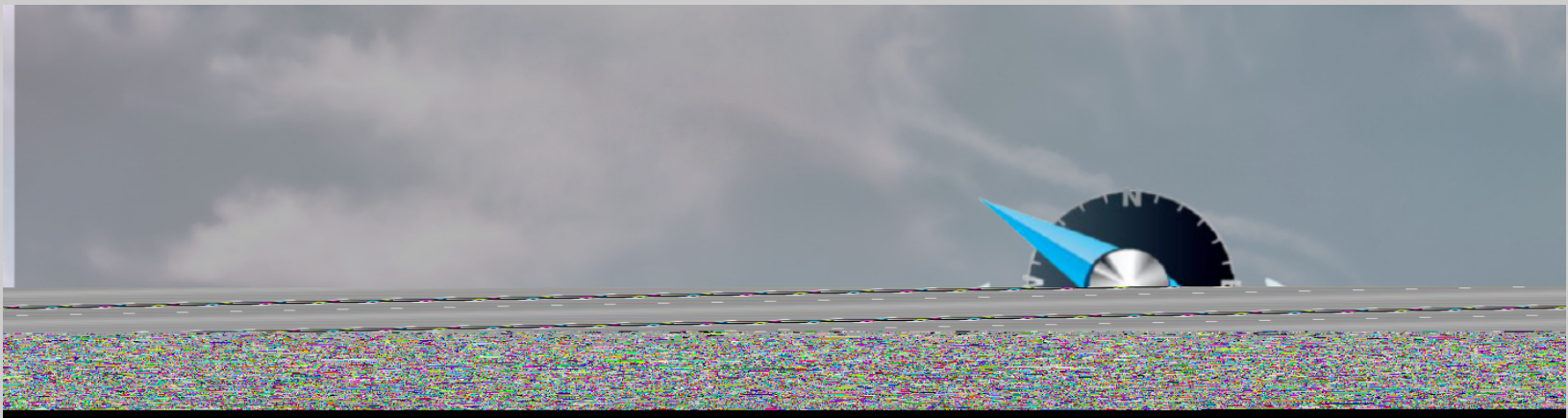

**School of Engineering and Applied Science,**
**Ahmedabad University**

# CSE300 Software Engineering
# Software Requirements Documentation Report

**Guided By: Prof Khusru Doctor**
**TA Muskan Matwani**
**TA Anupama Nair**


N

# Table of Contents

# 1. Introduction

## 1.1. Purpose

SRD is a document which mainly covers the topics related to purpose, overall description of the project and the requirements specific to system capabilities that can meet the clients' needs by taking into account the functioning and the users' goals into consideration. This document contains an in-depth description of the system developed. The main goals achieved by a well-written SRD are a follows:

- Deliver the feedback to the client and make sure that the issue related to the software is understood and resolved by the developer team or the company
- Help to break down the problem into subproblems by writing down the requirements at one place.
- Help to understand the crucial aspects of the project deeply and to deliver the design solutions accordingly and effectively.
- Speeds up the testing process.

## 1.2. Document Conventions

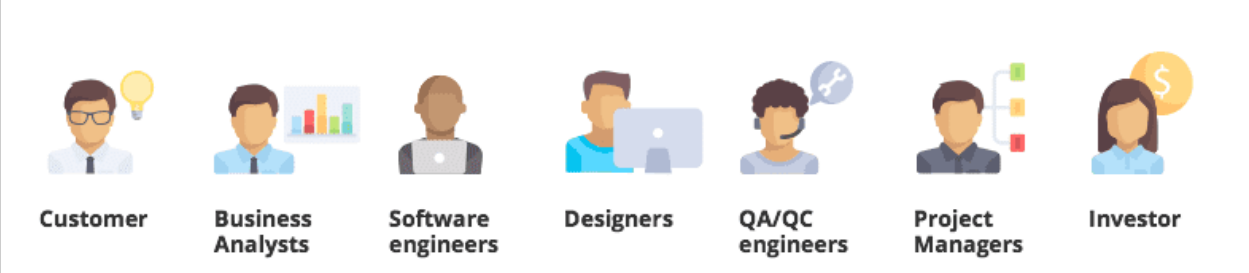
| **Abbreviation** | **Description** |
|---|---|
| SRD | Software Requirements Documentation |
| Client | A person for whom the system is designed and can use the system |

| | |
|---|---|
| User | A person who uses the system(same as client) |
| ERD | Entity-Relationship Diagram |
| DFD | Data Flow Diagram |
| Stakeholder | A person who has invested in the development of the system |
| Product Owner | A person who wants a system for their targeted clients |

# 1.3. Intended Audience

The SRD documentation is generally for the audience involved in the communication, planning, modelling, construction, deployment steps involved in the development of the project. The audience is as in the following image:

- Customers(Clients) : Needs SRD to be sure that the requirements meet the business goals
- Business Analysts : Deeply analyse and evaluate the project
- Software Engineers : Use SRD to identify the techstack and plan the process of deployment
- Designers : Use SRD to understand the project data and provide design solutions
- QA/QC(Testing) Engineers : Test the system based on the requirements provided in SRD
- Project Managers : User SRD to understand the system and manage the development process
- Investors(Stakeholders) : Reads SRD to understand the unique system functionality and make a decision




# 1.4. Scope of Project

Travoyager is a software where users can plan their trip according to their budget, number of people and generate an itinerary based on the age range of the people in the group. Using this itinerary one can get recommendations of hotels to stay in, restaurants to dine in, tourist spots to visit, route and medium of transportation etc; all at their favourable budget range.

The main elements of the project involves:

- Choice of some predefined travel plans for well-known destinations.

- Recommendations for choosing their travel preferences based on the season and weather conditions of that place.
- Selection of dates, place, stay, transport, budget and other information to develop an itinerary.
- Customisable itinerary based on needs and preferences.

# 2. Product Description

## 2.1. End-Product Perspective

The Travel Recommendation System helps users to solve the tedious task of one has to plan for the destination, mode of travel, hotel accommodation, what places to visit, how many days to spend etc. all in a decided budget and also sometimes travelling on one's own self can be a difficult task for finding hotels, restaurants, places to visit etc. This system is for ease of use for those travellers by making a whole trip plan on one portal.

This system is a web application through which users can interact with the system. They can make the itinerary by using the available destination's data. This data is added by the system administrator. The admin can add other admins and also the details of available destinations, places & hotels in the specific destination, place Images, etc. The web portal will communicate with the database and will add and modify data. All of the database communication will go over the Internet. By fetching the inputs from the user like source, destination, number of persons, type of place one likes to visit etc. the system will automatically generate the itinerary. The user can modify that itinerary and book the hotel according to availability. Now the user can cancel booking or make payment for confirming it. After the successful booking user can view the plan and can cancel also.

## 2.2. End-Product Characteristics

The system provides an auto-generated itinerary based on the user's interest. The system generates an itinerary based on what type of places the users wants to visit, the number of days of visit, budget and place ratings. The system will allow users to modify by adding or updating any place for any slot by giving the place explore option. In place explore user can see all the places which are not in his/her

current itinerary.(If all the place covered then show all place with warning that now duplications can be possible)

The system verifies at the time of user registration that it has unique username and email. The wallet generates automatically after the successful registration of the user. For old users, the system verifies the proper match of username and password. After the login the load the trips which are complete(history) and provide the option to make one. If already any trip generation is going on then it won't allow the user to make another until the payment has been done or trip cancels.

## 2.3. End-Product Functionalities

The main purpose is to reduce unnecessary search for making trip plans, fetching places & hotel details and their pre-booking etc. After getting all the functionalities at the same portal, the search time and complexity for the user will be reduced.

The user can register/login to the system, explore places, make itinerary, book hotel, make payment, see previous trips, see the ongoing trip, review places, see and update the profile. The cancellation of booking is also allowed for the user. The user can make the itinerary according to his/her budget, interested types of places and number of people to visit. The wallet account will be given to any user as long as he/she registers to the system. User can view and manage the profile and update the balance,

The admin can do CRUD operations for destination, place, hotel database entities. The admin can manage user details and also can generate reports out of it.

## 2.4. Type of Users

There are two kinds of website users: i) User and ii) System administrator. As the user can be a new user or already registered user. The new user can register to the system and generate the itinerary but logged in users can view the previous history or continue with their last booking also.

The necessary data in the system is added by system admins. The admin can do CRUD operation on those data and manage the user details also.

## 2.5. Operating Environment

This is a web based application, so it needs client server GUI.The website should run on any browser like Google Chrome or Firefox. The database should be hosted on localhost. The database should work in sync with the server. The website also has to be accessed from any other devices like phone, tablet, etc. According to screen resolution the GUI should be device friendly. The software should support all the operating systems such Windows, Mac or Linux.

## 2.6. Design Constraints

This system supports only English language, so it doesn't have multi language support. The good internet connection is required because the server fetches the data from the database.

The system can't generate more than 2 itinerary at same time for the same user. Means the user has to complete one itinerary before the generation of others. The trip starts from the same day of starting date, meaning the user can book from tomorrow's data till he wants. There are only 3 slots available for each day of the trip. There is no place that takes time to visit for more than 2 slots.

The hotel room selection is not given to the user. The update of wallet balance through net banking is still pending.

## 2.7. Assumptions & Dependencies

The users of the website know how to register for any website and can read & understand the next procedures. The user has at least a mobile device and good internet to access the website. The dependency is that it requires Internet based servers for its web-based features to be displayed over the website

The travel can happen from city to city only. The user can travel by his own from source to destination location and also to visit the places. The user has already reached the destination before the starting date of his/her trip. The places are not sorted based on location. The user can afford all the travel expenses. Room capacity and quality of all the rooms of the hotel are the same. User hasn’t flexibility to choose the room at any specific location. The wallet balance update is pending.

The main dependencies are between Itinerary generation and place fetch, hotel booking and Payment, completed trips and giving place review, place ratings and user given ratings, destination and place database, destination and hotel database, user inputs and itinerary database and hotel and hotel booking database.

# 3. External Requirements

## 3.1. Software Requirements

- Access to stable and secure internet connection.
- Availability of python environment, libraries related to django framework and other UI/UX related modules and installations.
- Working web-browser such as Google Chrome, Mozilla Firefox etc.

## 3.2. Hardware Requirements

The developed system does not require any specific hardware. It can run on any device including PC, laptop and mobile devices. Still enough memory and good internet connections would be helpful for a good user experience.

## 3.3. Communication Requirements

The communication between the database and the web portal consists of operation concerning both reading and modifying the data, while the communication between the database and the web application consists of only reading operation. The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems for the web portal.

# 4. Functional Requirements

## 4.1. Business requirements

### 4.1.1. Accessibility

The system should be easily accessible by the users and the stakeholders. There should not be any kind of difficulty in accessing the system.

### 4.1.2. Data Privacy

The information about the users should be protected by providing authentication in form of signin and signup options in the system. This data should not be publicly visible to its users.

### 4.1.3. System Updates

The system should be updated from time to time by the developer team in order to solve the issues and the bugs present in the system to maintain proper flow of the system for its users.

### 4.1.4. Integration

The stakeholders and the product owners should be able to access the system and its functionalities as a whole(Both as admin and user side)

### 4.1.5. User Satisfaction

The system should be designed keeping in mind the user demands and expectations. The user should be satisfied with the interface and the functionality of the system.

## 4.2. User requirements

### 4.2.1. User

#### 4.2.1.1. Functional Requirement 1.1

**Name**

Visibility of the system

**ID**

R1

#### Details

The system should be visible to the user on their devices.

#### Dependencies

None

### 4.2.1.2. Functional Requirement 1.2

#### Name

Sign-in / Sign-up

#### ID

R2

#### Details

If the user is new to the system then, that user should be able to sign-up into the system to access it. If the user has already registered, then the user should login into the system to access it.

#### Dependencies

R1

### 4.2.1.3. Functional Requirement 1.3

#### Name

View travel offers

#### ID

R3

#### Details

The user should be able to view different available travel offers and schemas, once signed-in or signed-up into the system.

#### Dependencies

R1, R2

#### 4.2.1.4. Functional Requirement 1.4

**Name**

Add trip details

**ID**

R4

**Details**

The user can add trip details as a form of input of trip destination, dates of trip, number of adults, number of children, budget for trip, types of places to visit etc for generating the itinerary.

**Dependencies**

R1, R2

#### 4.2.1.5. Functional Requirement 1.5

**Name**

View generated itinerary

**ID**

R5

**Details**

The user can view the itinerary generated by the system as per the inputs/details provided by the user about the trip requirements.

**Dependencies**

R1, R2, R3, R4

#### 4.2.1.6. Functional Requirement 1.6

**Name**

Add or update place in the itinerary

**ID**

R6

#### Details

The user can update the place to visit as per his/her choice for a given place in a slot in the itinerary and can add a place to visit for an empty slot in the itinerary.

#### Dependencies

R1, R2, R4, R5

### 4.2.1.7. Functional Requirement 1.7

#### Name

Update the generated itinerary

#### ID

R7

#### Details

The user can modify the itinerary by adding the places to visit in the generated itinerary as per his/her choice and thus, update the itinerary.

#### Dependencies

R1, R2, R4, R5, R6

### 4.2.1.8. Functional Requirement 1.8

#### Name

Add hotel booking to itinerary

#### ID

R8

#### Details

The system can book a hotel for the trip from a given list of hotels for the entered destination.

#### Dependencies

R1, R2, R4, R5

#### 4.2.1.9. Functional Requirement 1.9

**Name**

Make payment for the trip

**ID**

R9

**Details**

The user can do payment for the trip from his/her system wallet and confirm the booking.

**Dependencies**

R1, R2, R4, R5, R8

#### 4.2.1.10. Functional Requirement 1.10

**Name**

Cancel the trip booking

**ID**

R10

**Details**

The user can cancel the booking for the trip if he/she does not want to continue the trip before the trip starts.

**Dependencies**

R1, R2, R4, R5, R8, R9

#### 4.2.1.11. Functional Requirement 1.11

**Name**

View the trips history

**ID**

R11

#### Details

The user can view the details of the trip generated by himself/herself after the payment is done. The user can view trip details of the trip which is on-going during that date or has been completed previously.

#### Dependencies

R1, R2, R4, R5, R8, R9

### 4.2.1.12. Functional Requirement 1.12

#### Name

Update the wallet balance

#### ID

R12

#### Details

The user can update the wallet balance if he/she fills that the balance is not enough for the trip payment.

#### Dependencies

R1, R2

### 4.2.1.13. Functional Requirement 1.13

#### Name

Review a place

#### ID

R13

#### Details

The user can provide a review of the place visited during their trip at that place. The user can review the place after that trip has been completed.

#### Dependencies

R1, R2, R4, R5, R8, R9, R11

## 4.2.2. Admin

### 4.2.2.1. Functional Requirement 2.1

**Name**

Sign-in to the system

**ID**

R14

**Details**

The system administrator should be able to login into the system to access the system.

**Dependencies**

R1

### 4.2.2.2. Functional Requirement 2.2

**Name**

View the list of registered users

**ID**

R15

**Details**

The administrator can view the list of all the users who have registered into the system and their corresponding trip history.

**Dependencies**

R1, R14

### 4.2.2.3. Functional Requirement 2.3

**Name**

CRUD operations on destinations

**ID**

R16

#### Details

The administrator can add, update and delete any destination as per the needs of the users and make it visible to the system users for generating trips over that destination.

#### Dependencies

R1, R14

### 4.2.2.4. Functional Requirement 2.4

#### Name

CRUD operations on places

#### ID

R17

#### Details

The administrator can add, update and delete the places and its description in terms of charges, location etc for the destinations available in the database.

#### Dependencies

R1, R14, R16

### 4.2.2.5. Functional Requirement 2.5

#### Name

CRUD operations on place images

#### ID

R18

#### Details

The administrator can add, update and delete the images for any place for the destinations available in the database.

#### Dependencies

R1, R14, R16, R17

#### 4.2.2.6. Functional Requirement 2.6

**Name**

CRUD operations on hotels

**ID**

R19

**Details**

The administrator can add, update and delete any hotel and its description in terms of charges, location, services, accommodation etc for the destination available in the database.

**Dependencies**

R1, R14, R16

#### 4.2.2.7. Functional Requirement 2.7

**Name**

Try all user operations

**ID**

R20

**Details**

The administrator can perform all the operations in the system which can be done by the user.

**Dependencies**

R1, R14

#### 4.2.2.8. Functional Requirement 2.8

**Name**

Update the place reviews and ratings

**ID**

R21

#### Details

The administrator can update the ratings of any destination place after any user gives reviews of that place and make the user reviews for that place visible to other users.

#### Dependencies

R1, R14, R15

### 4.2.2.9. Functional Requirement 2.9

#### Name

Create user wallet

#### ID

R22

#### Details

The administrator can create a wallet for any newly registered user into the system.

#### Dependencies

R1, R14, R15

| **Requirement of** | **Requirement ID** | **Dependency** |
|---|---|---|
| **User** | R1 | - |
| | R2 | R1 |
| | R3 | R1, R2 |
| | R4 | R1, R2 |
| | R5 | R1, R2, R3, R4 |
| | R6 | R1, R2, R4, R5 |
| | R7 | R1, R2, R4, R5, R6 |
| | R8 | R1, R2, R4, R5 |
| | R9 | R1, R2, R4, R5, R8 |
| | R10 | R1, R2, R4, R5, R8, R9 |
| | R11 | R1, R2, R4, R5, R8, R9 |

| | | |
|---|---|---|
| | R12 | R1, R2 |
| | R13 | R1, R2, R4, R5, R8, R9, R11 |
| **Admin** | R14 | R1 |
| | R15 | R1, R14 |
| | R16 | R1, R14 |
| | R17 | R1, R14, R16 |
| | R18 | R1, R14, R16, R17 |
| | R19 | R1, R14, R16 |
| | R20 | R1, R14 |
| | R21 | R1, R14, R15 |
| | R22 | R1, R14, R15 |

# 5. Non-Functional Requirements

## 5.1. Usability

It refers to the ease in access and use of the system by the clients and the stakeholders as well. This can help to achieve the required goals of the project effectively. The use of this system can be done by any kind of clients who wish to travel and by the administration in form of the developer team, stakeholders and the product owners.

## 5.2. Availability

It refers to the available period of the system. It is the period in which the system will be fully functional and can be operated by its clients. Ideally the system should be available 24*7 for 365 days. This system is available anytime to its clients as long as either the bookings are accepted by the hotels for the corresponding destinations or the wallet balance limit does not exceed the final payment.

## 5.3. Performance

It refers to the responsiveness of the system whenever any user is interacting with the system. The system should be able to respond within a fixed time span to have the user keeping using the system. The delay between the user action and the system response should be as minimum as possible. This system is responsive while the client uses the system.

## 5.4. Maintainability

Any change in the system by the administration side should be reflected on the client side as well. The system should remain up-to-date with the available changes and modifications in the new versions. On refreshing, this system takes in all the updates and makes it visible to its client.

## 5.5. Security

It mainly focuses on providing data protection and prevention of unauthorised data access. This system resists the unauthorised access to any client by providing the login option for entering into the system.

## 5.6. Reliability

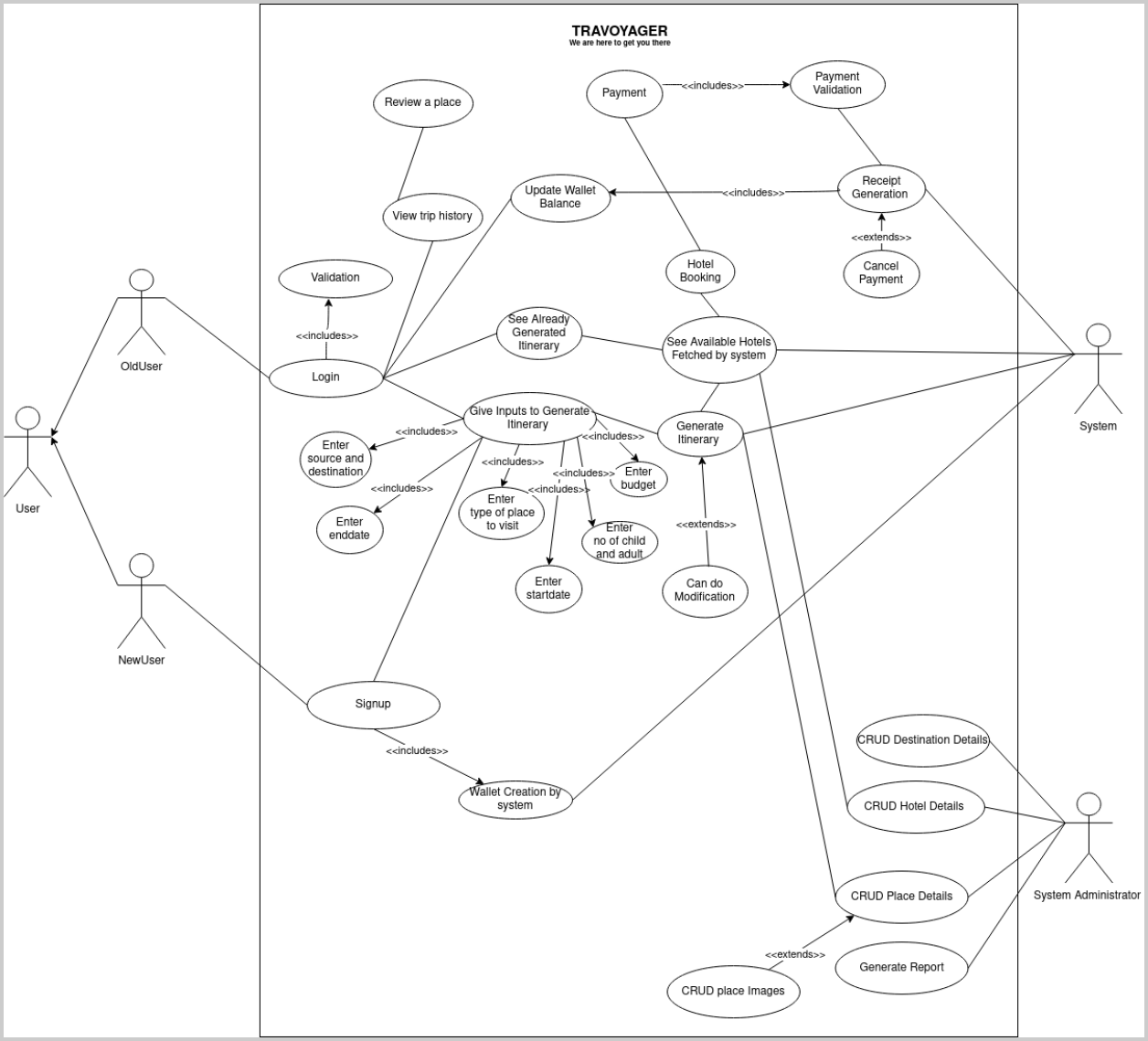
It describes the ability of the system to operate under specific conditions over a period of time. It implies the system to store the details of the clients and their inputs within the system which can be useful in future in case of any kind of failure. This system captures the client input and also makes it available in the history section such that clients can review it anytime.

## 5.7. Extensionability

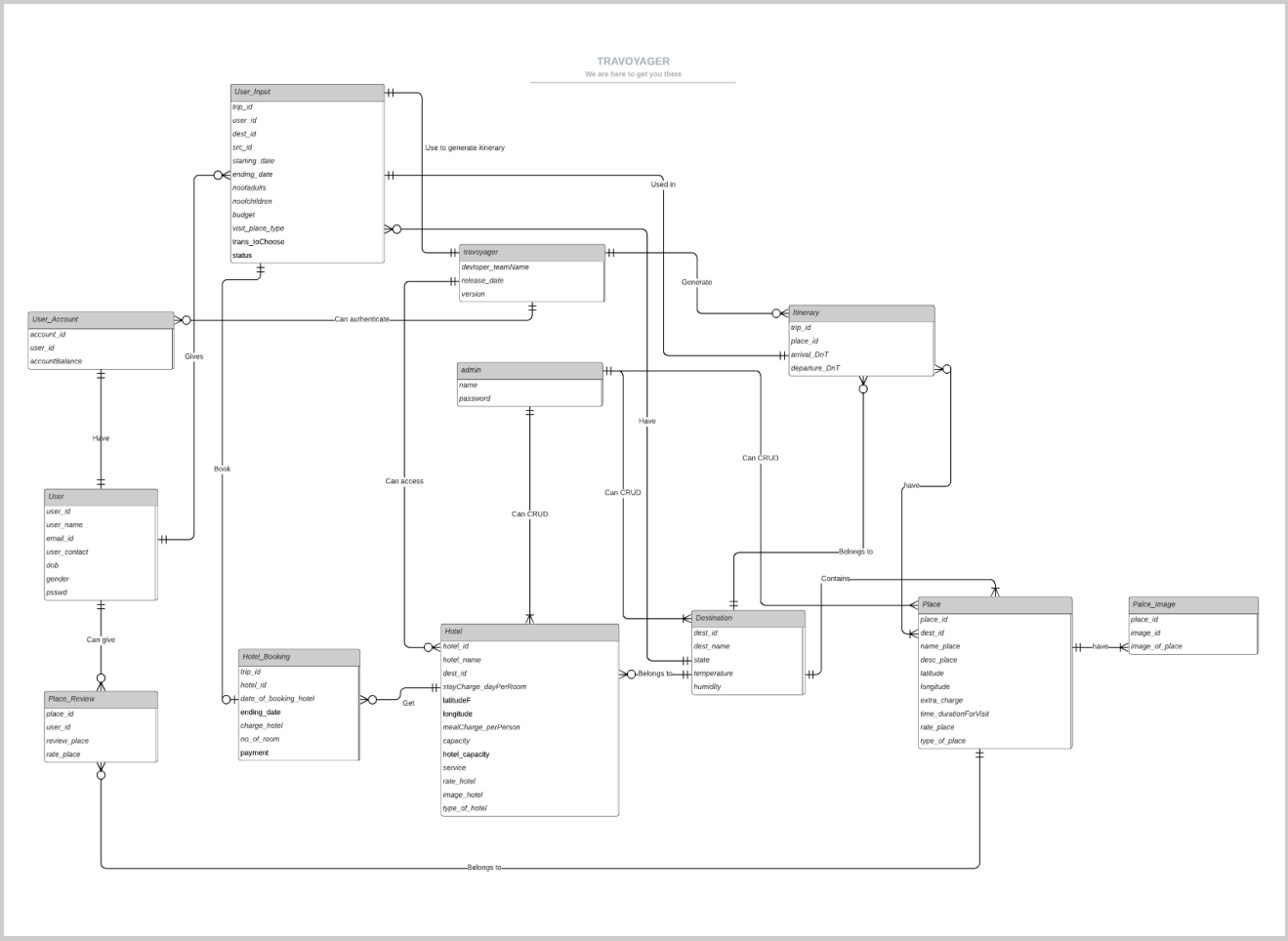
It is a measure of ability of the system to extend itself and make it compatible for all the clients. The extension can include improvement of the system, new functionalities, different modifications etc. The extension can provide enhancement in the system performance. This system includes extending and providing support across different web browsers, operating systems and mobile devices.

# 6. Diagrams

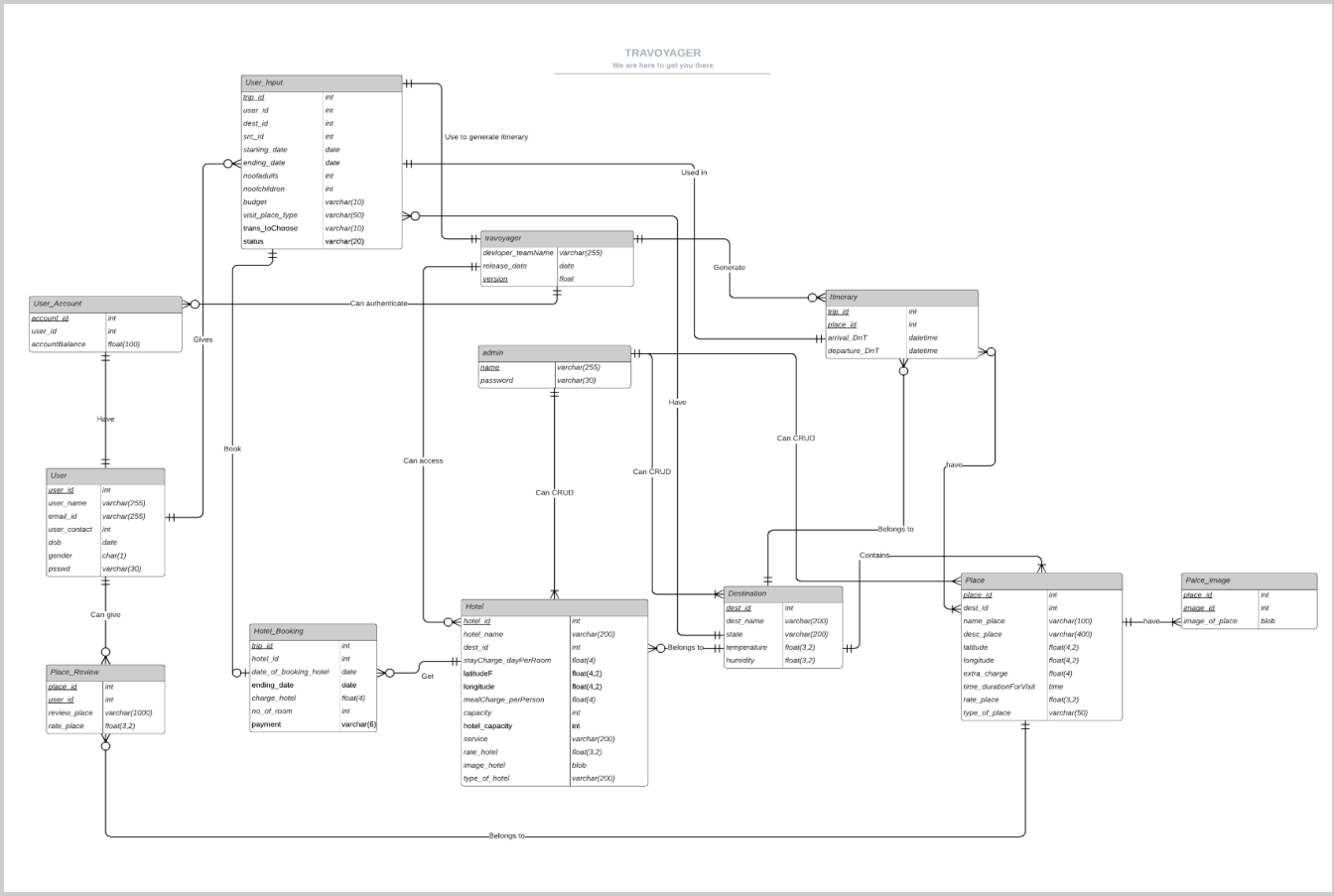
## 6.1. Use Case Diagram



# 6.2. Logical ER Diagram

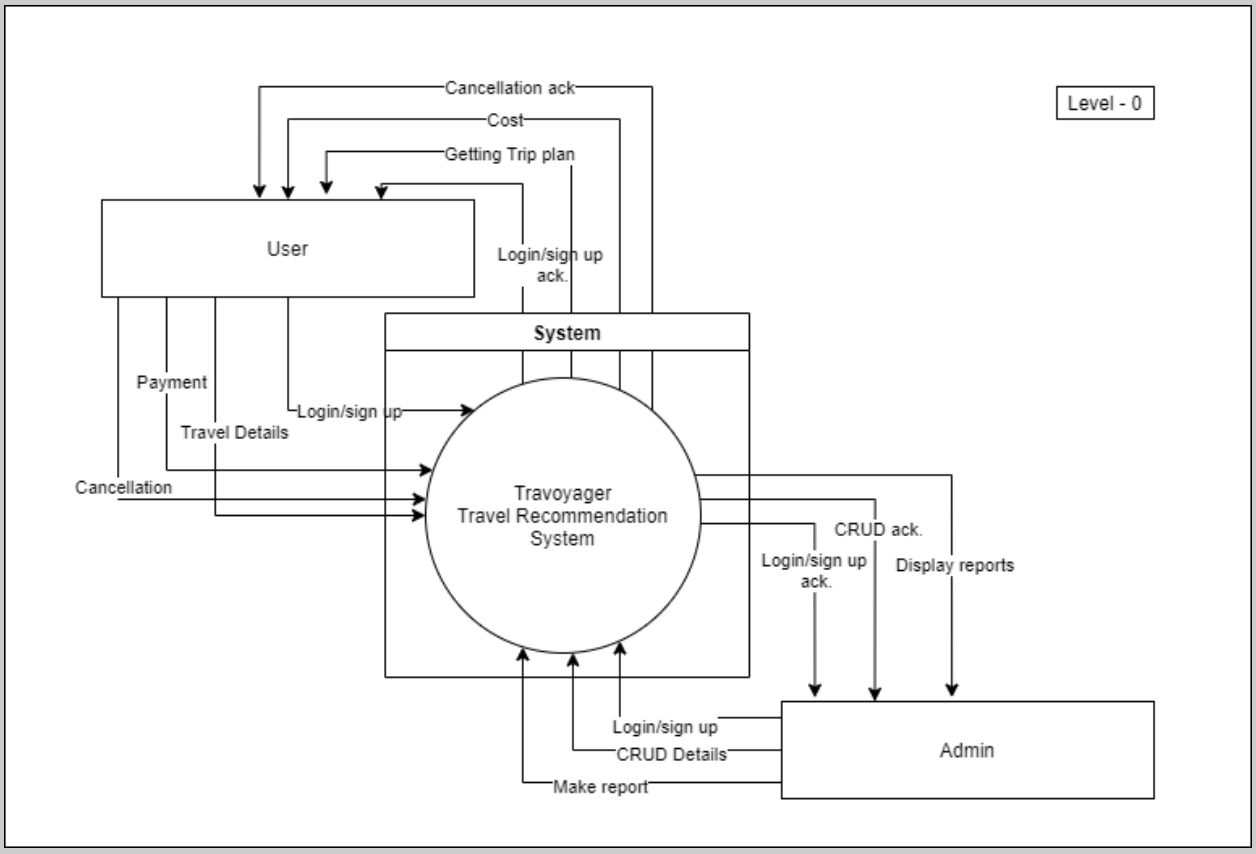


# 6.3. Physical ER Diagram

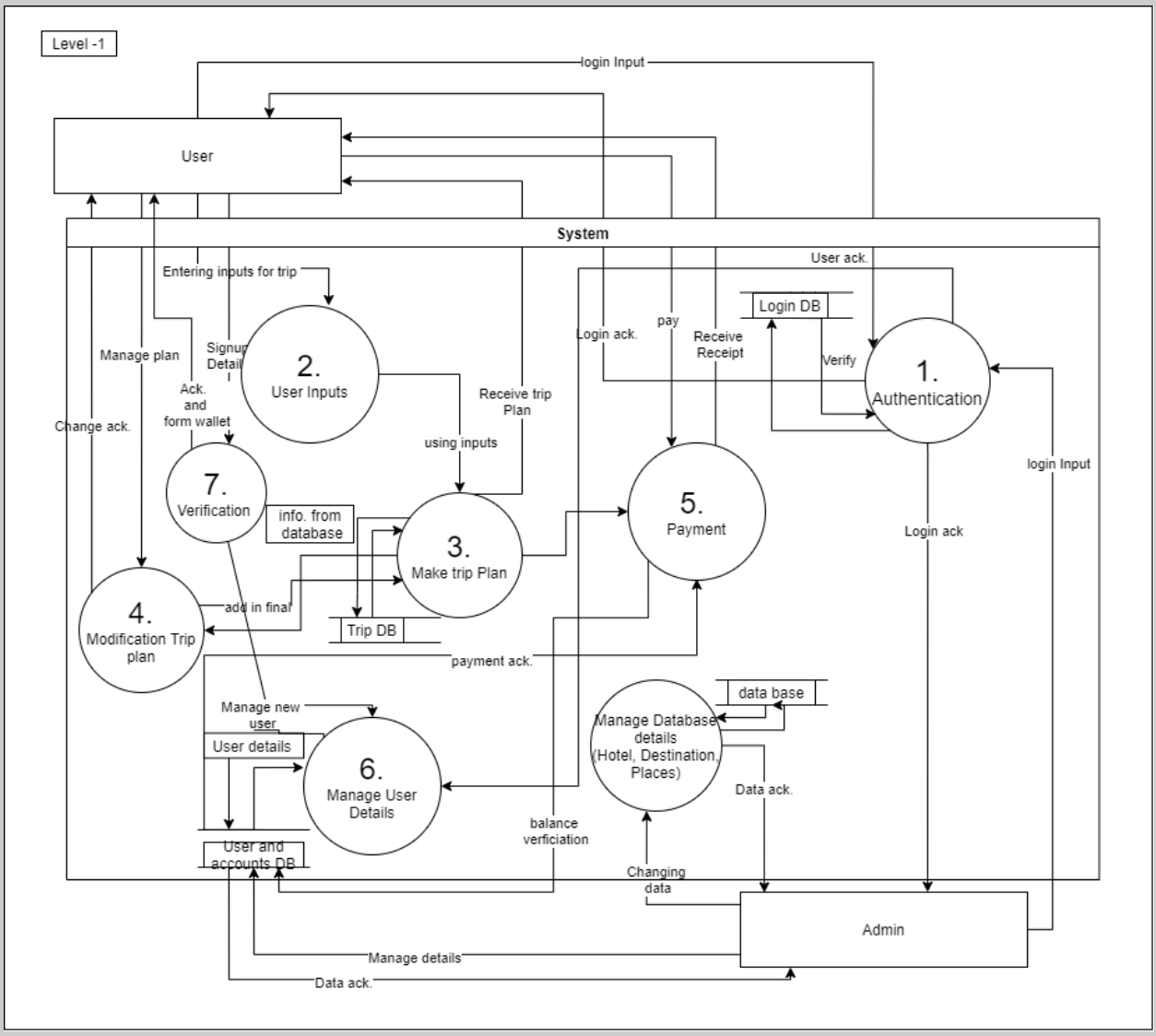


# 6.4. DFD Diagram

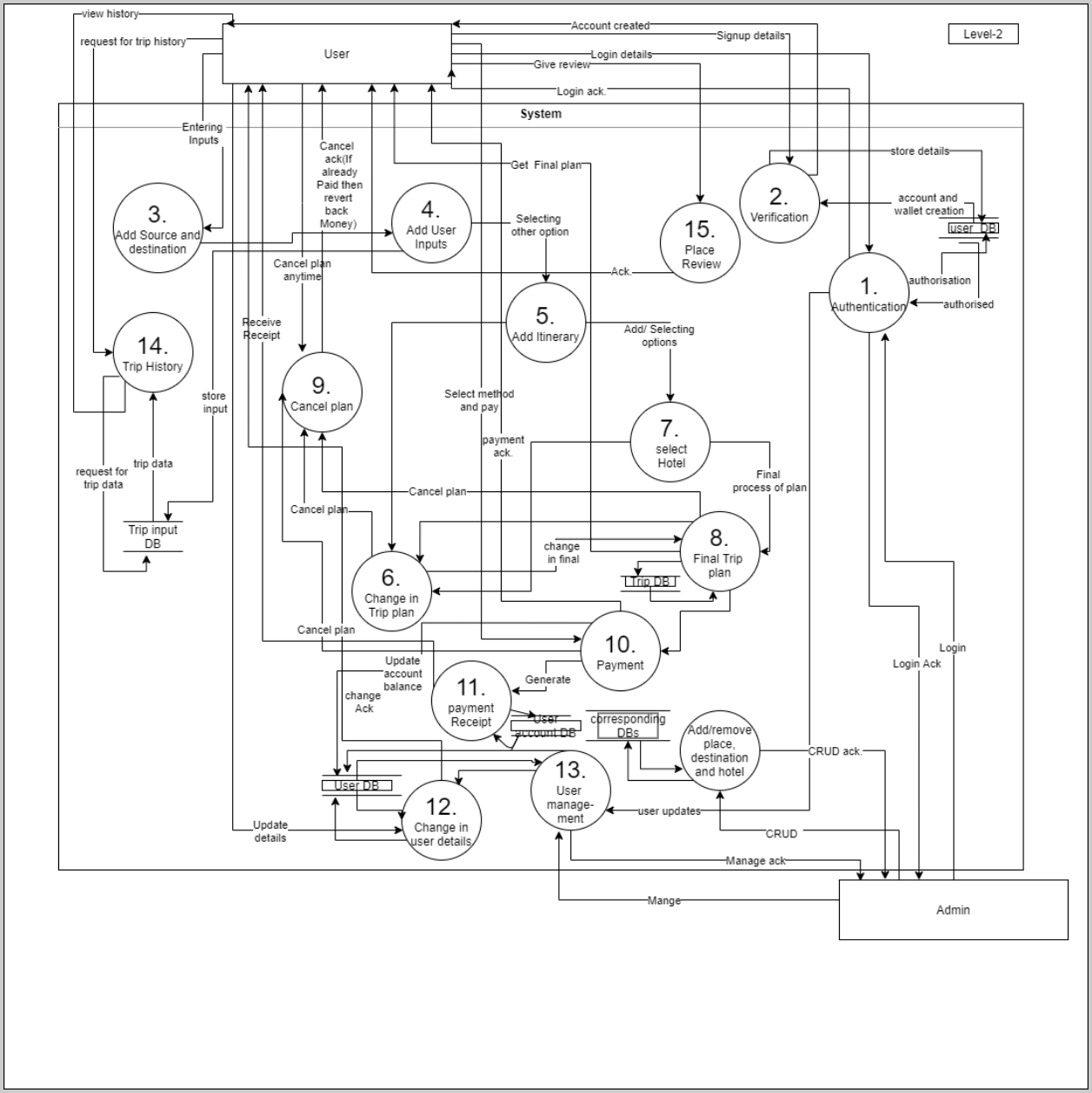
## 6.4.1. Level 0



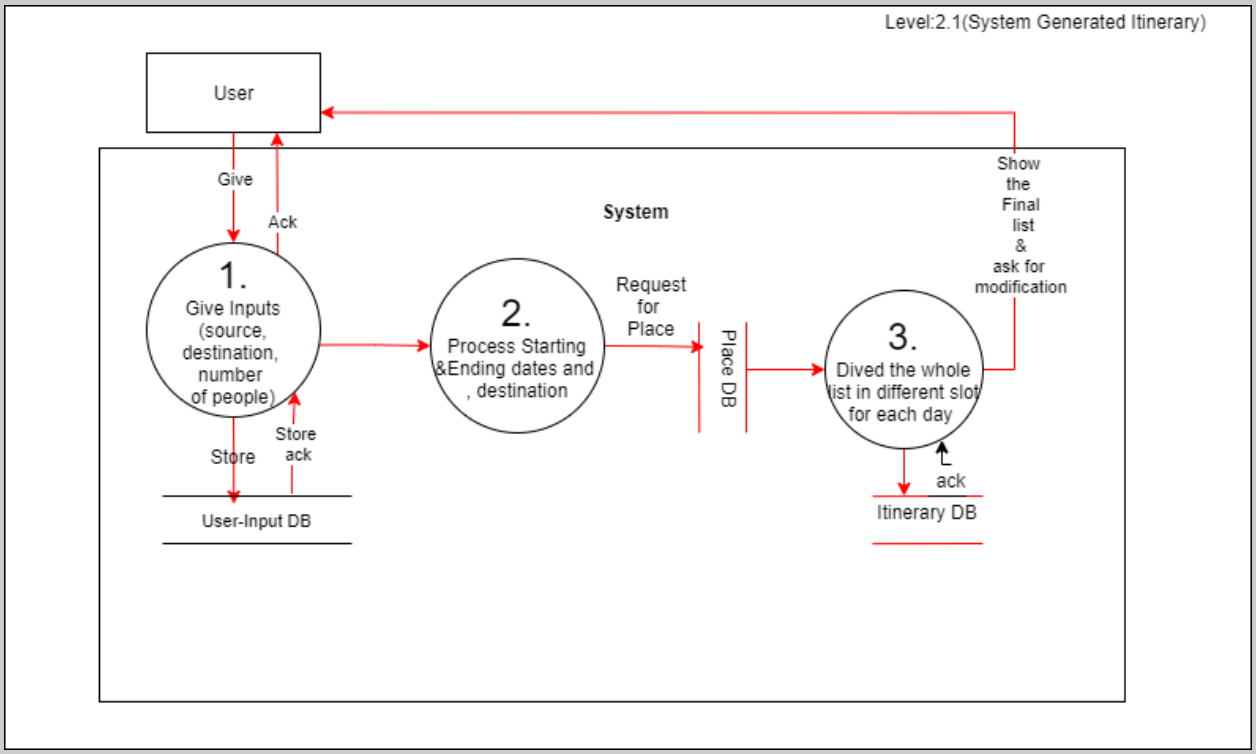
## 6.4.2. Level 1


Level -1
login Input
User
System
User ack.
Entering inputs for trip
Login DB
pay
Login ack.
Receive Receipt
Verify
Manage plan
Signup Detail
2. User Inputs
Receive trip Plan
1. Authentication
Ack. and form wallet
Change ack.
using inputs
login Input
7. Verification
info. from database
5. Payment
3. Make trip Plan
Login ack
4. Modification Trip plan
add in final
Trip DB
payment ack.
data base
Manage new user
Manage Database details (Hotel, Destination, Places)
User details
6. Manage User Details
Data ack.
balance verficiation
User and accounts DB
Changing data
Admin
Manage details
Data ack.

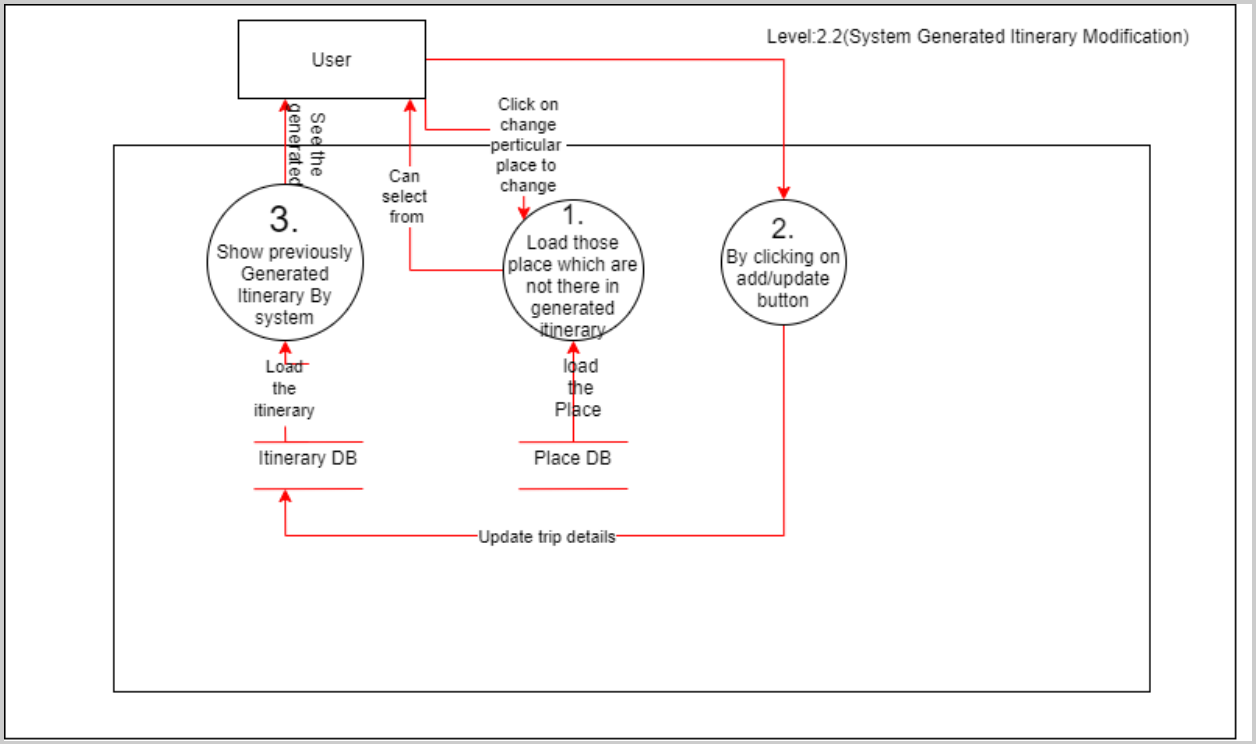
## 6.4.3. Level 2


Level-2
User
System
Admin
view history
request for trip history
Account created
Signup details
Login details
Give review
Login ack.
Entering Inputs
Cancel ack(If already Paid then revert back Money)
Get Final plan
store details
account and wallet creation
user DB
authorisation
authorised
1. Authentication
2. Verification
3. Add Source and destination
4. Add User Inputs
Selecting other option
15. Place Review
Ack.
Cancel plan anytime
5. Add Itinerary
Add/ Selecting options
14. Trip History
Receive Receipt
9. Cancel plan
store input
Select method and pay
payment ack.
7. select Hotel
Final process of plan
request for trip data
trip data
Trip input DB
Cancel plan
change in final
8. Final Trip plan
Trip DB
6. Change in Trip plan
10. Payment
Update account balance
change Ack
11. payment Receipt
Generate
User account DB
corresponding DBs
Add/remove place, destination and hotel
CRUD ack.
User DB
13. User manage-ment
12. Change in user details
Update details
user updates
CRUD
Manage ack
Mange
Login
Login Ack

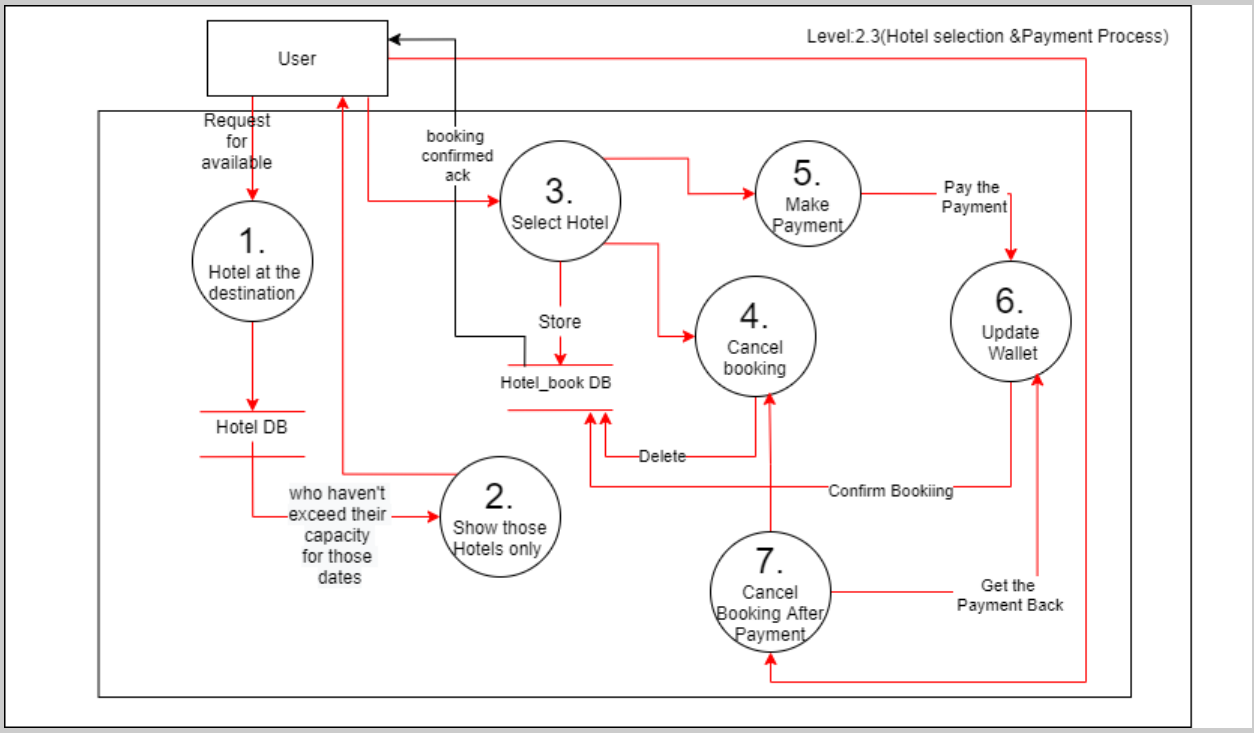
### 6.4.3.1. Level 2.1




### 6.4.3.2. Level 2.2

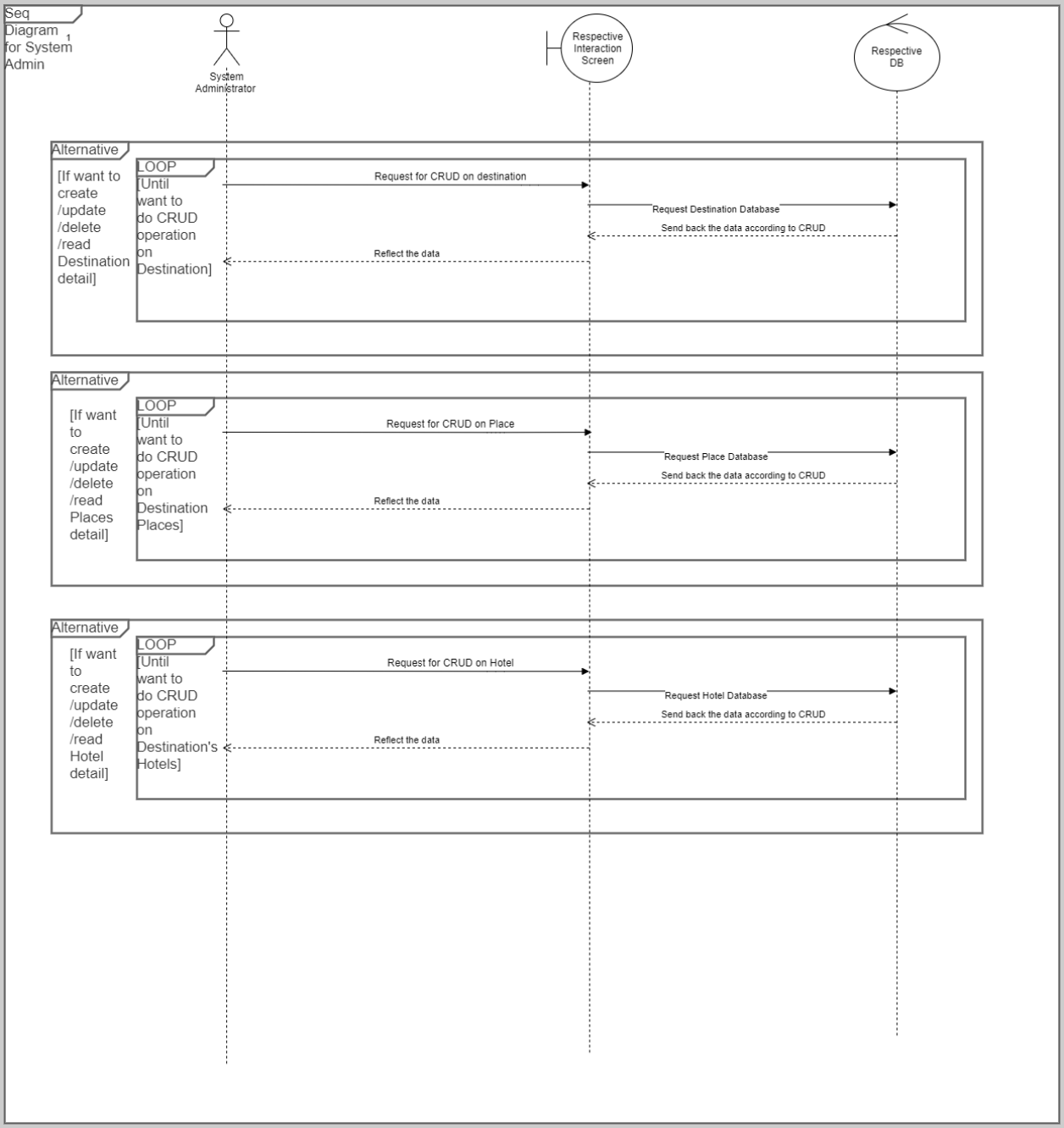


### 6.4.3.3. Level 2.3



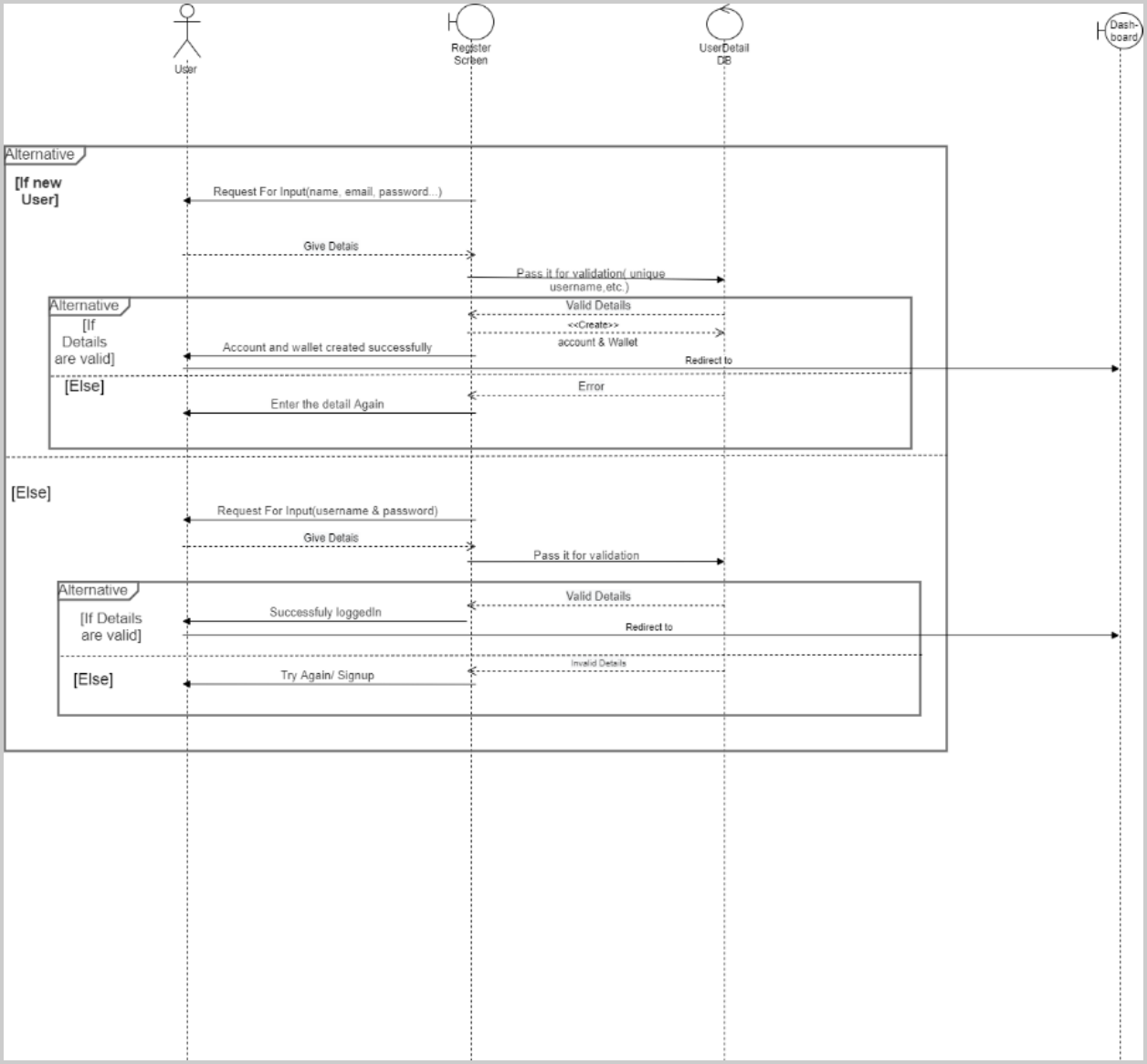
# 6.5. Sequence Diagram

## 6.5.1. Admin Side

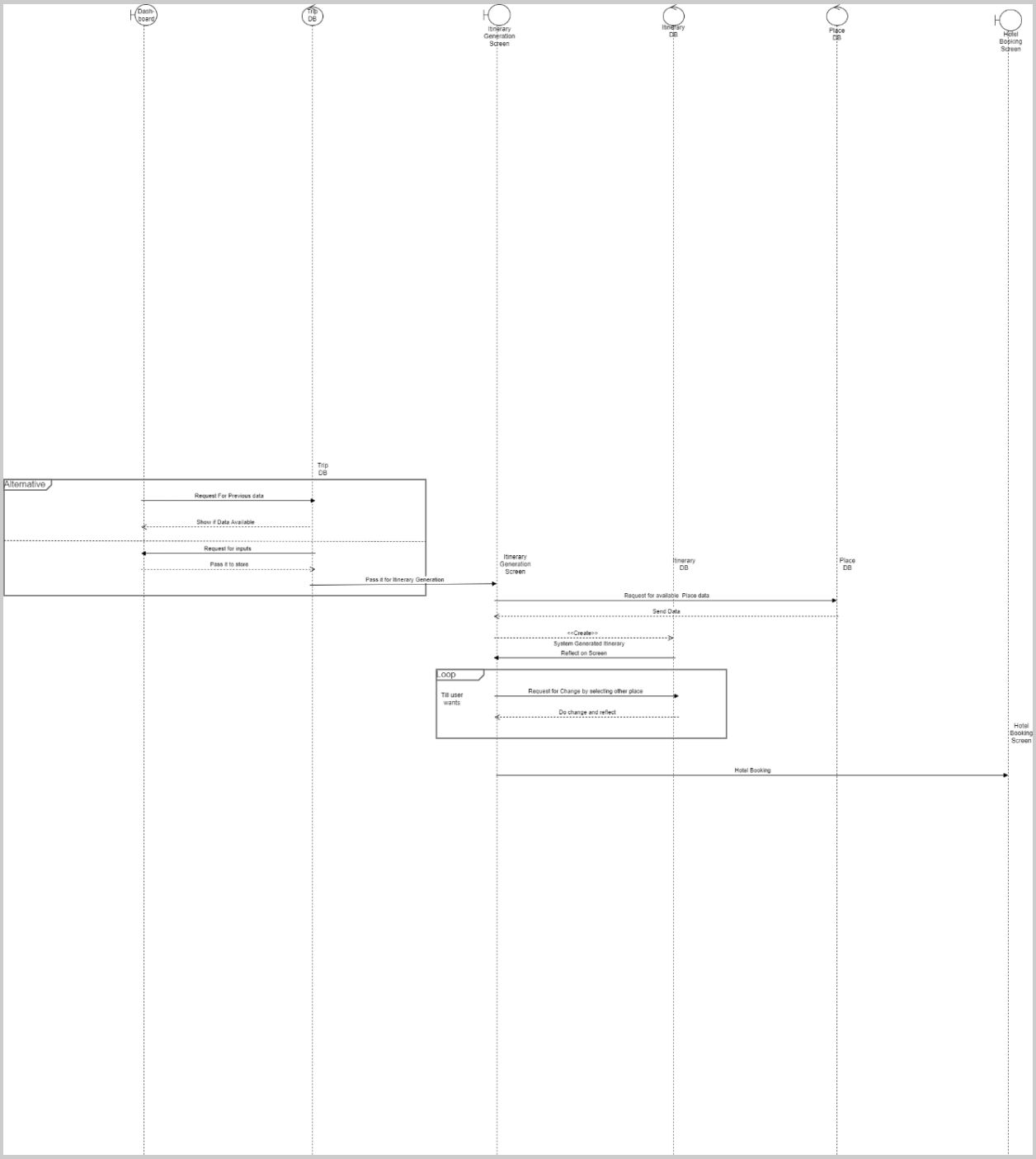


## 6.5.2. User Side

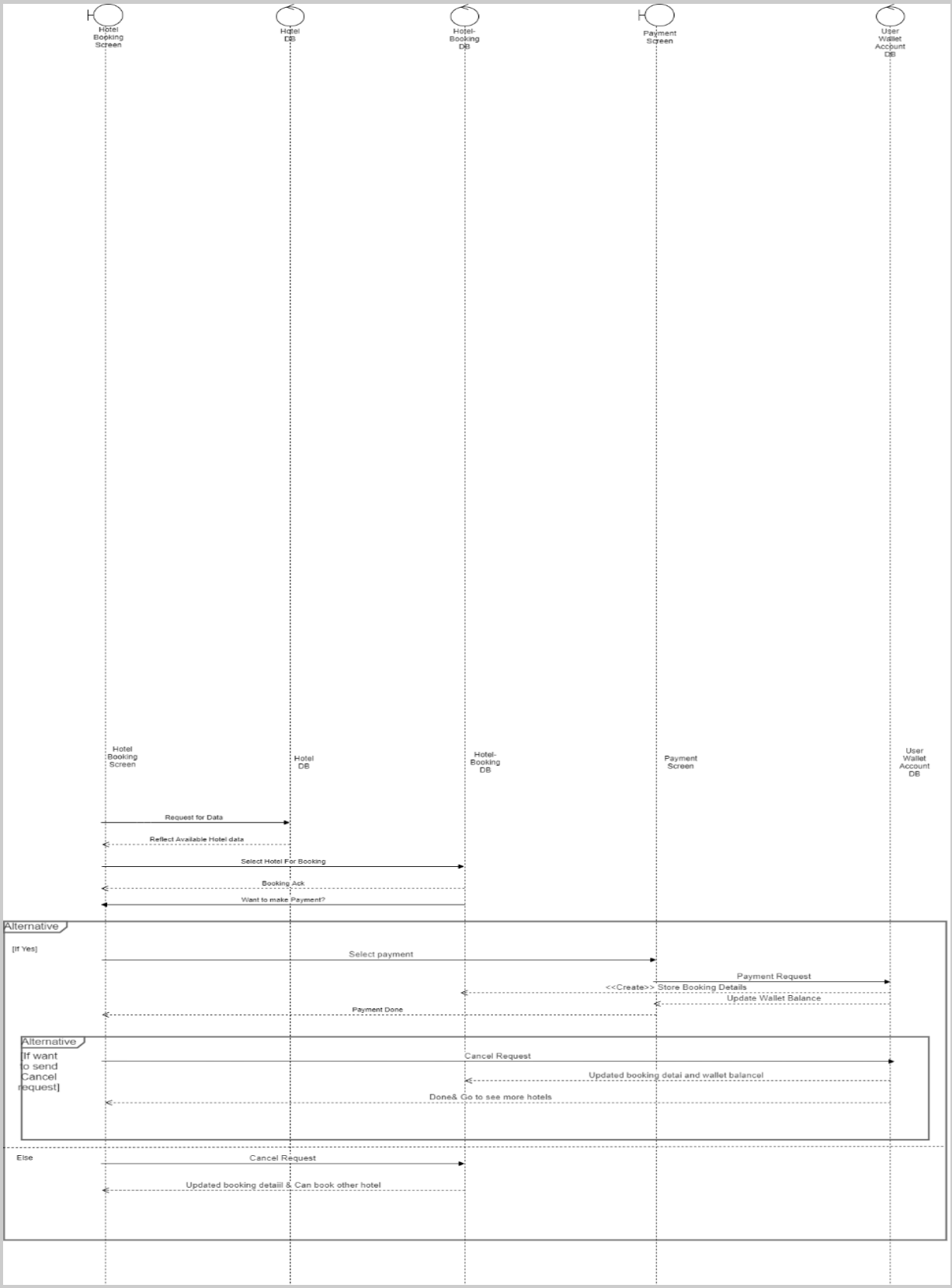
### 6.5.2.1. User Sign-in/Sign-up



## 6.5.2.2. User Itinerary Generation

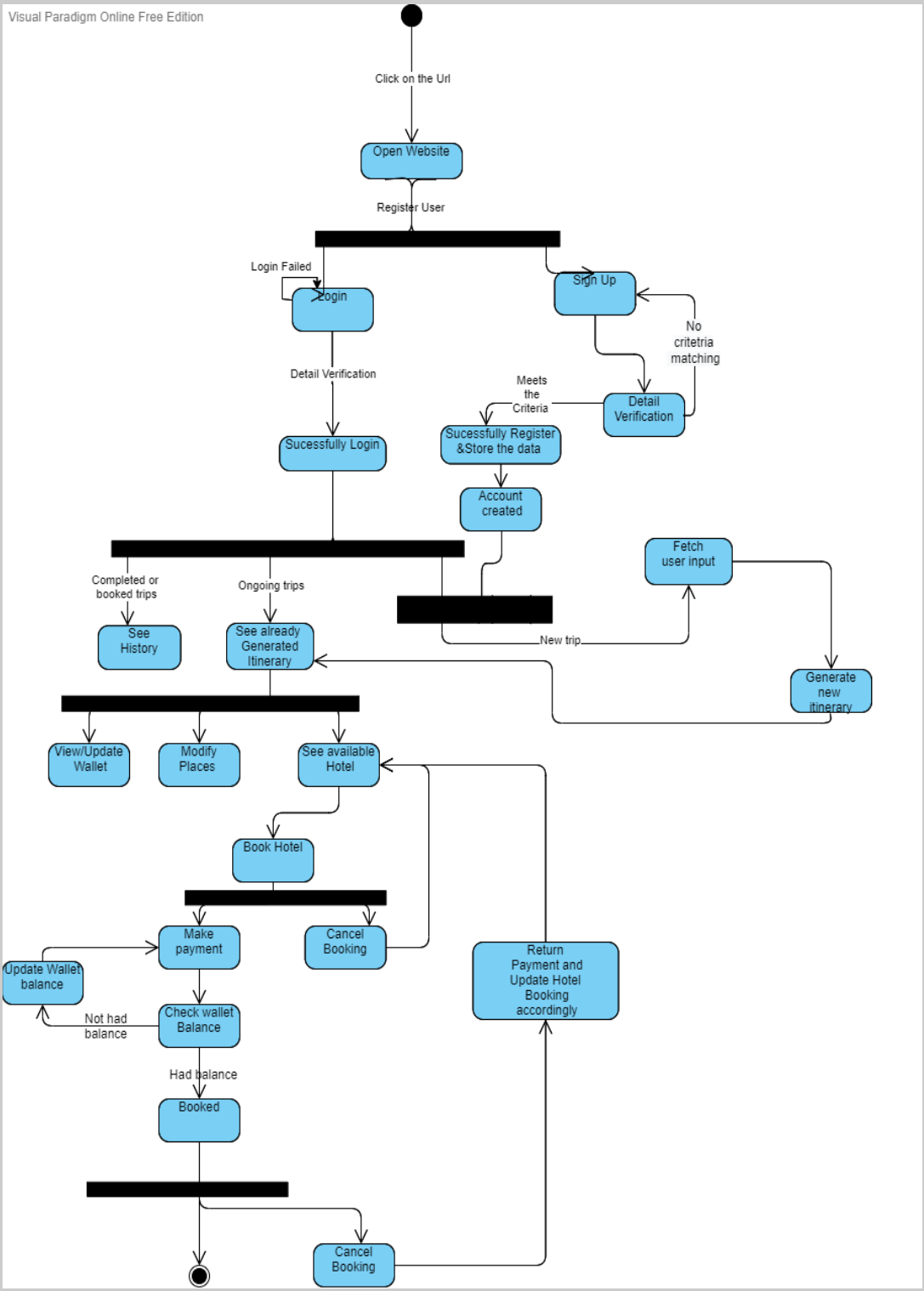


## 6.5.2.3. User Final Booking

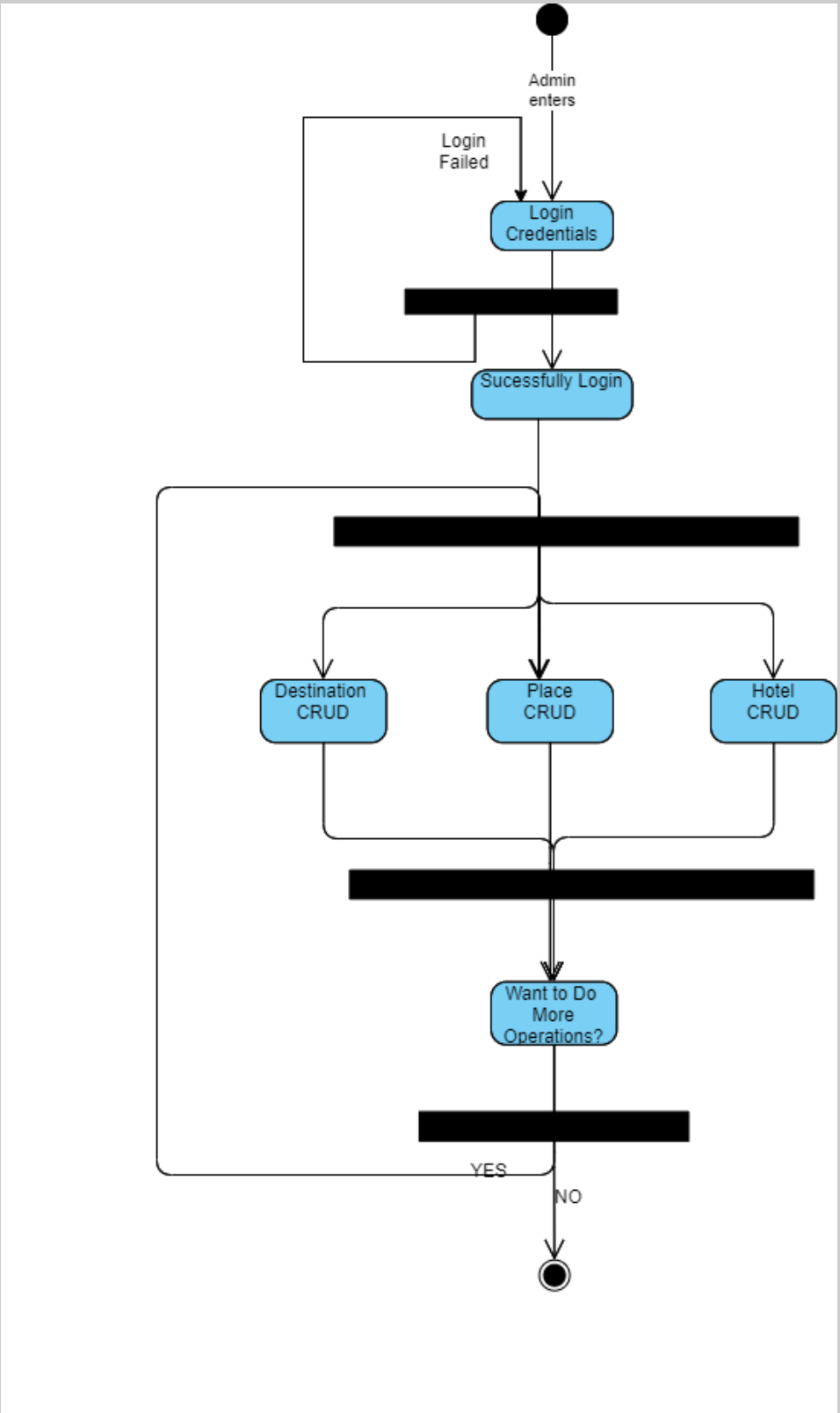


# 6.6. State Diagram

## 6.6.1. User Side



### 6.6.2. Admin Side



# 7. Summary

The SRD gives us detailed information about what are the requirements of the system, which are the functionalities of the system and how the system interacts with other systems and its users. It provides information about the specifications of the system and the user expectations. It describes the appearance of the system to its users and stakeholders. It also depicts the system with user case scenario - use case diagrams, shows how data in the system will flow with - data flow diagram, changing of states due to action of events - using state diagram, relationship among entities of the system for database - Entity Relationship diagram and Flow of activities using- Activity diagram. It helps the developer team to gain understanding about the system which can help them to implement and develop the system well.

# 8. References


[1]https://ddi-dev.com/blog/programming/how-to-write-software-requirement-specification-for-your-project/
[2]https://relevant.software/blog/software-requirements-specification-srs-document/
[3]https://netmind.net/en/business-vs-functional-requirements-who-cares/
[4]https://diceus.com/custom-software-requirements-specification/
[5]https://www.lucidchart.com/blog/software-requirements-documentation