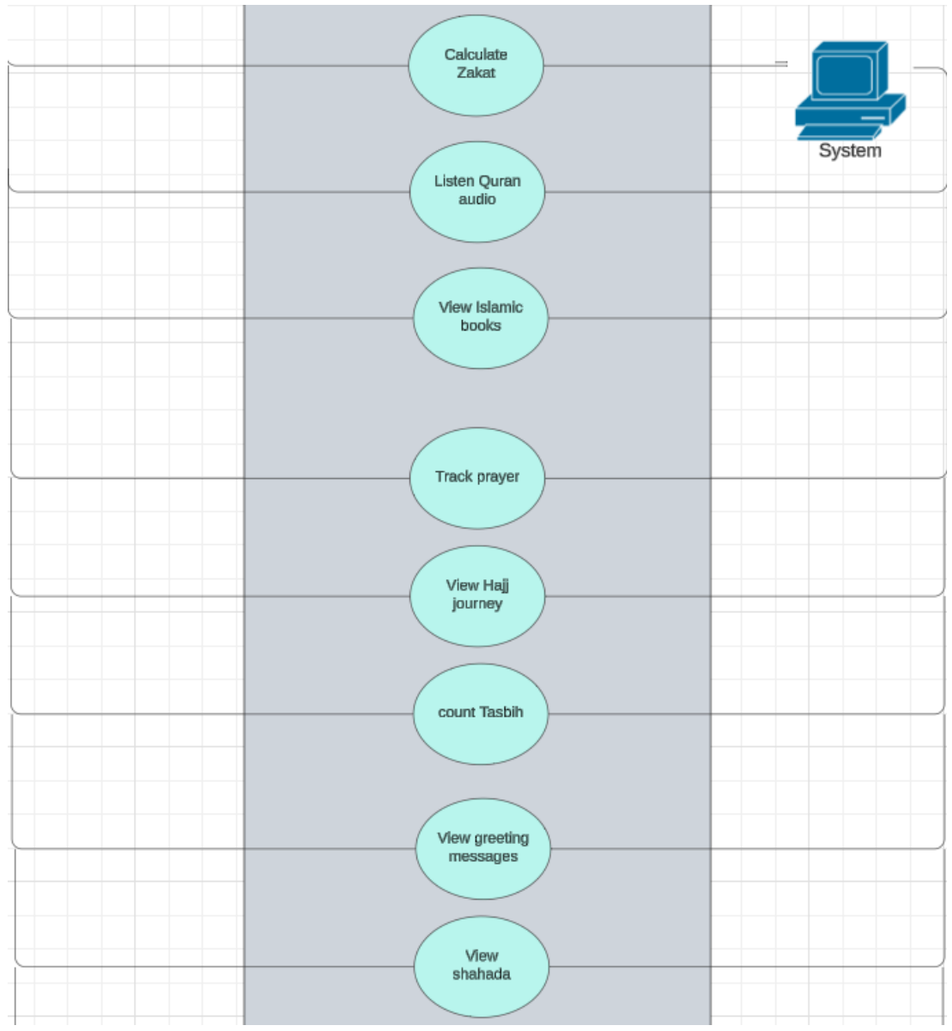
- The system has to do its functions without failure.

3.1-Usecase diagram:

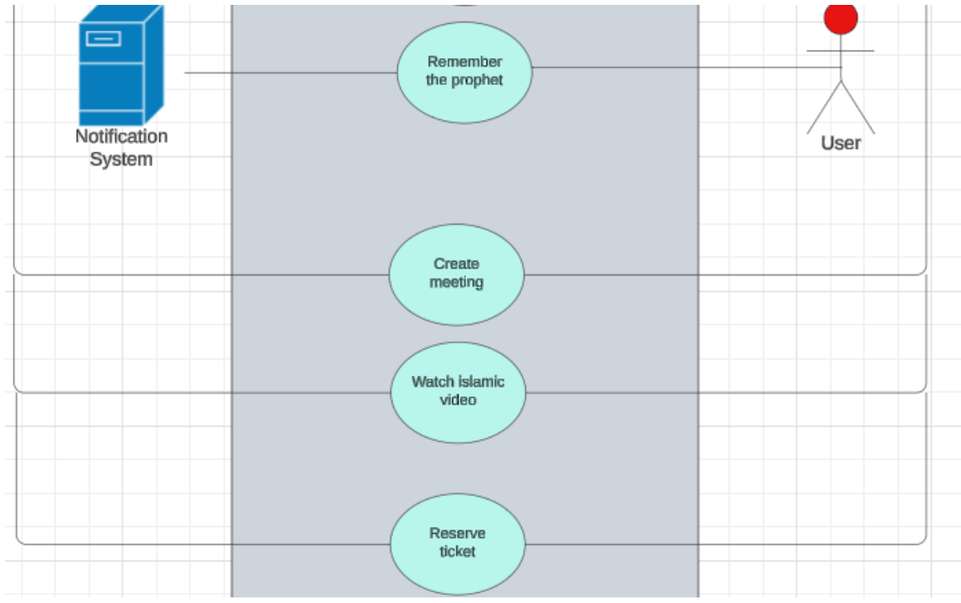
User use case:


Islamiat system
Error
Extend
Include
Verify Password
Login
User
Application Database
Register
Manage user profile
Sign quran progress
View Halal Restraunts
View Mosques
View Daily Duaa
View Daily Quote
View Daily Verse

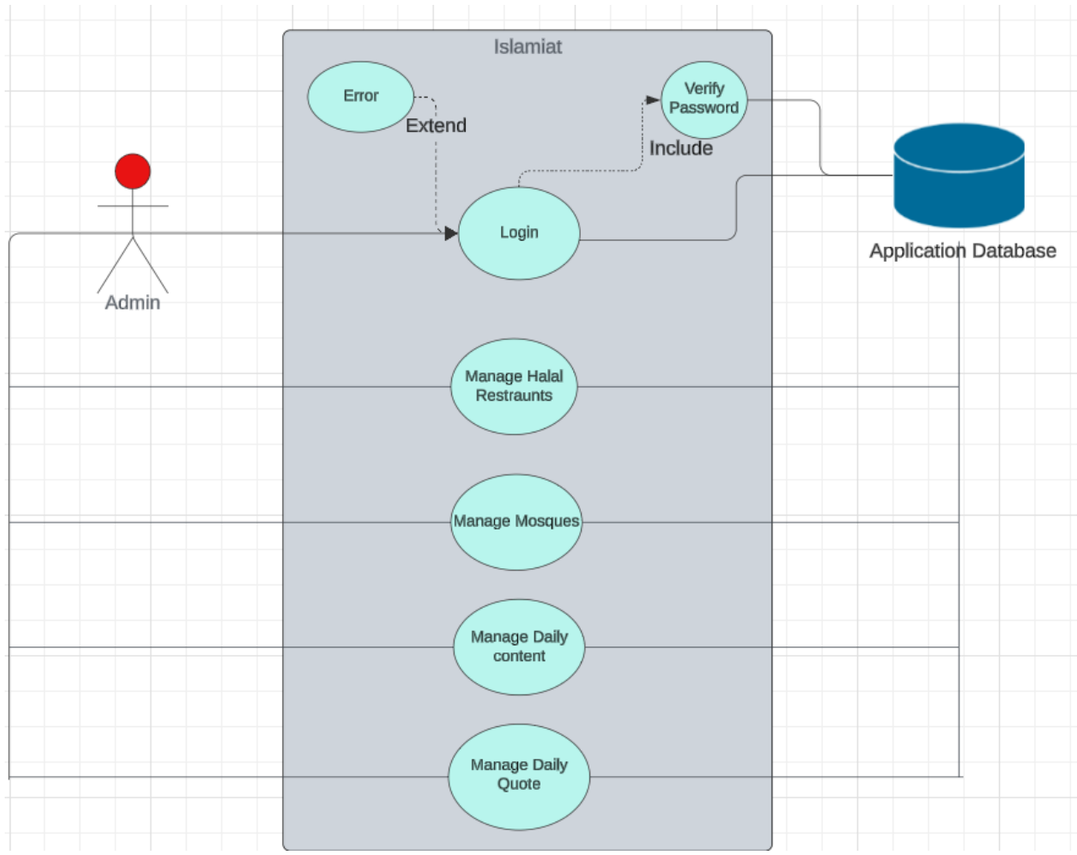

View Daily Hadith
Count Khatma
Uncheck/Check Day as fasted successfully
See Prayers Time
Include
Azan notification
Check Qibla Direction
View Hijri Calendar
PrayerTime API
User
Notification System
Qibla Direction Api
Hijri Calendar API


Calculate Zakat
Listen Quran audio
View Islamic books
Track prayer
View Hajj journey
count Tasbih
View greeting messages
View shahada
System

View quran tafseer
Donations
Todo list
Holy quran
View asma Al-Husna
Watch makkah live stream
View hajj-umrah guide
View morning/evening prayer
Ask question



Admin use case:



3.2Use Case Description

For azan application

(USER & ADMIN)

User use case descriptions:

Use case name: Register

-Summary: If the user doesn't already have an account, the system uses this to record the user's data in the database (and check if they follow the criterea), so that the user logs into the system whenever he tries with the data he submitted.

-Actors: User (Primary Actor), Application database (Secondary Actor)

-Precondition: Display a login form, and a create account button.

-Main Sequence:

1. The user clicks on the "Create account" button.
2. The user is now forwarded to the registration form.
3. The user enters the "Email, Username, Password, Phone" data.
4. The new account is now added to the database.
5. User profile created in the database.

-Alternative sequence:

1. If the data of the user doesn't match the criteria, an error message is displayed.

-Post-Condition: When the data is added successfully to the database, the user logs in, and a welcome message is displayed.

Use case name: Login

-Summary: The system uses this to check that the user has account in the system by username and password or tell the user that there's an error in the username or the password so that he can log into the system as user

-Actors: User (Primary Actor), Application database (Secondary Actor)

-Precondition: Display a login form, and a create account button.

-Main Sequence:

1. **The user will write the username.**
2. **The user will write the password.**
3. **Then the data will be verified by comparing with a matching record in the database.**
4. **The system gets user location.**

-Alternative sequence:

1. **If the data the user entered is wrong, an error message is displayed.**

-Post-Condition: When the data matches a record in the database successfully a welcome message is displayed.

use case name: Listen Quran audio

-Summary: The user will want to listen to the recitation of the Quran and will choose the Surah then choose a Qari' and then the selected record is played.

-Actors: User (Primary Actor), System (Secondary Actor)

-Precondition: The user opens the "Library" from navbar then click on Quran audio.

-Main Sequence:

1. The user chooses the desired Sheikh.
2. The user chooses the desired Surah.

-Post-Condition: The track of this Surah is played.

use case name: View Halal Restaurants

-Summary: The user will want to see the available Halal restaurants in a specific location, the system will use his location then list of the available Halal restaurants in this location will be displayed.

-Actors: User (Primary Actor), Application database (Secondary Actor)

-Precondition: The user clicks on the “View Halal Restaurants” option in homepage.

-Main Sequence:

1. The List of the matching records in our database of all the halal restaurants in the same location is displayed.

-Alternative sequence:

1. If there’s no halal restaurant stored in the database for the location, it will not recommend any restaurant.

-Post-Condition: A list of the available halal restaurants in this city, each with its address line will be displayed to the user.

use case name: View Mosques

-Summary: The user will want to see the available Mosques in a specific location, so he’ll allow the system to access his current location and then a list of the available Halal restaurants in this location will be displayed.

-Actors: User (Primary Actor), Application database (Secondary Actor)

-Precondition: The user clicks on the “View Mosques” option in homepage.

-Main Sequence:

2. The List of the matching records in our database of all the Mosques in the same location is displayed.

-Alternative sequence:

2. If there's no Mosques stored in the database for the location, it will not recommend any Mosque.

-Post-Condition: A list of the available mosques in this city, each with its address line will be displayed to the user.

-Post-Condition: A list of the available mosques in this city, each with its address line will be displayed to the user.

use case name: View Daily Quote

-Summary: The user will see the Islamic quote of the day as a part of his daily routine in our application

-actors: User (Primary Actor), Application database (Secondary Actor)

-Precondition: The user clicks on the "Library" navbar then clicks on daily content.

-Main Sequence:

The daily quote is displayed on the screen from the database.

-Post-Condition: A quote is displayed on the screen.

use case name: View popular verse

-Summary: The user will see popular verses of the day automatically as a part of his daily routine in the application

-actors: User (Primary Actor), Application database (Secondary Actor)

-Precondition: The user clicks on the “Library” navbar then clicks on daily content.

-Main Sequence:

1. The system gets popular verses from database.
2. The system displays popular verses to the user.

-Post-Condition: Verses are displayed on the screen

use case name: View daily hadith

-Summary: The user will see the daily Hadith of the day as a part of his daily routine in our application

-actors: User (Primary Actor), Application database (Secondary Actor)

-Precondition: The user clicks on the “Library” navbar then clicks on daily content.

-Main Sequence:

1. The daily hadith is displayed on the screen from the database.

-Post-Condition: A hadith is displayed on the screen.

use case name: Uncheck/Check Prayer name

-Summary: The user can track his prayers for this day using check boxes of the 5 prayers and check on the ones that he completed.

-actors: User (Primary Actor), System (Secondary Actor)

-Precondition: The user clicks on the “Prayer Tracker” button on the home page.

-Main Sequence:

1. The system create 5 prayer objects.
2. The user has the opportunity to check them until the end the of the day.

-Alternative Sequence:

1. The user edits the prayer spot he checked as completed if he misses checked it.

-Post-Condition: the prayer is checked for the day as completed.

use case name: Uncheck/Check Day as fasted

-Summary: The user can track his fasting days for the month using check boxes of the month days and check on the ones that he completed.

-actors: User (Primary Actor), Application database (Secondary Actor)

-Precondition: The user clicks on the “Fasting Tracker” button home page.

-Main Sequence:

1. The user checks/unchecks the day he fasted on the screen for a month.
2. The system modifies the new information in user profile automatically.

-Post-Condition: The day is checked as fasted day.

use case name: View prayer times

-Summary: The user can check the timing of all prayers in the current location.

-actors: User (Primary Actor), Prayer Time API (Secondary Actor)

-Precondition: The user enters the “prayer time” option on the homepage.

-Main Sequence:

1. The current location of the user is used.
2. The system display all prayer times to user according to his location.

-Post-Condition: The prayer times are shown on the screen.

Use case name: Azan Notification:

-Summary: The system will show an alert when the current time zone of the user has an active prayer.

-actors: “Notification System” (Primary Actor), User(Secondary Actor)

-Precondition: The user is logged in successfully

-Main Sequence:

1. The system access user location.
2. When it’s time for prayer according to the location, the system alert the user with notification.

-Post-Condition: The prayer alert with the name of the prayer is shown on the screen.

Use case name: Qibla Guide

-Summary: The user can check the Qibla towards the Kaaba from wherever he is.

-actors: User (Primary Actor), Qibla Direction Service (Secondary Actor)

-Precondition: The user clicks on the “Qibla guide” button in the Home.

-Main Sequence:

1. The system access user location.
2. The direction of the Qibla from his current location towards Al-Kaaba is shown on the screen.

-Alternative Sequence:

1. The user changes the location to one which he wants to know the Qibla in

-Post-Condition: The line of the Qibla is shown on the screen.

use case name: View Hijri Calendar

-Summary: The user sees the Hijri Calendar.

-Actors: User (Primary Actor), Hijri Calendar API (Secondary Actor)

-Precondition: The user enters the “Library” option in homepage.

-Main Sequence:

1. The user enters Hijri calendar in library section.
2. The current Hijri day and the whole month are shown on the screen

-Post-Condition: The current Hijri day and the whole month are shown on the screen.

Use Case Name: Calculate Zakat

Summary: The user can calculate the Zakat amount they owe based on their assets and income according to Islamic principles.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user selects the "Calculate Zakat" option in the application's menu and provides necessary financial information such as assets and income.

Main Sequence:

1. The user enters their financial details including savings, investments, and income in one variable.
2. The Zakat Calculator System computes the Zakat amount based on the provided information and the applicable Zakat rates.
3. The calculated Zakat amount is displayed to the user on the screen.

Post-Condition: The user is provided with the calculated Zakat amount according to Islamic guidelines based on the information provided, facilitating adherence to religious obligations.

Usecase name: View user profile

Summary: The user can know his information stored in the system.

Actors: User (Primary Actor), Database (Secondary Actor)

Precondition: The user selects the "profile icon" in the application

Main Sequence:

1- The system gets user information according to his sign info from database.
2- The system displays user information on the screen.

Post-Condition: The user can see his recorded information in the system.

Use case name: Sign Quran progress

Summary: The user can modify his Quran progress to keep his progress on the line.

Actors: User (Primary Actor), Database (Secondary Actor)

Precondition: The user selects the "profile icon" in the application

Main Sequence:

1- The user modifies his information in the Quran progress section.
2- The user submits his modification.
3- The system applies the change in the database.
4- The system changes the status of the user profile.

Post-Condition: The user submits modification, and the system stores the change.

Use case: Count khatma

Summary: Every time user finish khatma within month, he records it in the system.

Actors: User (Primary Actor), Database (Secondary Actor)

Precondition: The user selects the "profile icon" in the application

Main Sequence:

1- The user modifies his information in Count khatma section.
2- The user submits his modification.
3- The system applies the change in the database.
4- The system changes the status of user profile.

Post-Condition: The user submits modification and the system stores the change.

Use case: View Islamic books

Summary: The user can increase his knowledge by surfing in Islamic books provided in the system.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user enters the "Library" option in homepage.

Main Sequence:

1- The user enters islamic books section.
2- The user choose one of book from existing books.
3- The system sends the user to another tab that has more info on this book.

Post-Condition: The user got all book details he needs.

Use case: View Hajj journey

Summary: The system Provide Hajj journey steps.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user enters the "Library" option in homepage.

Main Sequence:

1- **The user enter Hajj journey section.**
2- **The system displays Hajj journey steps for the user.**

Post-Condition: The user remembers/ knows keeps Hajj journey steps.

Use case: Count Tasbih

Summary: The user can count his tasbih worship once in time.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user selects the "Count tasbih" option in the homepage.

Main Sequence:

1- The user presses the counter button.
2- The System increases the counter each time user presses the counter button.
3- The System keep the change until user get out from Count tasbih option.

Alternative option

1- If the user press reset button
2- The counter back to zero

Post-Condition: The user keeps following his tasbih.

Use case: View greeting messages

Summary: The user knows our greeting messages in Islam, so he knows more about communication in Islam.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user enters the "Library" option in the homepage.

Main Sequence:

1- The user enters greeting message section.
2- The system displays all greeting messages stored.

Post-Condition: The system displays all greeting messages.

Use case: View shahada

Summary: The system provides our shahada in Islam.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user selects the "Library" option in the homepage.

Main Sequence:

1- The user enters Shahada section.
2- The system displays Shahada for the user.

Post-Condition: The system displays Shahada for the user.

Use case: Donations

Summary: The system provides for user donations sources that he can use it.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user selects the "Library" option in the homepage.

Main Sequence:

1- The user enters donation section.
2- The system displays donations resources.
3- The user chooses one of the resources.

Post-Condition: The system displays Shahada for the user.

Use case: Todo list

Summary: The user obligates himself with some good things he wants to do in the day.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user selects the "Notes" option in the home page.

Main Sequence:

1- The user adds mission.
2- The system create object for the user.
3- The user enter what he want to do.
4- The system keeps the content until the end of the day.
5- The user has to check if he did it or not.

Alternative sequence:

1- If the user miss check, he can uncheck it

Post-Condition: The System saves user content until the end of the day.

Use case: View Asma’ Allah Al-Husna

Summary: The system provides asma Al-Husna for Allah.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user selects the "Library" option in the homepage.

Main Sequence:

1- **The user enters Asma’ Allah Al-Husna section.**
2- **The system displays Asma Al-Husna**

Post-Condition: The user reads Asma Al-Husna.

Use case: Watch Makkah live stream

Summary: The system guide the user to watch makkah live stream(24 hours).

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user selects the "Makkah stream" option in the homepage.

Main Sequence:

1- The system sends the user to Makkah live YouTube.

Post-Condition: The user can watch Makkah live stream

Use case: Create meeting

Summary: When user want to create google meet to communication with other people, the system provides google meet for the user

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The user selects the "Create meeting" option in the homepage.

Main Sequence:

1- The User asks for creating meeting.
2- The system sends the user to google meet and provides the link.

Post-Condition: The user get thee google meet and can send it to other people.

Use case: Remember the prophet:

Summary: When every time user goes to home page, the system alert the user to remember the prophet.

Actors: User (Primary Actor), System (Secondary Actor)

Precondition: The User available on the home page.

Main Sequence:

1- The user signs to the application.
2- The system sends the alert to the user to remember the prophet.

Alternative sequence:

1- The user back to homepage.
2- The system sends the alert to the user to remember the prophet.

Post-Condition: The system display the alert message.

3.2Admin use case descriptions:

use case name: Login

-Summary: we use this to check that the admin has account in the system by login and the right account using another use case it's verified password that checks if the password is true and the extended use case tell the admin there's an error in the password and to can treat with the system as admin

-actors: Admin (Primary Actor), Application database (Secondary Actor)

-Precondition: display message welcome

-Main Sequence:

1. The admin will write the id
2. The admin will write the password

3. Then the email will be verified by matching with a matching record in the database

-alternative sequence:

1. If the credentials the admin entered are wrong, an error message is displayed.

-Post-Condition: When the credentials are matched successfully with the database, greeting message displayed to the admin

Use case: Manage halal restaurants

Summary: Admins can manage information about halal restaurants by adding or deleting restaurants or updating information for specific ones.

Actors: Admin (Primary Actor), Application Database (Secondary Actor)

Precondition: The admin accesses the "Manage Halal Restaurants" section within the application.

Main Sequence:

1. The admin selects the option to add a new Halal restaurant.
2. The admin provides details such as name, country, city, address.

Alternative sequence:

1. If admin is updating, the admin selects a specific restaurant and modifies its information.

2. If admin is deleting, the admin selects the restaurant to be removed from the database.

3. The admin updates the database accordingly.

Post-Condition: The database reflects the latest additions, deletions, and updates, providing users with accurate and up-to-date information for dining choices.

Use case: Manage mosques

Summary: Admins can manage information about mosques by adding or deleting mosques or updating information for specific ones.

Actors: Admin (Primary Actor), Application Database(Secondary Actor)

Precondition: The Admin accesses the "Manage Mosques" section within the application.

Main Sequence:

1. The admin selects the option to add a new mosque and provides details such as name, country, city, address.
2. If admin is updating, the admin selects a specific mosque and modifies its information.
3. If deleting, the Admin selects the mosque to be removed from the database.
4. The Admin updates the database accordingly.

Post-Condition: The database reflects the latest additions, deletions, and updates, providing users with accurate and up-to-date information for their religious needs.

Use Case Name: Manage Daily content

Summary: The admin can add or delete daily content from the database available in the application.

Actors: Admin (Primary Actor), Application Database (Secondary Actor)

Precondition: The admin accesses the "Manage Daily content" feature in the application's settings or admin panel.

Main Sequence:

1. The Admin selects the option to add a new daily content (versus, duaa, hadith) or delete an existing one from the database.
2. when Admin wants to manage any daily content, it's a must to make the change on all daily content categories.
3. If adding a new daily content, the Admin provides the text and any relevant details of the content to be added.
4. If deleting daily content, the Admin selects the content from the list of available content.
5. The Admin updates the database accordingly.

Alternative Sequence:

1. If the Admin attempts to add new daily content that already exists in the database, the system prompts them with a message indicating the duplication.

Post-Condition: The database of daily Duas is updated with the addition or deletion of the selected Duaa, ensuring the availability of desired prayers for users.

Use Case Name: Manage Daily Quote

Summary: The Admin can add or delete quotes from the database of daily quotes available in the application.

Actors: Admin (Primary Actor), Application Database (Secondary Actor)

Precondition: The Admin accesses the "Manage Daily Quote" feature in the application's settings or admin panel.

Main Sequence:

1. The Admin selects the option to add a new quote or delete an existing one from the database.
2. If adding a new quote, the Admin provides the content and reference of the quote to be added.
3. If deleting a quote, the user selects the quote from the list of available quotes.
4. The Admin updates the database accordingly.

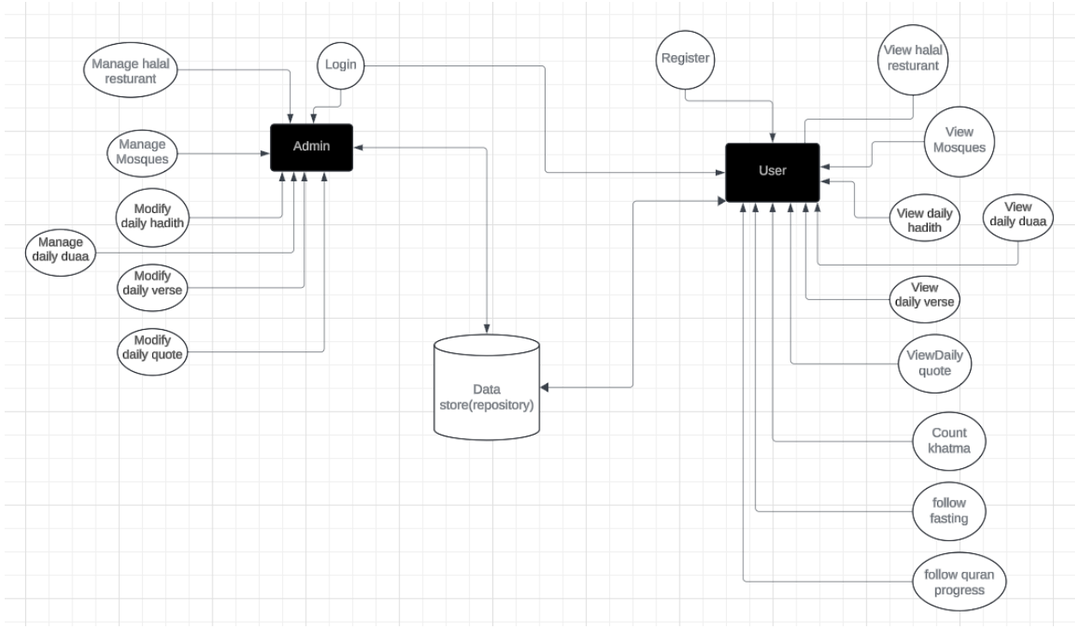
Alternative Sequence:

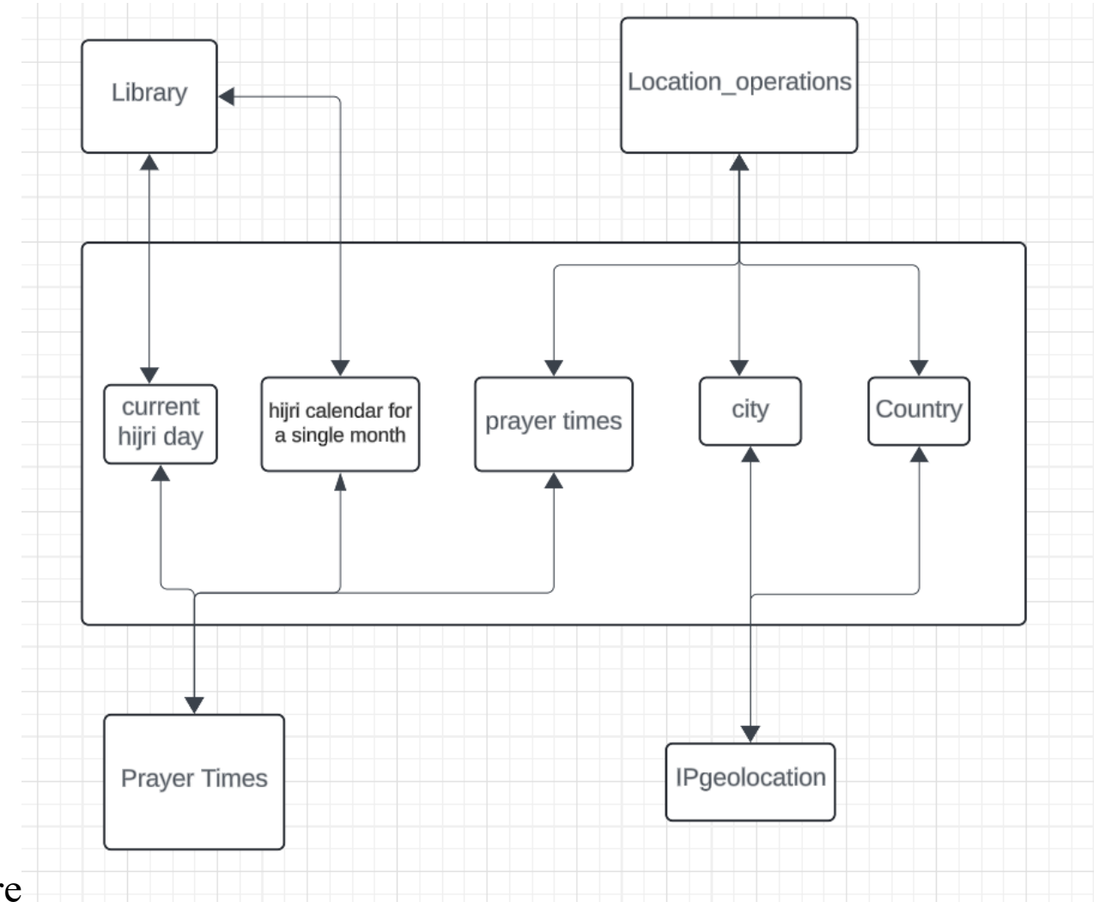
1. If the Admin attempts to add a quote that already exists in the database, the system prompts them with a message indicating the duplication.

Post-Condition: The database of daily quotes is updated with the addition or deletion of the selected quote, ensuring the availability of inspiring content for users.

4-System architecture:

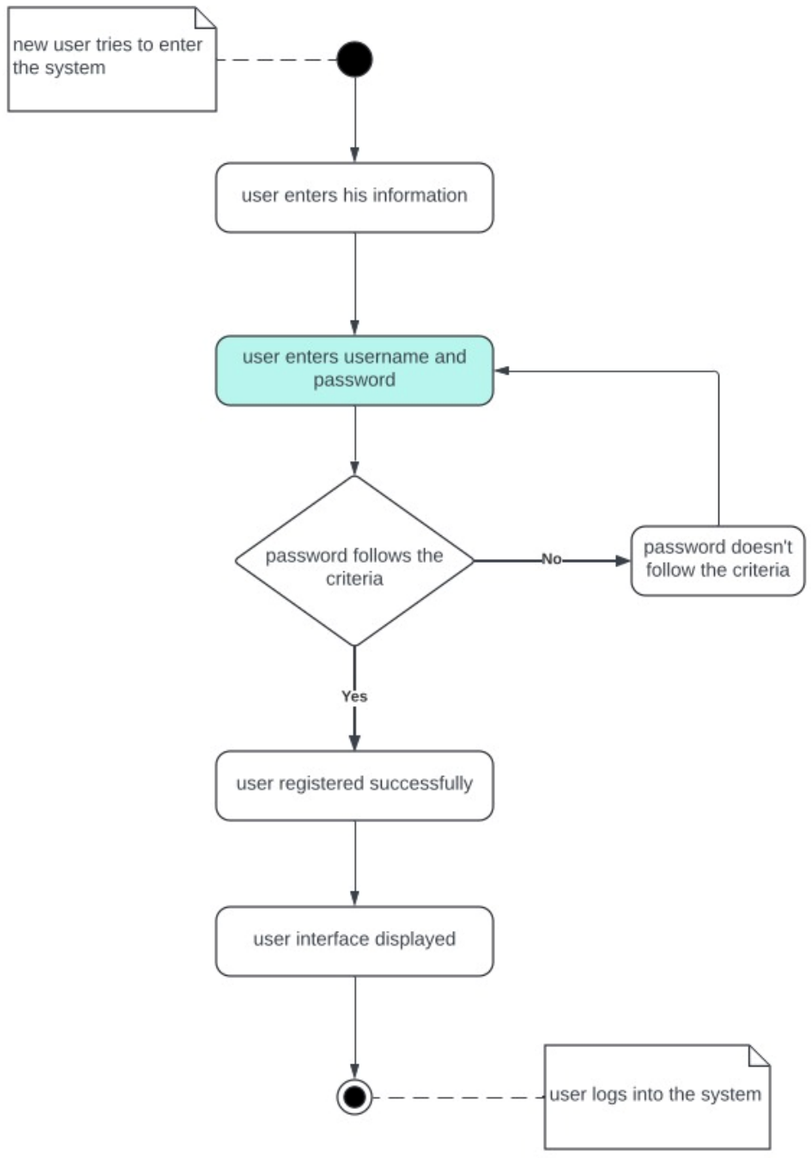
4.1- Data centered

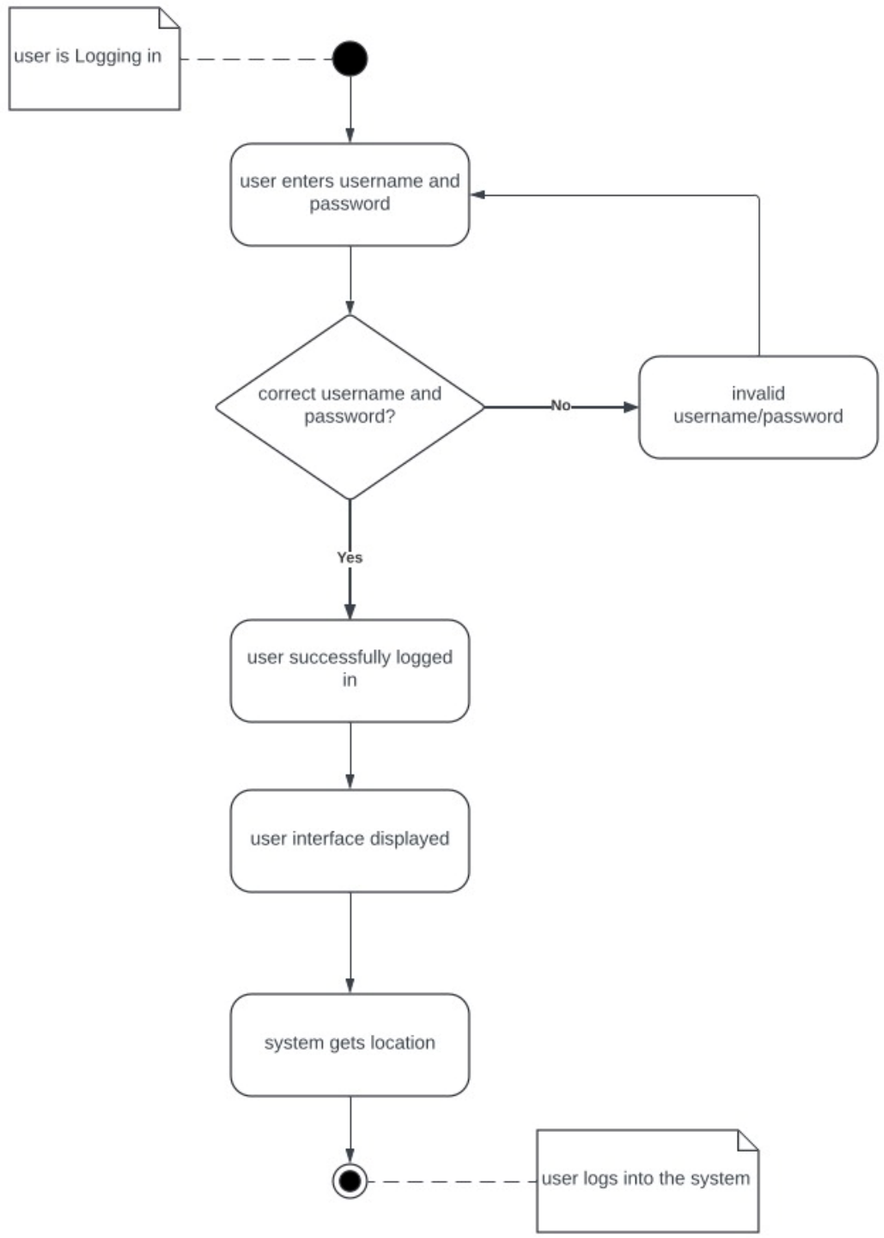

Manage halal resturant
Login
Manage Mosques
Admin
Modify daily hadith
Manage daily duaa
Modify daily verse
Modify daily quote
Data store(repository)
Register
View halal resturant
View Mosques
User
View daily hadith
View daily duaa
View daily verse
ViewDaily quote
Count khatma
follow fasting
follow quran progress

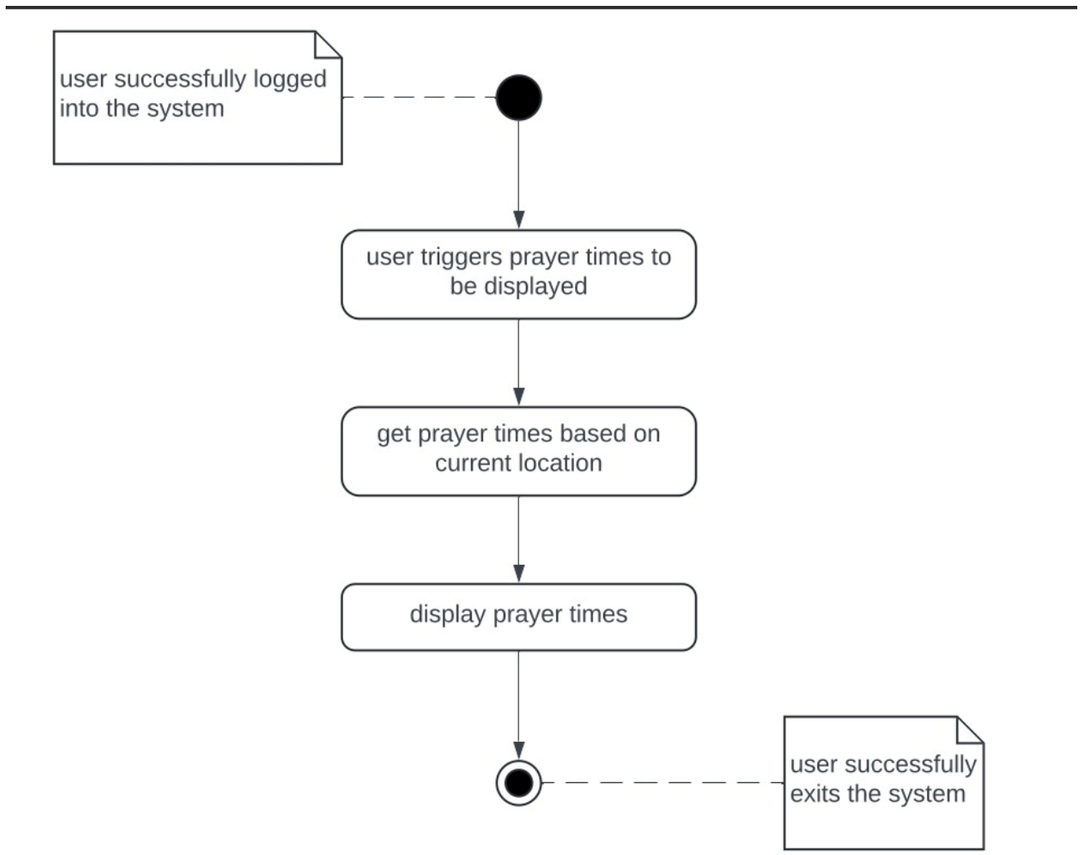


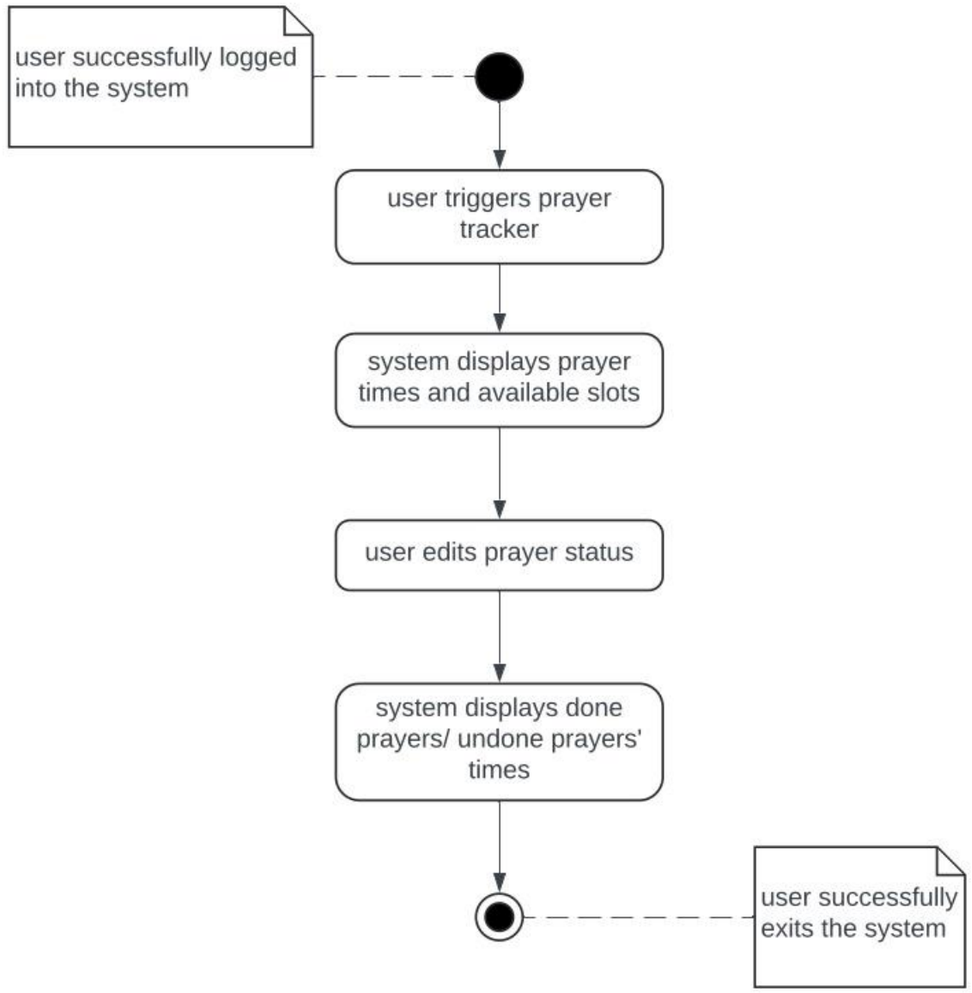
4.2 Service oriented architecture

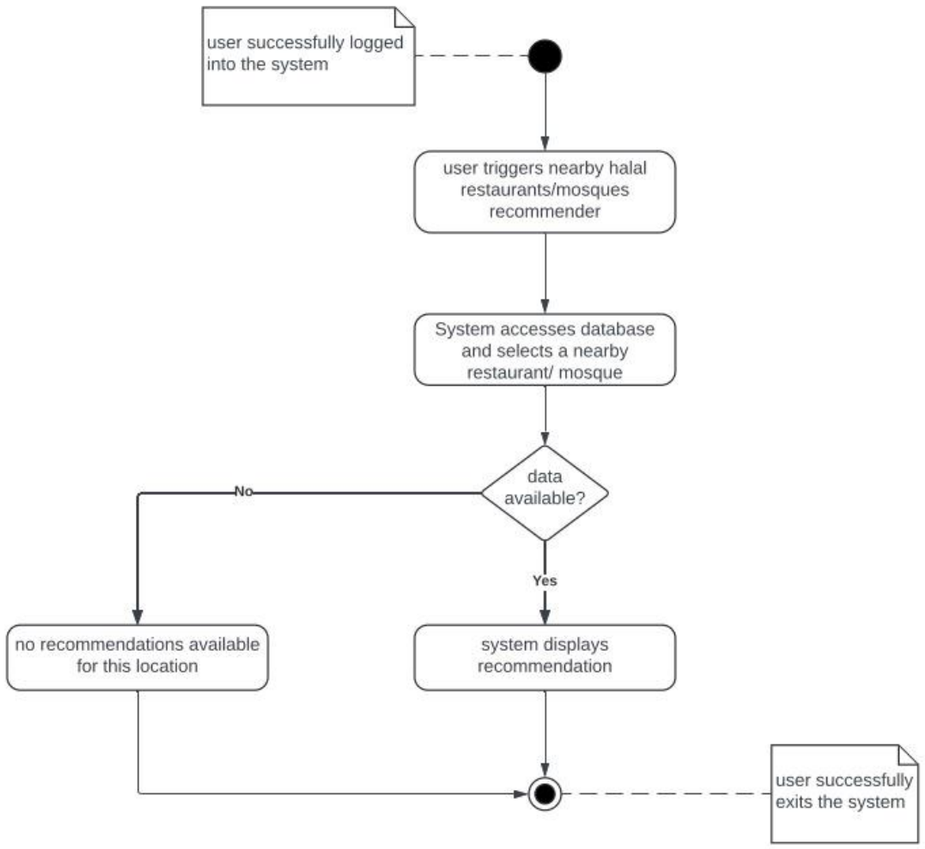
5-Activity diagram

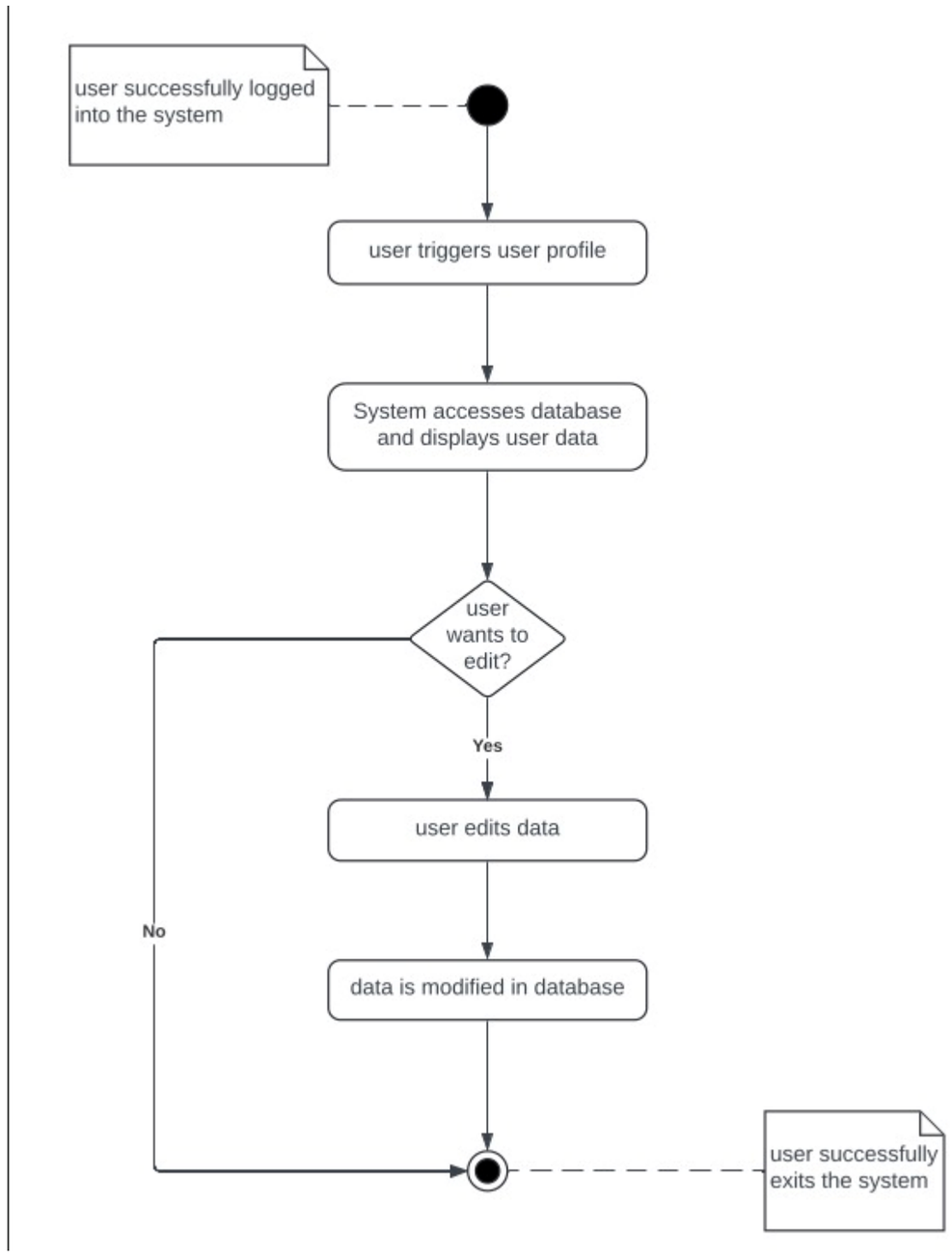

new user tries to enter the system
user enters his information
user enters username and password
password follows the criteria
No
password doesn't follow the criteria
Yes
user registered successfully
user interface displayed
user logs into the system


user is Logging in
user enters username and password
correct username and password?
No
invalid username/password
Yes
user successfully logged in
user interface displayed
system gets location
user logs into the system

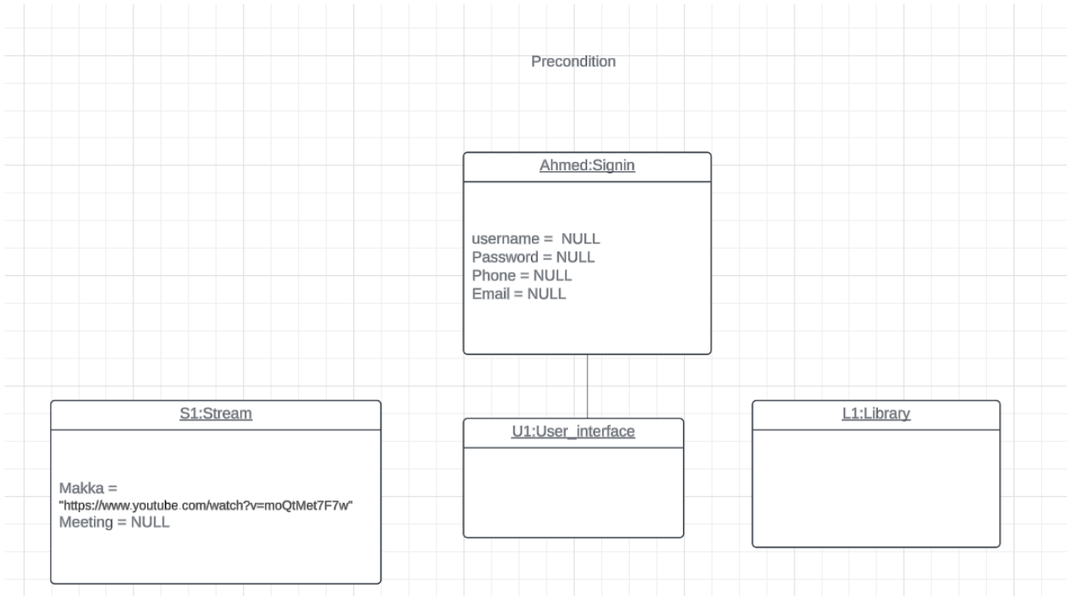

user successfully logged into the system
user triggers prayer times to be displayed
get prayer times based on current location
display prayer times
user successfully exits the system

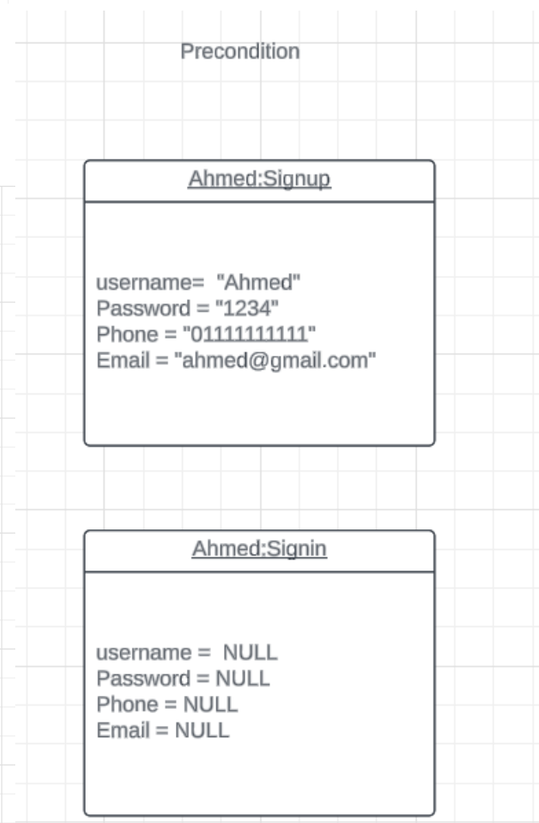

user successfully logged into the system
user triggers prayer tracker
system displays prayer times and available slots
user edits prayer status
system displays done prayers/ undone prayers' times
user successfully exits the system

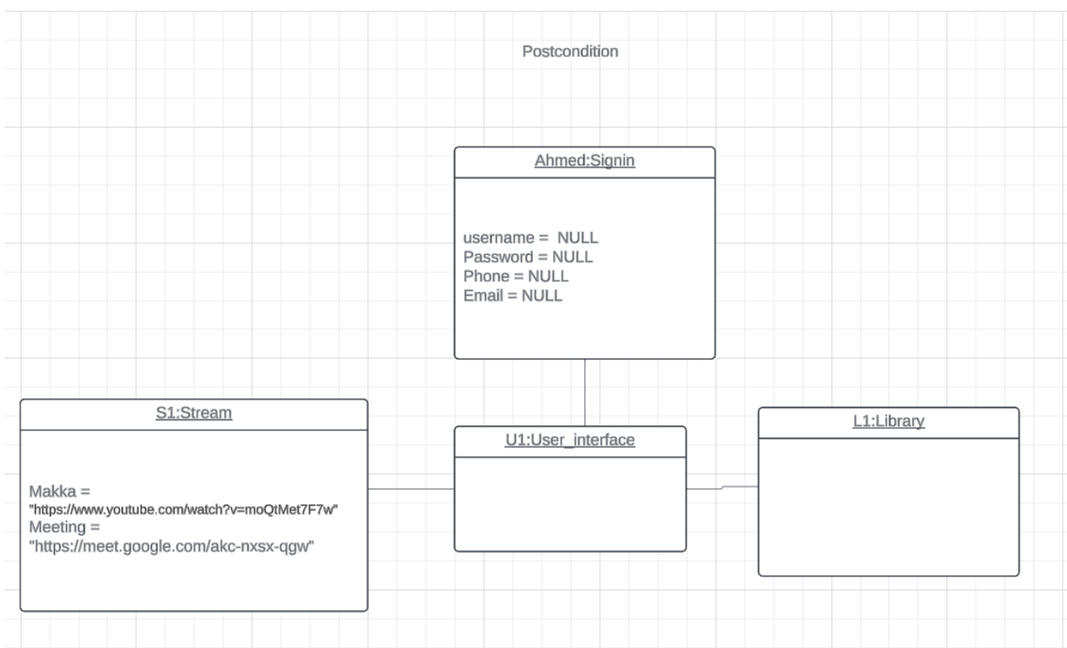

user successfully logged into the system
user triggers nearby halal restaurants/mosques recommender
System accesses database and selects a nearby restaurant/ mosque
data available?
No
Yes
no recommendations available for this location
system displays recommendation
user successfully exits the system

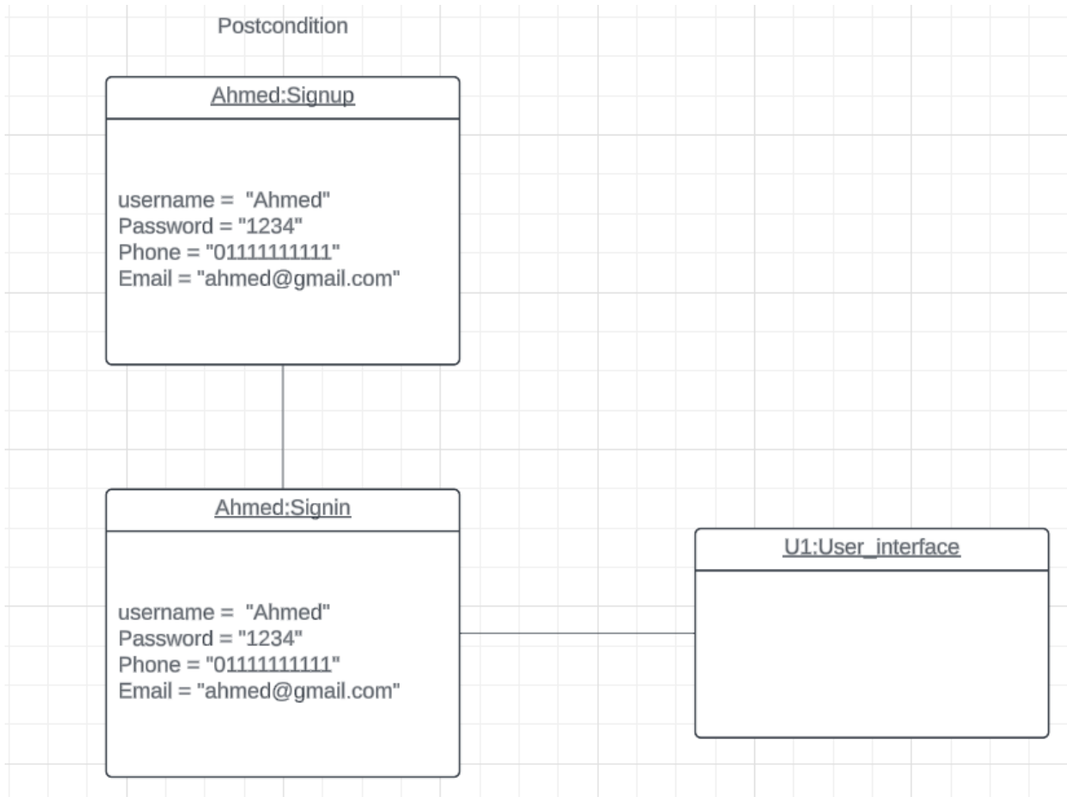


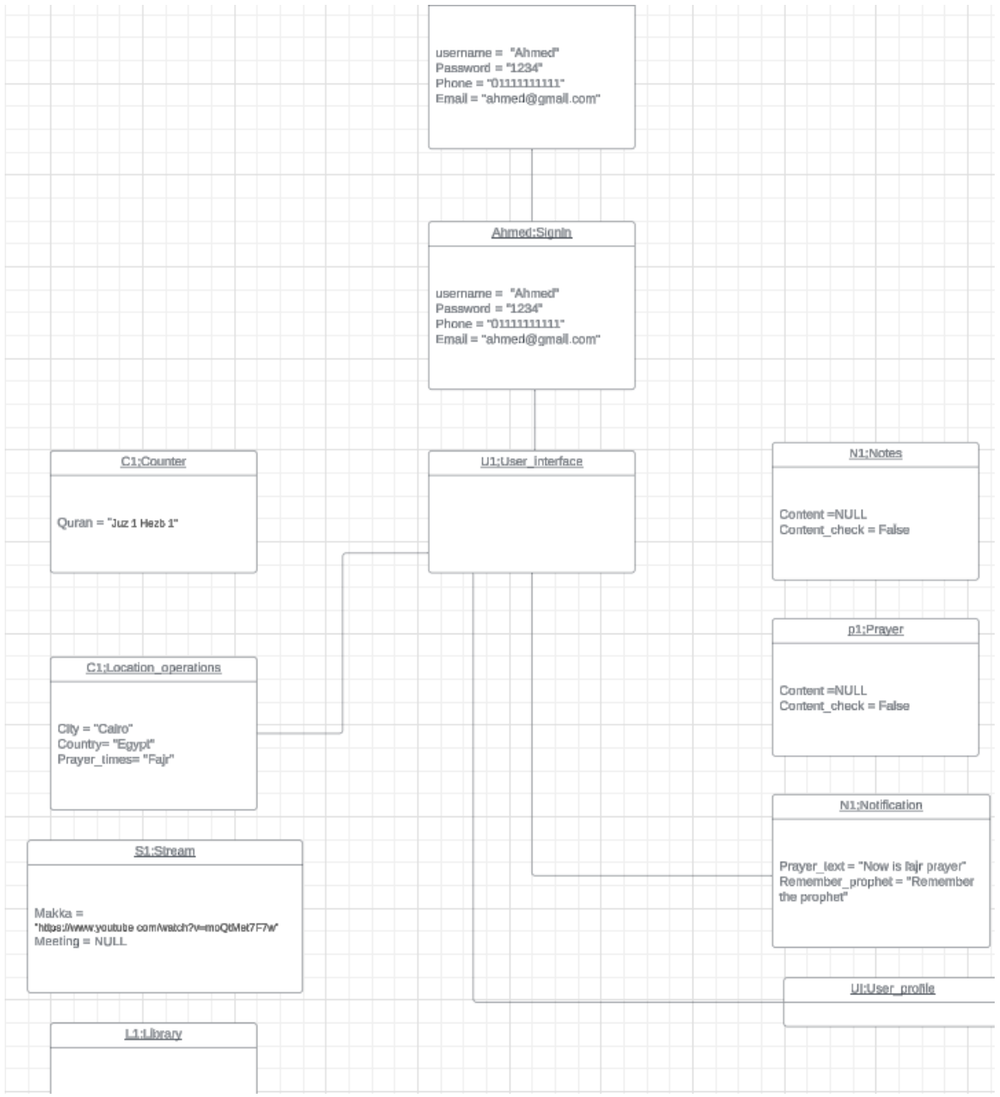
6-Object diagram

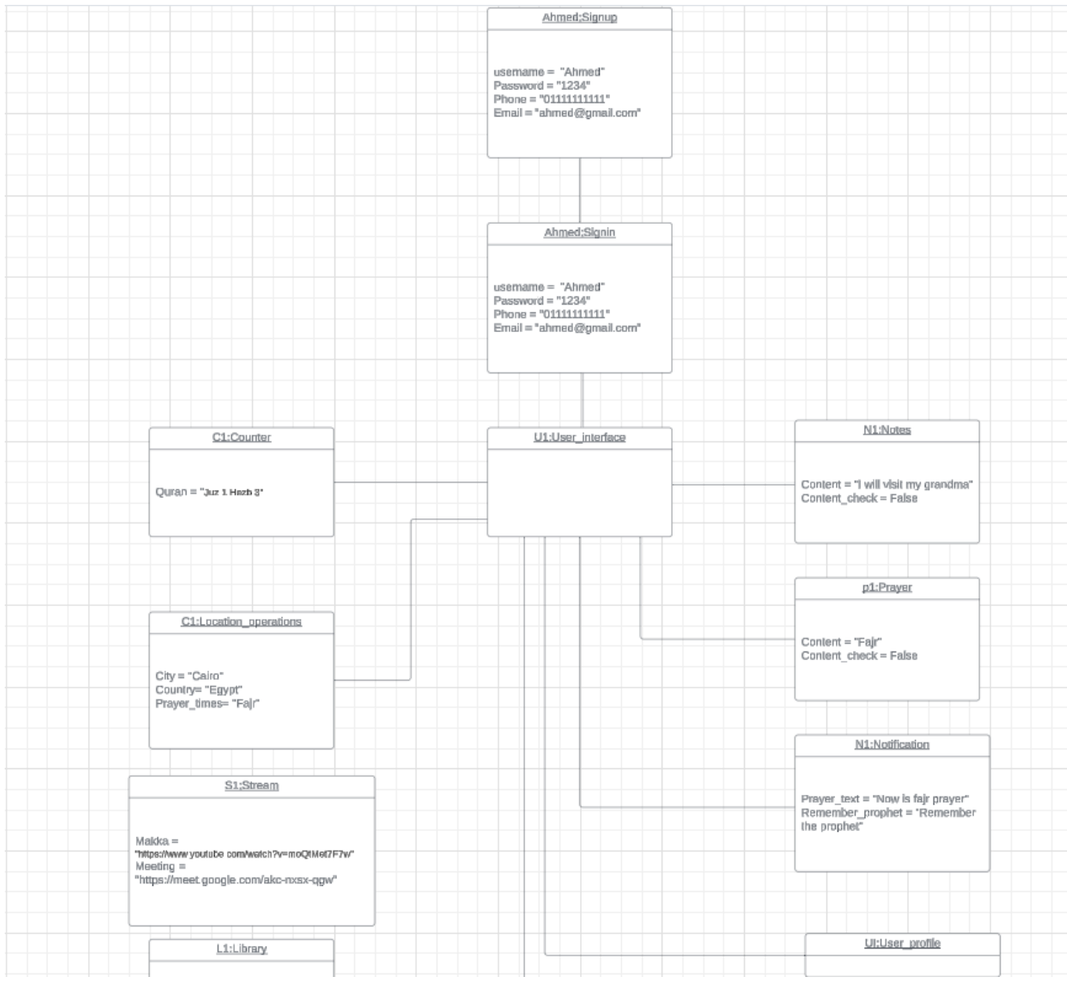

Precondition
Ahmed:Signin
username = NULL
Password = NULL
Phone = NULL
Email = NULL
S1:Stream
Makka =
"https://www.youtube.com/watch?v=moQtMet7F7w"
Meeting = NULL
U1:User_interface
L1:Library

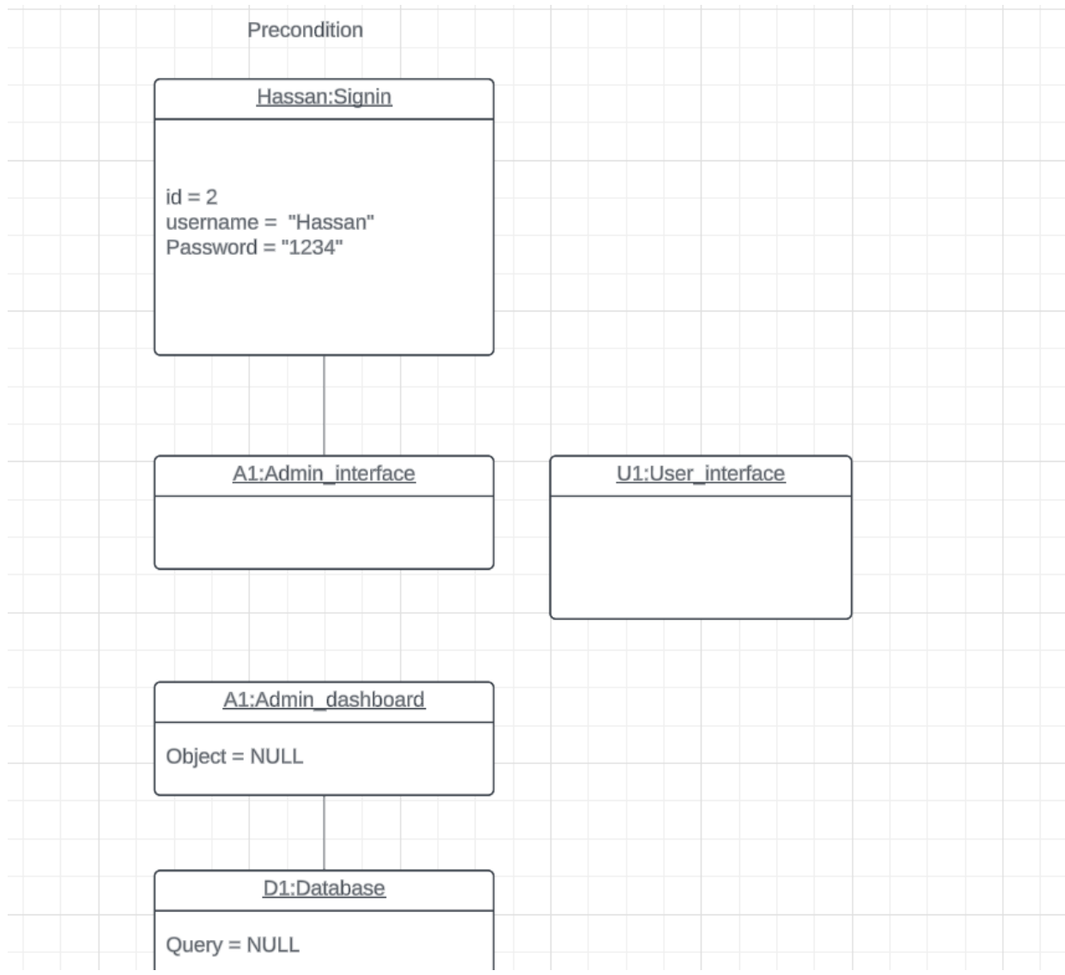

Precondition
Ahmed:Signup
username= "Ahmed"
Password = "1234"
Phone = "01111111111"
Email = "ahmed@gmail.com"
Ahmed:Signin
username = NULL
Password = NULL
Phone = NULL
Email = NULL

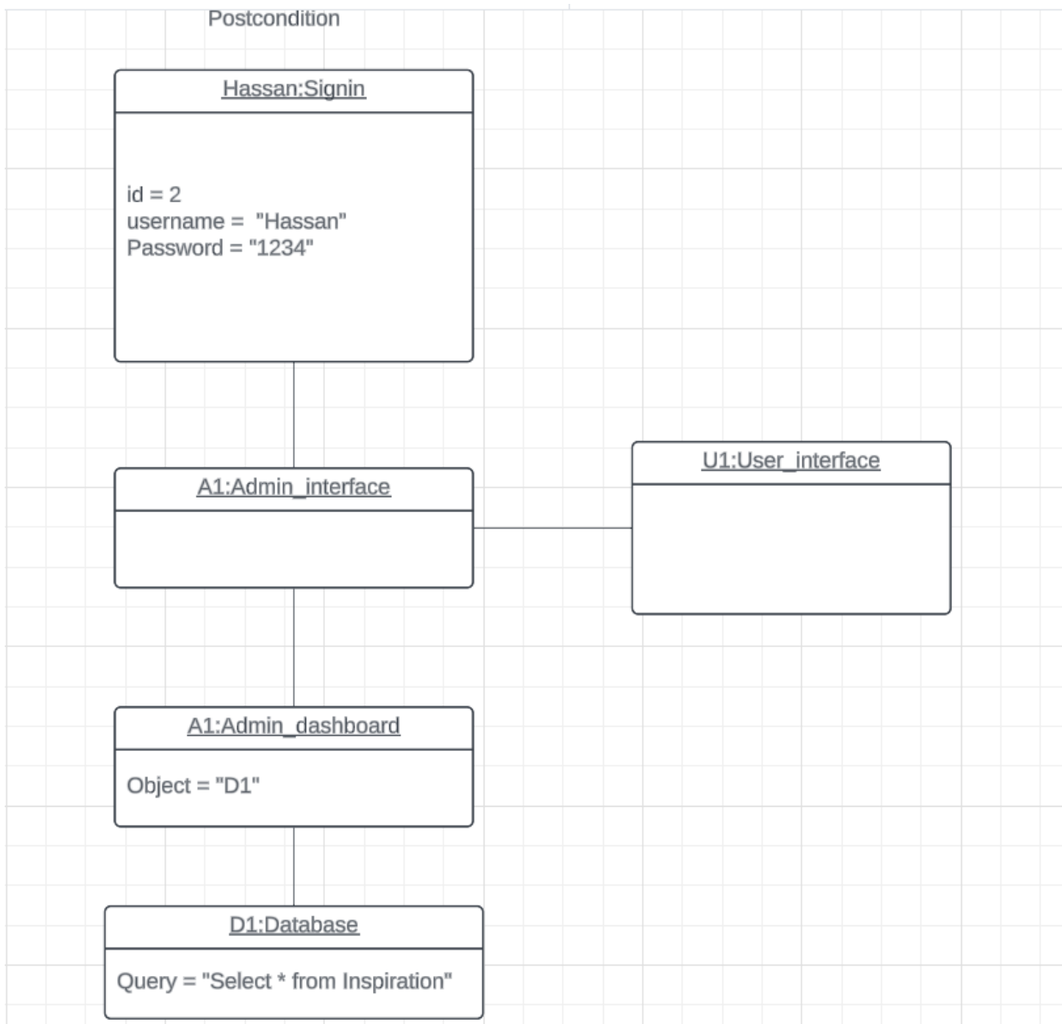

Postcondition
Ahmed:Signin
username = NULL
Password = NULL
Phone = NULL
Email = NULL
S1:Stream
Makka =
"https://www.youtube.com/watch?v=moQtMet7F7w"
Meeting =
"https://meet.google.com/akc-nxsx-qgw"
U1:User_interface
L1:Library


Postcondition
Ahmed:Signup
username = "Ahmed"
Password = "1234"
Phone = "01111111111"
Email = "ahmed@gmail.com"
Ahmed:Signin
username = "Ahmed"
Password = "1234"
Phone = "01111111111"
Email = "ahmed@gmail.com"
U1:User_interface

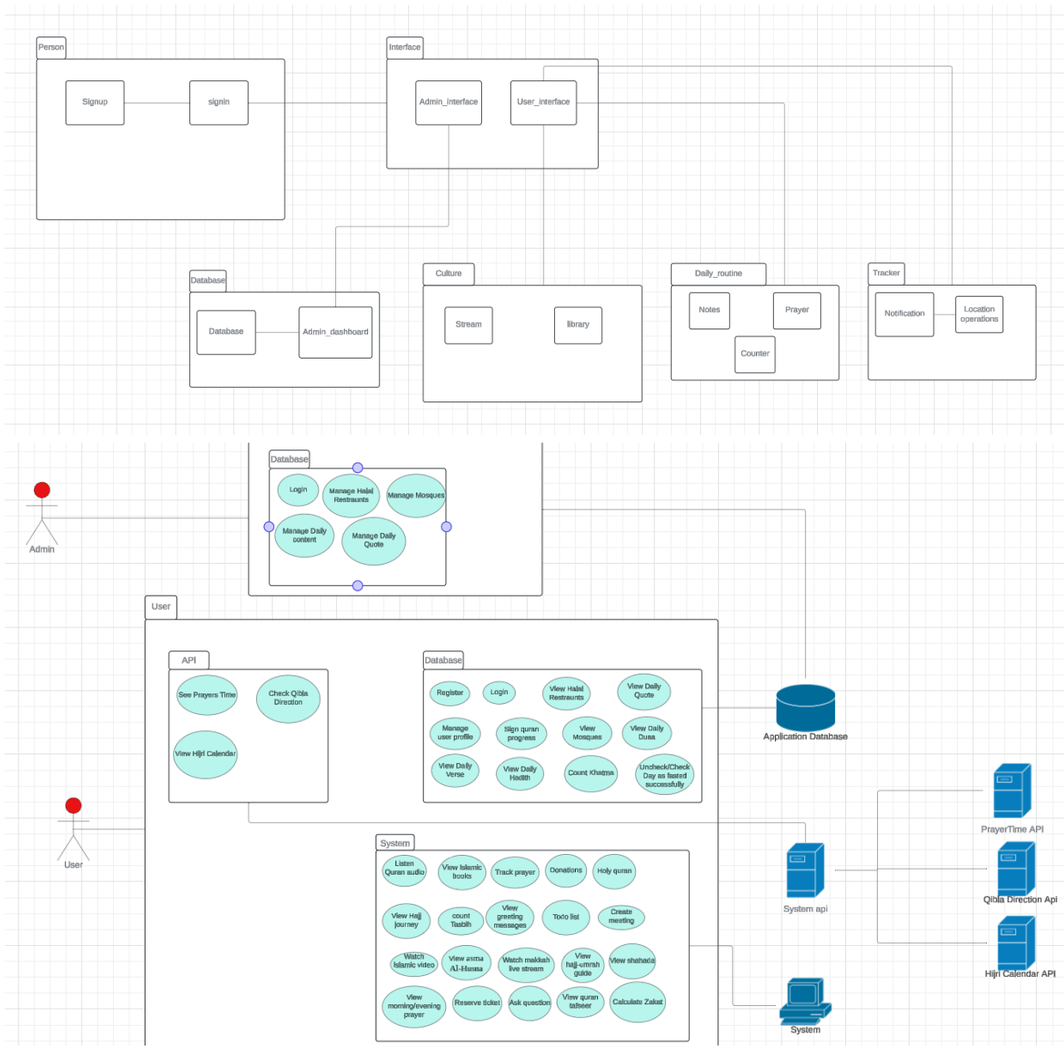

username = "Ahmed"
Password = "1234"
Phone = "01111111111"
Email = "ahmed@gmail.com"
Ahmed:Signin
username = "Ahmed"
Password = "1234"
Phone = "01111111111"
Email = "ahmed@gmail.com"
C1:Counter
Quran = "Juz 1 Hezb 1"
U1:User_interface
N1:Notes
Content =NULL
Content_check = False
p1:Prayer
Content =NULL
Content_check = False
C1:Location_operations
City = "Cairo"
Country= "Egypt"
Prayer_times= "Fajr"
N1:Notification
Prayer_text = "Now is fajr prayer"
Remember_prophet = "Remember the prophet"
S1:Stream
Makka =
"https://www.youtube.com/watch?v=moQdMet7F7w"
Meeting = NULL
U1:User_profile
L1:Library


Ahmed:Signup
username = "Ahmed"
Password = "1234"
Phone = "01111111111"
Email = "ahmed@gmail.com"
Ahmed:Signin
username = "Ahmed"
Password = "1234"
Phone = "01111111111"
Email = "ahmed@gmail.com"
C1:Counter
Quran = "Juz 1 Hezb 3"
U1:User_interface
N1:Notes
Content = "I will visit my grandma"
Content_check = False
p1:Prayer
Content = "Fajr"
Content_check = False
C1:Location_operations
City = "Cairo"
Country= "Egypt"
Prayer_times= "Fajr"
N1:Notification
Prayer_text = "Now is fajr prayer"
Remember_prophet = "Remember the prophet"
S1:Stream
Makka =
"https://www.youtube.com/watch?v=moQtMot7F7w"
Meeting =
"https://meet.google.com/akc-nxsx-qgw"
L1:Library
UI:User_profile

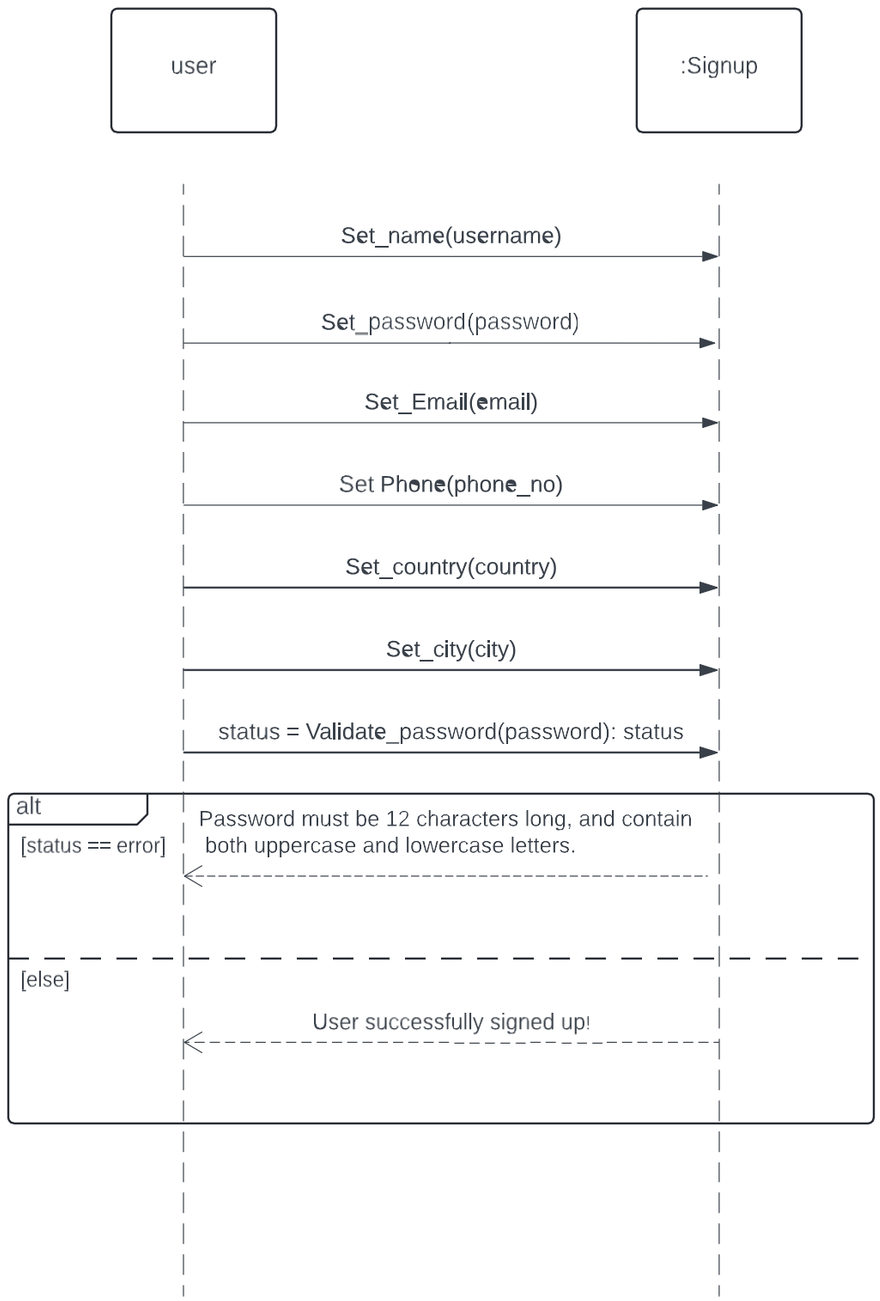

Precondition
Hassan:Signin
id = 2
username = "Hassan"
Password = "1234"
A1:Admin_interface
U1:User_interface
A1:Admin_dashboard
Object = NULL
D1:Database
Query = NULL

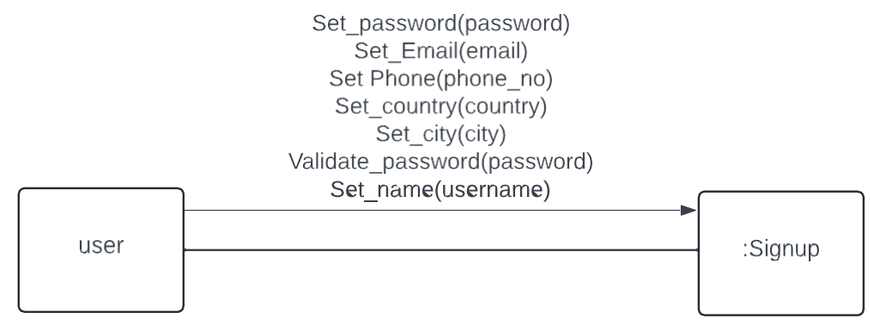

Postcondition
Hassan:Signin
id = 2
username = "Hassan"
Password = "1234"
A1:Admin_interface
U1:User_interface
A1:Admin_dashboard
Object = "D1"
D1:Database
Query = "Select * from Inspiration"

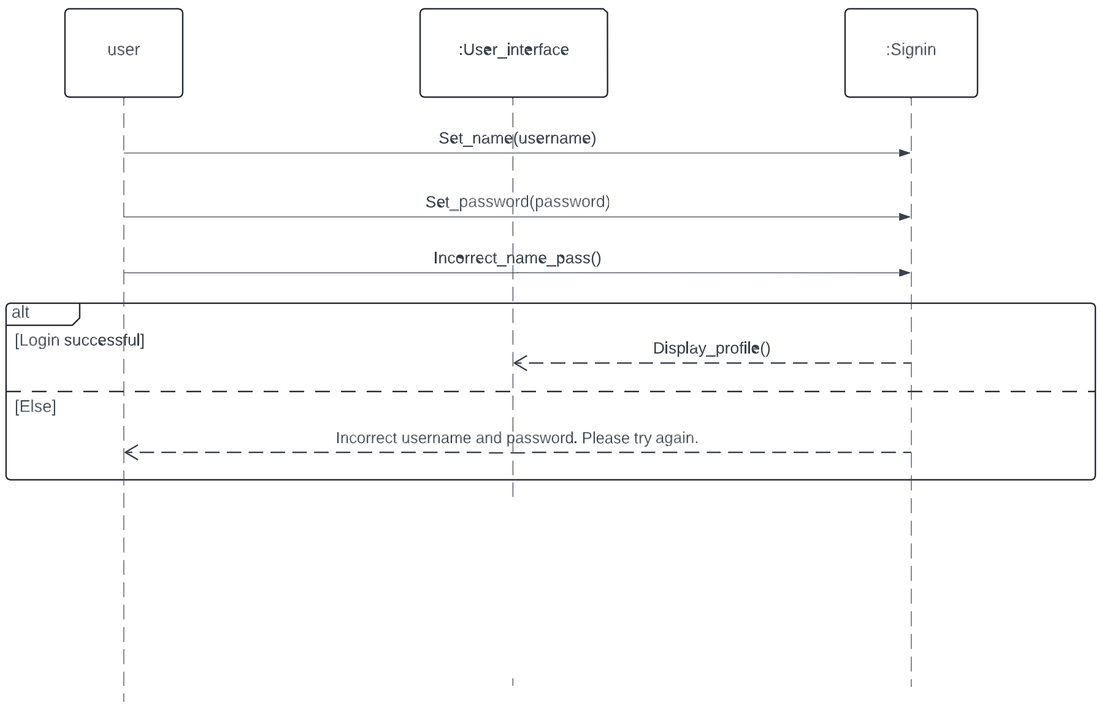
7-Package diagram



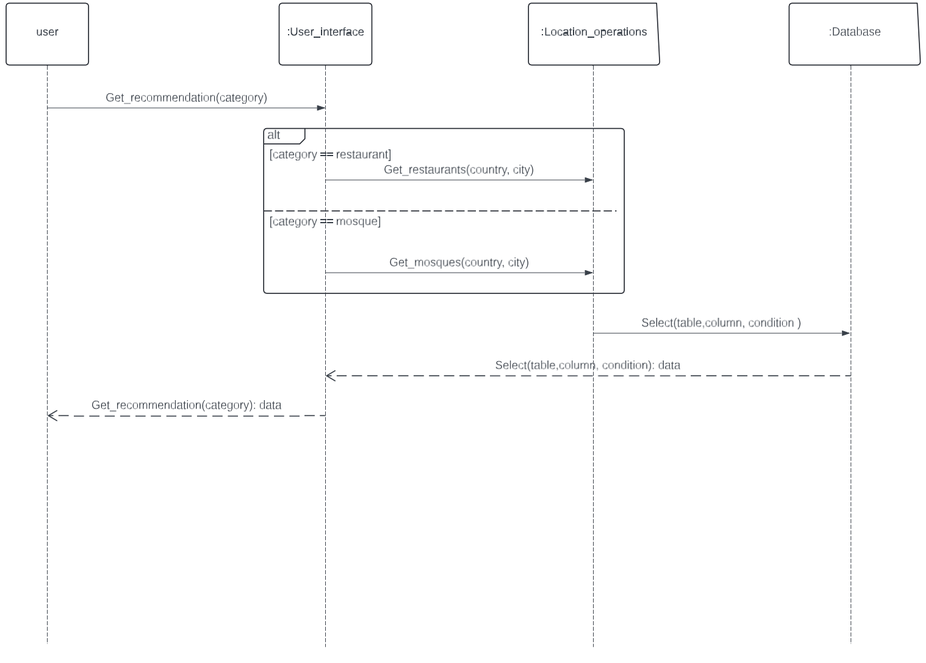
8- Sequence diagram and collaboration



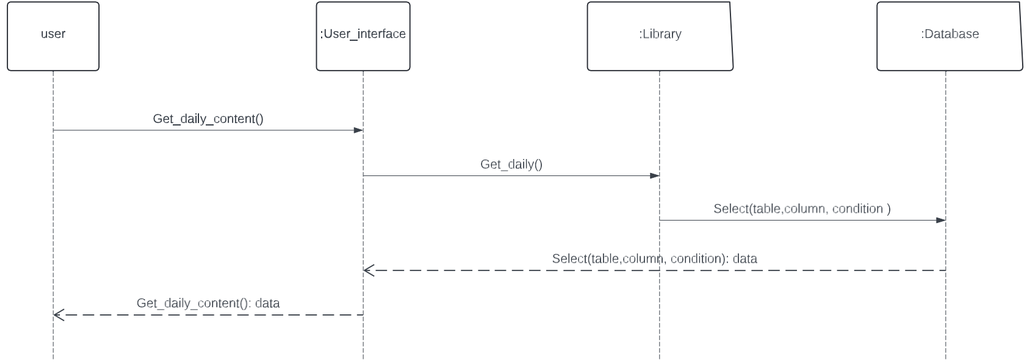



user
:User_interface
:Signin
Set_name(username)
Set_password(password)
Incorrect_name_pass()
alt
[Login successful]
Display_profile()
[Else]
Incorrect username and password. Please try again.

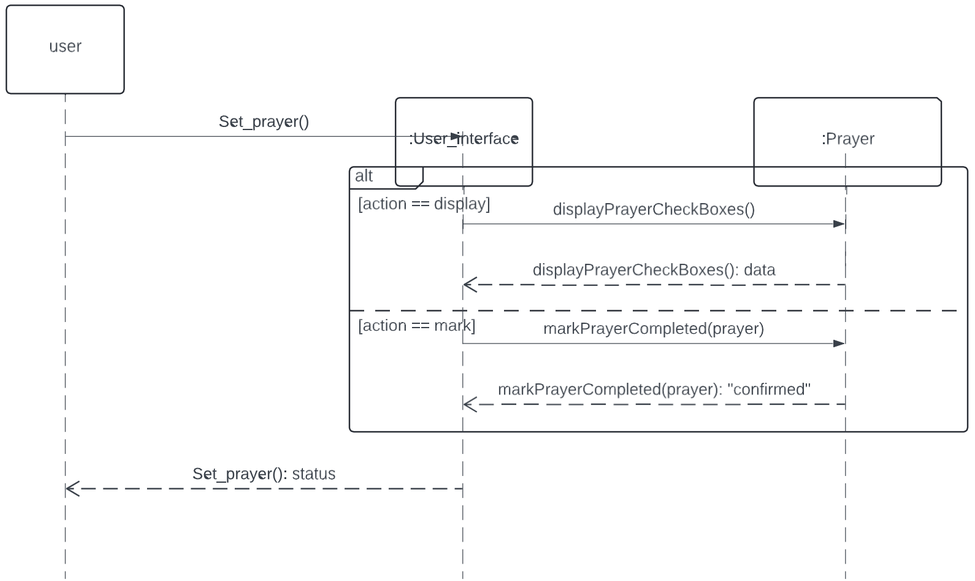
user
Set_name(username)
Set_password(password)
Incorrect_name_pass()
:Signin
Display_profile()
:User_interface

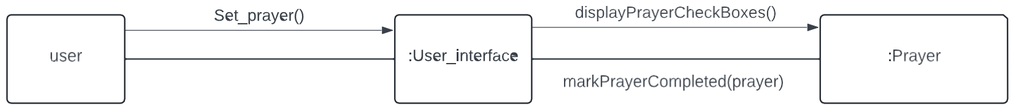

user
:User_interface
:Location_operations
:Database
Get_recommendation(category)
alt
[category == restaurant]
Get_restaurants(country, city)
[category == mosque]
Get_mosques(country, city)
Select(table,column, condition)
Select(table,column, condition): data
Get_recommendation(category): data

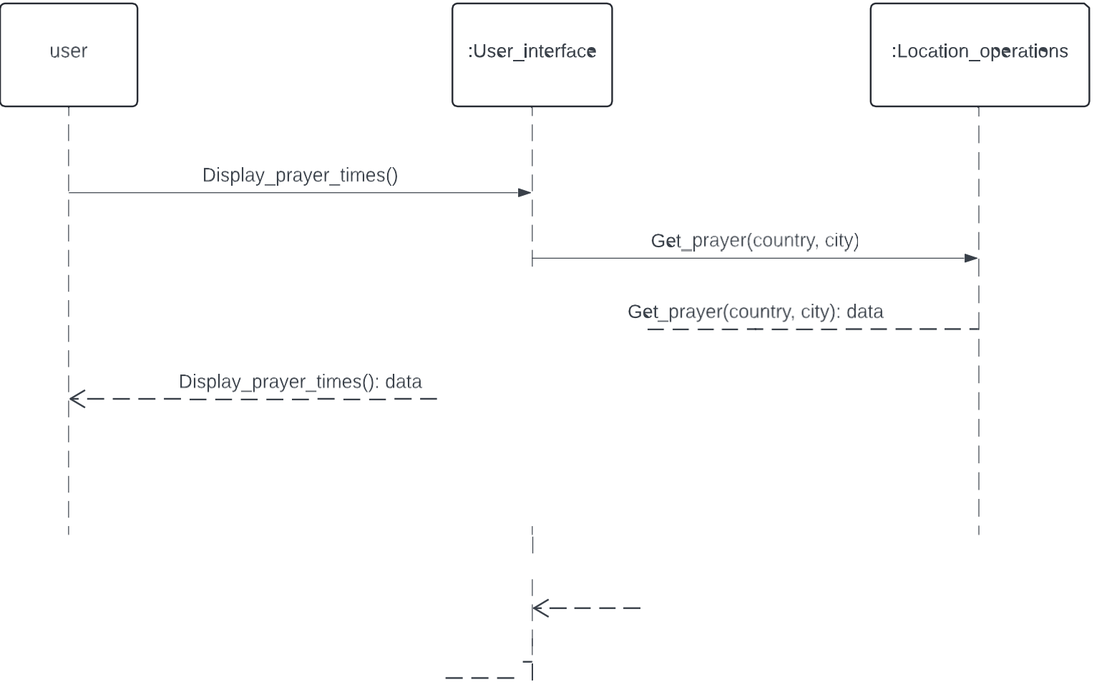
user
Get_recommendation(category)
:User_interface
Get_mosques(country, city)
Get_restaurants(country, city)
:Location_operations
Select(table,column,condition)
:Database


user
:User_interface
:Library
:Database
Get_daily_content()
Get_daily()
Select(table,column, condition)
Select(table,column, condition): data
Get_daily_content(): data

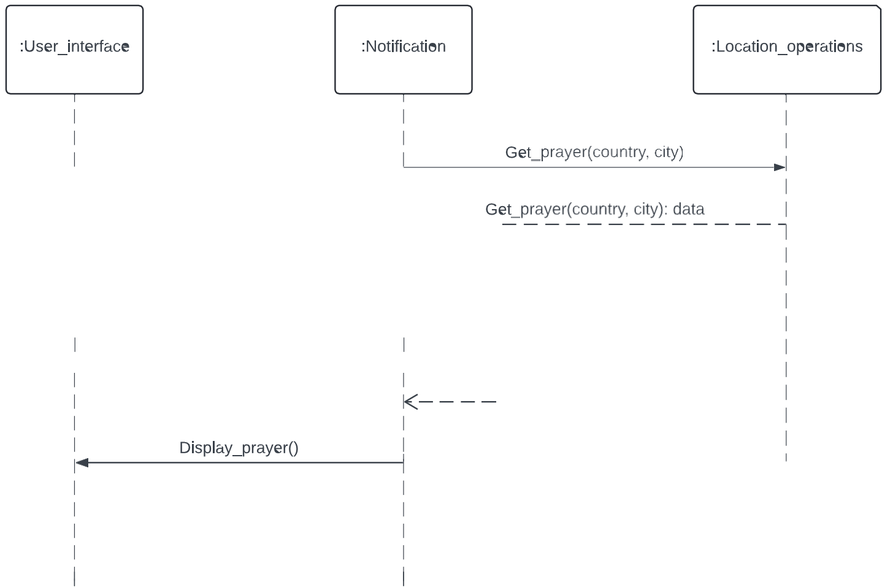
user
Get_daily_content()
:User_interface
Get_daily()
:Library
Select(table,column,condition)
:Database


user
:User_interface
:Prayer
Set_prayer()
alt
[action == display]
displayPrayerCheckBoxes()
displayPrayerCheckBoxes(): data
[action == mark]
markPrayerCompleted(prayer)
markPrayerCompleted(prayer): "confirmed"
Set_prayer(): status

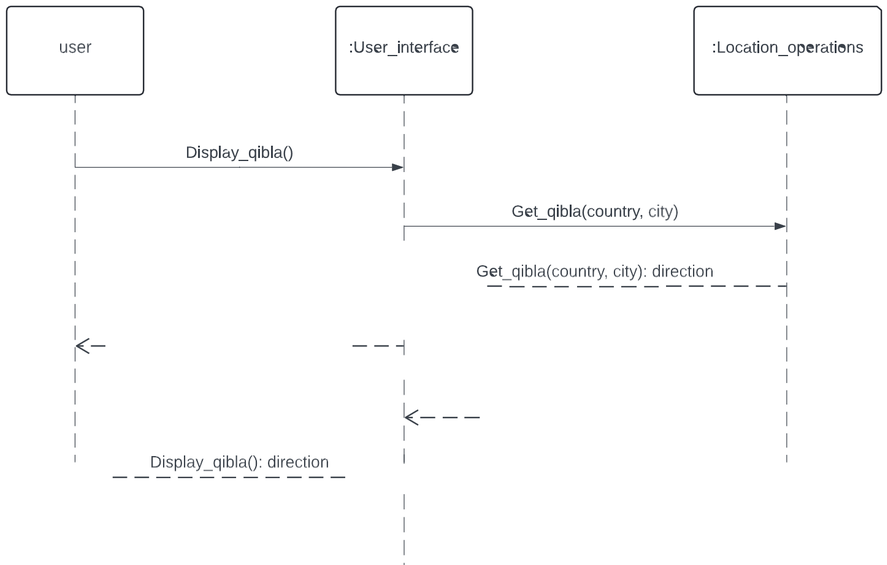

user
Set_prayer()
:User_interface
displayPrayerCheckBoxes()
markPrayerCompleted(prayer)
:Prayer

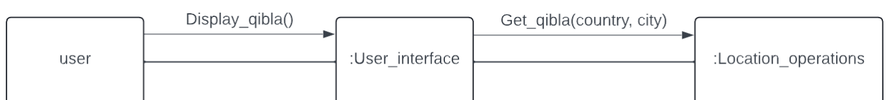

user
:User_interface
:Location_operations
Display_prayer_times()
Get_prayer(country, city)
Get_prayer(country, city): data
Display_prayer_times(): data

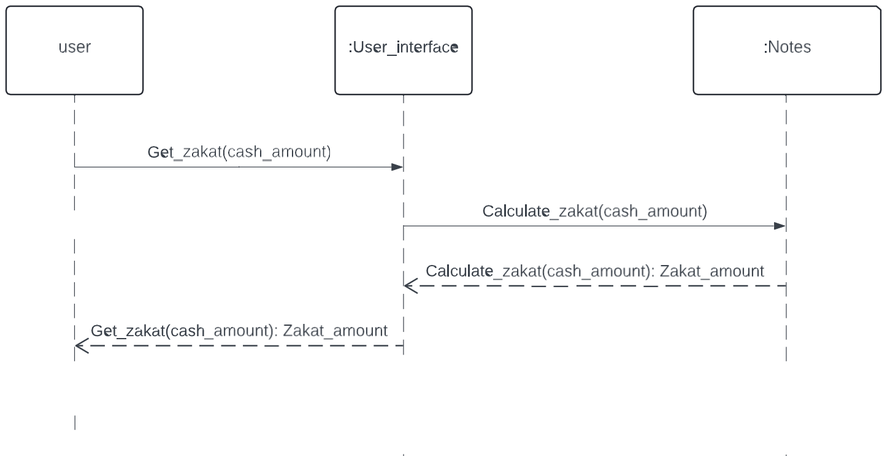
user
Display_prayer_times()
:User_interface
Get_prayer(country, city)
:Location_operations

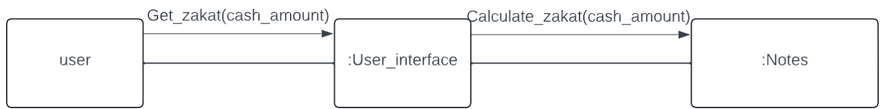

:User_interface
:Notification
:Location_operations
Get_prayer(country, city)
Get_prayer(country, city): data
Display_prayer()

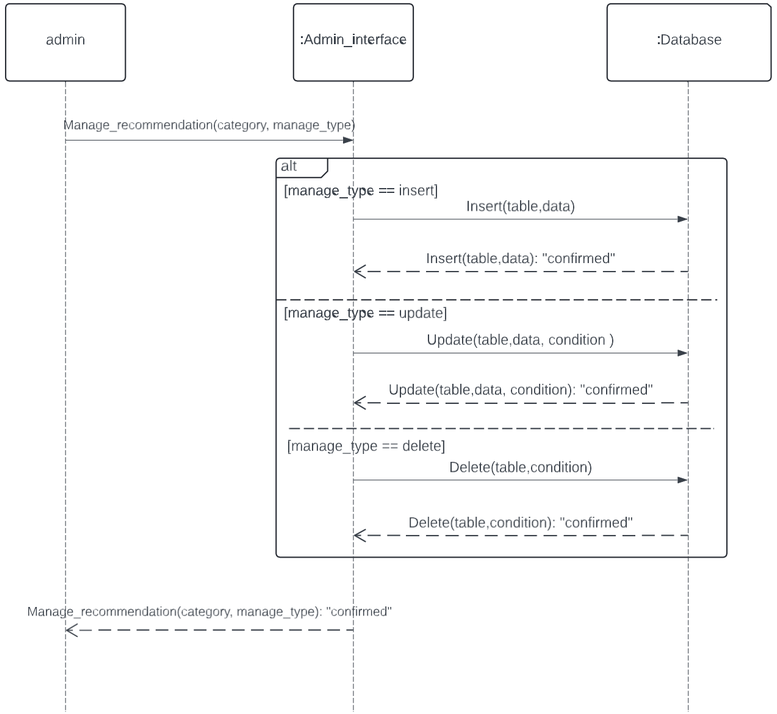
:User_interface
:Notification
:Location_operations
1. Get_prayer(country,city)
1.1. Display_prayer()

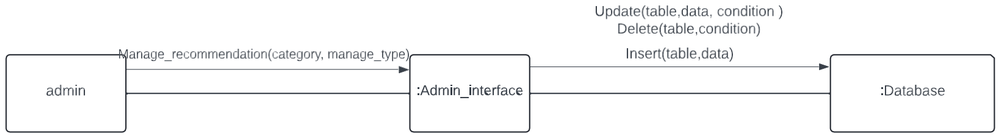

user
:User_interface
:Location_operations
Display_qibla()
Get_qibla(country, city)
Get_qibla(country, city): direction
Display_qibla(): direction

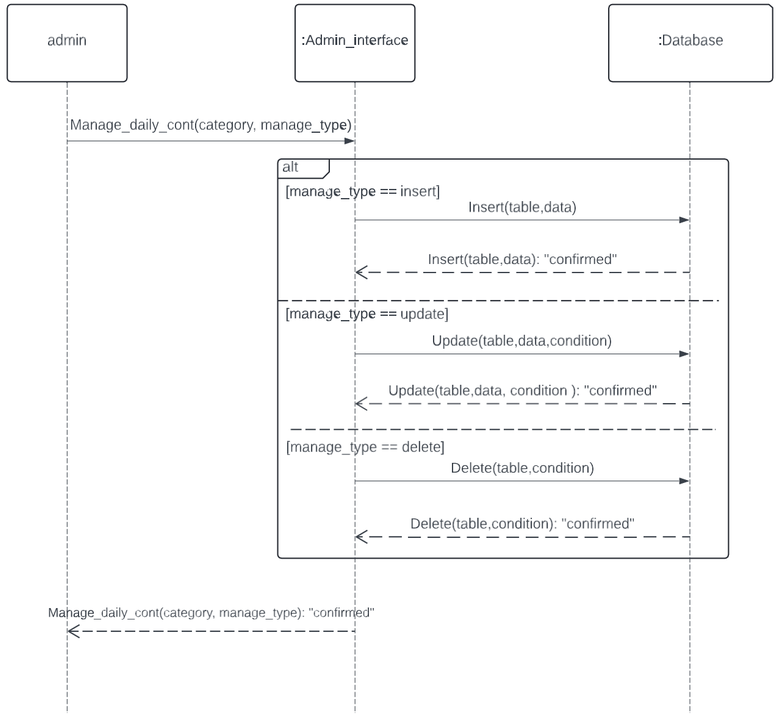

user
Display_qibla()
:User_interface
Get_qibla(country, city)
:Location_operations

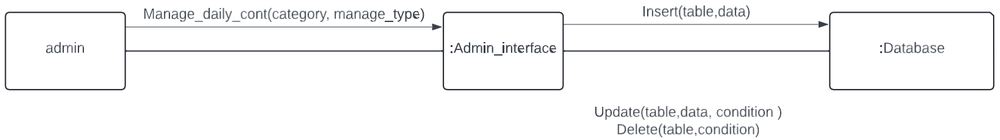

user
:User_interface
:Notes
Get_zakat(cash_amount)
Calculate_zakat(cash_amount)
Calculate_zakat(cash_amount): Zakat_amount
Get_zakat(cash_amount): Zakat_amount

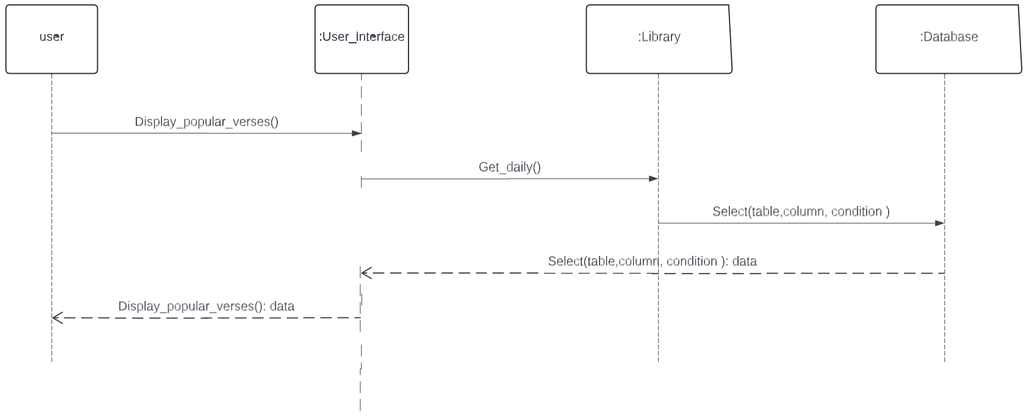

user
Get_zakat(cash_amount)
:User_interface
Calculate_zakat(cash_amount)
:Notes


admin
:Admin_interface
:Database
Manage_recommendation(category, manage_type)
alt
[manage_type == insert]
Insert(table,data)
Insert(table,data): "confirmed"
[manage_type == update]
Update(table,data, condition)
Update(table,data, condition): "confirmed"
[manage_type == delete]
Delete(table,condition)
Delete(table,condition): "confirmed"
Manage_recommendation(category, manage_type): "confirmed"

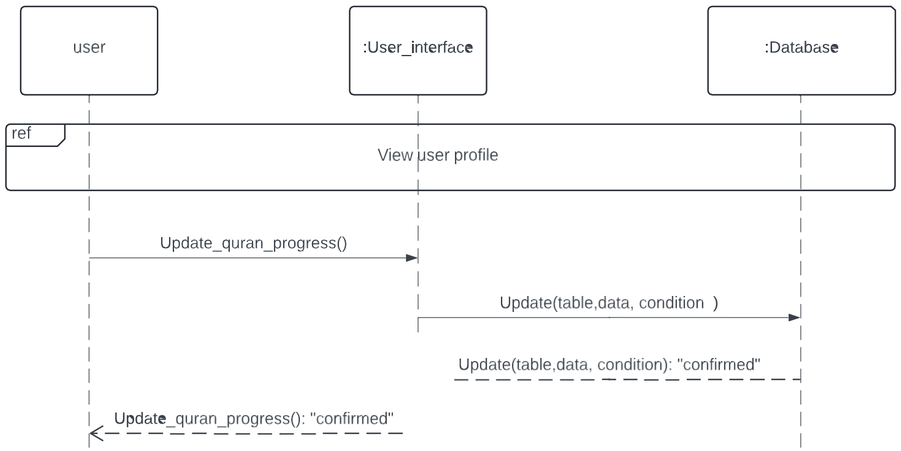

admin
Manage_recommendation(category, manage_type)
:Admin_interface
Update(table,data, condition)
Delete(table,condition)
Insert(table,data)
:Database

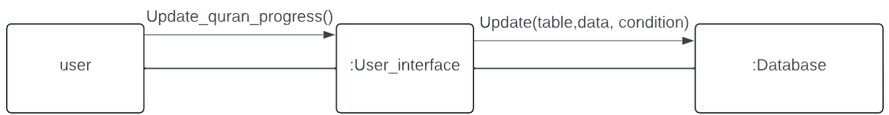

admin
:Admin_interface
:Database
Manage_daily_cont(category, manage_type)
alt
[manage_type == insert]
Insert(table,data)
Insert(table,data): "confirmed"
[manage_type == update]
Update(table,data,condition)
Update(table,data, condition): "confirmed"
[manage_type == delete]
Delete(table,condition)
Delete(table,condition): "confirmed"
Manage_daily_cont(category, manage_type): "confirmed"

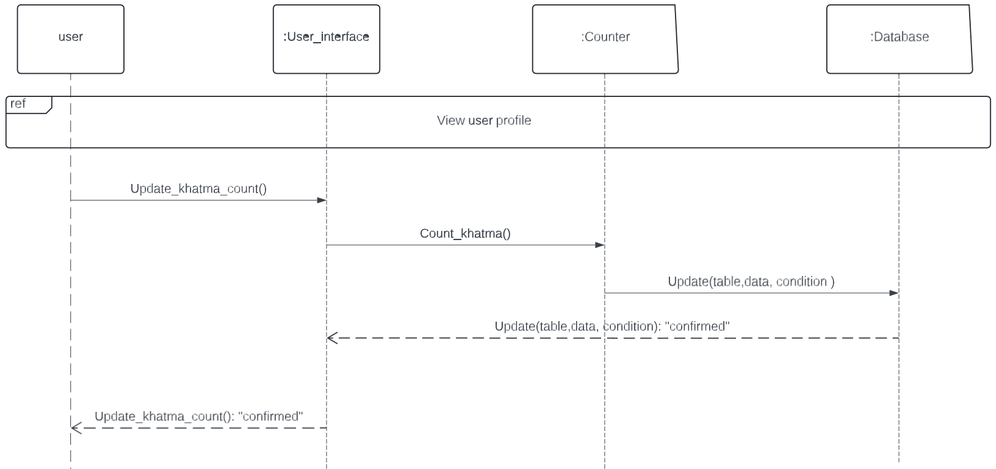

admin
Manage_daily_cont(category, manage_type)
:Admin_interface
Insert(table,data)
:Database
Update(table,data, condition)
Delete(table,condition)

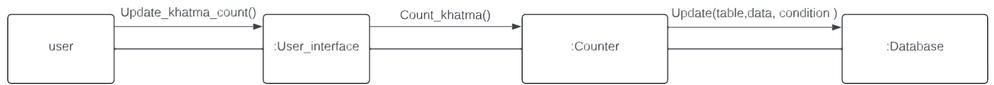

user
:User_interface
:Library
:Database
Display_popular_verses()
Get_daily()
Select(table,column, condition)
Select(table,column, condition): data
Display_popular_verses(): data

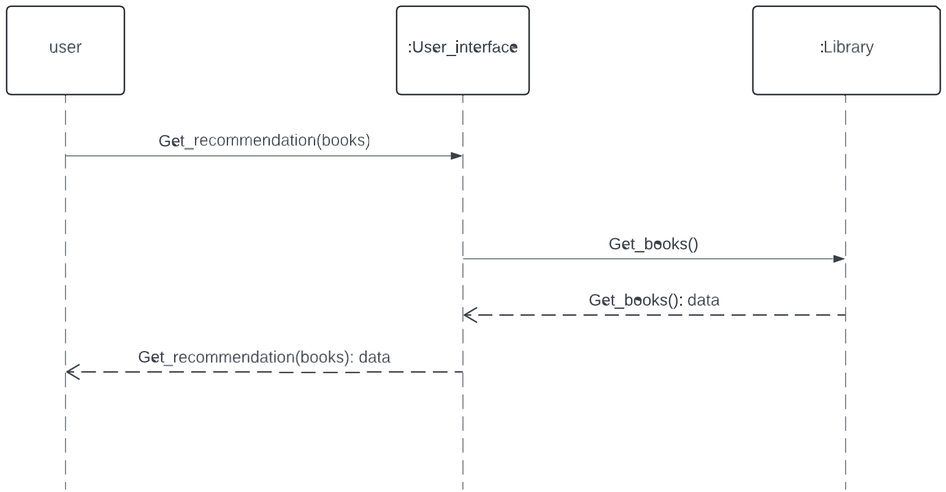
user
Display_popular_verses()
:User_interface
Get_daily()
:Library
Select(table,column, condition)
:Database


user
:User_interface
:Database
ref
View user profile
Update_quran_progress()
Update(table,data, condition)
Update(table,data, condition): "confirmed"
Update_quran_progress(): "confirmed"

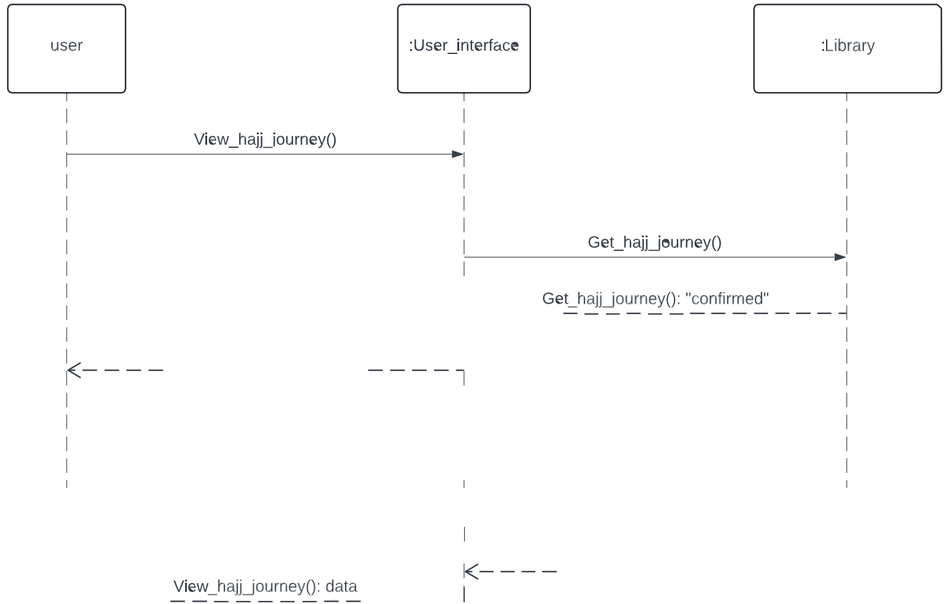

user
Update_quran_progress()
:User_interface
Update(table,data, condition)
:Database


user
:User_interface
:Counter
:Database
ref
View user profile
Update_khatma_count()
Count_khatma()
Update(table,data, condition)
Update(table,data, condition): "confirmed"
Update_khatma_count(): "confirmed"

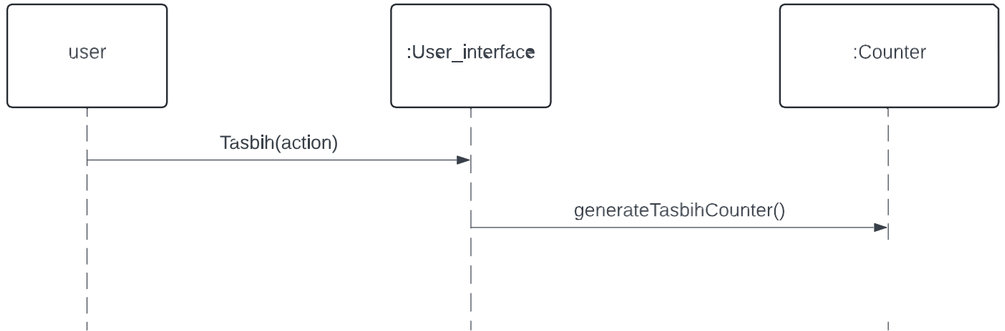

user
Update_khatma_count()
:User_interface
Count_khatma()
:Counter
Update(table,data, condition)
:Database

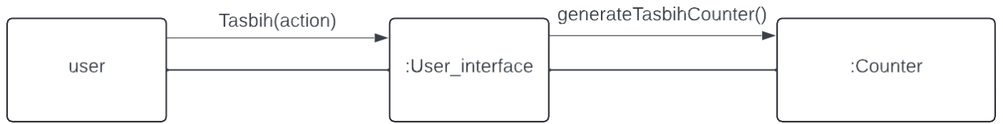

user
:User_interface
:Library
Get_recommendation(books)
Get_books()
Get_books(): data
Get_recommendation(books): data

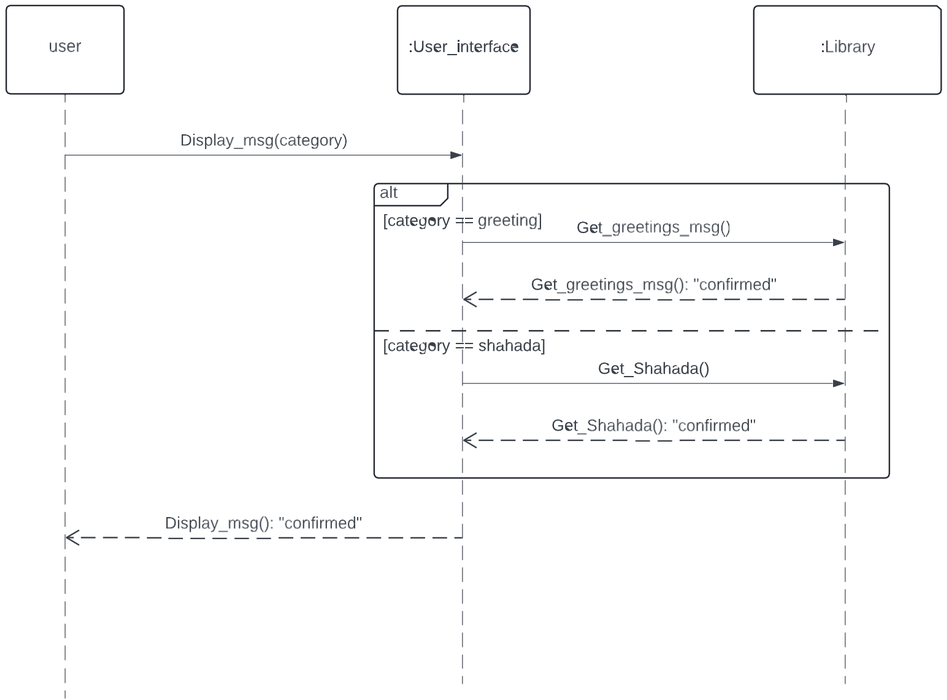
user
Get_recommendation(books)
:User_interface
Get_books()
:Library

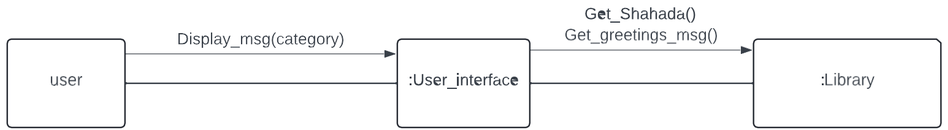

user
:User_interface
:Library
View_hajj_journey()
Get_hajj_journey()
Get_hajj_journey(): "confirmed"
View_hajj_journey(): data

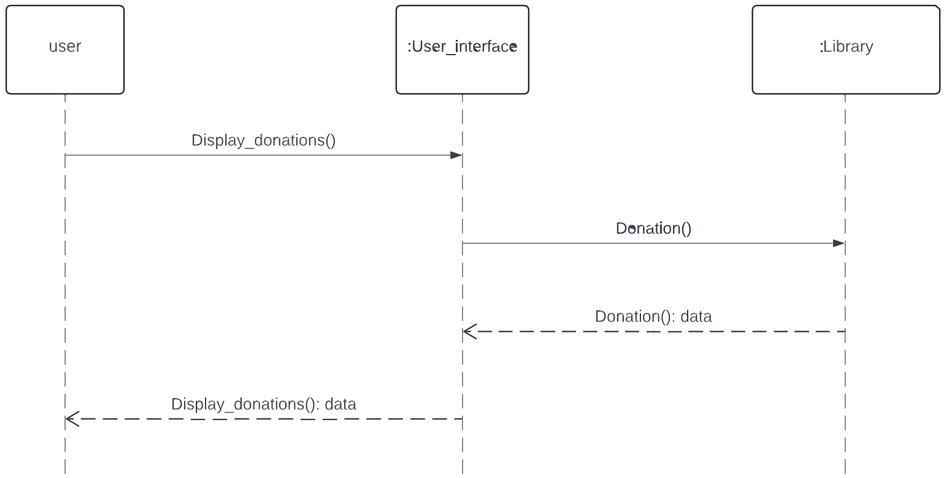
user
View_hajj_journey()
:User_interface
Get_hajj_journey()
:Library


user
:User_interface
:Counter
Tasbih(action)
generateTasbihCounter()

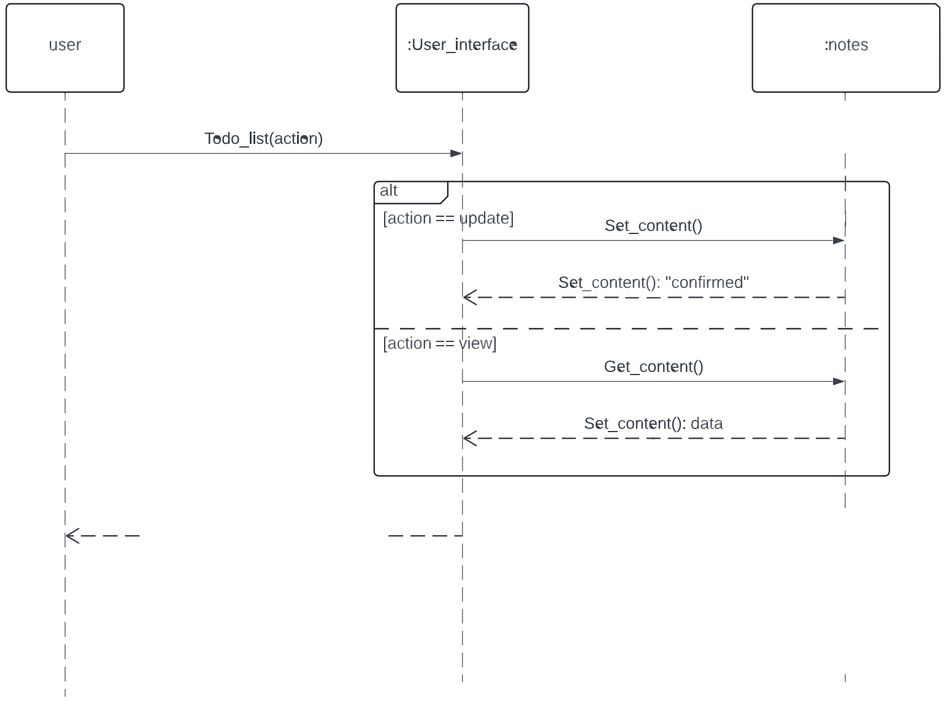

user
Tasbih(action)
:User_interface
generateTasbihCounter()
:Counter


user
:User_interface
:Library
Display_msg(category)
alt
[category == greeting]
Get_greetings_msg()
Get_greetings_msg(): "confirmed"
[category == shahada]
Get_Shahada()
Get_Shahada(): "confirmed"
Display_msg(): "confirmed"


user
Display_msg(category)
:User_interface
Get_Shahada()
Get_greetings_msg()
:Library


user
:User_interface
:Library
Display_donations()
Donation()
Donation(): data
Display_donations(): data

user
Display_donations()
:User_interface
Donation()
:Library


user
:User_interface
:notes
Todo_list(action)
alt
[action == update]
Set_content()
Set_content(): "confirmed"
[action == view]
Get_content()
Set_content(): data

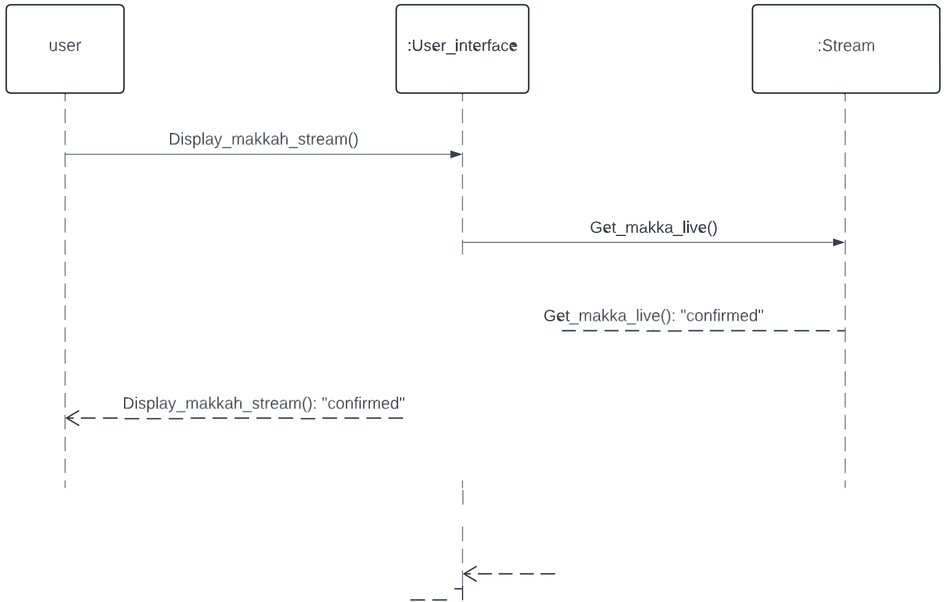
user
Todo_list(action)
:User_interface
Set_content()
Get_content()
:notes

Todo_list(action): "confirmed"

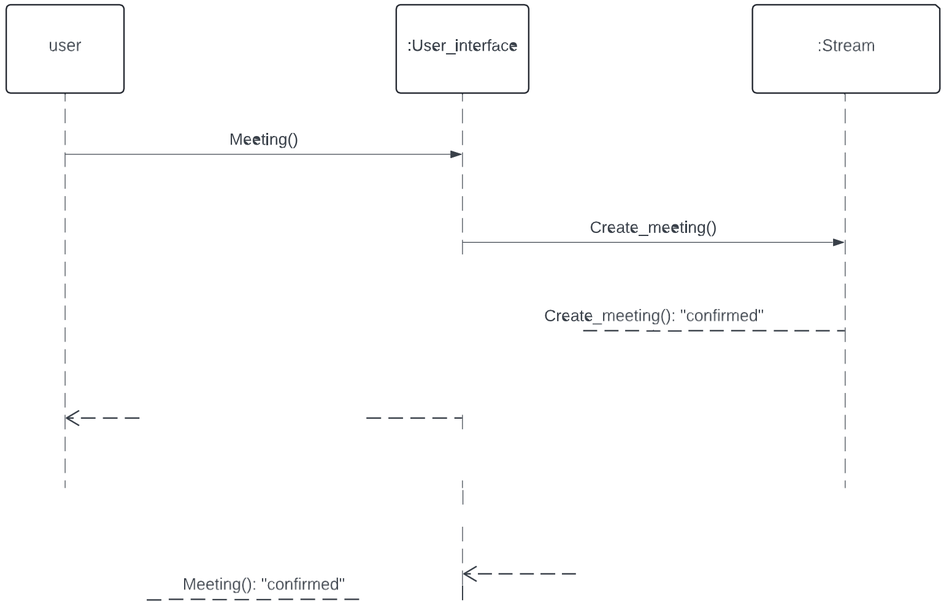
user
:User_interface
:Library
Display_asmaa_husna()
Get_asmaa_husna()
Get_asmaa_husna(): data

user
Display_asmaa_husna()
:User_interface
Get_asmaa_husna()
:Library

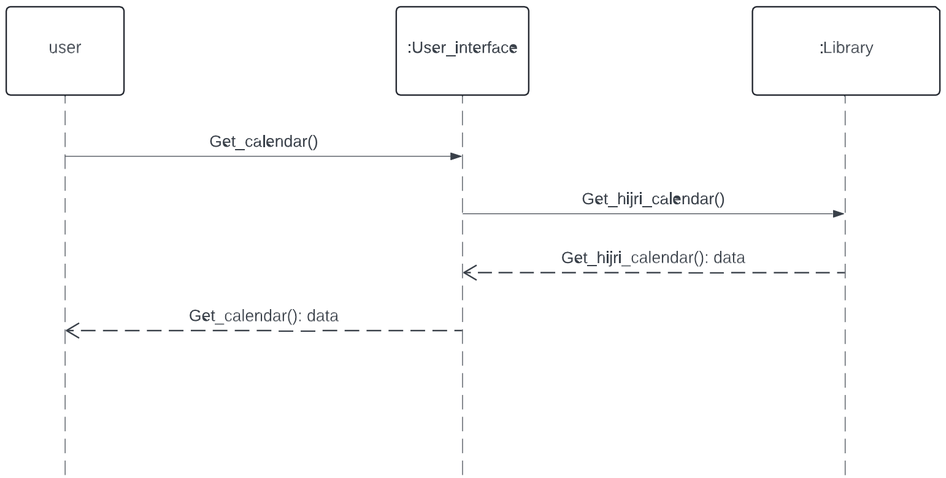
Display_asmaa_husna(): data


user
:User_interface
:Stream
Display_makkah_stream()
Get_makka_live()
Get_makka_live(): "confirmed"
Display_makkah_stream(): "confirmed"

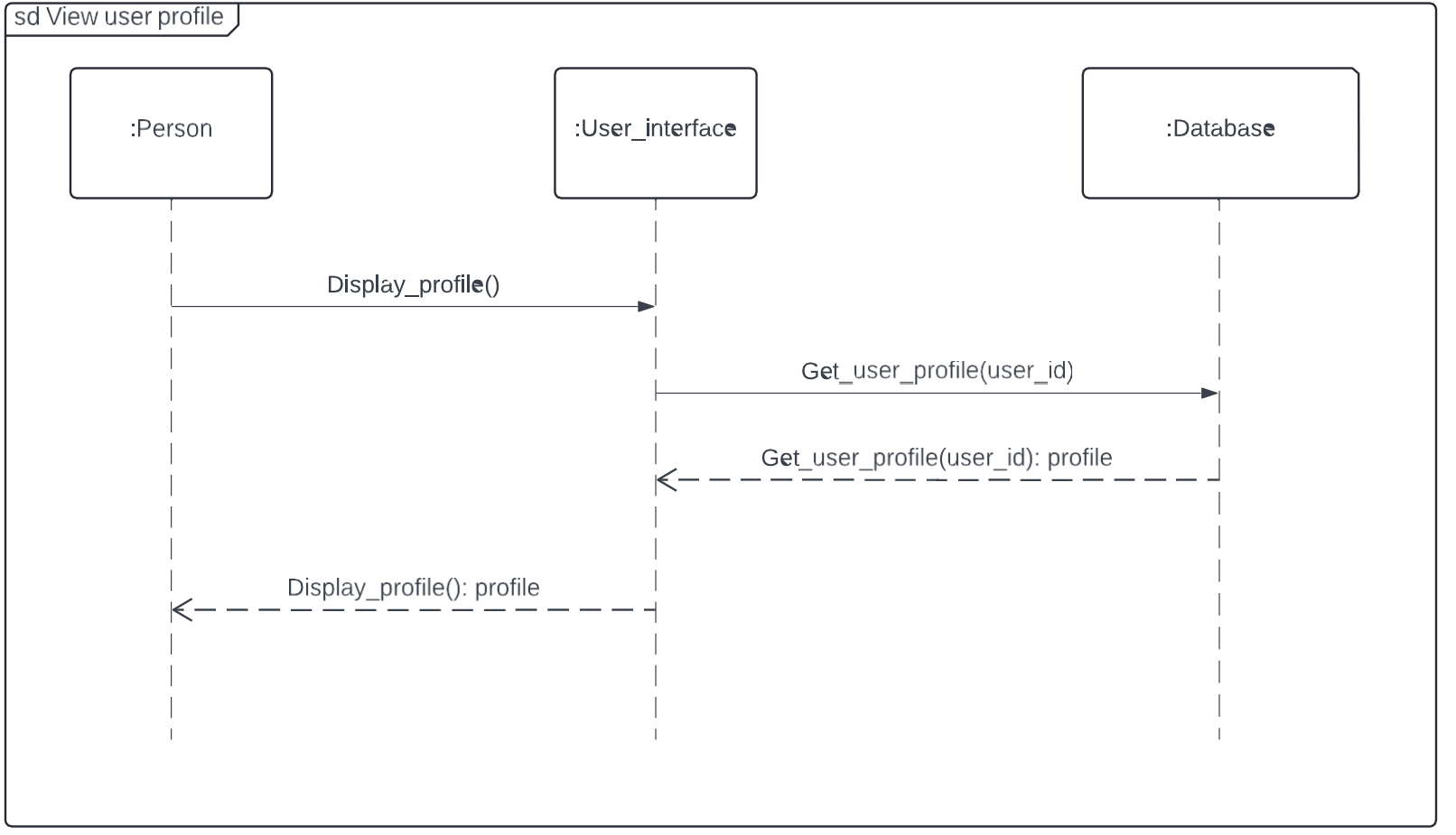
user
Display_makkah_stream()
:User_interface
Get_makka_live()
:Stream

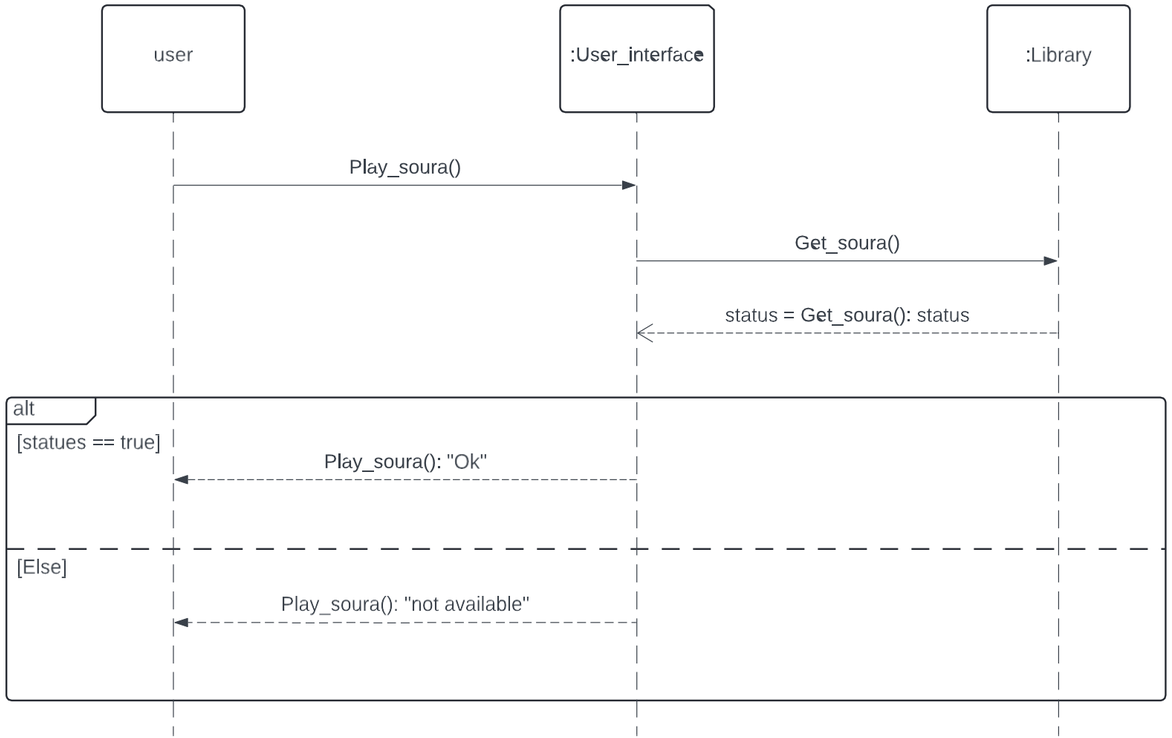

user
:User_interface
:Stream
Meeting()
Create_meeting()
Create_meeting(): "confirmed"
Meeting(): "confirmed"

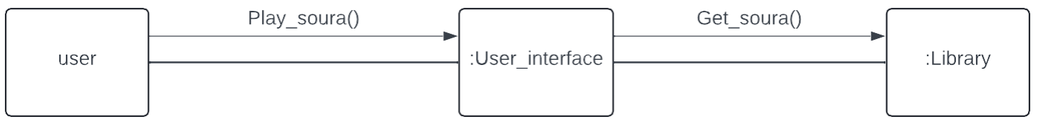
user
Meeting()
:User_interface
Create_meeting()
:Stream

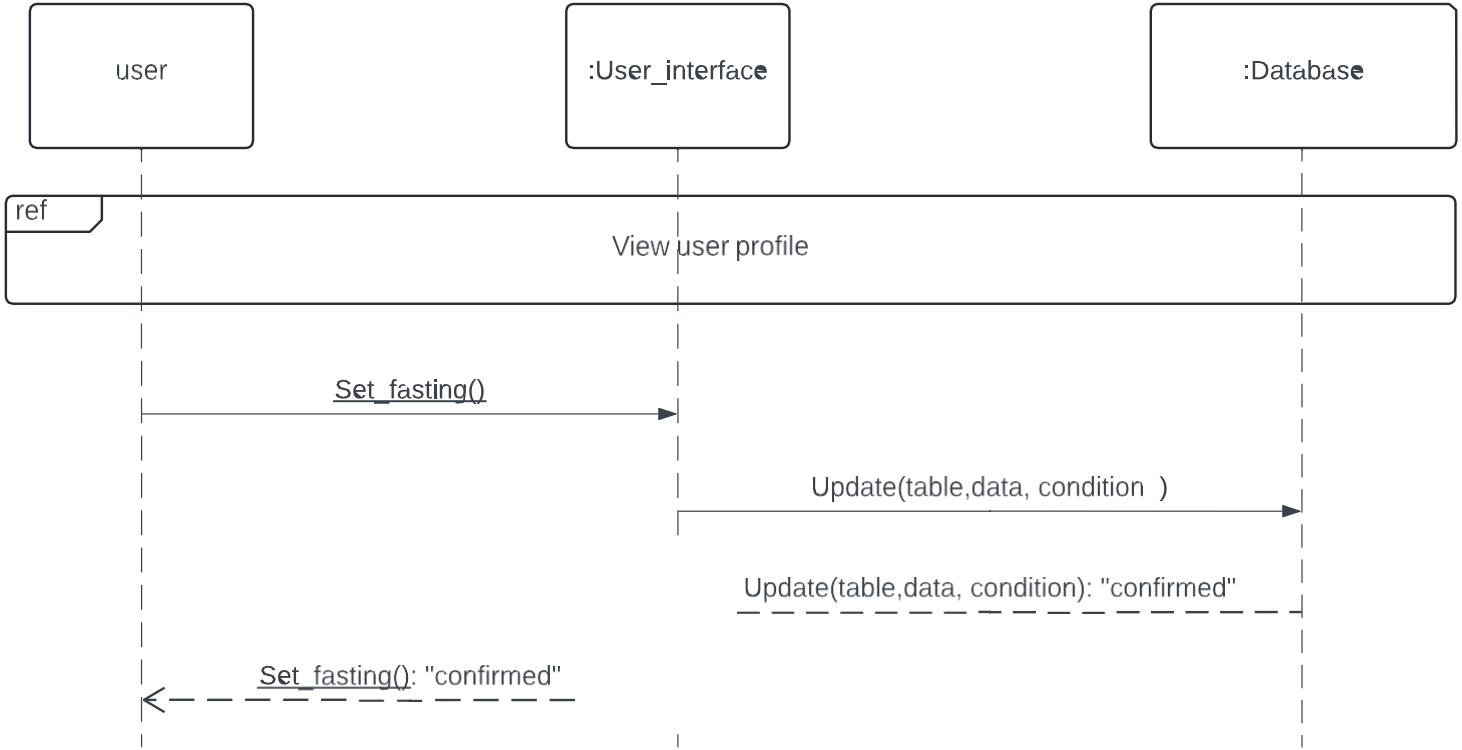

user
:User_interface
:Library
Get_calendar()
Get_hijri_calendar()
Get_hijri_calendar(): data
Get_calendar(): data

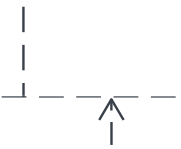
user
Get_calendar()
:User_interface
Get_hijri_calendar()
:Library


sd View user profile
:Person
:User_interface
:Database
Display_profile()
Get_user_profile(user_id)
Get_user_profile(user_id): profile
Display_profile(): profile

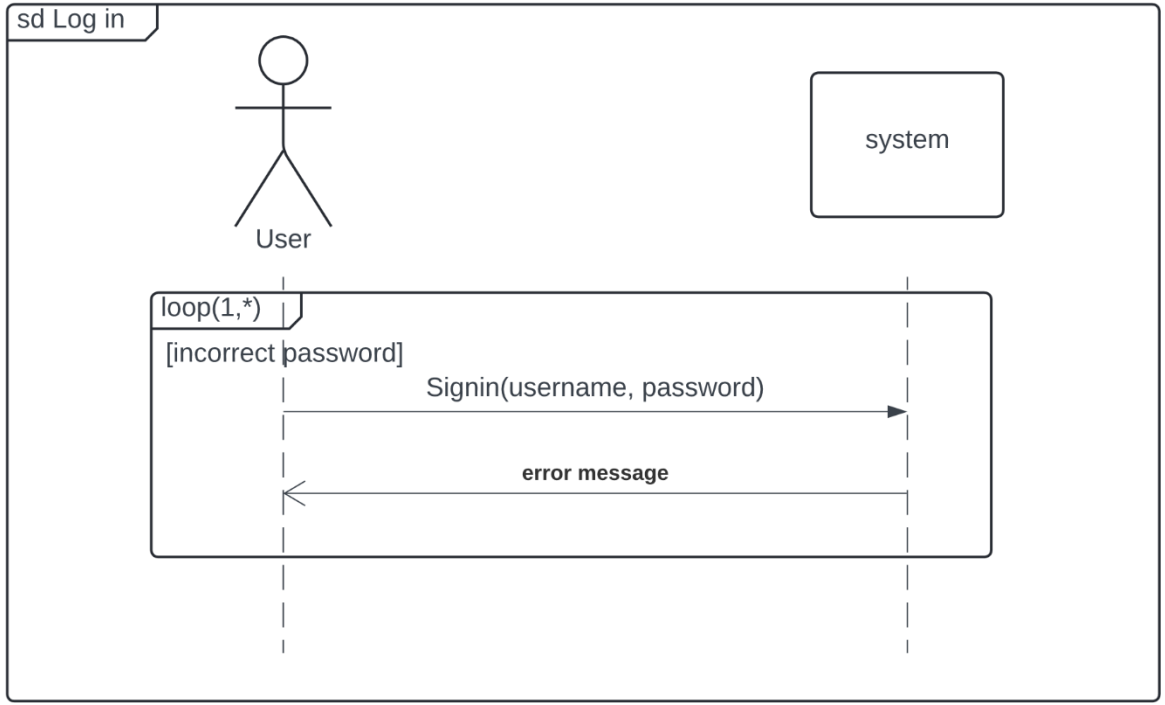

user
:User_interface
:Library
Play_soura()
Get_soura()
status = Get_soura(): status
alt
[statues == true]
Play_soura(): "Ok"
[Else]
Play_soura(): "not available"

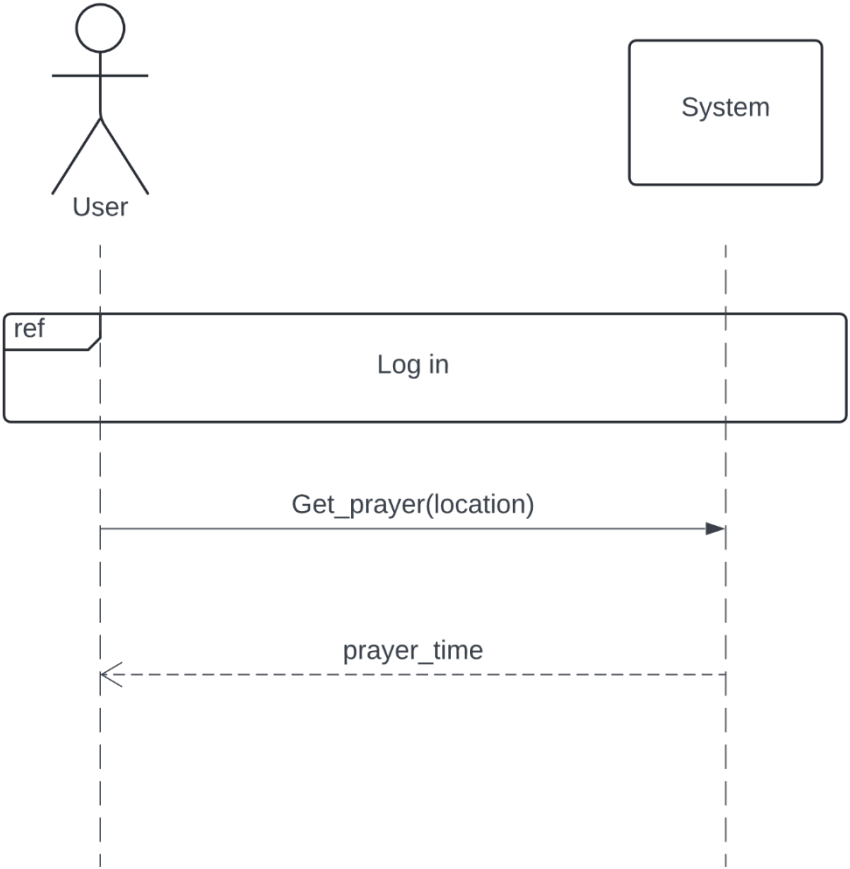

user
Play_soura()
:User_interface
Get_soura()
:Library

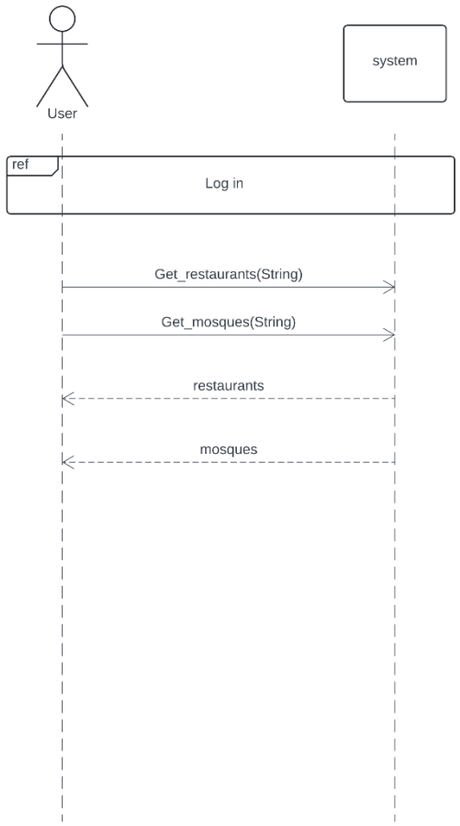

user
:User_interface
:Database
ref
View user profile
Set_fasting()
Update(table,data, condition)
Update(table,data, condition): "confirmed"
Set_fasting(): "confirmed"

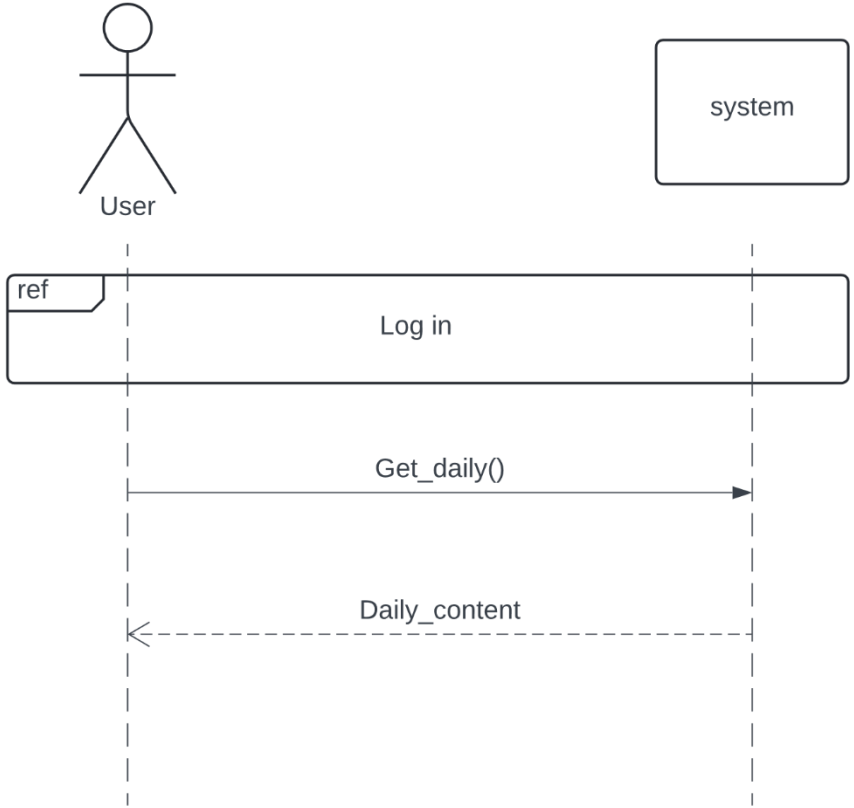


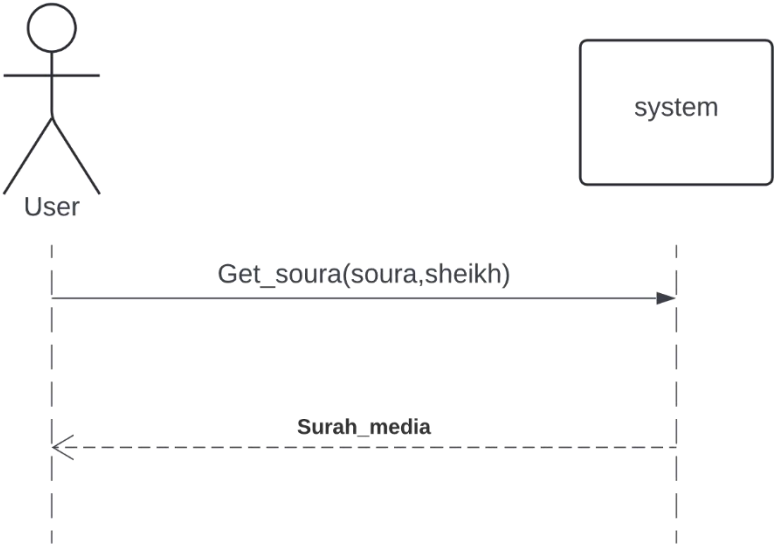
9-System sequence diagram

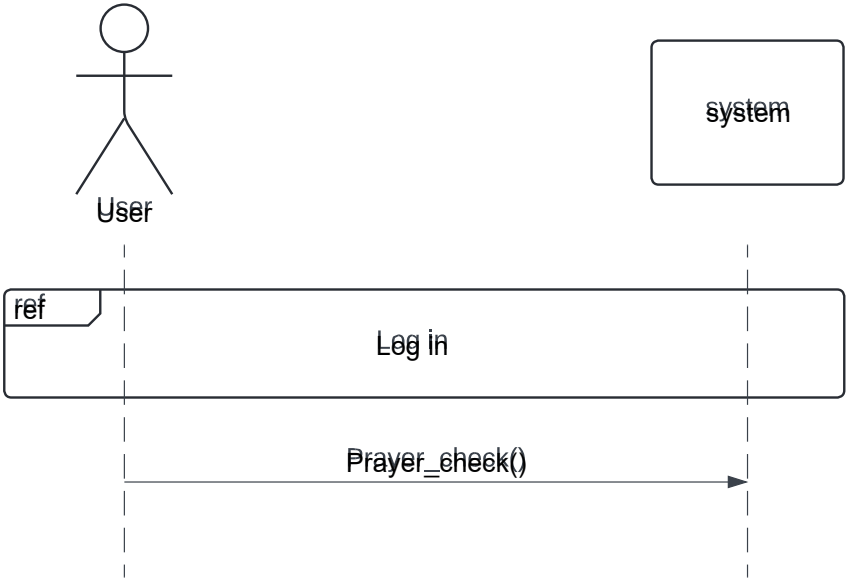

sd Log in
User
system
loop(1,*)
[incorrect password]
Signin(username, password)
error message

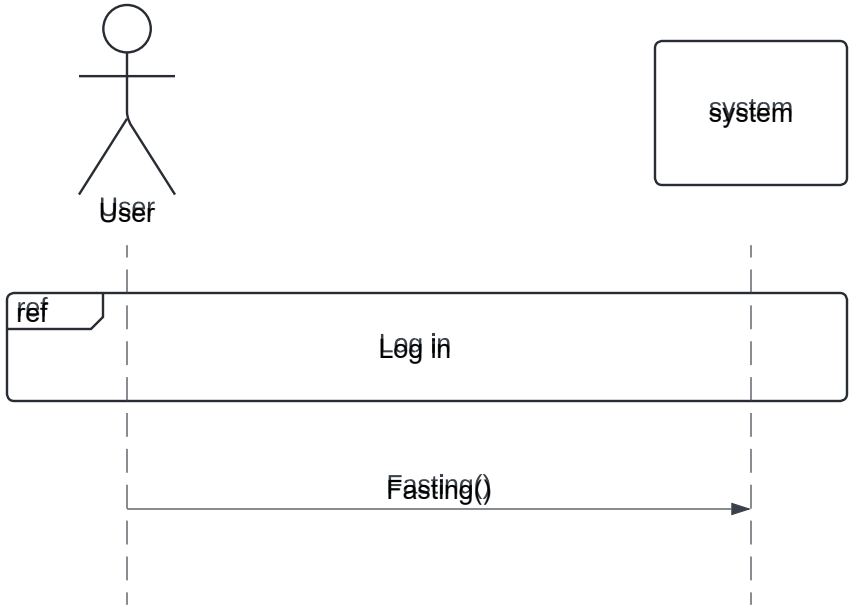

User
System
ref
Log in
Get_prayer(location)
prayer_time

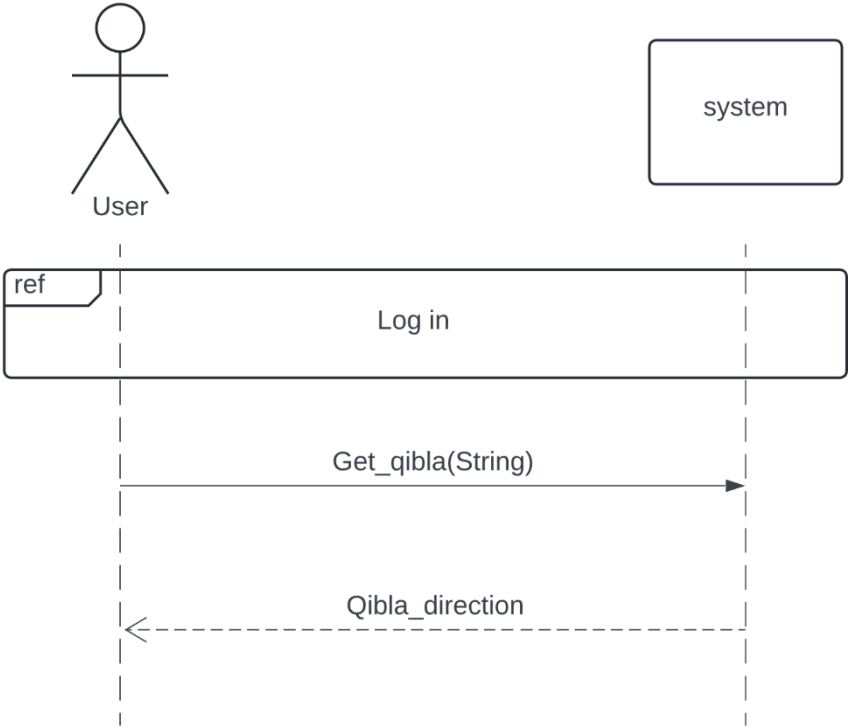

User
system
ref
Log in
Get_restaurants(String)
Get_mosques(String)
restaurants
mosques

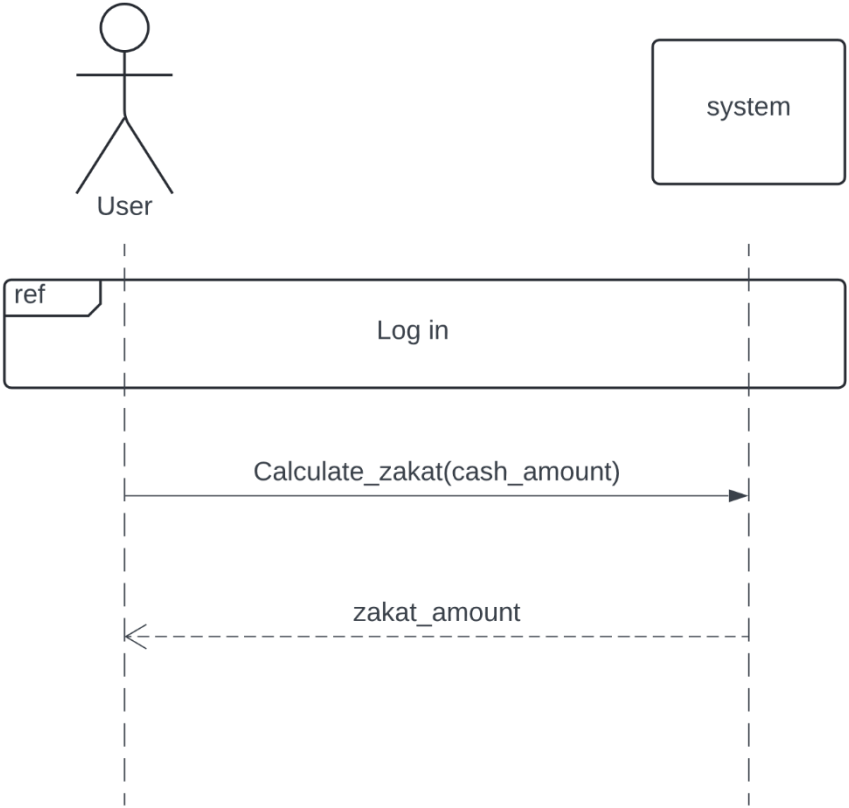

User
system
ref
Log in
Get_daily()
Daily_content

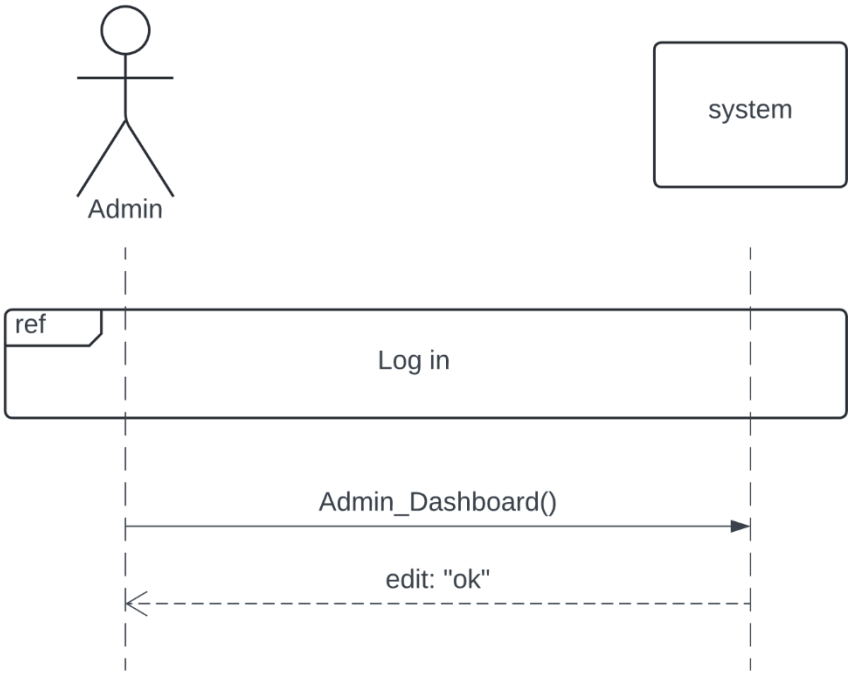

User
system
Get_soura(soura,sheikh)
Surah_media


User
system
ref
Log in
Prayer_check()


User
system
ref
Log in
Fasting()

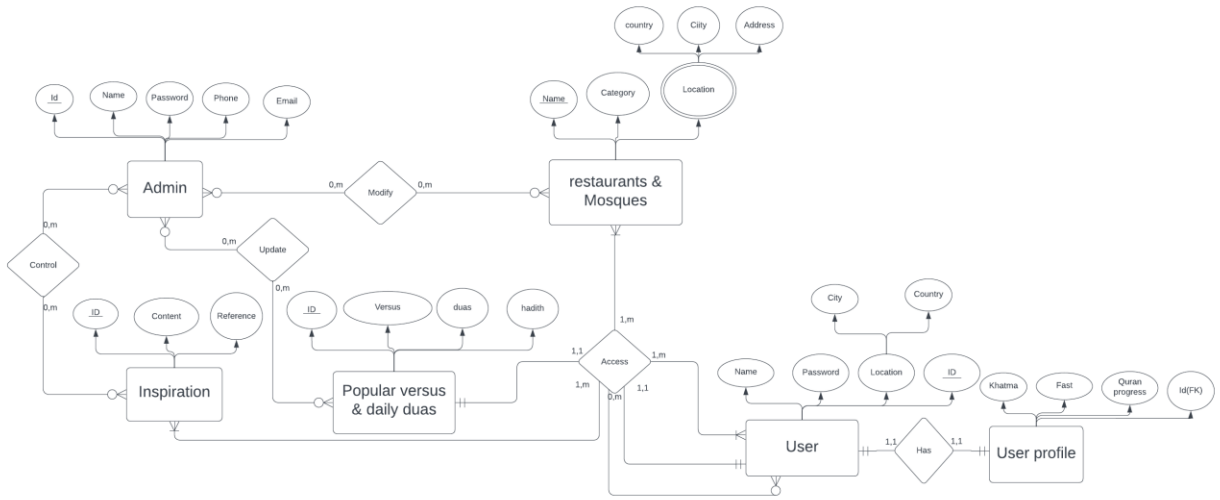

User
system
ref
Log in
Get_qibla(String)
Qibla_direction


User
system
ref
Log in
Calculate_zakat(cash_amount)
zakat_amount

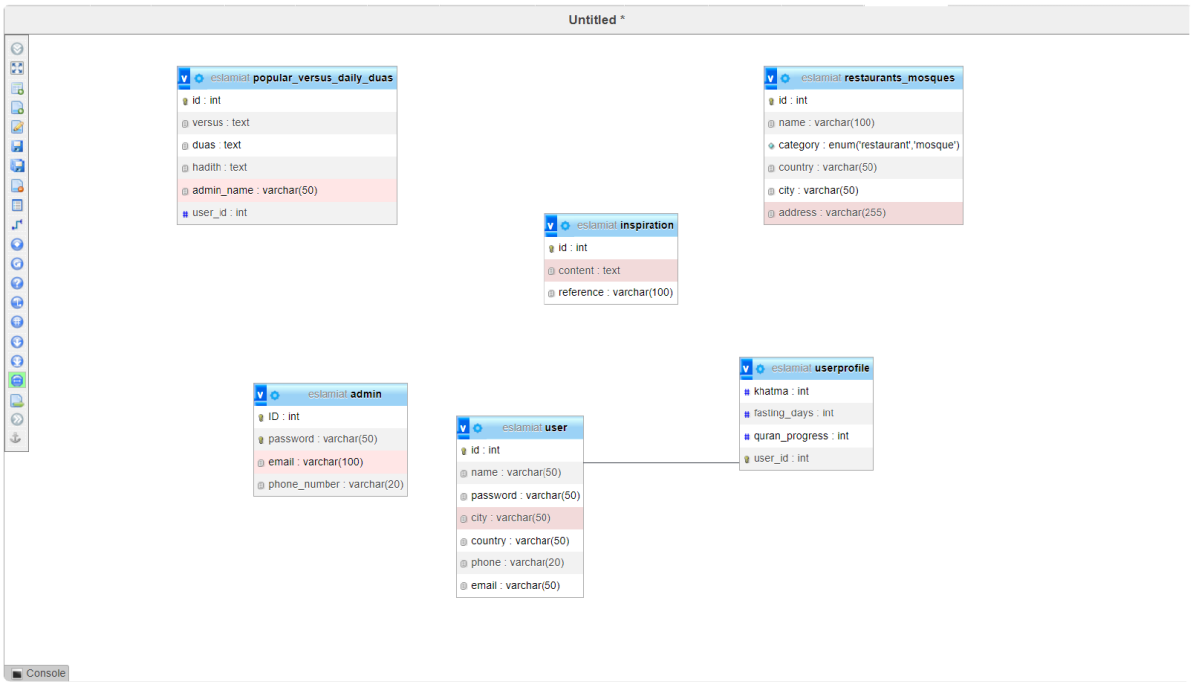

Admin
system
ref
Log in
Admin_Dashboard()
edit: "ok"

10. Database Specification

10.1 ERD

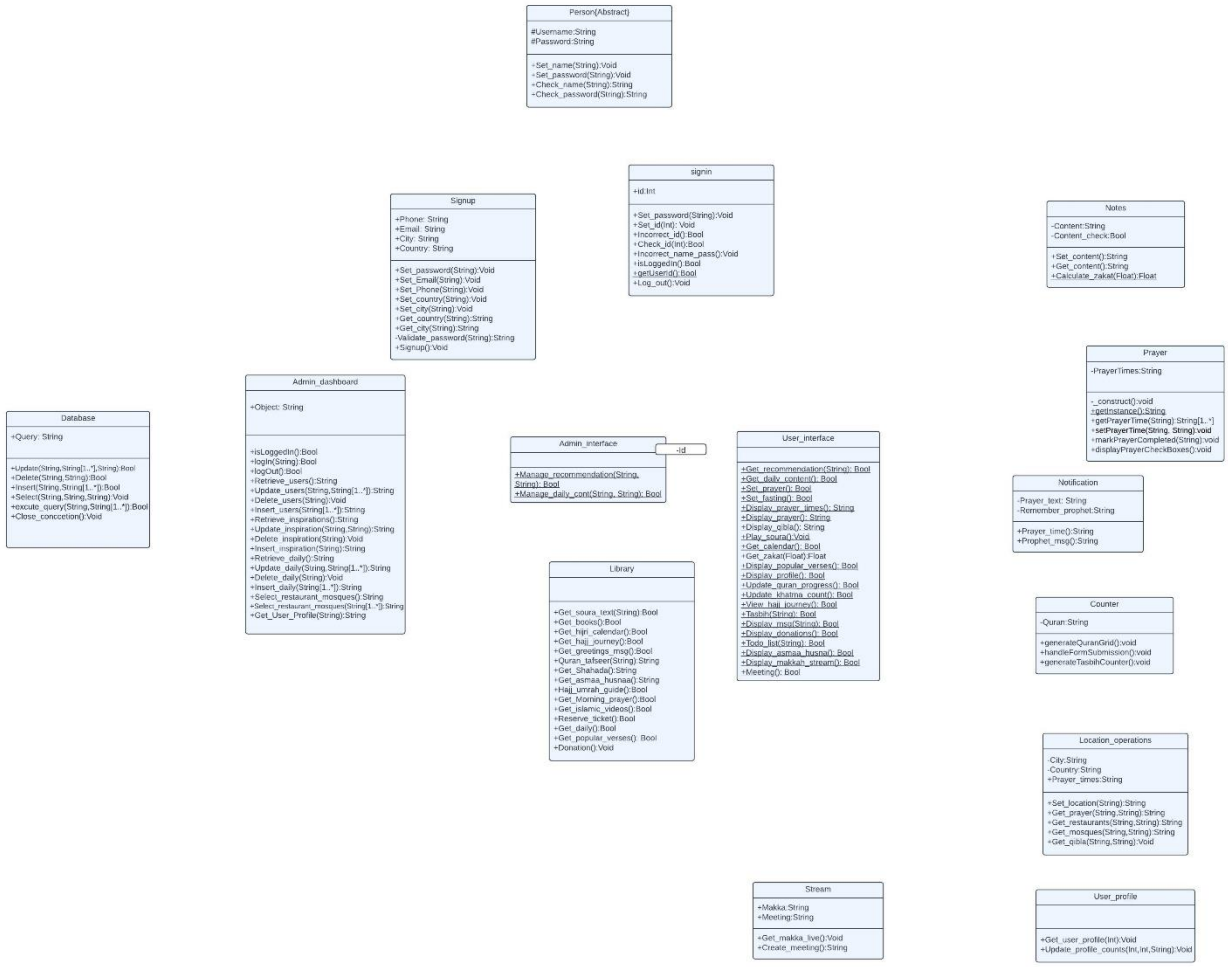


10.2 Table

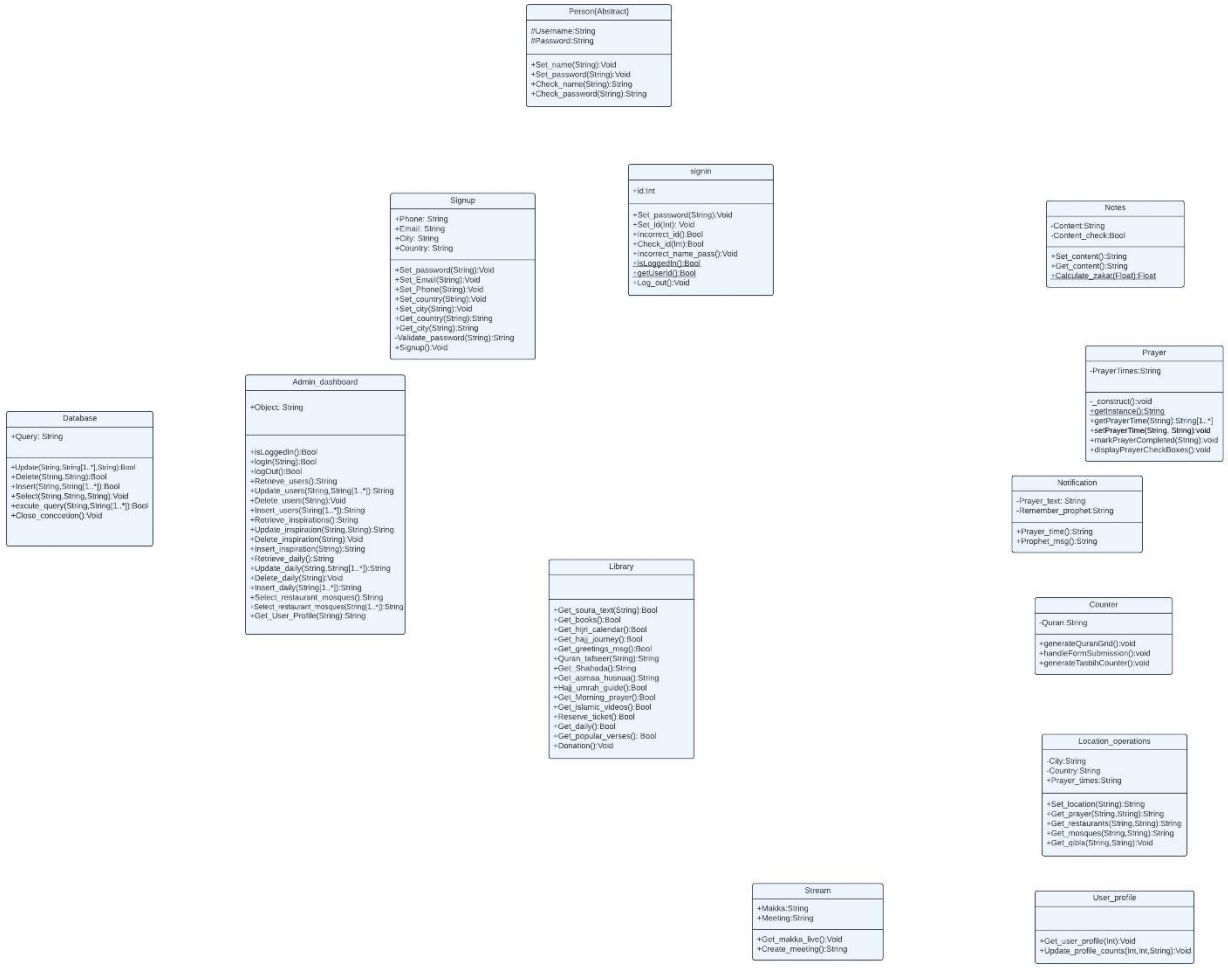

Untitled *
eslamiat popular_versus_daily_duas
id : int
versus : text
duas : text
hadith : text
admin_name : varchar(50)
user_id : int
eslamiat restaurants_mosques
id : int
name : varchar(100)
category : enum('restaurant','mosque')
country : varchar(50)
city : varchar(50)
address : varchar(255)
eslamiat inspiration
id : int
content : text
reference : varchar(100)
eslamiat admin
ID : int
password : varchar(50)
email : varchar(100)
phone_number : varchar(20)
eslamiat user
id : int
name : varchar(50)
password : varchar(50)
city : varchar(50)
country : varchar(50)
phone : varchar(20)
email : varchar(50)
eslamiat userprofile
khatma : int
fasting_days : int
quran_progress : int
user_id : int
Console

11.Class diagram

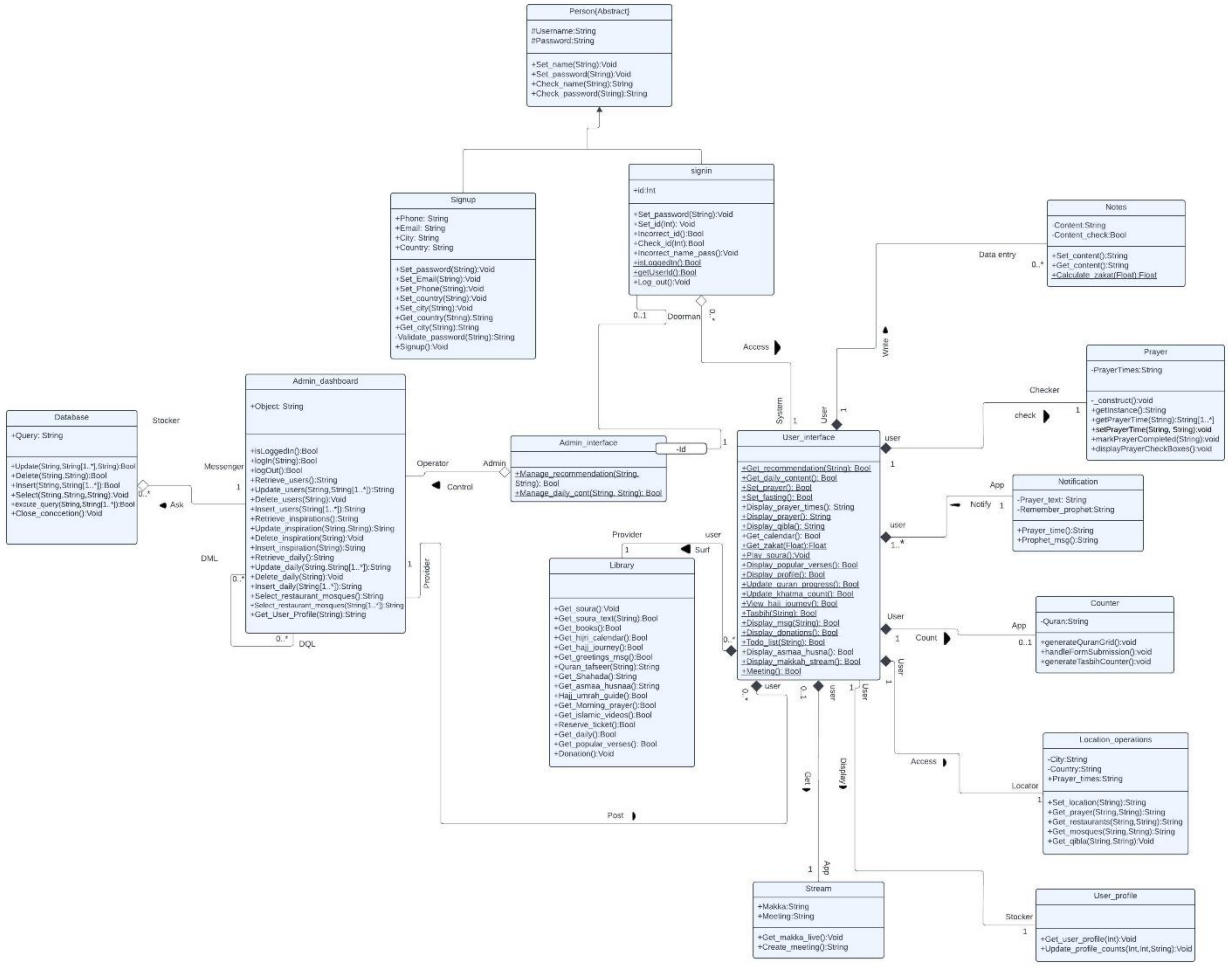
11.1 version 1



11.2 version 2



11.3 Version 3



12-Design patterns:

12.1Singleton Design Pattern for Prayer Management:

Intent:

Ensure that only one instance of the Prayer class is created to manage the prayer times efficiently. The class provides a single interface for accessing and updating the prayer times array.

Motivation:

Since there are only five prayers a day, it's practical to have a single instance managing prayer times. This approach ensures consistency and avoids redundant instances.

Structure:

- The Prayer class has a private constructor to prevent direct instantiation.
- It provides a static method `getInstance()` to access the single instance.
- The class maintains an associative array to store prayer times.
- It offers methods to retrieve and update prayer times.

Consequences:

- Guarantees a single instance of the Prayer class throughout the application.
- Ensures synchronized access to the prayer times array.
- Simplifies the management of prayer times by centralizing them in one place.
-

12.2Builder Pattern for User Sign-Up:

Intent:

The Builder pattern separates the construction of a complex object from its representation, allowing the same construction process to create different representations. In the context of user sign-up, the Builder pattern can be used to create user objects with various attributes while keeping the construction process flexible and readable.

Motivation:

User sign-up processes often involve multiple optional parameters (e.g., username, email, password, profile picture), and not all users may provide all of them. Using the Builder pattern allows us to construct user objects step by step, setting only the attributes that are provided by the user.

Structure:

- UserBuilder: Defines an interface for building user objects step by step.
- ConcreteBuilder: Implements the UserBuilder interface to provide methods for setting user attributes.
- Director: Controls the construction process by orchestrating the steps defined in the UserBuilder interface.
- User: Represents the complex object being constructed.

Consequences:

- Provides a clear and flexible way to construct user objects with optional attributes.
- Simplifies the creation of complex user objects while keeping the construction process readable.
- Allows for variations in user attributes without modifying the user class or constructor.

12.3Factory Design Pattern for User Sign-up:

Intent:

Create a factory class responsible for creating instances of user objects based on different user types. This pattern allows for flexible creation of user objects while encapsulating the object creation logic.

Motivation:

In a system where users can sign up with different roles or types (e.g., regular users, admins, moderators), using a factory pattern helps centralize the user creation process and abstracts the details of object instantiation.

Structure:

- The UserFactory class acts as a factory for creating user objects.
- It provides methods for creating different types of users.
- Concrete user classes inherit from a common User class or interface.
- The factory encapsulates the logic for determining which type of user to create.

Consequences:

- Simplifies the user creation process by providing a centralized factory.
- Encapsulates object creation logic, making it easier to manage and extend.
- Promotes code reuse and maintainability by abstracting object creation details.