Software Requirements Specification

For

LifeSync

Version 1.0 approved

Prepared by:
Abdul Rafay Chohan
Zophiel Suleman
Arslan Kashif

March 30th 2024

Table of Contents

Contents

1. Introduction 1
1.1 Purpose 1
1.2 Document Conventions 1
1.3 Intended Audience and Reading Suggestions 1
1.4 Product Scope 1
1.5 References 1
2. Overall Description 2
2.1 Product Perspective 2
2.2 Product Functions 3
2.3 User Classes and Characteristics 5
2.4 Operating Environment 5
2.5 Design and Implementation Constraints 5
3. External Interface Requirements 5
3.1 User Interfaces 5
3.2 Hardware Interfaces 6
3.3 Software Interfaces 6
3.4 Communications Interfaces 6
4. System Features 6
4.1 Task Management 6
4.2 Journal 9
4.3 Budget Tracking 12
4.4 Login/Signup 15
5. Other Nonfunctional Requirements 19
5.1 Security: 19
5.2 Usability: 19
5.3 Reliability: 19
6. Other Requirements 19

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

*This SRS, based on our project, '**LifeSync'** version 1.0, aims to identify the basic functional and nonfunctional requirements of the project, as well as the associated use cases and diagrams to represent the flow and working of the various functions.*

1.2 Document Conventions

The font that is followed in this document is Arial and size 11. Special highlighting is done by making the text bold so that important keywords can be easily differentiated. Every requirement stated in this document has its own unique priority and every functionality is equally important.

1.3 Intended Audience and Reading Suggestions

This SRS is intended for educational purposes and the intended audience is the teachers and teacher assistants. This SRS contains details about the project LifeSync, it has the overall description first followed by the functionalities and use cases.

1.4 Product Scope

In today's fast-paced and interconnected world, individuals are constantly handling multiple responsibilities, from work deadlines to personal commitments and financial obligations. As the demands of modern life continue to increase, many people struggle to effectively manage their time, finances, and emotional well-being.
The motivation behind selecting this project comes from the recognition of the increasing need for a comprehensive solution that addresses the multifaceted challenges individuals face in managing their productivity, finances, and personal growth. By harnessing the power of artificial intelligence (AI), we aim to develop a mobile application that goes beyond traditional task management and budget tracking tools, offering users a personalized and intelligent assistant to help them lead more organized and fulfilling lives. Our main features include: Task management, Budgeting goals, personal reflection.

1.5 References

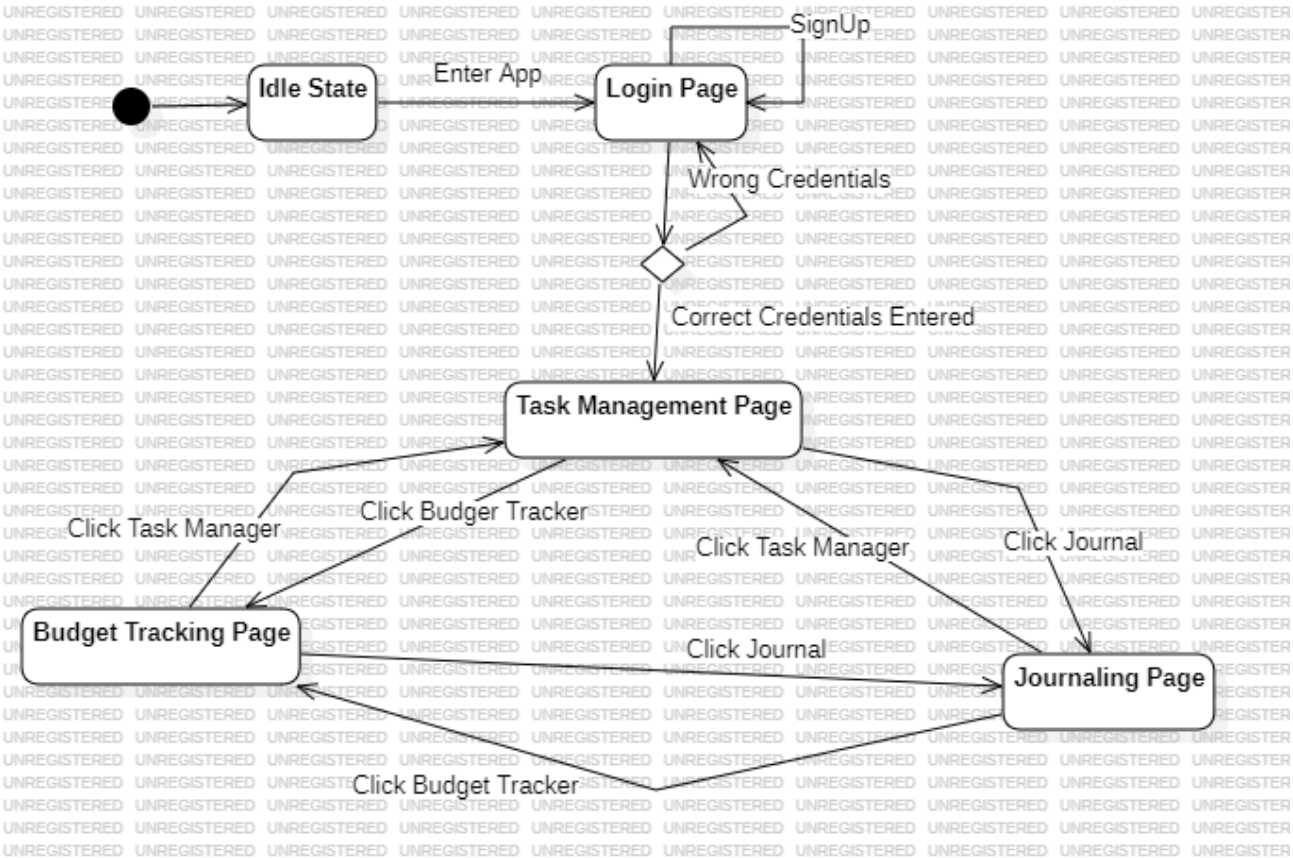
Management Studies and Business Journal (PRODUCTIVITY), 1(1):123–129, Jan. 2024. doi: 10.62207/s298rx18. URL https://journal.ppipbr.com/index.php/ productivity/article/view/35.
Cesar Bravo, Luigi Saputelli, Francklin Rivas, Anna Perez de Rivas, Michael Nikolaou, Georg ´ zangl, Neil Guzman, Shahab Mohaghegh, and Gustavo Nunez. State-of-the-art application of artificial intelligence and trends in the ep industry: A technology survey. 03 2012. doi: 10.2118/150314-MS.
M.D.R.-Moreno and P. Kearney. Integrating ai planning techniques with workflow management system. Knowledge-Based Systems, 15(5):285–291, 2002. ISSN 0950-7051. doi: https: //doi.org/10.1016/S0950-7051(01)00167-8. URL https://www.sciencedirect.com/

science/article/pii/S0950705101001678.
Karen Myers, Pauline Berry, Jim Blythe, Ken Conley, Melinda Gervasio, Deborah L. McGuinness, David Morley, Avi Pfeffer, Martha Pollack, and Milind Tambe. An intelligent personal assistant for task and time management. AI Magazine, 28(2):47, Jun. 2007. doi: 10.1609/aimag.v28i2.2039. URL https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2039.
Mahammad Khalid Shaik Vadla, Mahima Agumbe Suresh, and Vimal K. Viswanathan. Enhancing product design through ai-driven sentiment analysis of amazon reviews using bert. Algorithms, 17(2), 2024. ISSN 1999-4893. doi: 10.3390/a17020059. URL https://www.mdpi.com/1999-4893/17/2/59.

2. Overall Description

2.1 Product Perspective

This SRS aims to define a new self-contained product named LifeSync with the functions and use cases mentioned in the SRS. This state chart diagram shows the major components of the overall system and subsystem interconnections.

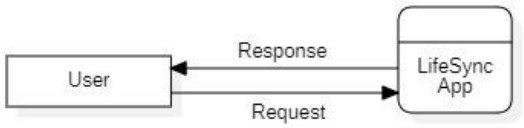


2.2 Product Functions

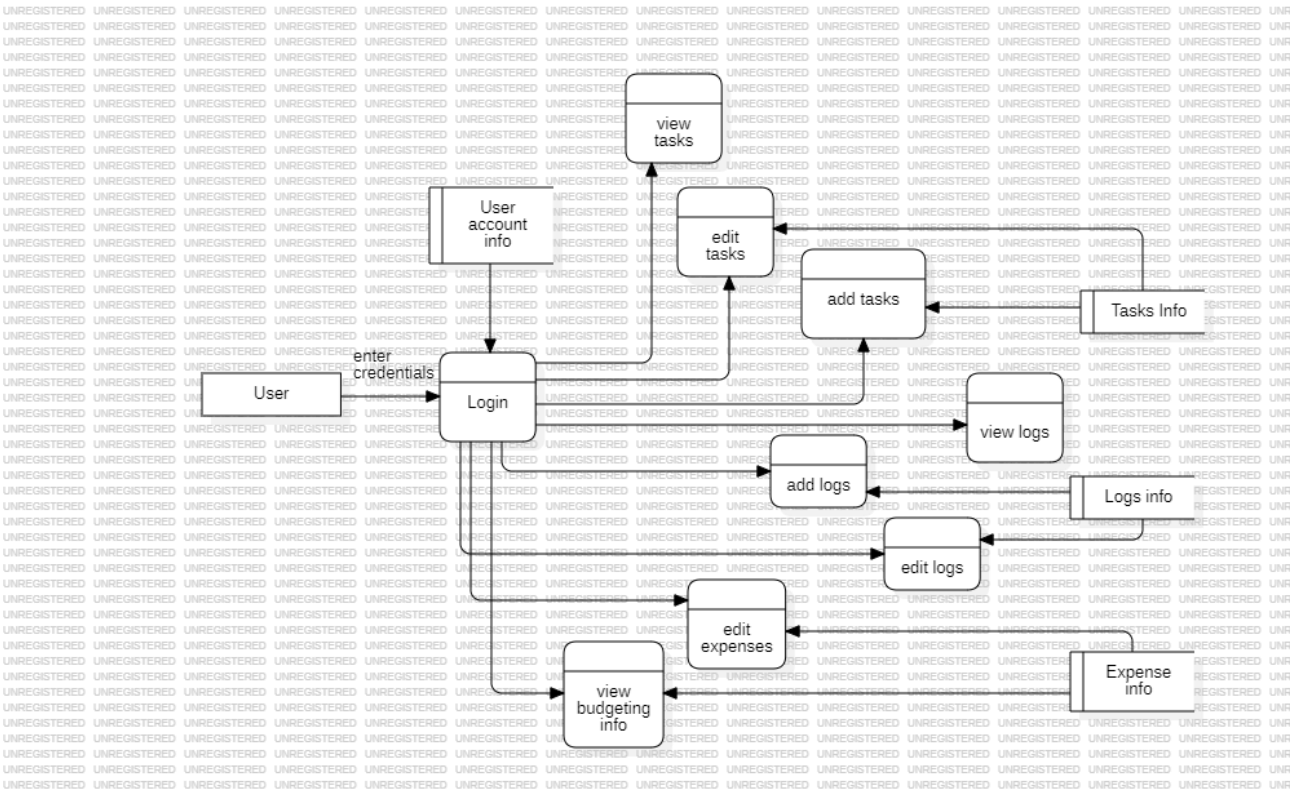
1. Creating, assigning, and tracking Tasks
2. Set deadlines
3. Priority levels for Tasks
4. Track income, expenses, and savings goals
5. Colorful visualizations and insightful reports
6. Mood Tracking
7. Attach photos or media
8. Personalized Space locked behind locks

The following Data Flow Diagrams represent the general flow of our application and model the functionalities

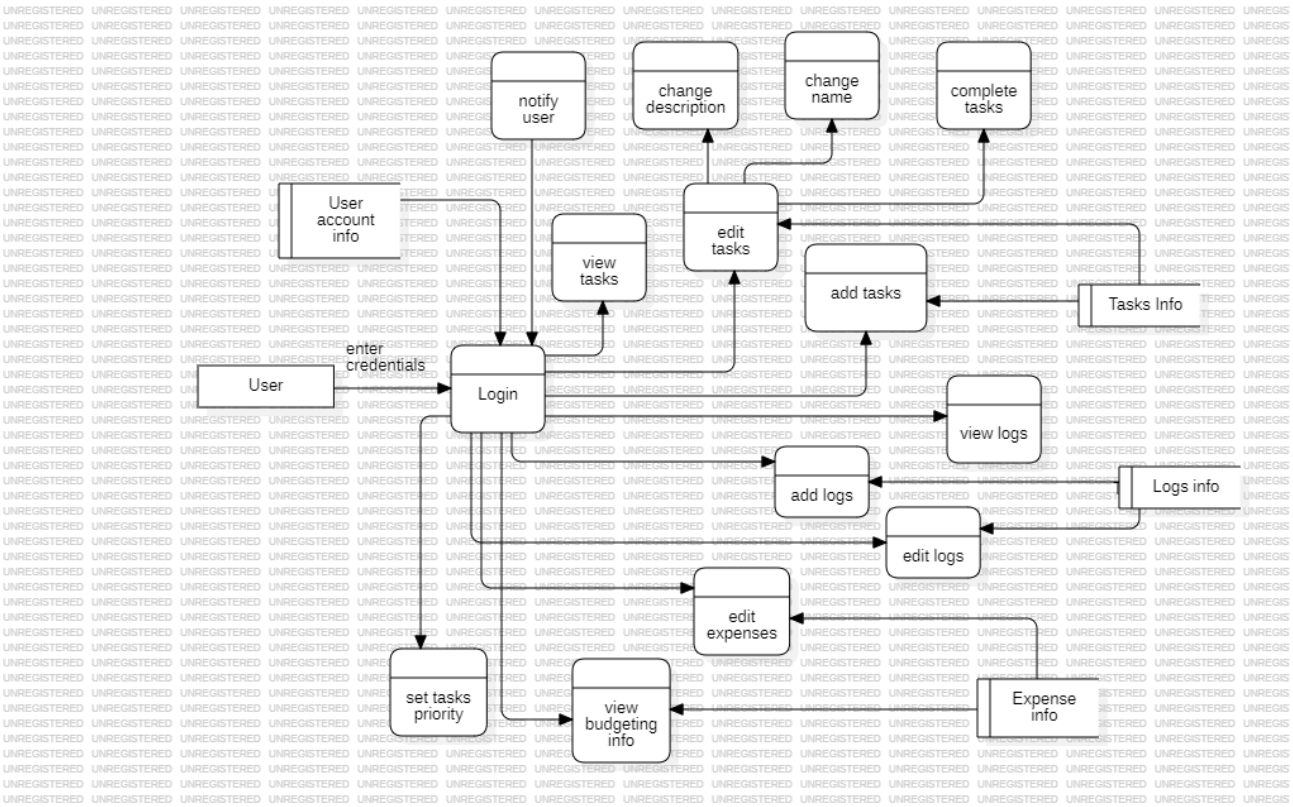
Level 0 DFD Diagram



Level 1 DFD Diagram

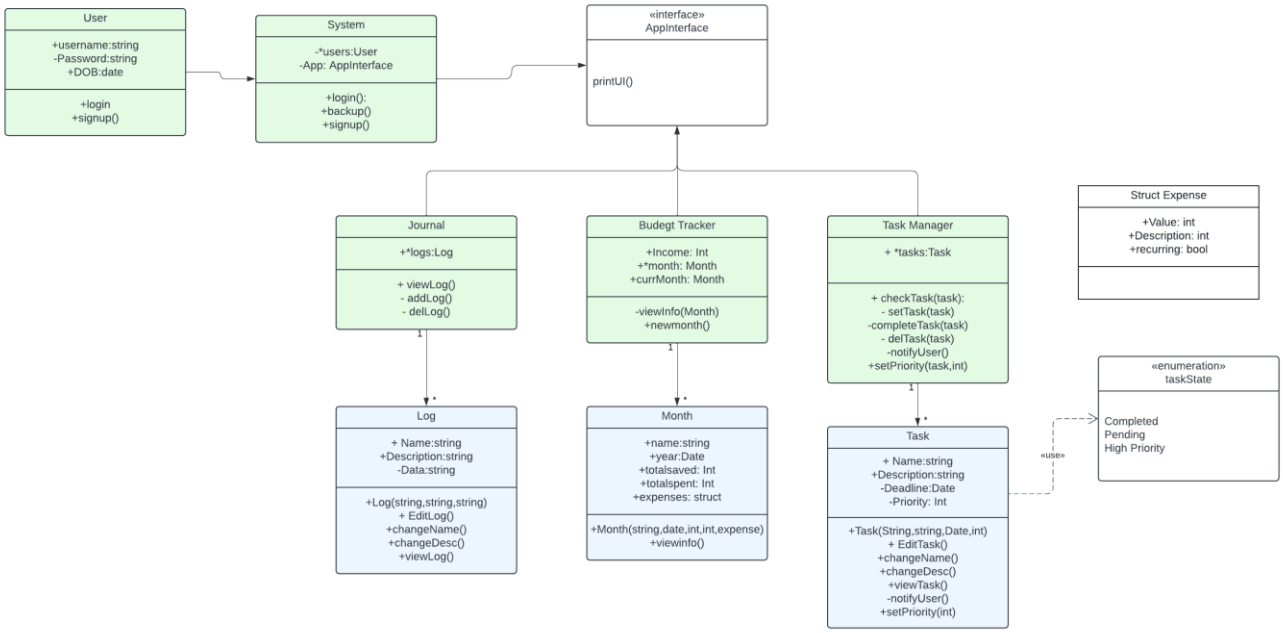


Level 2 DFD Diagram



2.3 User Classes and Characteristics

Given is the class Diagram that shows the classes used and their relations:



2.4 Operating Environment

The product is an Android application that will be compatible on Android devices having version 7.0 and above.

2.5 Design and Implementation Constraints

Memory Constraint: As the user keeps adding new entries in the app over the course of time it can start taking significant memory which will also effect the size of backups.

Cloud Access: In future if this project is to be scaled up to cloud storage it would require access to a cloud. Also cloud storage may bring security concerns with it.

3. External Interface Requirements

3.1 User Interfaces

The UI of this app includes multiple pages for each major function like task management, budget tracking and journaling. The pages will be accessed through a navbar or on-screen buttons. At the

start, users will get a screen where they can enter their information which can be used to access the app.

3.2 Hardware Interfaces

As the app is being developed for Android devices it requires the Android OS above version 7.0. The app will be optimized for low end devices as well. But would require significant storage for local backup of data on the devices.

3.3 Software Interfaces

The product is an Android application that will be compatible on android devices having version 7.0 and above. The app is planned to run offline without the need of any online sources all the database required for the app data will be stored in the user device (Note: This is scalable to cloud storage in the future).

3.4 Communications Interfaces

Currently the app is planned to run offline without the need of any online sources or communication with the web. In future plans the app will be integrating AI which will require basic communication with the web servers for the recommendation system.

4. System Features

4.1 Task Management

4.1.1 Description

Use Case Title: *Task Management*

Actor*: User*

Goal*: The user wants to add a new task to their list in the app.*

Preconditions*:*

- *The user is logged into the app.*
- *The user has navigated to the task management section of the app.*

Flow*:*

- *The user opens the task management section of the app.*
- *The app displays the user's existing task list or an empty task list if no tasks have been added yet.*

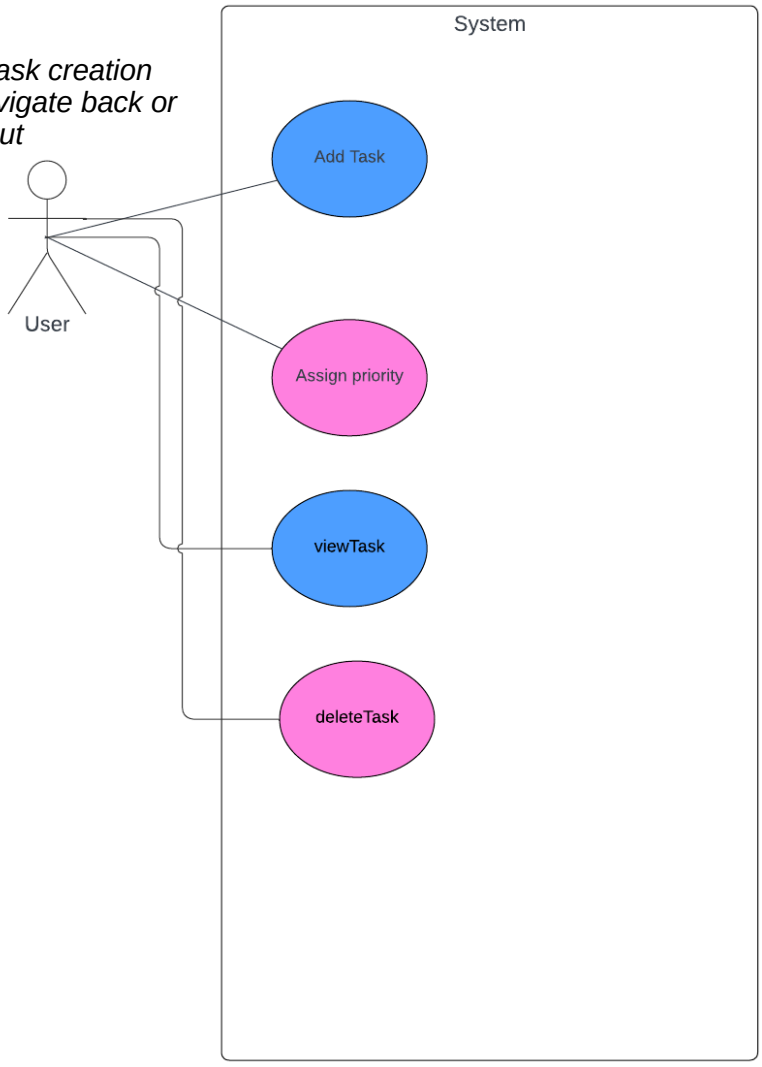
- *The user taps on the "Add Task" button or icon to initiate the task creation process.*
- *The app presents a form or input fields for the user to enter details about the new task, including title, description, due date, priority level.*
- *The user fills in the required task details and any optional information.*
- *The user confirms or submits the task details to add the task to their task list.*
- *The app validates the input to ensure all required fields are filled and the data is in the correct format.*
- *The user can now see the newly added task in their task list along with any other existing tasks.*

***Postconditions**:*

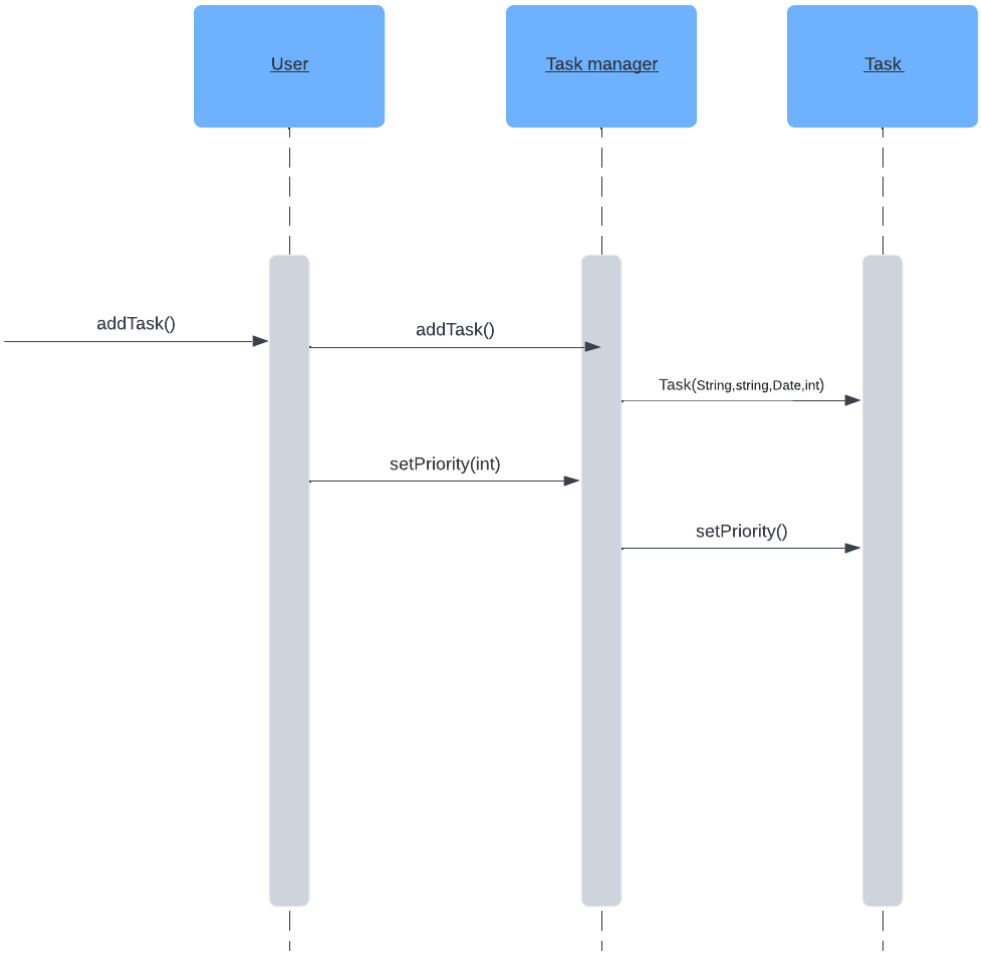
- *The new task is successfully added to the user's task list in the app. The user can view, edit, or delete the task as needed.*

***Alternate Flow**:*

- *If the user decides to cancel the task creation process at any point, they can navigate back or close the task creation form without submitting any data.*



4.1.2 Sequence Diagram



4.1.3 Functional Requirements

TBD

4.2 Journal

4.1.1 Description

Actor: *User*

Goal: *The user wants to write a new journal entry in the app's personal journaling feature.*

Preconditions:

- *The user is logged into the app.*
- *The user has navigated to the personal journaling section of the app.*

Flow:

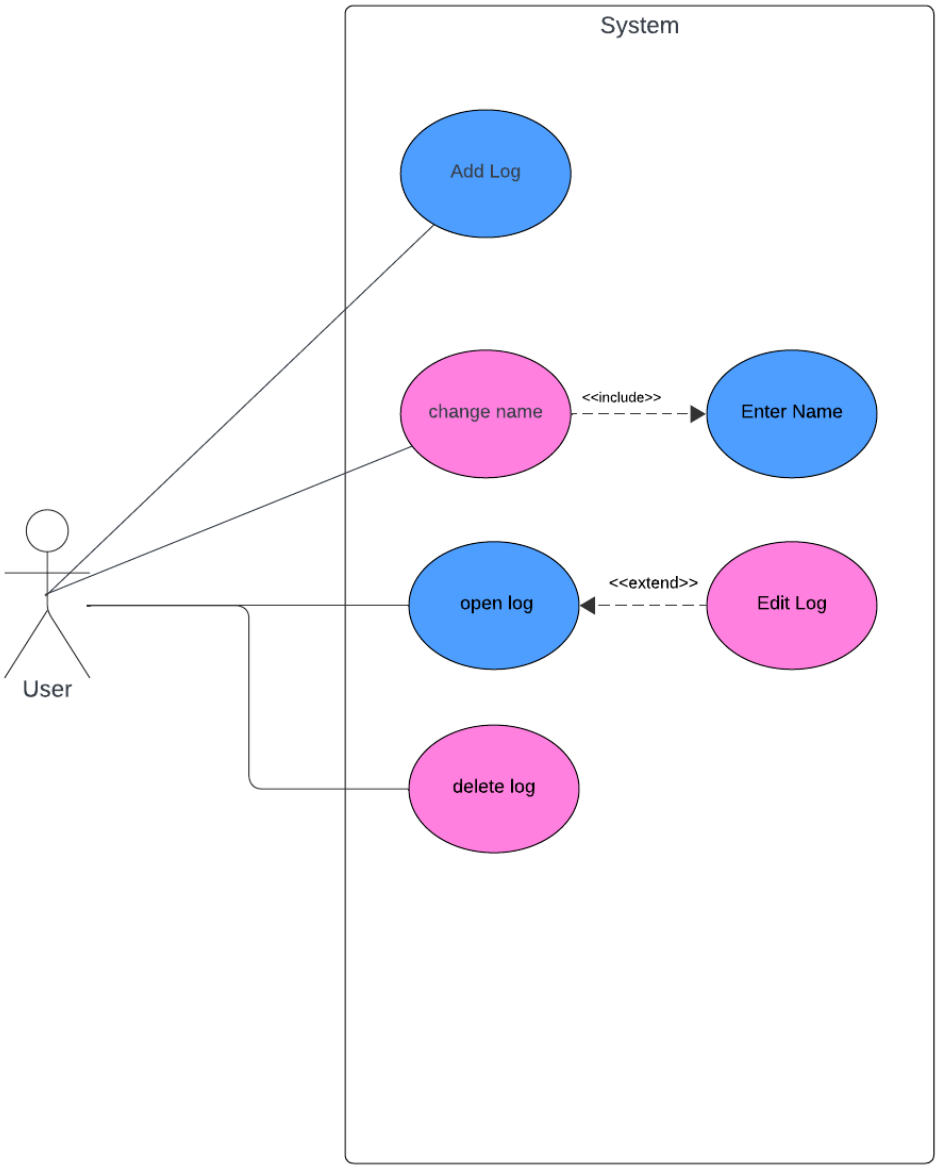
- *The user opens the personal journaling section of the app.*
- *The app displays the user's journal archive, showing past journal entries or an empty archive if no entries exist yet.*
- *The user taps on the " New Entry" button.*
- *The app presents a text editor or input field for the user to write their new journal entry.*
- *The user writes the content of their journal entry, including thoughts, feelings, experiences, or reflections*
- *The user saves the completed journal entry to their journal archive.*
- *The app validates the input to ensure the journal entry is not empty and meets any other requirements.*
- *The user can now see the newly written journal entry in their journal archive alongside any existing entries.*

Postconditions:

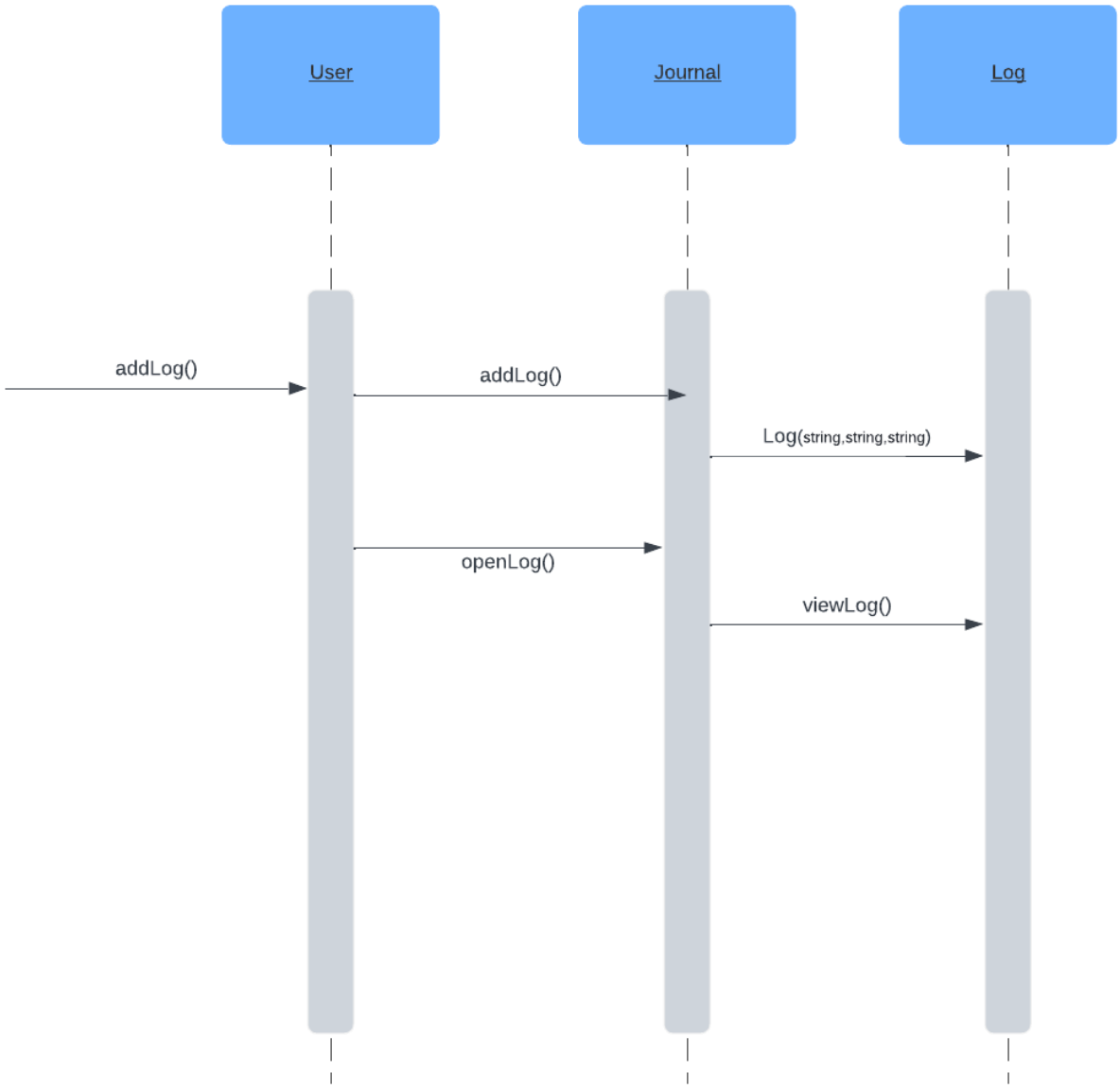
- *The new journal entry is successfully saved in the user's journal archive in the app.*

Alternate Flow:

- *If the user decides to cancel the journal entry creation process at any point, they can navigate back or close the journal entry form without saving any data.*


System
Add Log
change name
<<include>>
Enter Name
open log
<<extend>>
Edit Log
delete log
User

4.1.2 Sequences



4.1.3 Functional Requirements

TBD

4.3 Budget Tracking

4.3.1 Description

Use Case Title: *Budget Tracker*

Actor*: User*

Goal*: The user wants to record a new expense in the app's budget tracker.*

Preconditions*:*

- *The user is logged into the app.*
- *The user has navigated to the budget tracking section of the app.*

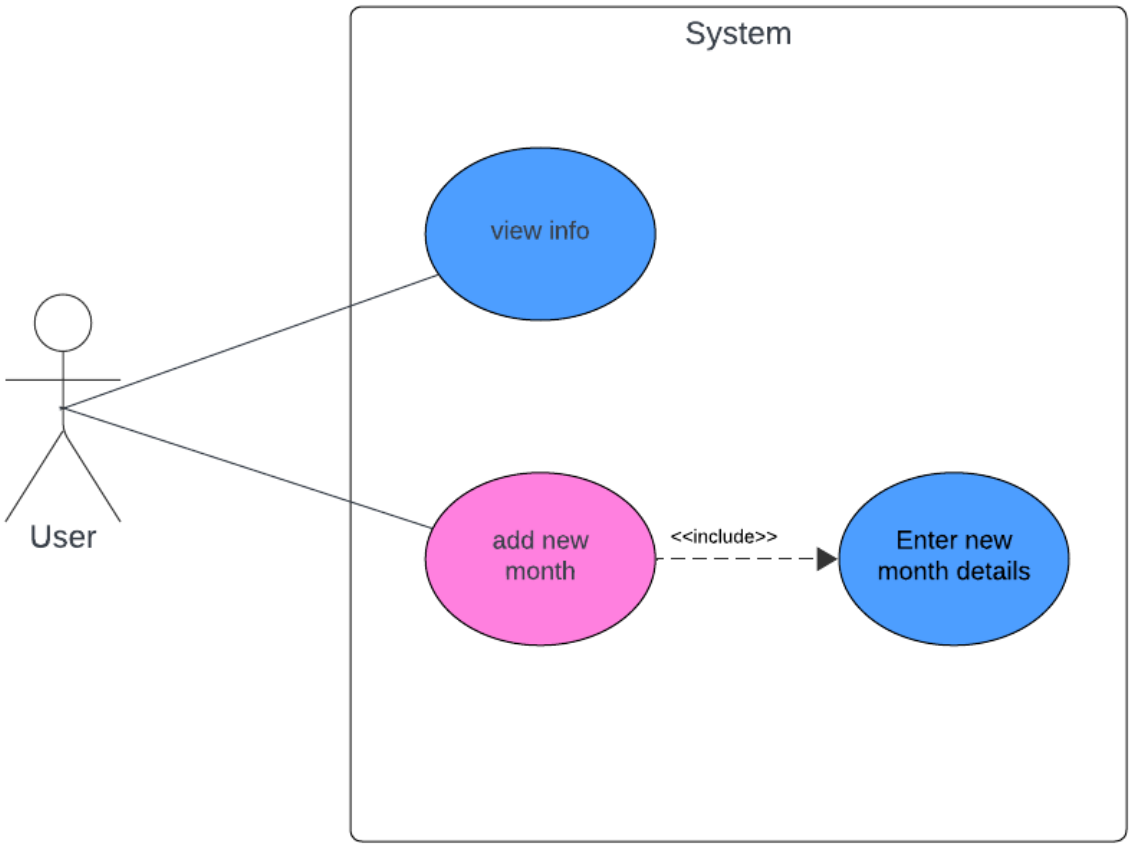
Flow*:*

- *The user opens the budget tracking section of the app.*
- *The app displays the user's budget overview.*
- *The user taps on the "Add Expense" button or icon.*
- *The app presents a form or input fields for the user to enter details about the new expense, including amount, category, date.*
- *The user fills in the required expense details and any optional information.*
- *The user confirms or submits the expense details to add the expense to their budget tracker.*
- *The app validates the input to ensure all required fields are filled and the data is in the correct format.*
- *The user can now see the newly recorded expense in their expense log along with any other existing expenses.*

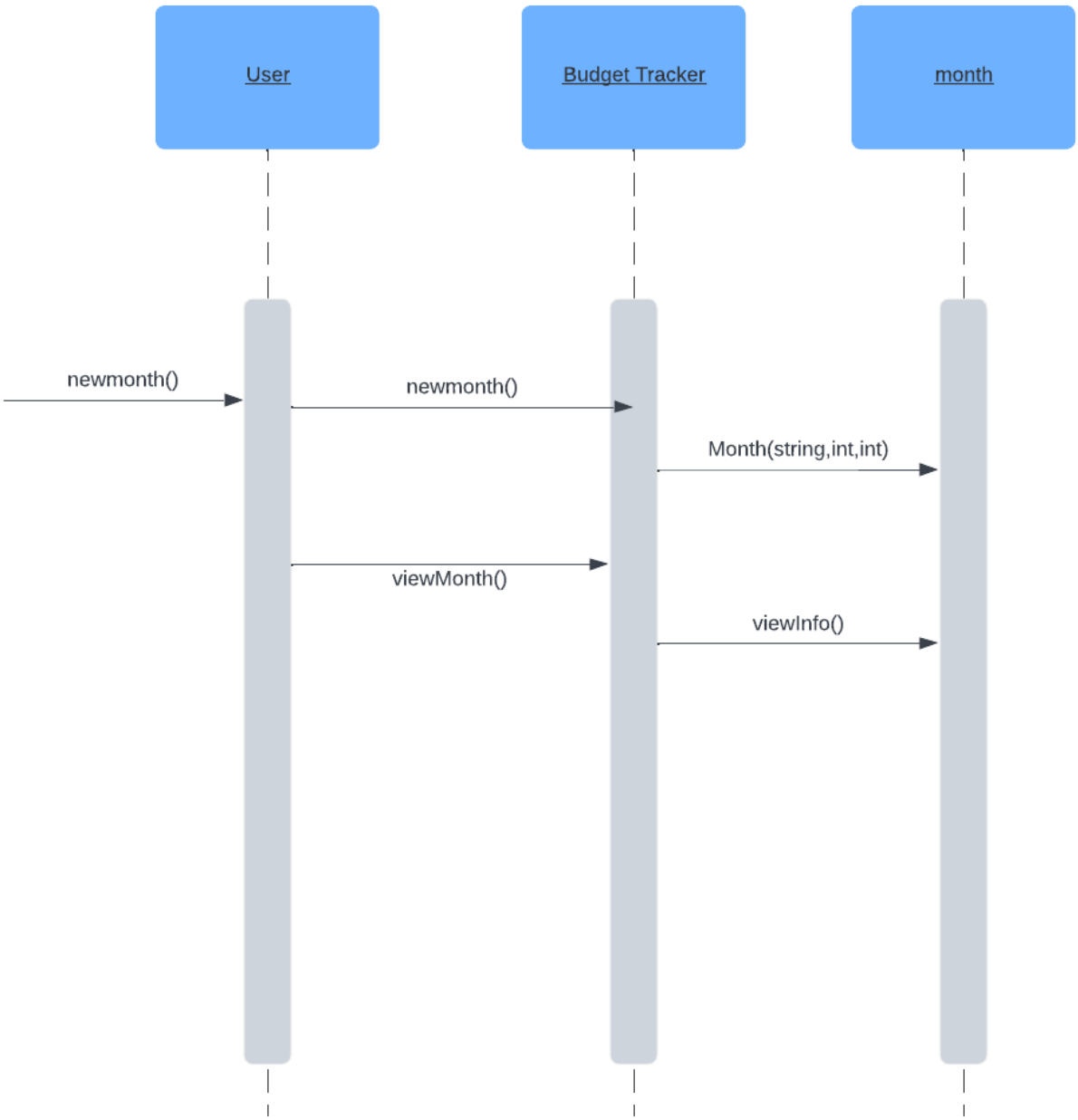
Postconditions:

- *The new expense is successfully recorded in the user's budget tracker in the app.*

.



4.1.2 Sequences



4.1.3 Functional Requirements

TBD

4.4 Login/Signup

4.1.1 Description

***Use Case Title**: User Authentication*

***Actor**: User*

***Goal**: The user wants to access the app by either logging into an existing account or creating a new account.*

***Preconditions**:*

- *The user has installed the app on their device.*
- *If the user is attempting to log in, they must have already registered an account.*

Flow(login):

- *The user opens the app on their device.*
- *The app displays the login screen, prompting the user to enter their credentials.*
- *The user enters their username and password.*
- *The user taps on the "Login" button to submit their credentials.*
- *The app sends the entered credentials to the authentication server for verification.*
- *The authentication server validates the user's credentials against the registered accounts database.*
- *If the credentials are valid, the authentication server sends a success response to the app.*
- *The app grants access to the user, allowing them to navigate to the app's main interface.*
- *The user can now access the app's features and functionalities.*

Flow(signup):

- *The user opens the app on their device.*
- *The app displays the signup screen, prompting the user to create a new account.*
- *The user enters their desired username, email address, and password.*
- *The user taps on the "Sign Up" button to submit their registration details.*
- *The app sends the entered registration details to the authentication server.*

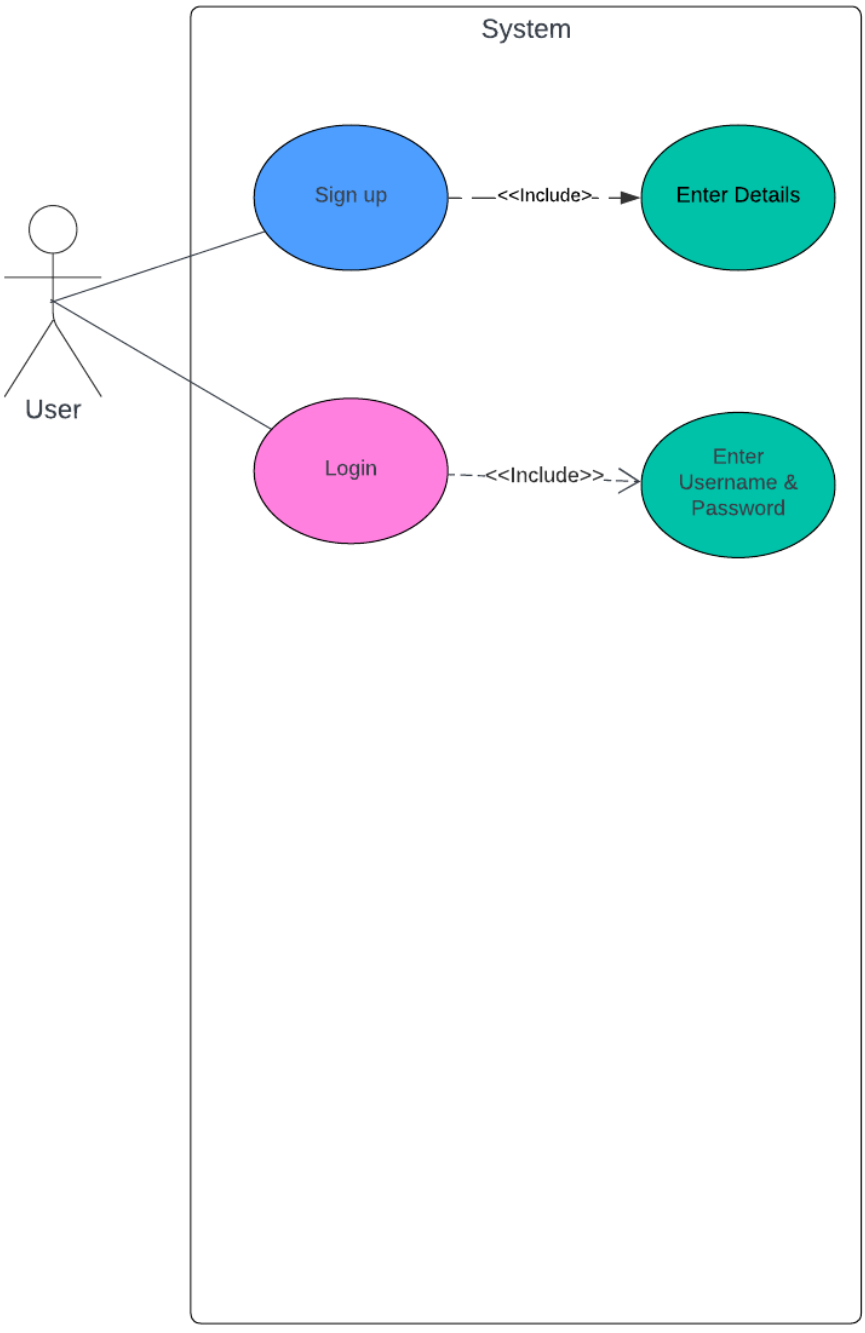
- *The authentication server checks if the provided email address is not already associated with an existing account.*
- *If the email address is available, the authentication server creates a new account with the provided credentials.*
- *The authentication server sends a success response to the app, indicating that the account has been successfully created.*
- *The app automatically logs in the user using the newly created account credentials.*

***Postconditions**:*

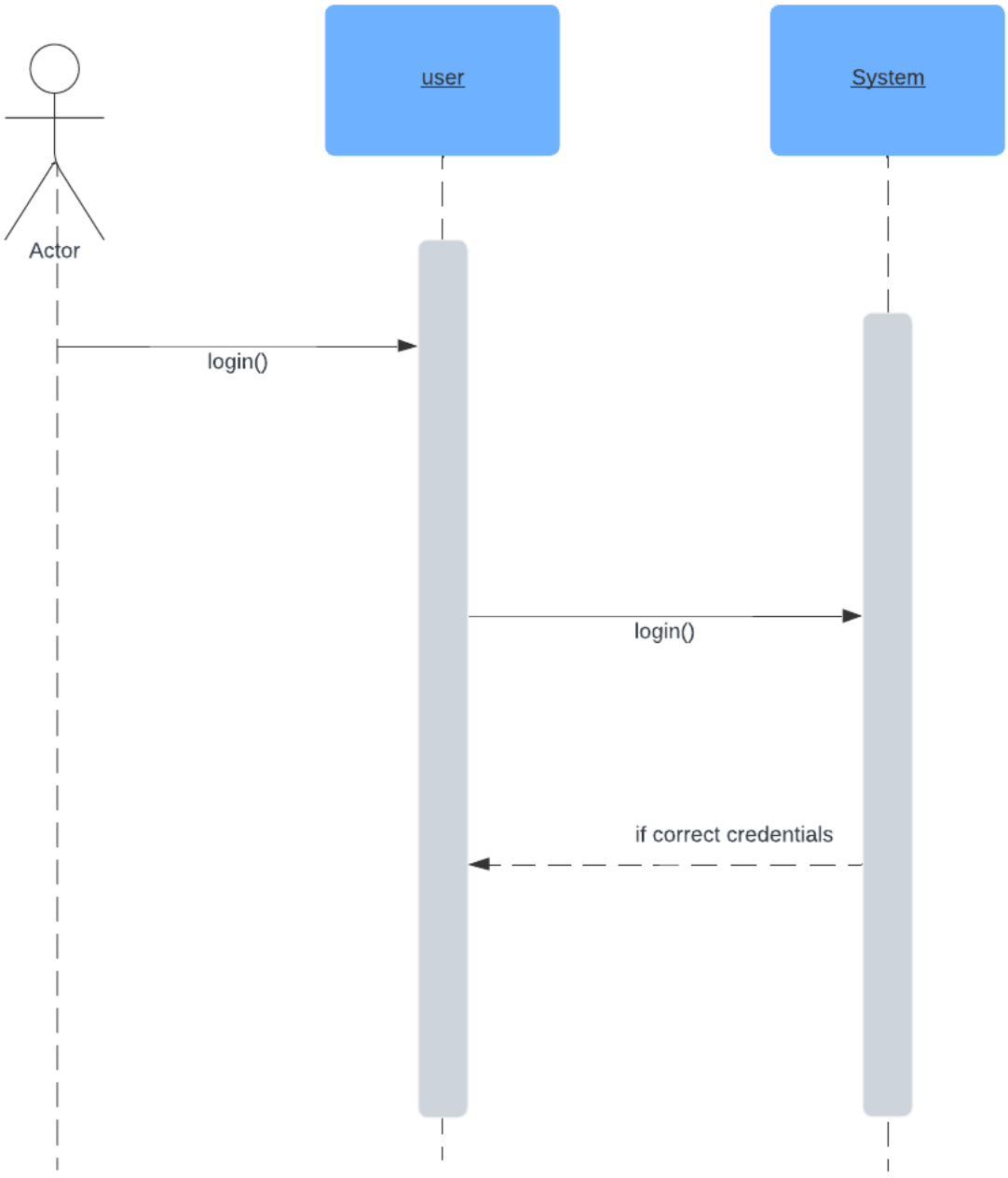
- *For the login scenario: The user is successfully authenticated and granted access to the app's features.*
- *For the signup scenario: The user's new account is successfully created, and they are automatically logged in.*

Alternate Flow:

- *If the user enters invalid credentials during login, the app displays an error message prompting the user to try again.*
- *If the user enters an email address that is already associated with an existing account during signup, the app prompts the user to choose a different email address.*


System
Sign up
<<Include>>
Enter Details
User
Login
<<Include>>
Enter Username & Password

4.1.2 Sequences



4.1.3 Functional Requirements

TBD

5. Other Nonfunctional Requirements

5.1 Security*:*

- *Ensure that sensitive user data, such as passwords, is stored securely,*

5.2 Usability*:*

- *Intuitive Design: Design an intuitive and user-friendly interface for easy navigation.*

5.3 Reliability*:*

- *Data Backup: Implement regular data backup mechanisms to prevent data loss.*
- *Error Handling: Develop robust error-handling mechanisms to gracefully handle unexpected situations without crashing the application.*

6. Other Requirements

Database will be used to manage the tasks, journal logs.