



Sri Lanka Institute of Information Technology

Travel Agency and Tour Planning Project Report

Information Technology Project 2021

Project ID: ITP2021_S2_MTR_G05

Submitted by:

1. IT20172350– (M.P.O.M Gomes)
2. IT20255510– (Y.N Jayasekara)
3. IT20186906– (Tharuka Gayashan F.)
4. IT20046552– (Abeywardahange S.R.D)
5. IT20204648– (Nethu Nipuna M.)
6. IT20163204– (Manimendra N. H.)
7. IT20142964– (Dulakshi Hansini R.M)
8. IT20142728– (Ranawaka T.D)

Submitted to:

Chamari silva

Date of submission

Declaration

We declare that this project report or part of it was not a copy of a document done by any organization, university any other institute or a previous student project group at SLIIT and was not copied from the Internet or other sources.

Project Details

Project Title	Travel Agency and Tour Planning
Project ID	ITP2021_S2_MTR_G05

Group Members

Reg. No	Name	Signature
IT20172350	M.P.O.M Gomes	<u>oshen</u>
IT20255510	Y.N Jayasekara	<u>yasiruJ</u>
IT20186906	Tharuka Gayashan F	<u>Gayshan</u>
IT20046552	Abeywardhanage S.R. D	<u>sachini</u>
IT20204648	Nethu Nipuna M	<u>nipuna</u>
IT20163204	Manimendra N. H.	<u>himanka</u>
IT20142964	Dulakshi Hansini R.M	<u>dulakshi</u>
IT20142728	Ranawaka T. D	<u>thenushka</u>

Abstract

Our client Uniwest wanted to create an attractive website for their travel Agency to make things easier to use for customers. Uniwest presented their needs to us, and our team created this website accordingly.

Therefore, thorough this website project is implemented under eight functions that cover the complete scope of the company. Those functions are accommodation reservation, ticket reservation, ground resource reservation, tour package reservation, travel document management, insurance reservation, event reservation. This website overcomes issues of a manual system such as less security, long access time, and difficulty to make changes.

To implement this project, we use the MERN stack. In the MERN stack, Mongo DB, React js, Express and Node js were used.

Acknowledgement

First of all, we extend our heartfelt gratitude to everyone those who supported and guided us in making our project a success one.

Especially, we need to give a big thanks and our uncommon much gratitude goes the instructor responsible for the Information Technology Project (ITP), the Ms. Chamari de silva for giving her significant time to explain our disarray and calling attention to our shortcomings. Also, we have to give our appreciate to her guidance to success our project.

We would also like to thank for our senior student Mr. Nimesh Madusanka who helps us by giving some instructions to solve our problems that emerged throughout the project and supported us to go ahead.

Also, we would like to give our kind express to Mr. Kasun Indika Kapugmaje the owner of Uniwest travel agency for giving us such attention and choosing us as their developers to create and deploy the automated system.

We want to show thanks to our team members, who have actively supported and emancipated us in making and accomplishing such a great project.

Finally, we would not forget to remember and thank the staff of faculty of computing for guiding us step by step. And for giving this golden opportunity to us.

Table of Contents

Abstract	i
Acknowledgement	iii
Declaration	i
Table of Contents	iv
List of Figures.....	v
List of Tables	x
List of Acronyms and Abbreviations.....	Error! Bookmark not defined.
1. Introduction.....	1
1.1 Problem Statement.....	1
1.2 Product Scope.....	2
1.3 Project Report Structure.....	4
2. Methodology	5
2.1 Requirements and Analysis	5
2.2 Design	255
2.3 Implementation.....	68
2.4 Testing	115
3. Conclusion	131
4. References.....	132

List of Figures

Figure 1: Use case diagram- User Account Management.....	5
Figure 2: Use case diagram- Accommodation reservation management.....	6
Figure 3: Use case diagram- Reservation of Ticket Management.....	7
Figure 4: Use case diagram- Reservation of Ground Resource Management.....	8
Figure 5: Use case diagram- Travel Document Management.....	9
Figure 6: Use case diagram- Tour Package Management.....	10
Figure 7: Use case diagram- Travel Insurance Management.....	11
Figure 8: Use case diagram- Event Management	12
Figure 9: Activity diagram- User Account Management	13
Figure 10: Activity diagram- Accommodation reservation management.....	14
Figure 11: Activity diagram- Reservation of Ticket Management	15
Figure 12: Activity diagram- Reservation of Ground Resource Management(User).....	16
Figure 13: Activity diagram- Reservation of Ground Resource Management(Admin).....	17
Figure 14: Activity diagram- Travel Document Management (User).....	18
Figure 15: Activity diagram- Travel Document Management (Admin).....	19
Figure 16: Activity diagram- Tour Package Management (User).....	20
Figure 17: Activity diagram- Tour Package Management (Admin).....	21
Figure 18: Activity diagram- Travel Insurance Management (User).....	22
Figure 19: Activity diagram- Travel Insurance Management (Admin).....	23
Figure 20: Activity diagram- Event Management.....	24
Figure 21: Entity relationship diagram.....	25
Figure 22: Sequence diagram- User Account Management(User).....	26
Figure 23 Sequence diagram- User Account Management(Admin).....	27
Figure 24: Sequence diagram- Accommodation reservation management (User).....	28
Figure 25: Sequence diagram- Accommodation reservation management (Admin).....	29
Figure 26: Sequence diagram- Reservation of Ticket Management (User).....	30
Figure 27: Sequence diagram- Reservation of Ticket Management (Admin).....	31

Figure 28: Sequence diagram- Reservation of Ground Resource Management (User).....	32
Figure 29: Sequence diagram- Reservation of Ground Resource Management (Admin)	33
Figure 30: Sequence diagram- Travel Document Management (User).....	34
Figure 31: Sequence diagram- Travel Document Management (Admin)	35
Figure 32: Sequence diagram- Tour Package Management (User).....	36
Figure 33: Sequence diagram- Tour Package Management (Admin)	37
Figure 34: Sequence diagram- Travel Insurance Management (User).....	38
Figure 35: Sequence diagram- Travel Insurance Management (Admin)	39
Figure 36: Sequence diagram- Event Management.....	40
Figure 37: User Interface – Login Page.....	41
Figure 38: User Interface – User Details Page.....	41
Figure 39: User Interface - User Details Edit Page.....	42
Figure 40: User Interface - Admin Dashboard page.....	42
Figure 41: User Interface – Admin User Management page.....	43
Figure 42: User Interface – Accommodation view page.....	43
Figure 43: User Interface – Selected Accommodation details view page.....	44
Figure 44: User Interface – User Details for Accommodation page.....	44
Figure 45: User Interface – User’s Reservation of Accommodation page.....	45
Figure 46: User Interface – List of Accommodation page.....	45
Figure 47: User Interface – Insert new Accommodation page.....	46
Figure 48: User Interface – Update Accommodation page.....	46
Figure 49: User Interface – Reservation list of Accommodation page.....	47
Figure 50: User Interface – Ticket Category page.....	47
Figure 51: User Interface – Airplane Ticket Reservation page.....	48
Figure 52: User Interface – Ticket Reservation Admin page.....	48
Figure 53: User Interface – Airplane Ticket insert Admin page.....	49
Figure 54: User Interface – Airplane Ticket view Admin page.....	49
Figure 55: User Interface – Airplane Ticket Update Admin page.....	50
Figure 56: User Interface – Airplane Ticket Delete Admin page.....	50
Figure 57: User Interface – Ground Resource category page.....	51
Figure 58: User Interface – Car reservation category page.....	51

Figure 59: User Interface – Car reservation page.....	52
Figure 60: User Interface – Ground resource Admin page.....	52
Figure 61: User Interface – Car reservation list view Admin page.....	53
Figure 62: User Interface – Car reservation update Admin page.....	53
Figure 63: User Interface – Car reservation delete Admin page.....	54
Figure 64: User Interface –Travel Document Add page.....	54
Figure 65: User Interface – Travel Document Admin page.....	55
Figure 66: User Interface – Customer’s Travel Document Added list view page.....	55
Figure 67: User Interface – Customer’s Travel Document update page.....	56
Figure 68: User Interface – Customer’s Travel Document Delete page.....	56
Figure 69: User Interface – Tour Package category view page.....	57
Figure 70: User Interface – Selected Tour Package Detail view page.....	58
Figure 71: User Interface – Selected Tour Package Payment page.....	58
Figure 72: User Interface – Tour package Admin page.....	59
Figure 73: User Interface – Tour package update Admin page.....	59
Figure 74: User Interface – Tour package Delete Admin page.....	60
Figure 75: User Interface – Tour Package reservation list view admin page.....	60
Figure 76: User Interface – Tour Package reservation update admin page	61
Figure 77: User Interface – Tour Package reservation delete admin page	61
Figure 78: User Interface – Insurance category view page	62
Figure 79: User Interface – Selected Insurance Details view page.....	62
Figure 80: User Interface – Selected Insurance Details Resevarition page.....	63
Figure 81: User Interface – Insurance Admin page.....	63
Figure 82: User Interface – Insurance type list view admin page.....	64
Figure 83: User Interface – Insurance type Add Admin page.....	64
Figure 84: User Interface – Insurance type update admin page.....	65
Figure 85: User Interface – Insurance type delete admin page.....	65
Figure 86: User Interface – Insurance Resevarition list Admin page	66
Figure 87: User Interface – Event category view page.....	66

Figure 88: User Interface – Event add page.....	67
Figure 89: User Interface – Event add list view page.....	67
Figure 90: Code – User Management(Models).....	68
Figure 90: Code – User Management(Models).....	69
Figure 91: Code – User Management(Routes).....	69
Figure 91: Code – User Management(Routes).....	70
Figure 92: Code – User Management (Frontend).....	70
Figure 92: Code – User Management (Frontend).....	71
Figure 92: Code – User Management (Frontend).....	72
Figure 93: Code – Accommodation reservation management (Models).....	72
Figure 93: Code – Accommodation reservation management (Models).....	73
Figure 94: Code – Accommodation reservation management (Routes).....	73
Figure 94: Code – Accommodation reservation management (Routes).....	74
Figure 95: Code – Accommodation reservation management (Frontend).....	74
Figure 95: Code – Accommodation reservation management (Frontend).....	75
Figure 95: Code – Accommodation reservation management (Frontend).....	76
Figure 95: Code – Accommodation reservation management (Frontend).....	77
Figure 95: Code – Accommodation reservation management (Frontend).....	78
Figure 95: Code – Accommodation reservation management (Frontend).....	79
Figure 96: Code – Reservation of Ticket Management (Models).....	80
Figure 97: Code – Reservation of Ticket Management (Routes).....	80
Figure 97: Code – Reservation of Ticket Management (Routes).....	81
Figure 98: Code – Reservation of Ticket Management (Frontend)	82
Figure 98: Code – Reservation of Ticket Management (Frontend)	83
Figure 98: Code – Reservation of Ticket Management (Frontend)	84
Figure 98: Code – Reservation of Ticket Management (Frontend)	85
Figure 99: Code – Reservation of Ground Resource Management (Models).....	86
Figure 100: Code – Reservation of Ground Resource Management (Routes).....	86
Figure 100: Code – Reservation of Ground Resource Management (Routes).....	87
Figure 101: Code – Reservation of Ground Resource Management (Frontend)	88
Figure 101: Code – Reservation of Ground Resource Management (Frontend)	89

Figure 101: Code – Reservation of Ground Resource Management (Frontend)	90
Figure 101 Code – Reservation of Ground Resource Management (Frontend)	91
Figure 102: Code – Travel Document Management (Models).....	91
Figure 103: Code – Travel Document Management (Routes).....	92
Figure 103: Code – Travel Document Management (Routes).....	93
Figure 104: Code – Travel Document Management (Frontend).....	93
Figure 104: Code – Travel Document Management (Frontend).....	94
Figure 104: Code – Travel Document Management (Frontend).....	95
Figure 105: Code – Tour Package Management (Models).	96
Figure 106: Code – Tour Package Management (Routes).....	97
Figure 106: Code – Tour Package Management (Routes).....	98
Figure 107: Code – Tour Package Management (Frontend).....	99
Figure 107: Code – Tour Package Management (Frontend).....	100
Figure 107: Code – Tour Package Management (Frontend).....	101
Figure 107: Code – Tour Package Management (Frontend).....	102
Figure 107: Code – Tour Package Management (Frontend).....	103
Figure 108: Code – Insurance Management (Models).....	104
Figure 109: Code – Insurance Management (Routes).....	104
Figure 109: Code – Insurance Management (Routes).....	105
Figure 110: Code – Insurance Management (Frontend).....	106
Figure 110: Code – Insurance Management (Frontend).....	107
Figure 110: Code – Insurance Management (Frontend).....	108
Figure 110: Code – Insurance Management (Frontend).....	109
Figure 110: Code – Insurance Management (Frontend).....	110
Figure 110: Code – Insurance Management (Frontend).....	111
Figure 111: Code – Event Management (Routes).....	112
Figure 111: Code – Event Management (Routes).....	113
Figure 112: Code – Event Management (Frontend)	113
Figure 112: Code – Event Management (Frontend)	114

List of Tables

Table 1: Testing – User Registration and Sign-in into the system	115
Table 2: Testing – Admin can delete and view user profiles	116
Table 3: Testing – Book an accommodation	117
Table 4: Testing – insert an accommodation	118
Table 5: Testing – Airline Tickets Booking	119
Table 6: Testing- Insert Bus Details	120
Table 7: Testing – Add New Travel Guide	121
Table 8: Testing – Reserve a car	122
Table 9: Testing – Add Travel Document	123
Table 10: Testing – Update User’s Travel Document	124
Table 11: Testing – <u>User input No of adults and child calculate the price for the tour package</u>	125
Table 12: Testing – <u>Admin can view, add, update, delete tour packages</u>	126
Table 13: Testing – Add new insurance type	127
Table 14: Testing – Insurance Reservation	128
Table 15: Testing – Add a new event	129
Table 16: Testing – Delete an added event	130

1. Introduction

1.1 Problem Statement

Client name is Uniwest travel agency which located in Nupe, Matara. It all started about ten years ago. This company which started by providing only the service of Student Recruitment for student visa. Now they are planning to provide all the services as a travel agency with that Student Recruitment for student visa. Therefore, they now entered into a complex business process. Therefore, they felt the need for a specific website as they wanted to provide a clear image for their organization to the customer while providing services as a more professional and reliable travel agency. In addition, they wanted to maintain a systematic database. So, they contacted us and explained their need. There were several requirements they put forward.

1. Creating a website that ensures customer attraction and trust.
2. Maintain a systematic database.
3. Give Various of categories for the customer to select for their accommodation reservations.
4. Issuing tickets for Airplanes, trains, buses, and events.
5. Facilitate customers to plan and reserve their tour packages.
6. To give the customer a complete understanding of the services provided by their company and their prices.
7. Facilitate customer to reserve ground resources for their needed.
8. Maintain travel documents of customers and give essential details about travel documents.
9. Issuing travel insurance for their travels with trusted insurance companies.
10. Facilitate customers to plan and reserve their events.

Our team worked tirelessly to meet these needs. As a result, Uniwest now has a well-maintained website and a dedicated database. Through this website, many of their basic needs were satisfied and they were able to save a lot of unnecessary expenses. This will enable Uniwest owner to promote their company while providing a more efficient and reliable service to their customers.

1.2 Product Scope

To ease the process of achieving the above-mentioned goals, the System for Uniwest travel agency was divided into eight functions.

1. User Account management
2. Accommodation reservation Management
3. Reservation of Tickets Management
4. Reservation of ground Management
5. Travel Document and Travel information Management
6. Tour Package Management
7. Travel Insurance Management
8. Event Management

1.2.1 User Account Management

This function provides hotel users to create their user profiles to get the service online. Users can log in using provided username and password, logout, change profile details anytime, and also they can delete their account from the system.

When the Employee login the system automatically redirects to the relevant user interfaces. Once a user account is created users can see the hotel dynamic web page.

1.2.2 Accommodation reservation Management

From this function manage accommodation reservations. Maintaining accommodations, prices and display to customer available accommodations. provide filters to advance search. Provide environment to book accommodations and payments. Manage the role of accommodation reservation manager (adding new accommodations to data base, update information, and manage reservation database)

1.2.3 Reservation of Tickets Management

In this function customer can select types of tickets easily. Bus tickets, Railway, airline tickets and events ticket available on. Customer can select each destination, starting points, fixed date they want. They can easily pay by Master and visa card to book their tickets. All the data save in our database as types of Ticket tables and Reservation table. Admin can add, Delete, and modify tickets and get a report about list of people who have booked tickets on daily basis

1.2.4 Reservation of ground Management

This function allows customers to book taxis and cars. In addition to booking a guide to make their destination easier. Also, the system admin can add new vehicles, guides, and judges to the system. And to change and remove their details. In addition, the system allows the admin to retrieve data as a report if desired.

1.2.5 Travel Document and Travel information Management

This function handling enquiries of travel destination details. Customer can know any doubts about travel information with this. And, customer want to apply a visa, they can send their travel document and set their visa from via through with our agency. That will make work easier for travelers.

As an admin side on this function, staff can communicate with their customer if they have any questions about travel information. And when customer send their travel document, staff can check their documents and process it to apply the visa.

1.2.6 Tour Package Management

This function allows to customer select and reserve a tour package at the discretion of the customer. As well customer can cancel his/her reservation with fill cancelation form.

From the admin side staff member can manage packages. Staff Member can add, update, and delete specific packages according to his/her opinion. And allows getting

1.2.7 Travel Insurance Management

This function allows the customer to choose the type of travel insurance they want. The types of travel insurance selected by the customer are stored in the travel insurance database. Also, on the admin side the administrator can manage the travel insurance type in travel insurance type. The administrator can add new travel insurance types, edit and delete existing ones. The administrator can also manage the travel insurance database and the administrator can obtain a report of the stored details if required

1.2.8 Event Management

This function allows to customers to be aware of entertaining events which including in their tour packages. Then customer can choose a tour package that including proper events. Also, the staff members and guides can be aware of their meetings and conferences.

Event manager can schedule meetings and conferences and he can add new events to the packages. Also, event manager updates the event and meeting schedules.

1.3 Project Report Structure

First Section:

As the first section it mainly focused on client's problems which they have while using manual system and client's requirements. Therefore, in this part gave the solutions for that problem and include the overall project plan and the scope of the project by us.

Second Section:

In this section focused on methodology part which contain the clarification of Requirement Analysis, Designs, Implementation, and the Testing. Use case Diagrams, Activity Diagrams and sequence diagrams are used for describing Requirement Analysis part. The Design part is classified utilizing the guide of the ER diagram and the snapshots of user Interfaces. For the Implementation part there are Snapshots of the Implementations and finally there are test cases for the test areas.

Third Section:

This section indicates the information of evaluation and accomplishment of the whole project and the new knowledge which are learned through this project.

Forth Section:

This section indicates all the references which used to full fill this project

2. Methodology

2.1 Requirements and Analysis

2.1.1 User Account management

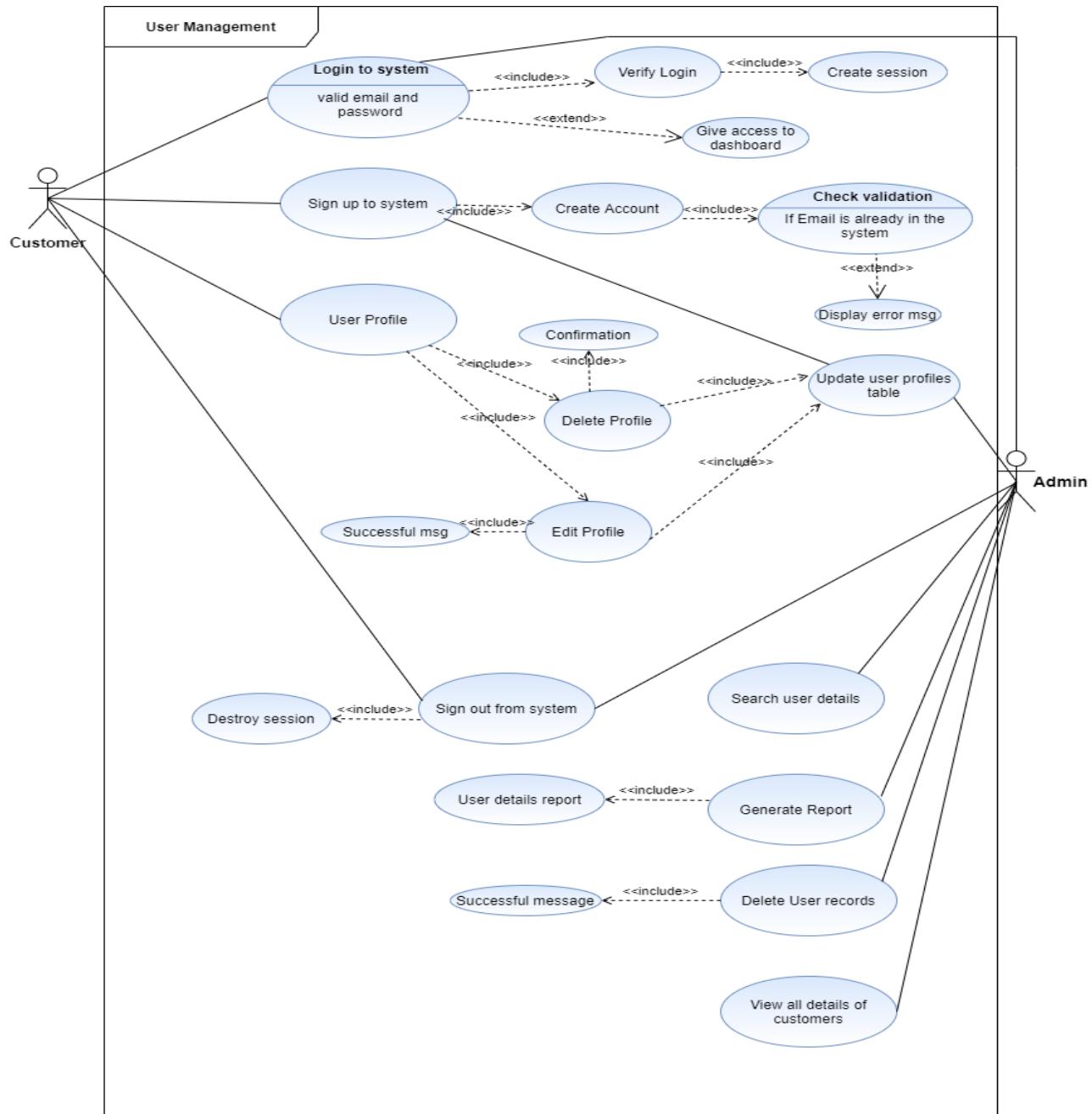


Figure 1

2.1.2 Accommodation reservation Management

Accommodation management



Figure 2

2.1.3 Reservation of Tickets Management

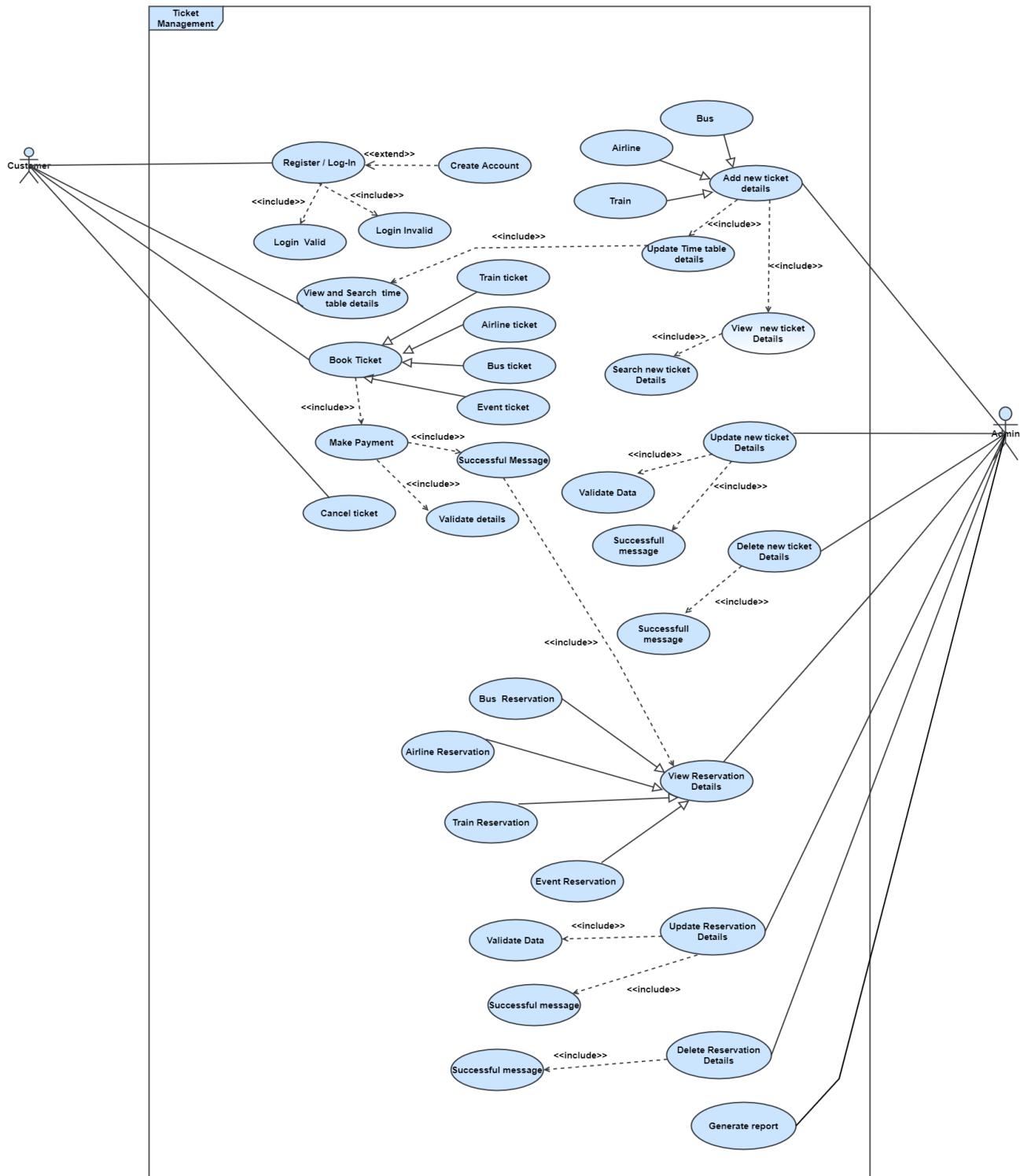


Figure 3

2.1.4 Reservation of ground Management



Figure 4

2.1.5 Travel Document and Travel information Management



Figure 5

2.1.6 Tour Package Management

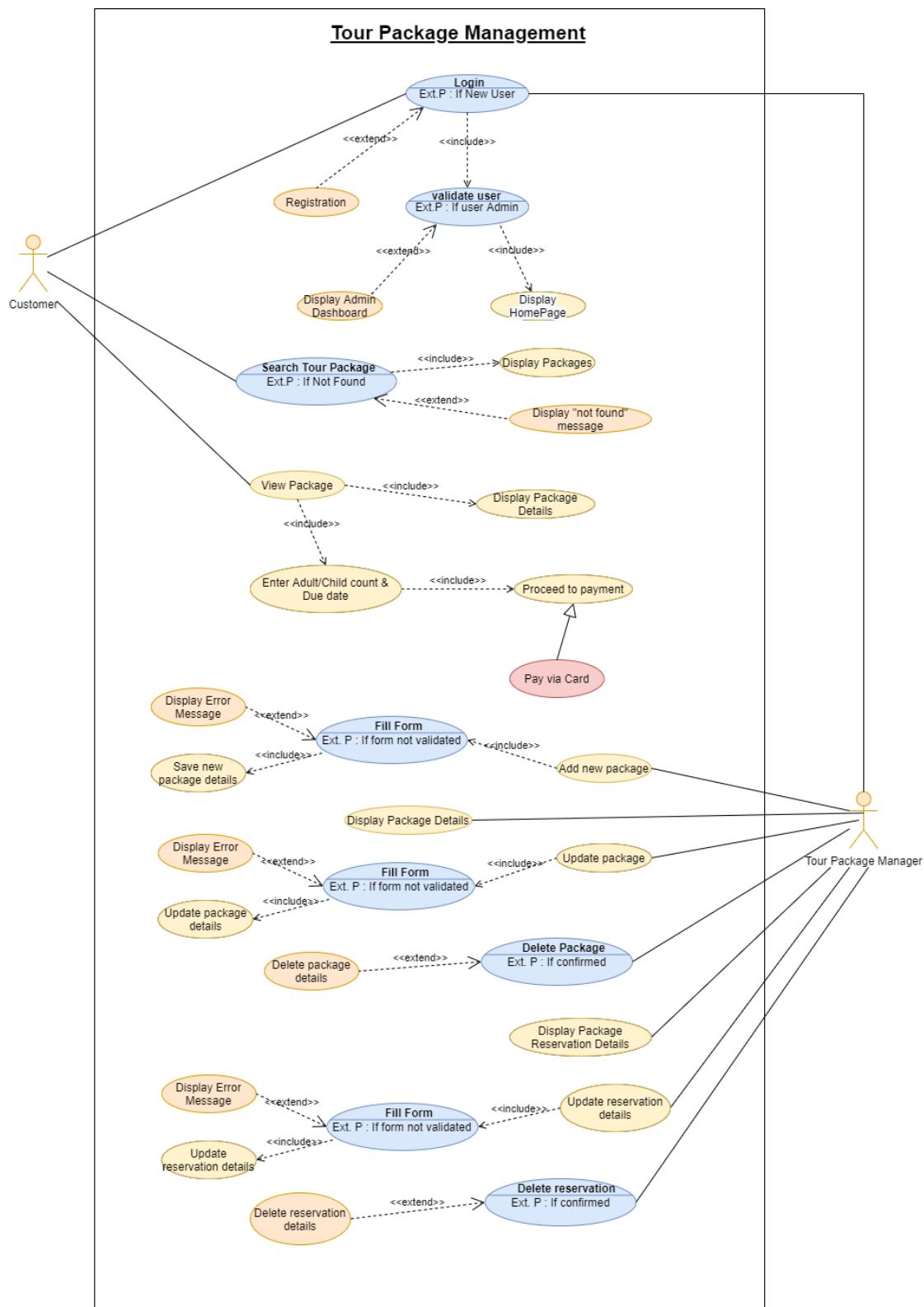


Figure 6

2.1.7 Travel Insurance Management

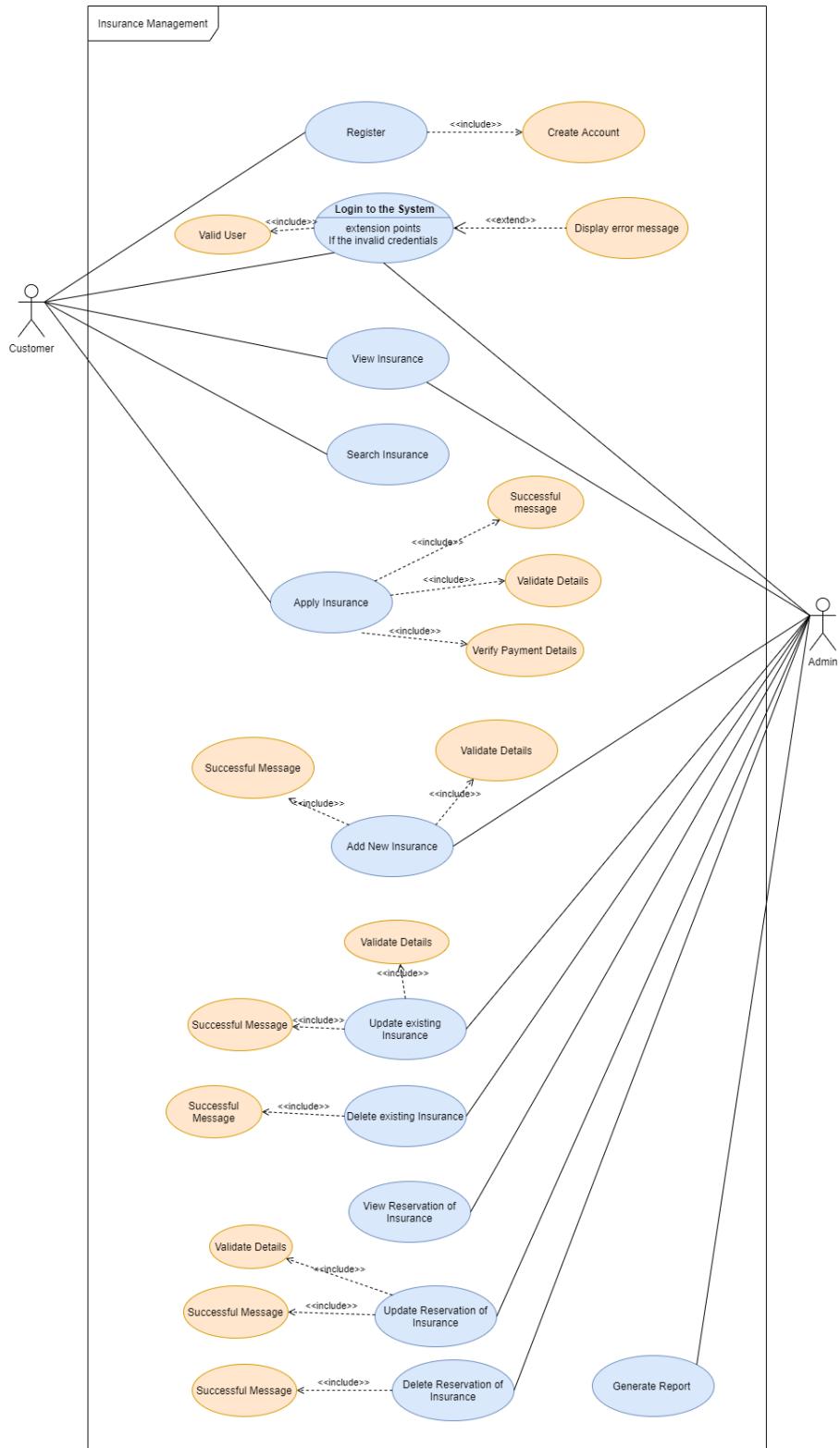


Figure 7

2.1.8 Event Management

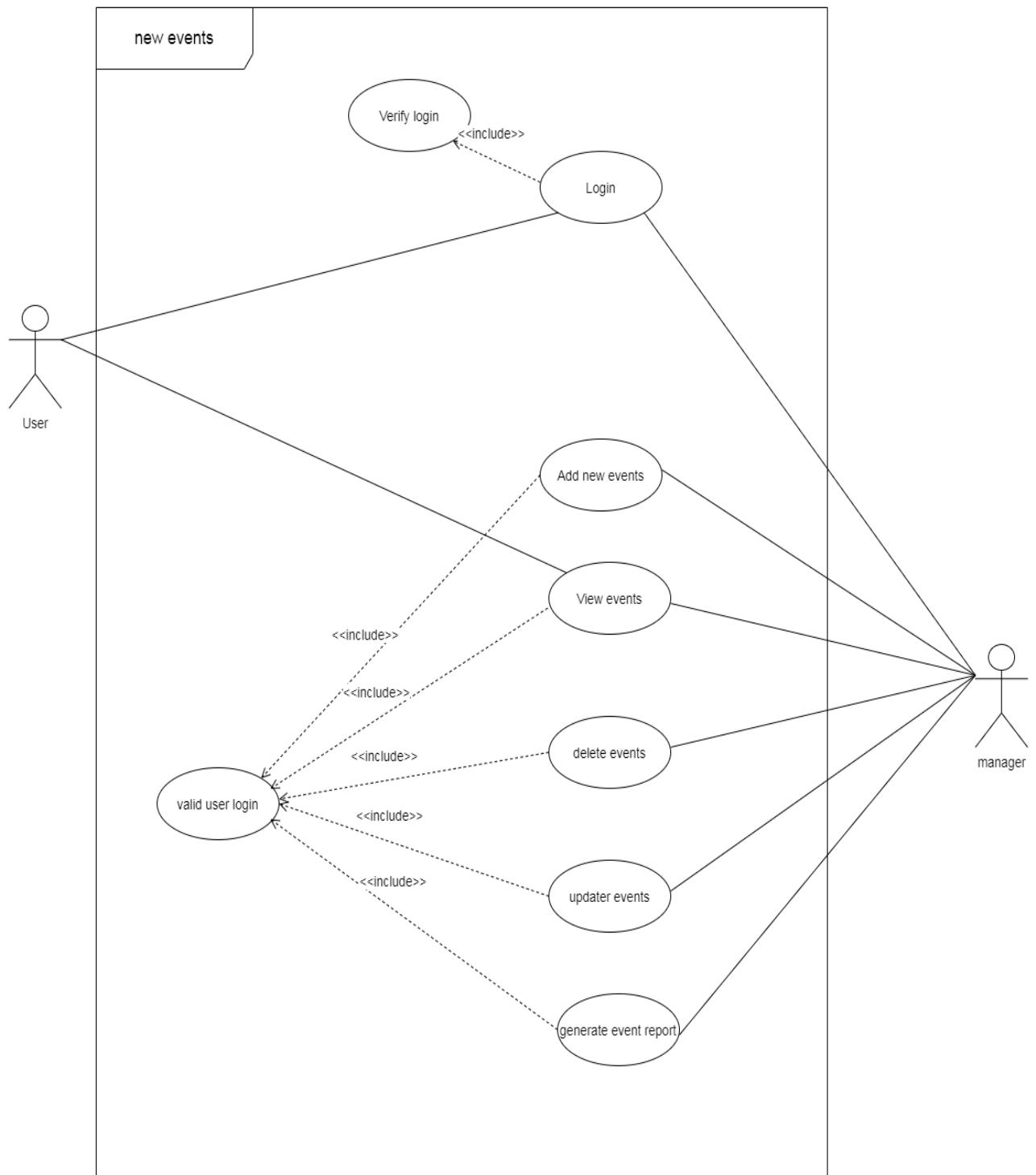
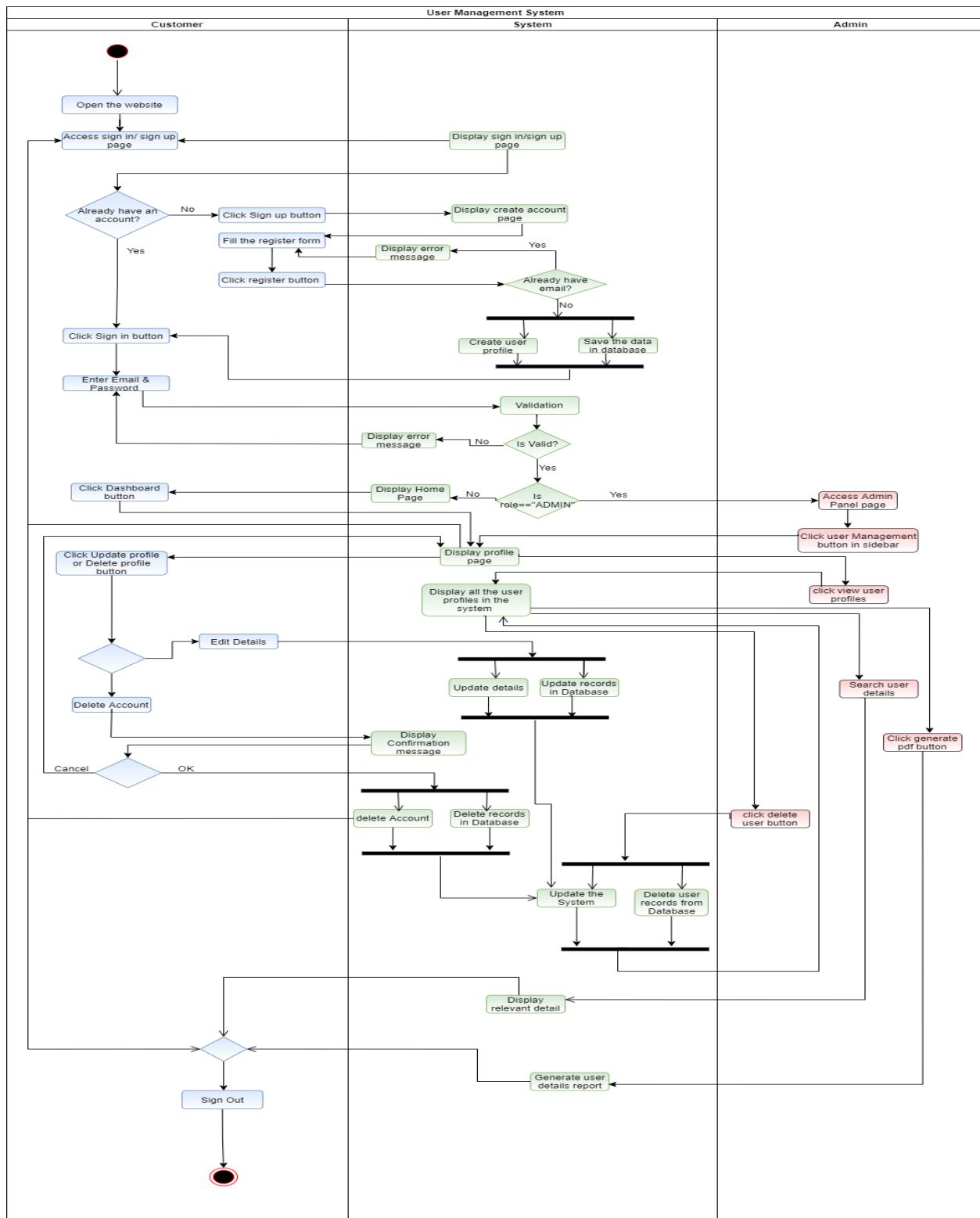


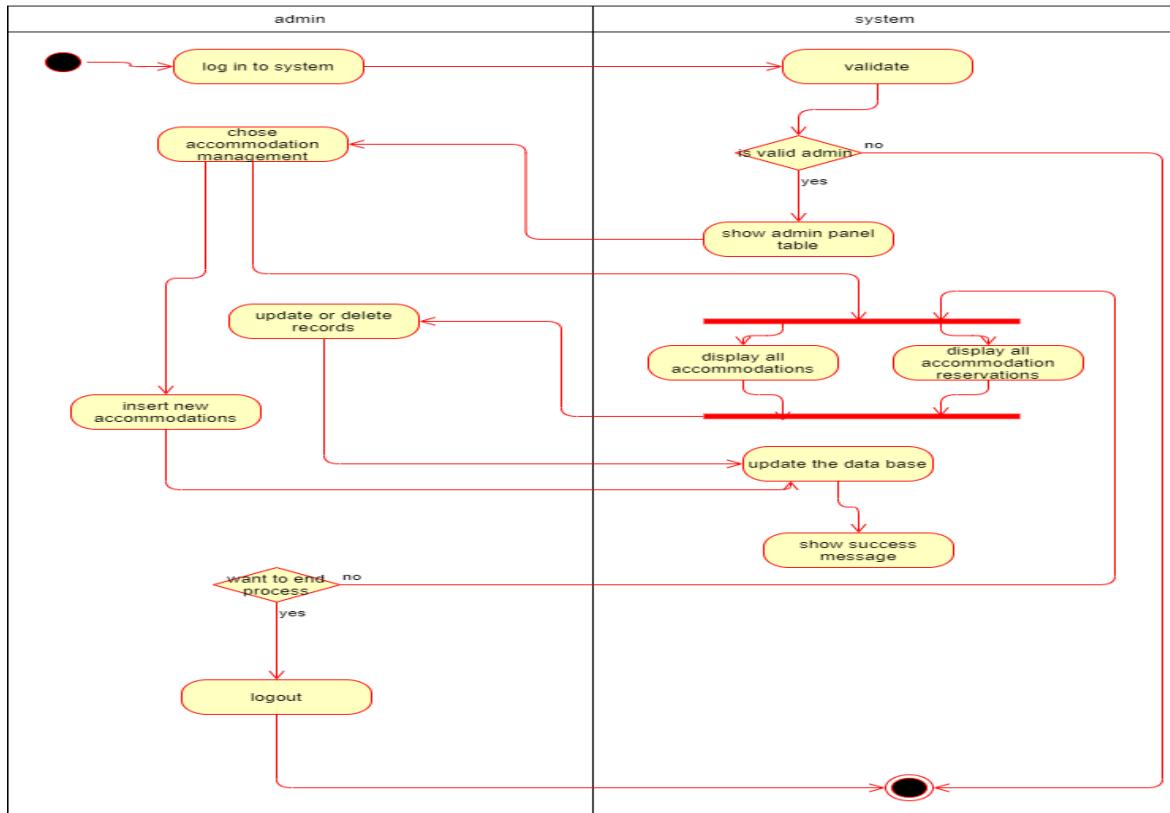
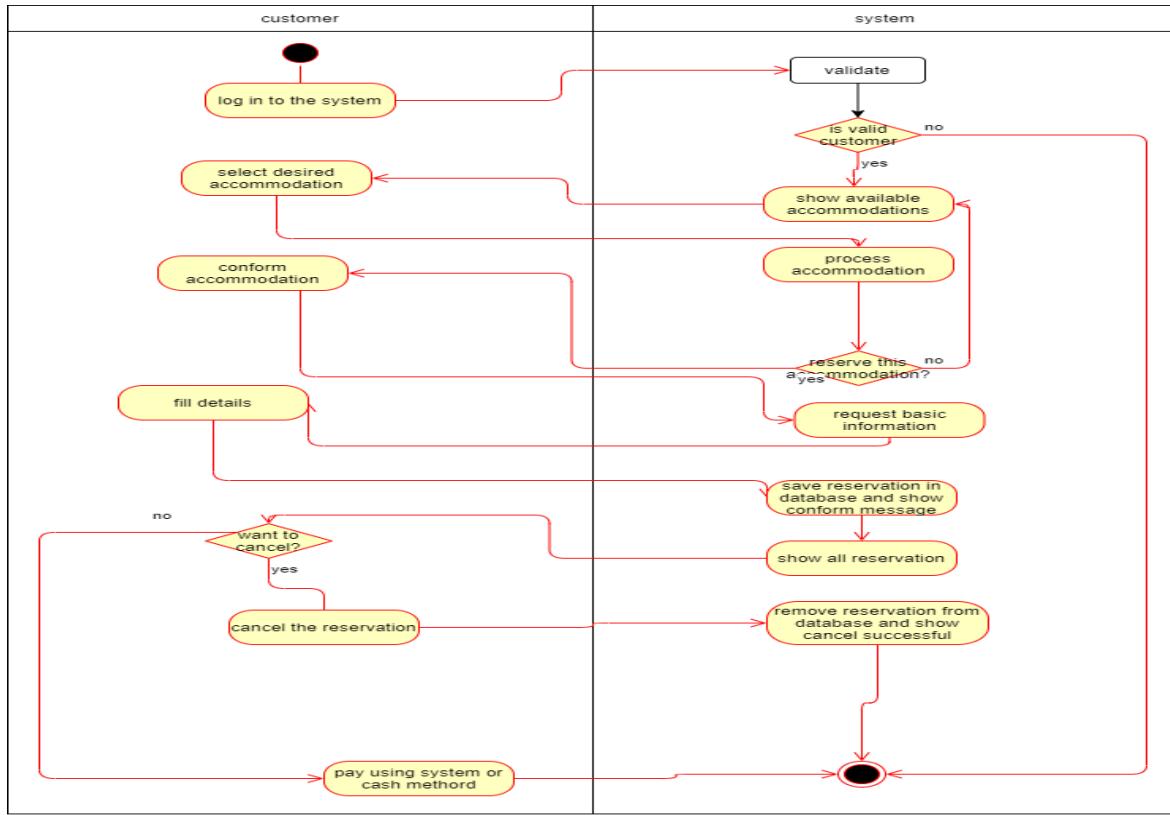
Figure 8

Activity Diagram



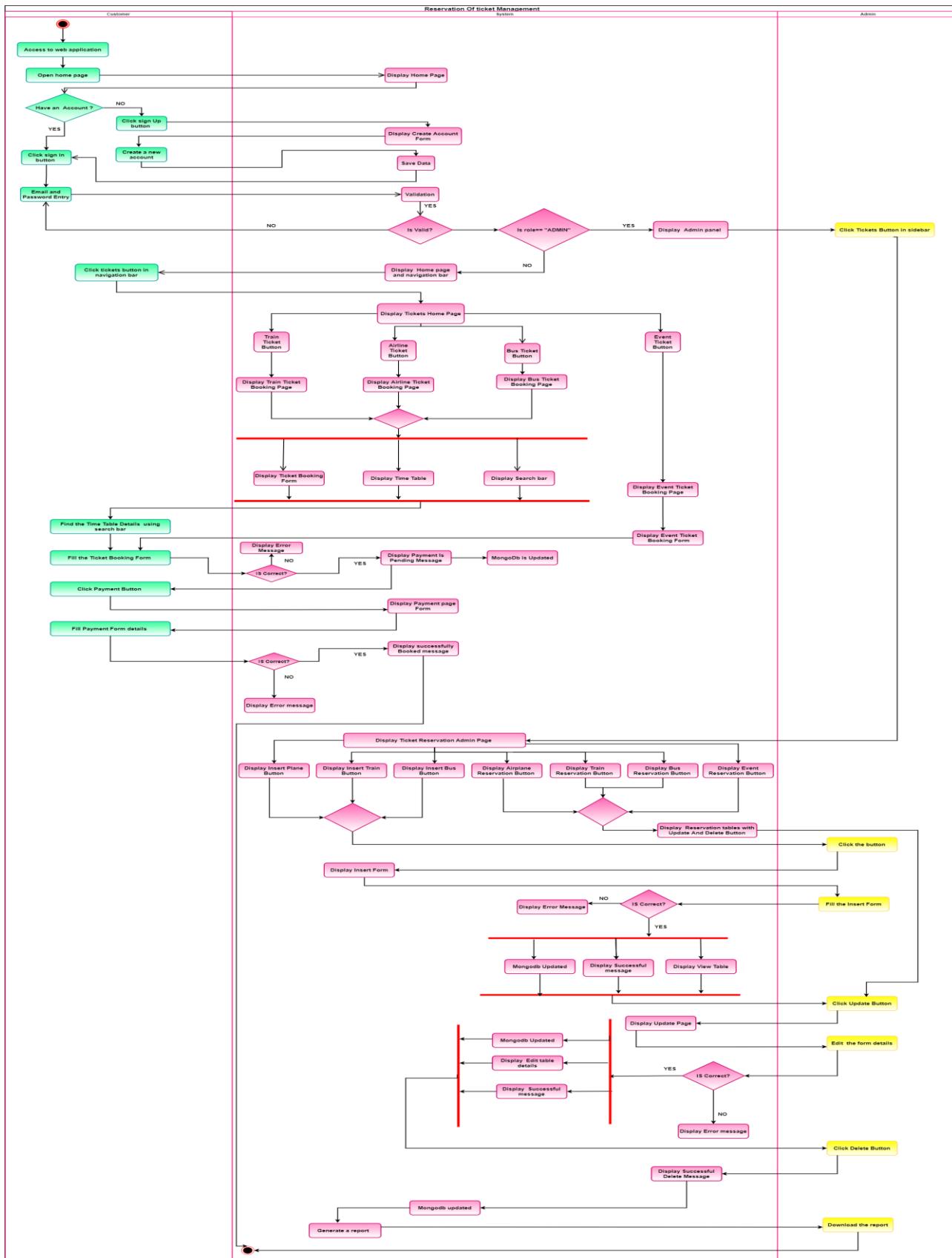
User Account Management

Figure 9



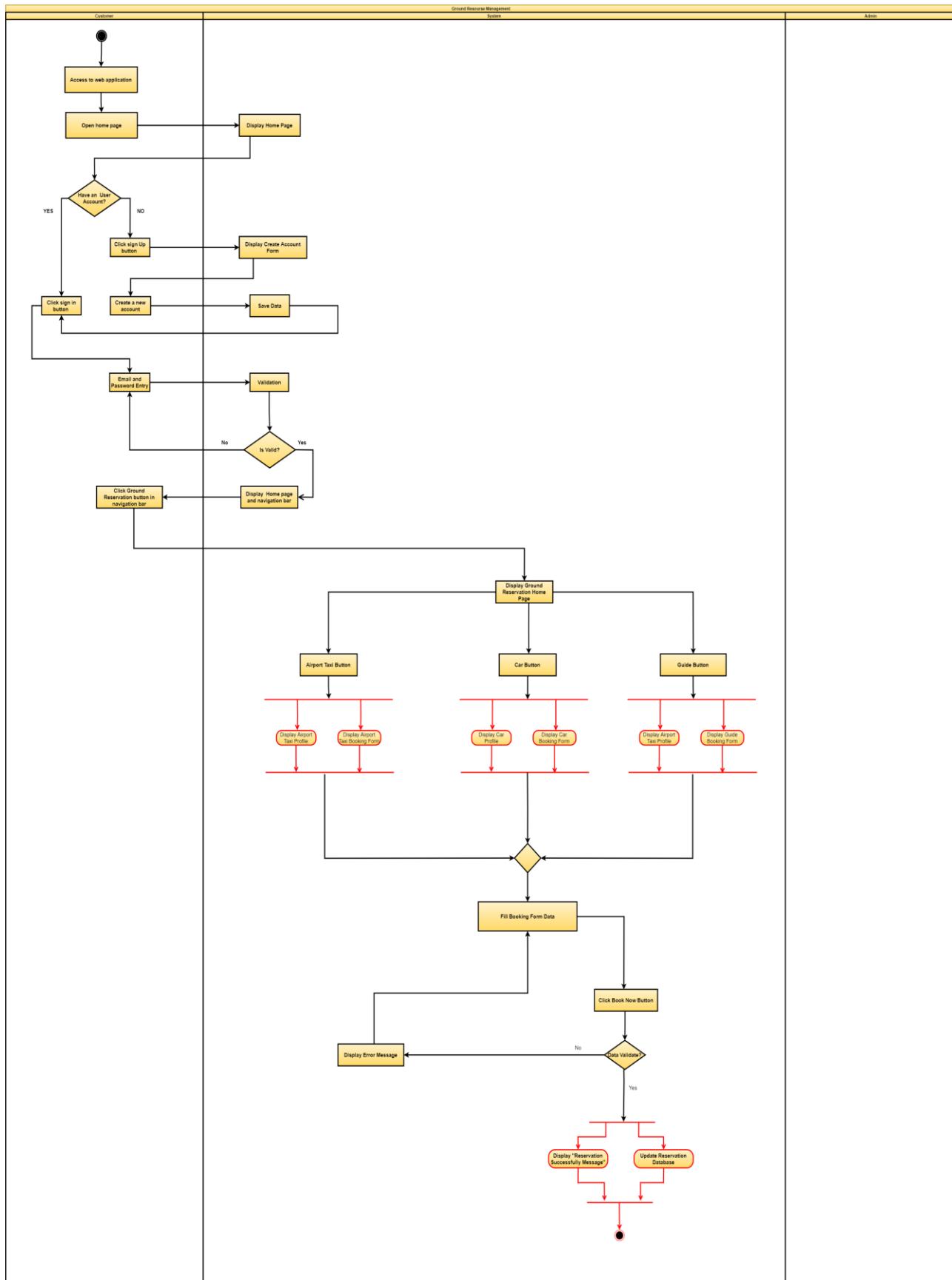
Accommodation reservation management

Figure 10



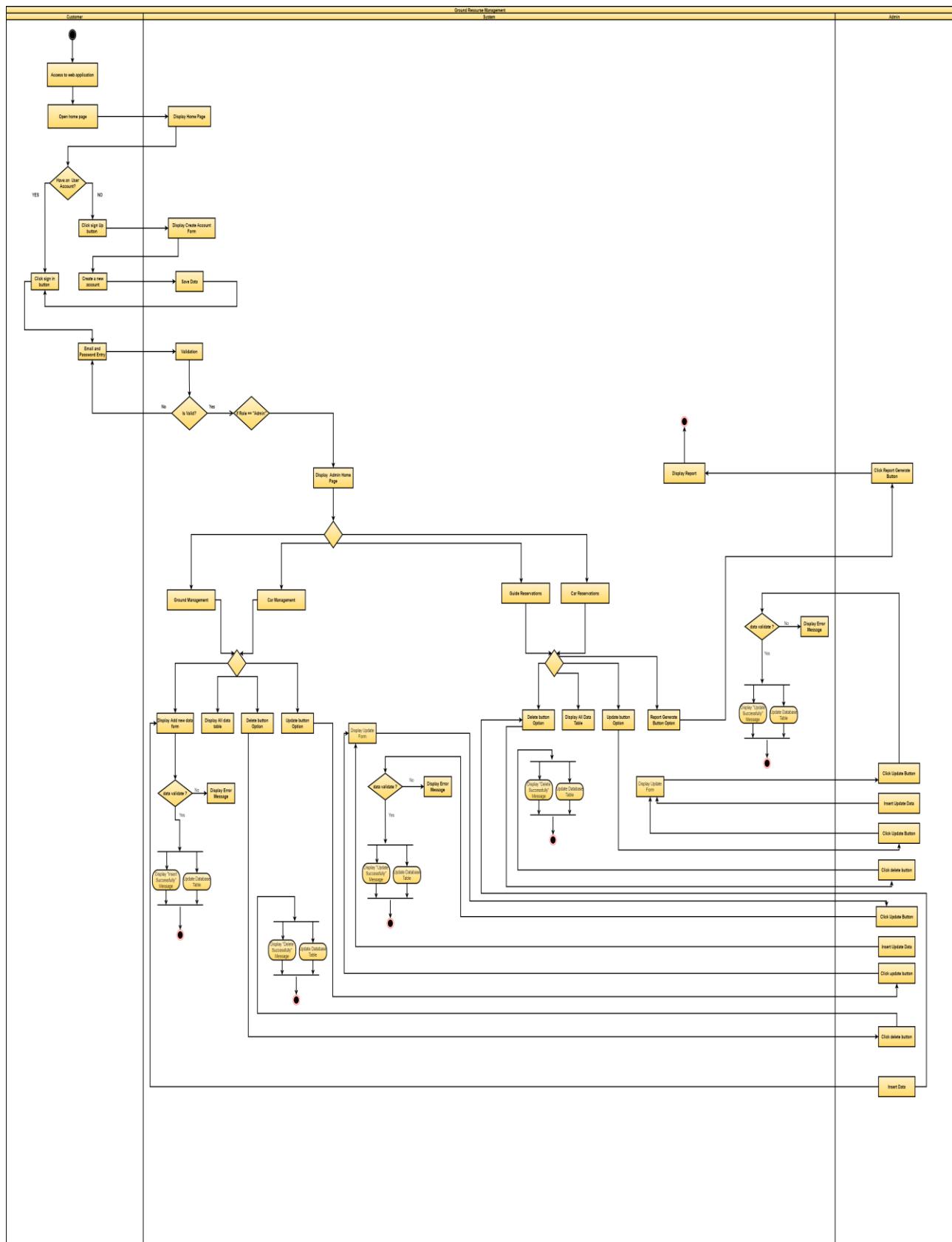
Reservation of Tickets Management

Figure 11



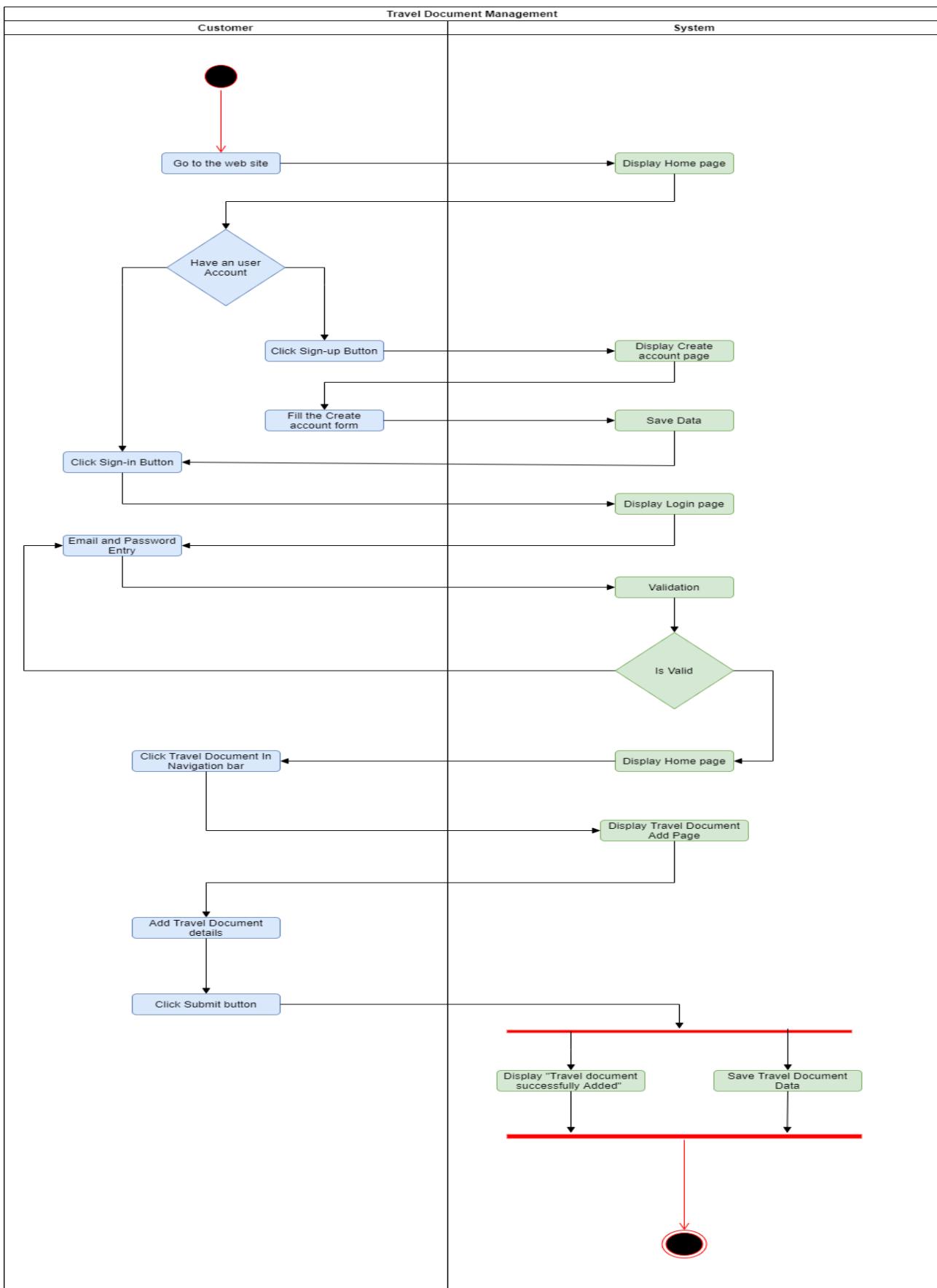
User Reservation of Ground Management

Figure 12



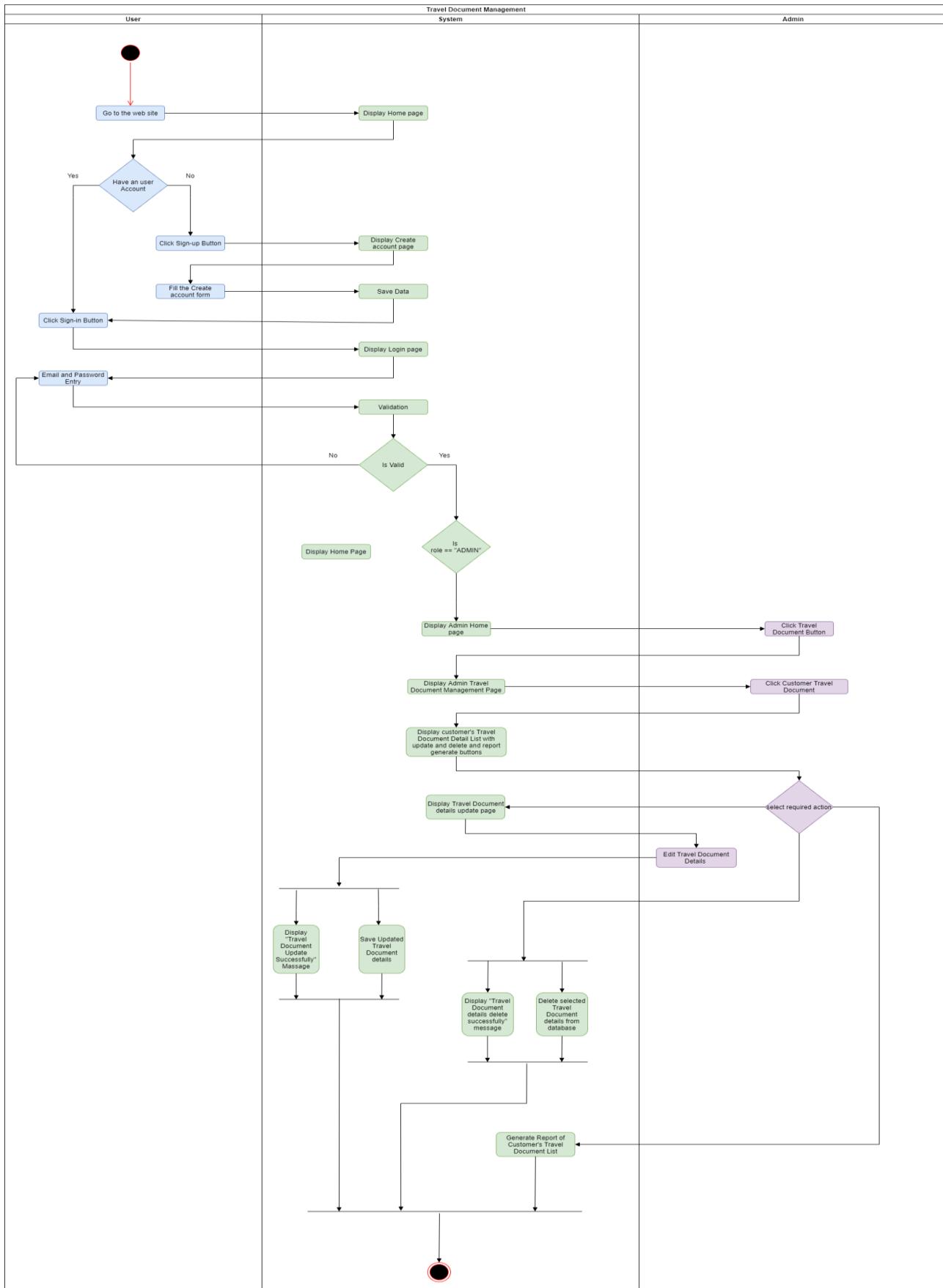
Admin Reservation of Ground Resources Management

Figure 13



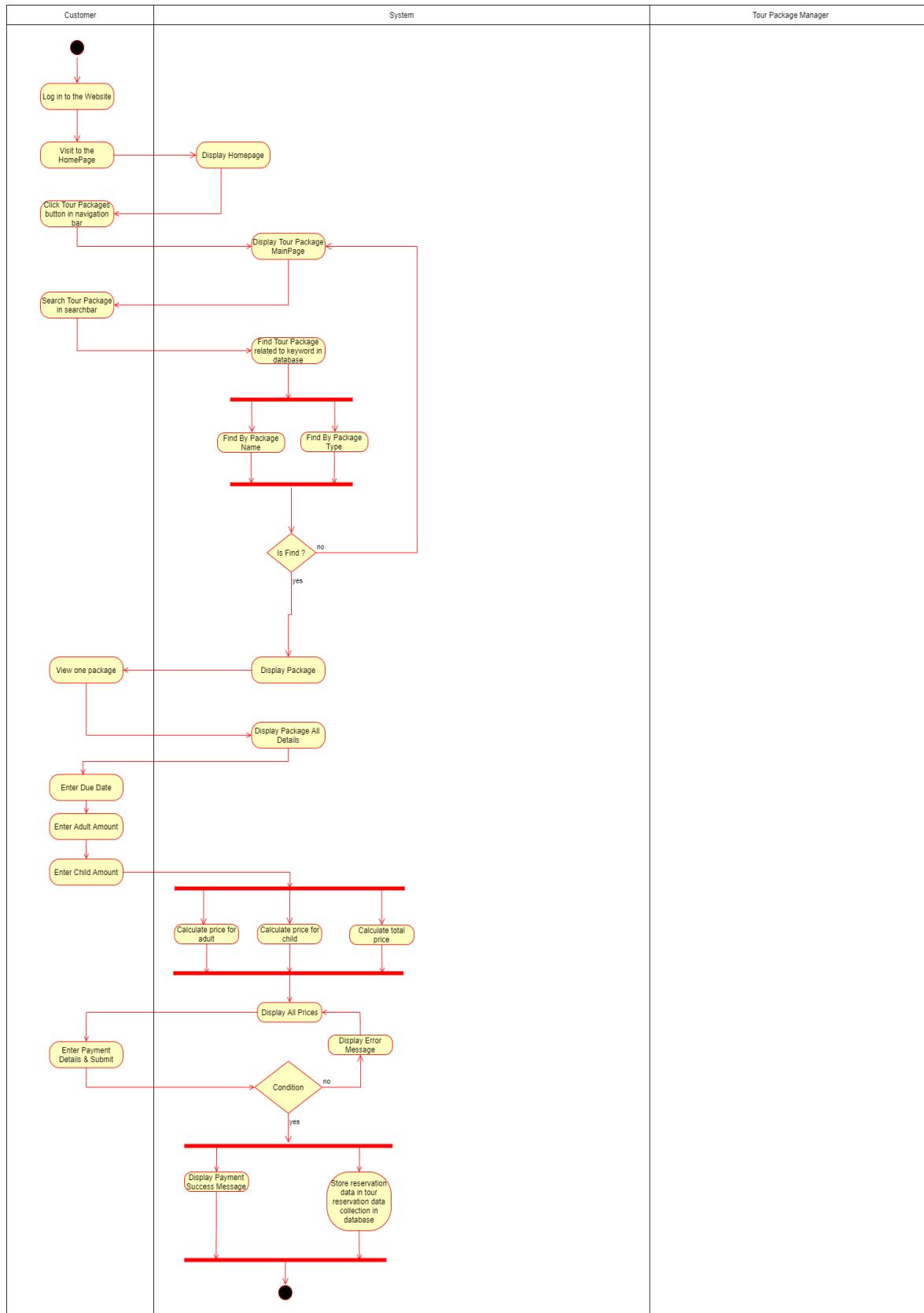
User Travel Document Management

Figure 14



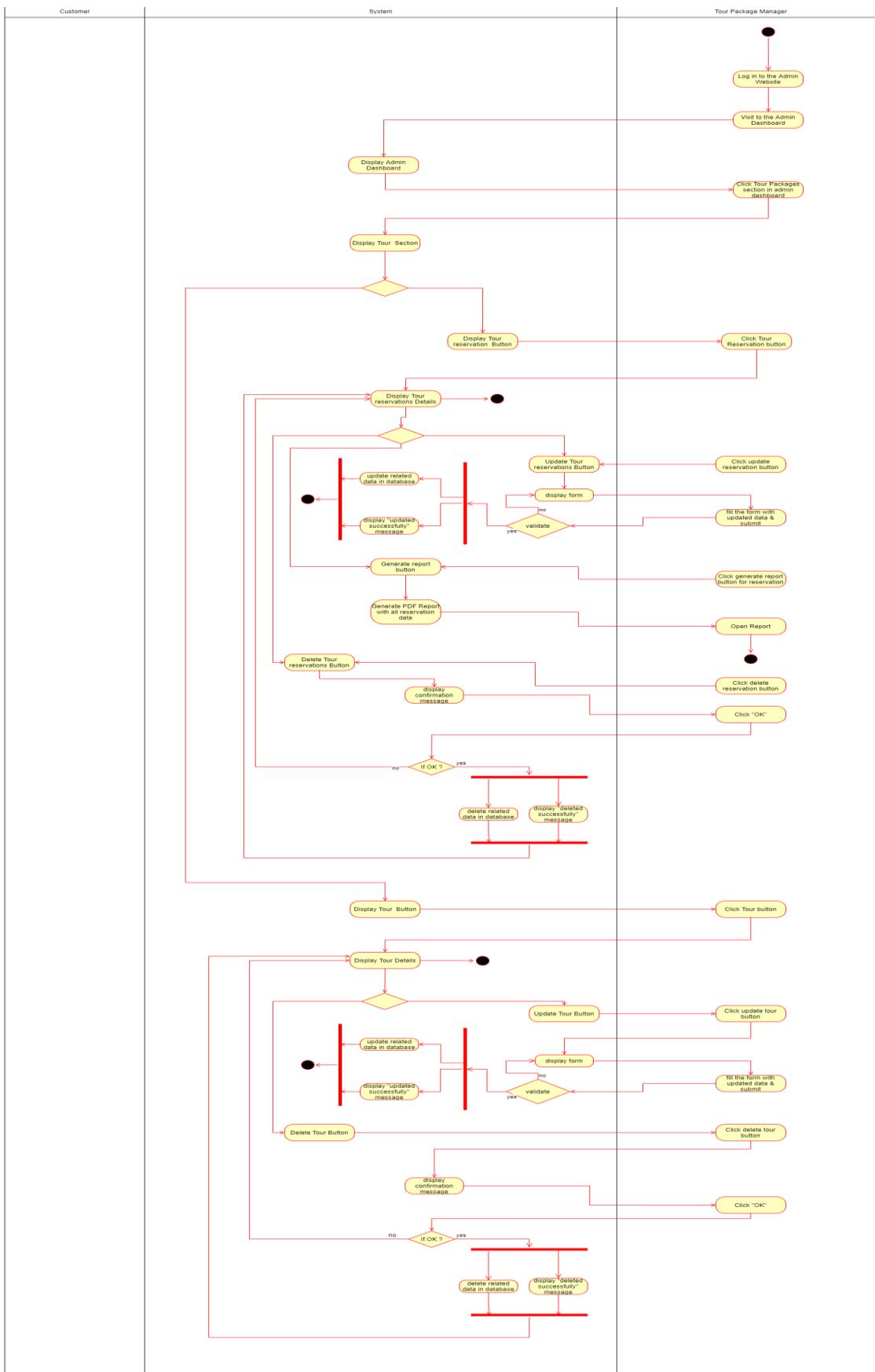
Admin Travel Document Management

Figure 15



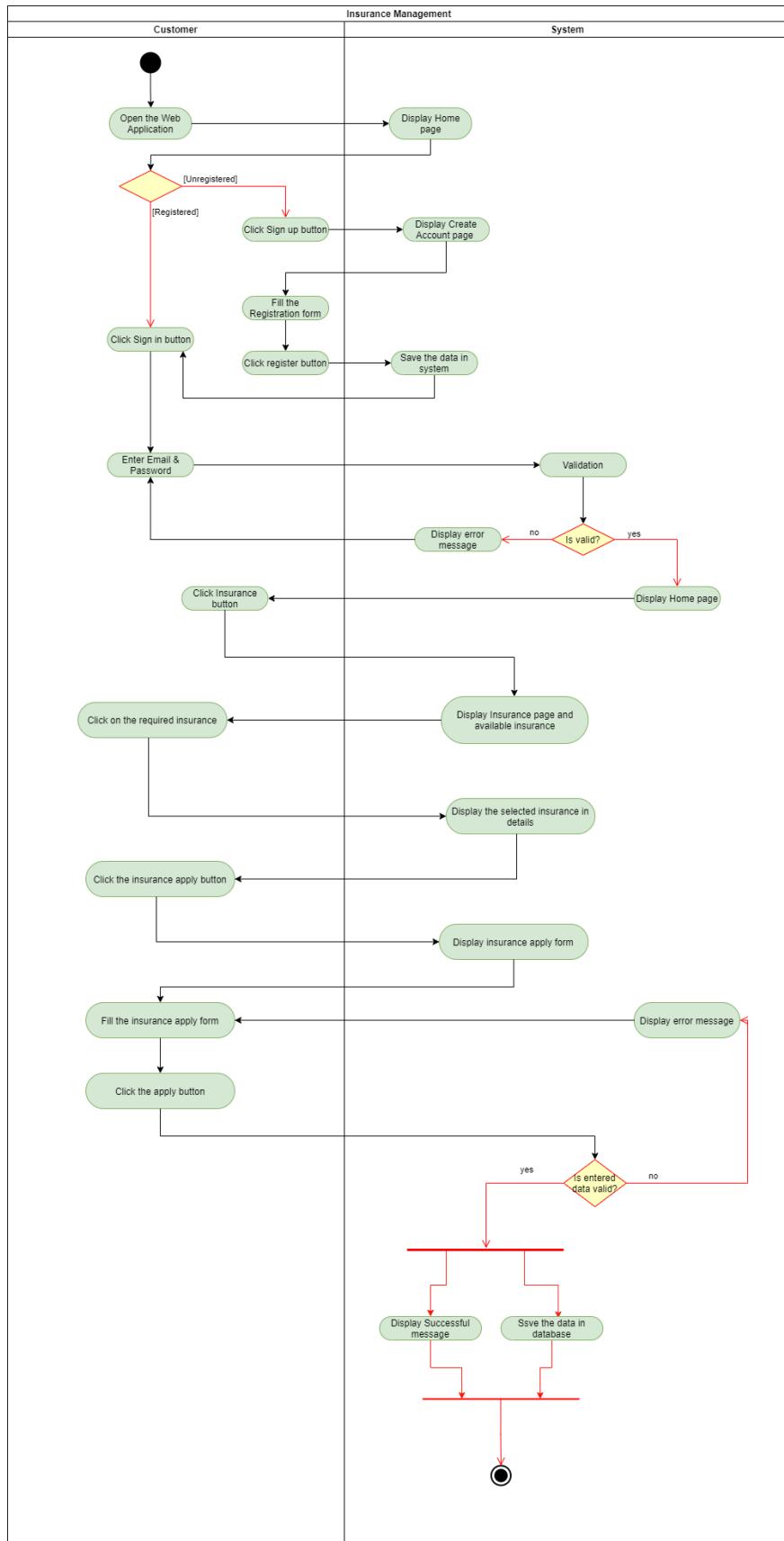
Tour Package Management (USER SIDE)

Figure 16



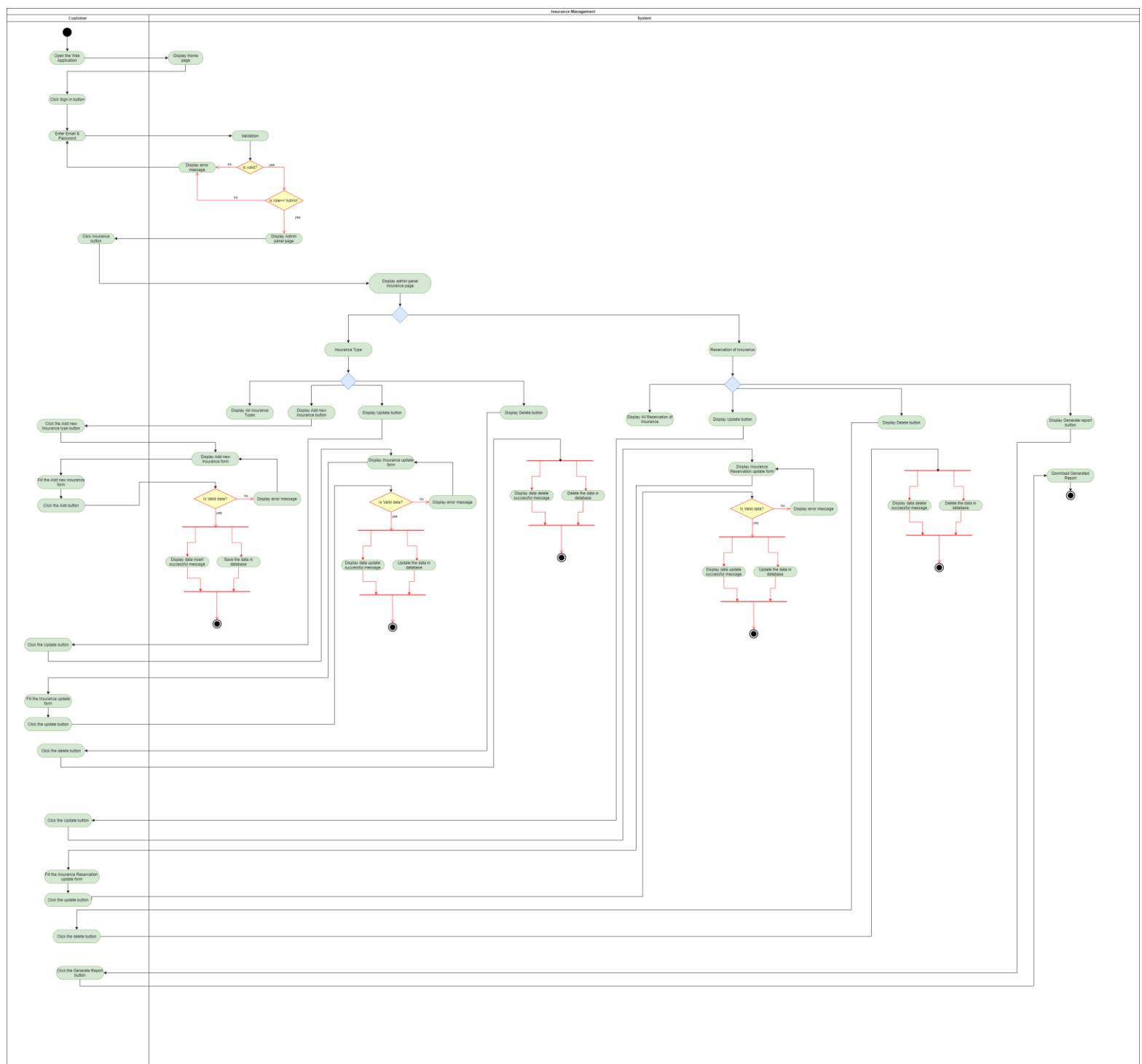
Tour Package Management (ADMIN SIDE)

Figure 17



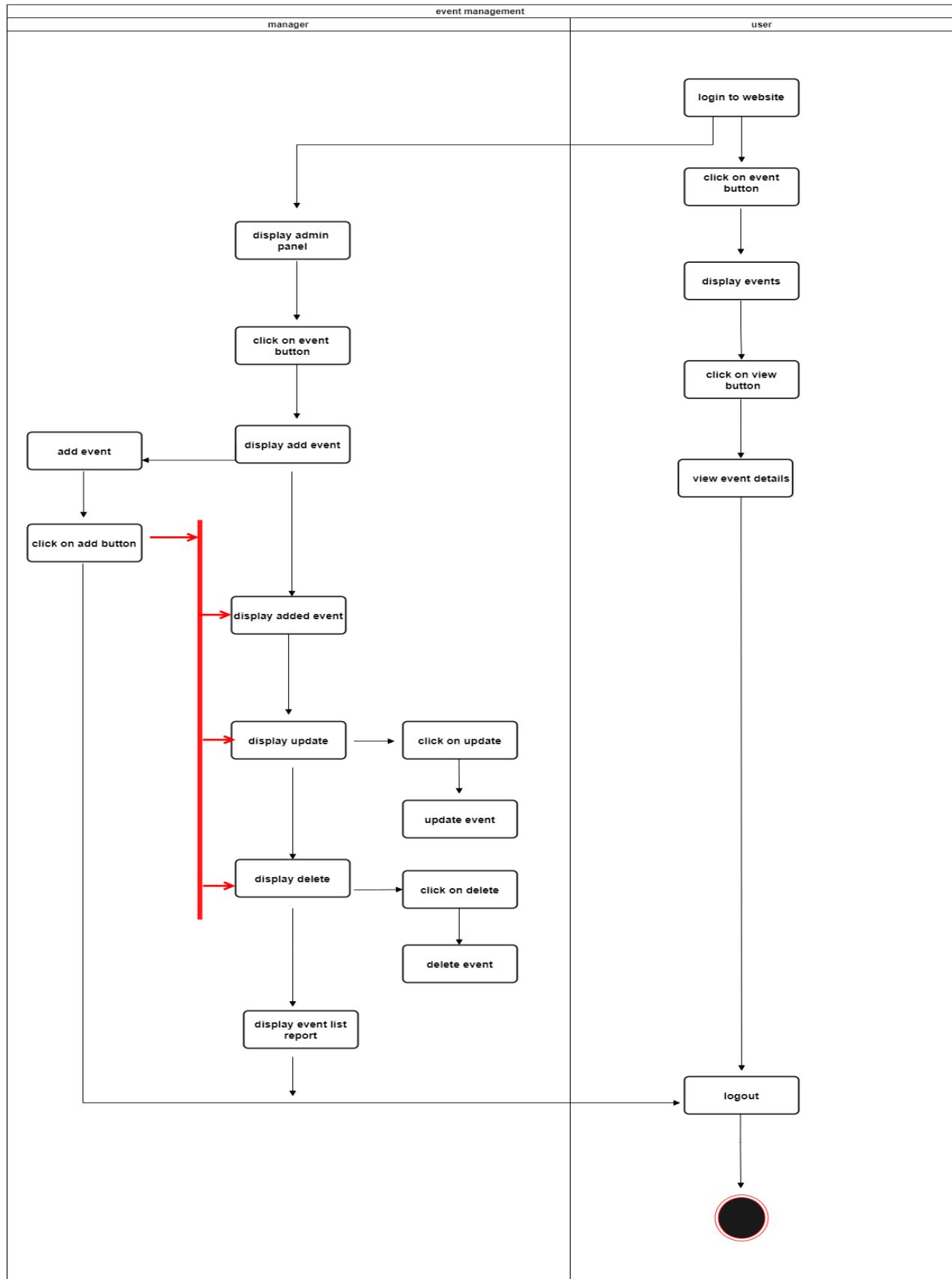
Travel Insurance Management (USER SIDE)

Figure 18



Travel Insurance Management (ADMIN SIDE)

Figure 19



Event Management

Figure 20

2.2 Design

2.2.1 ER Diagram

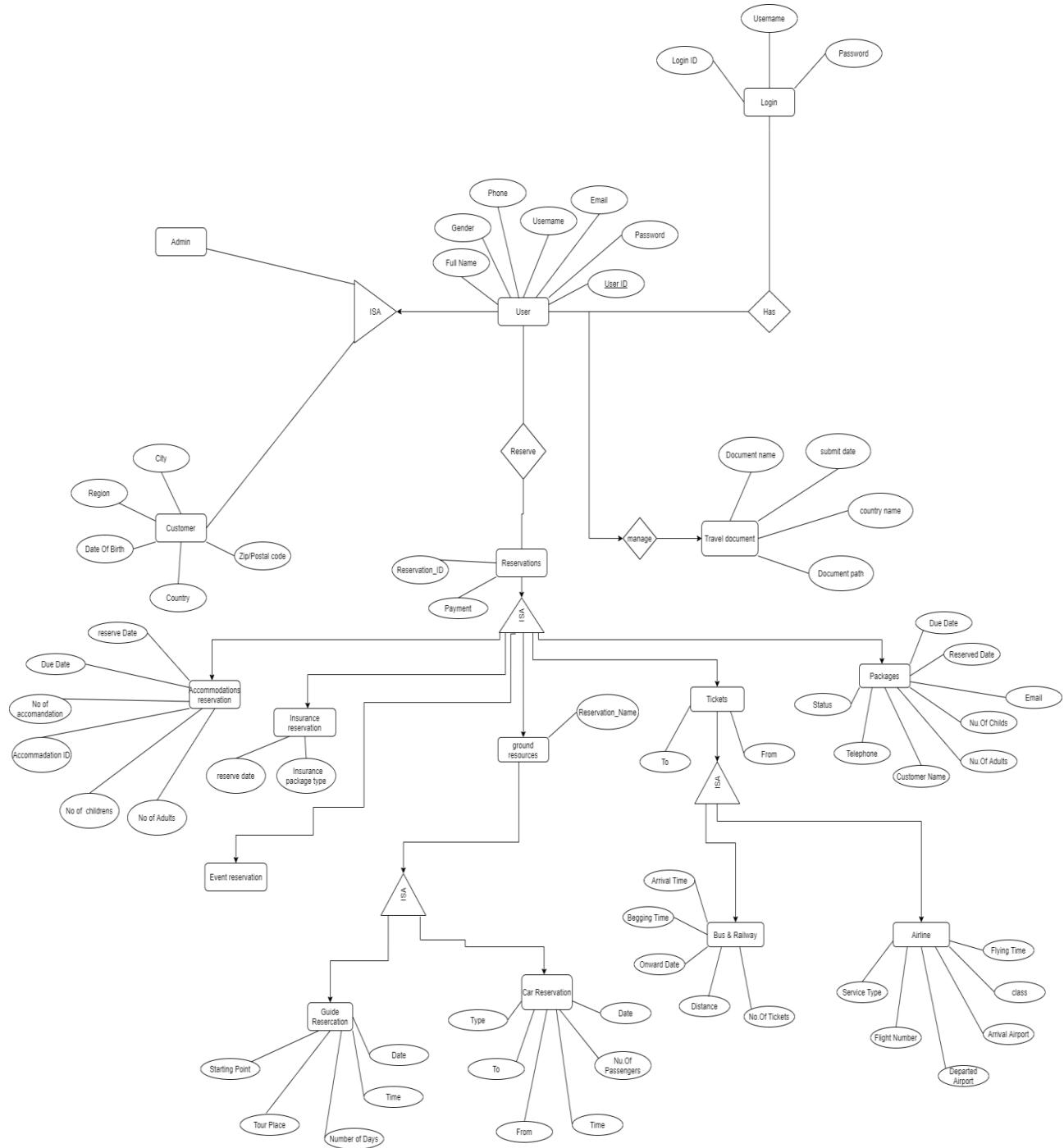


Figure 21

2.2.2 Sequence Diagram

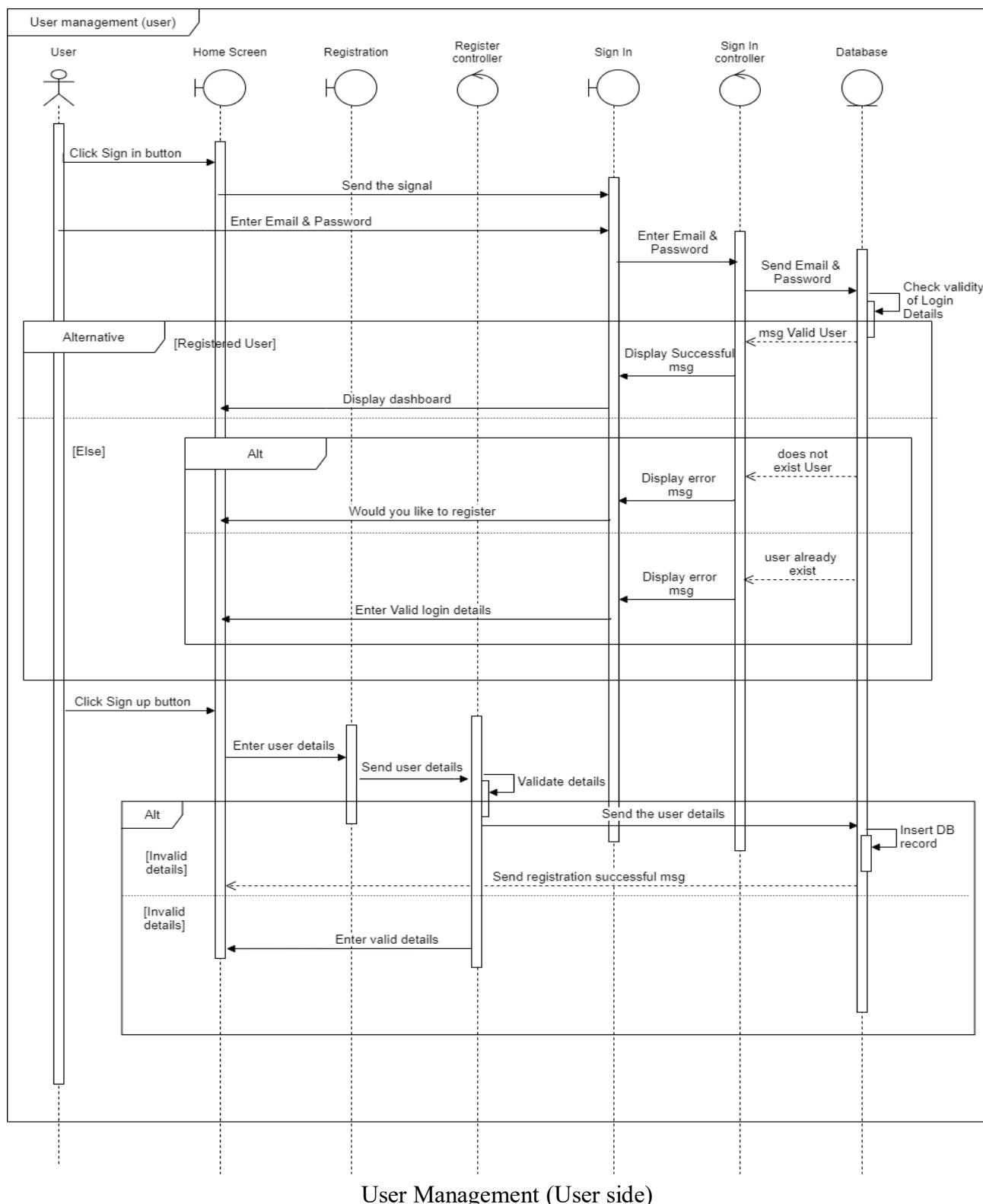
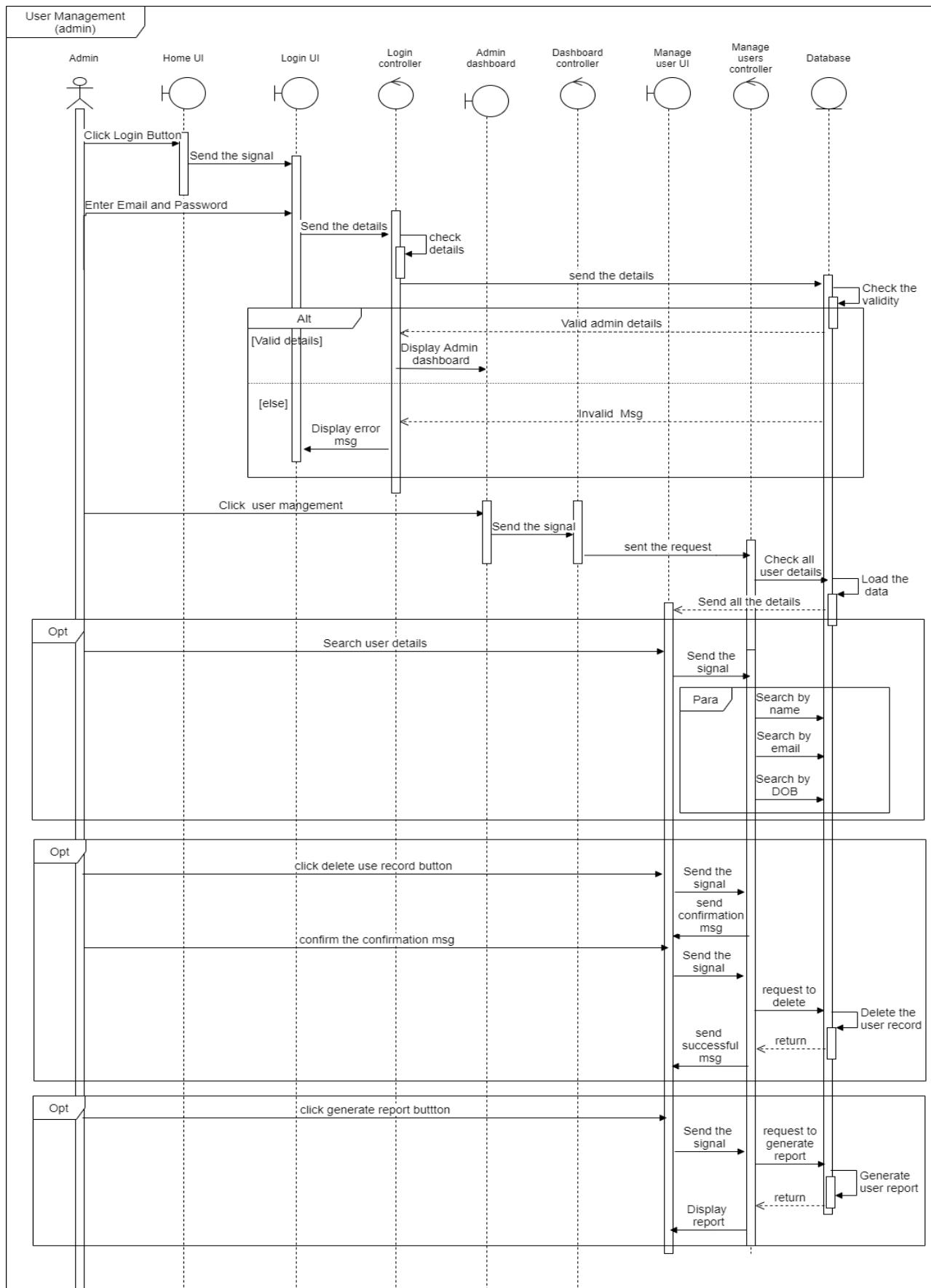
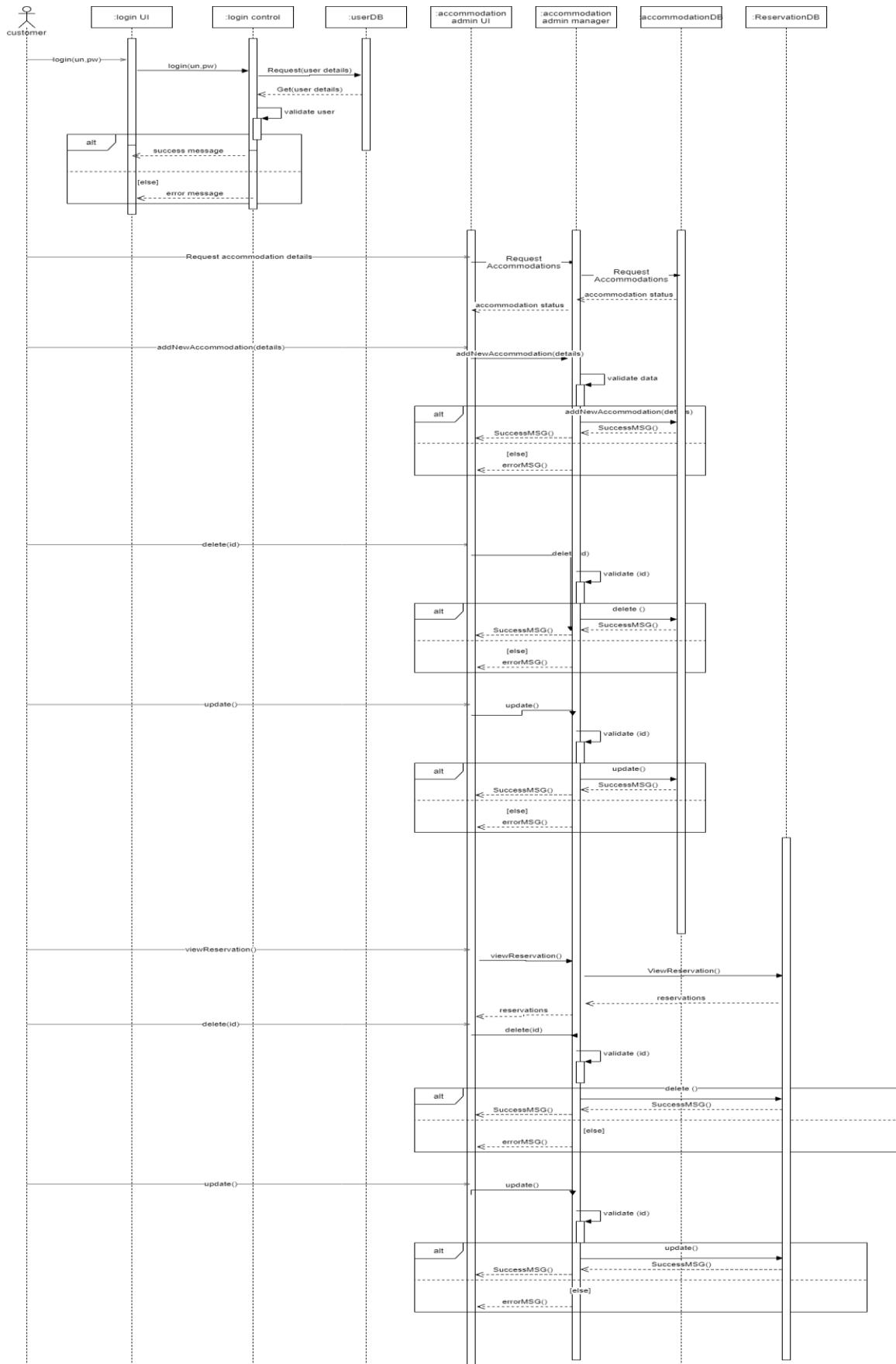


Figure 22



User Management (Admin side)

Figure 23



Accommodation reservation Management (User side)

Figure 24

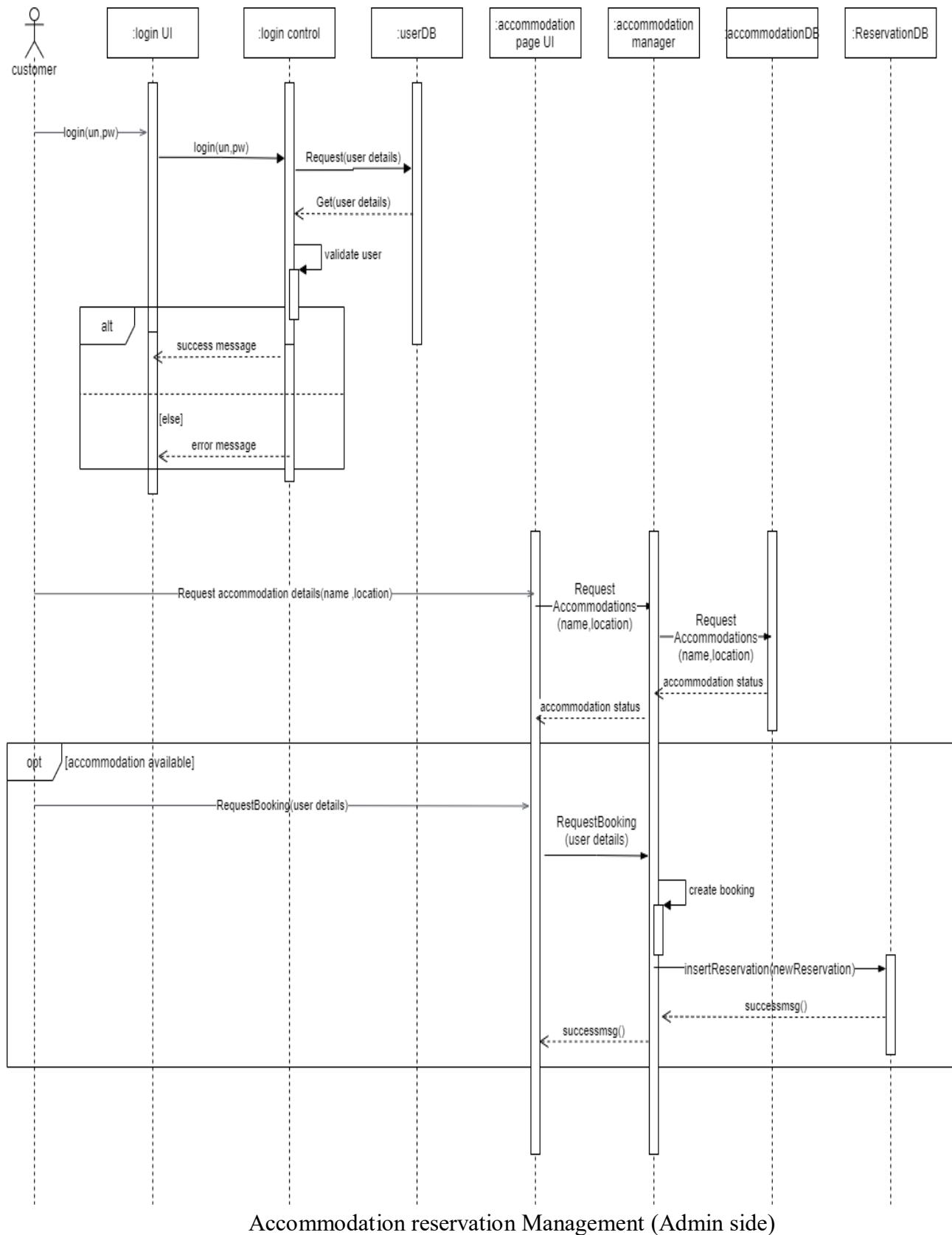
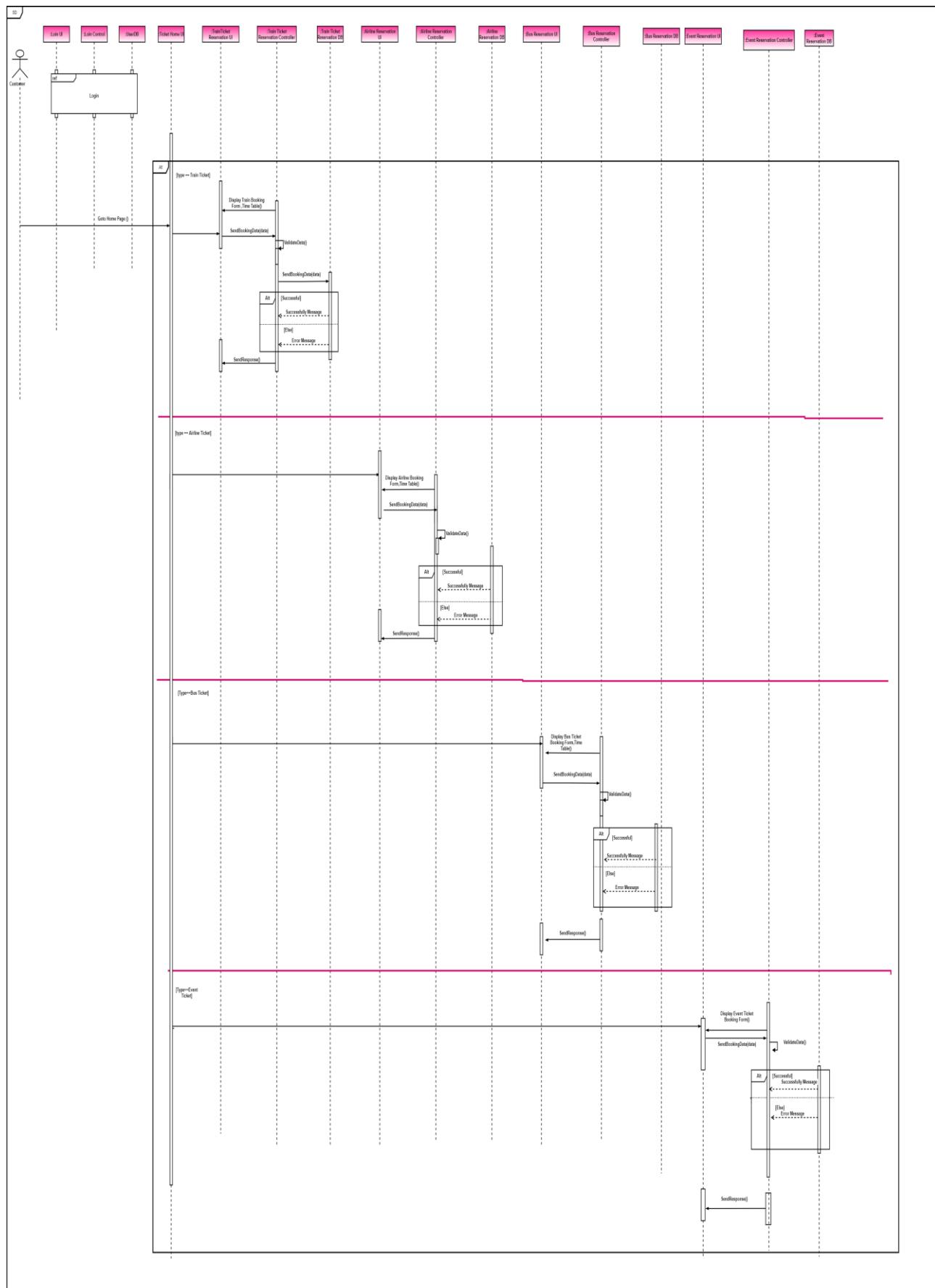
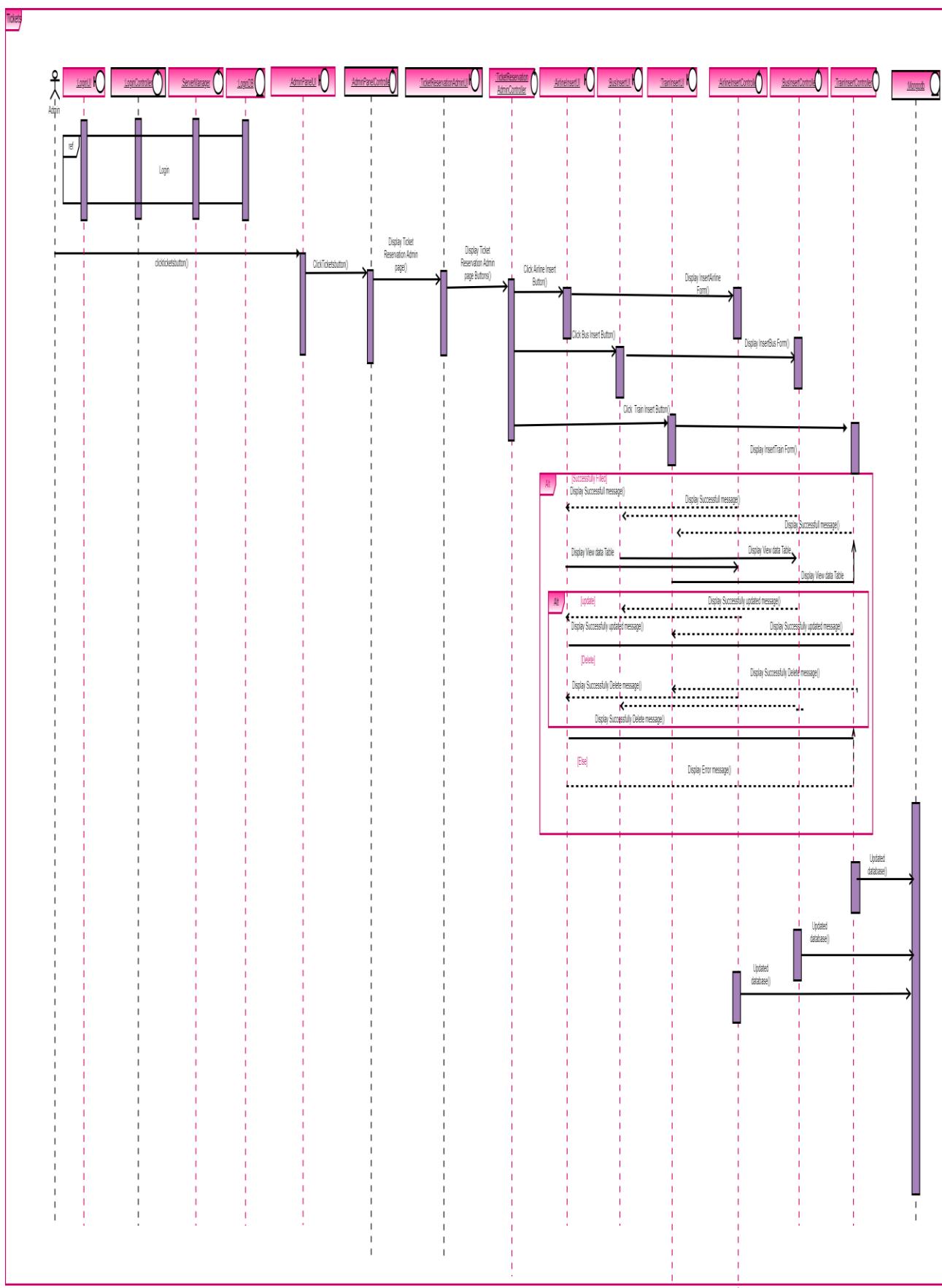


Figure 25



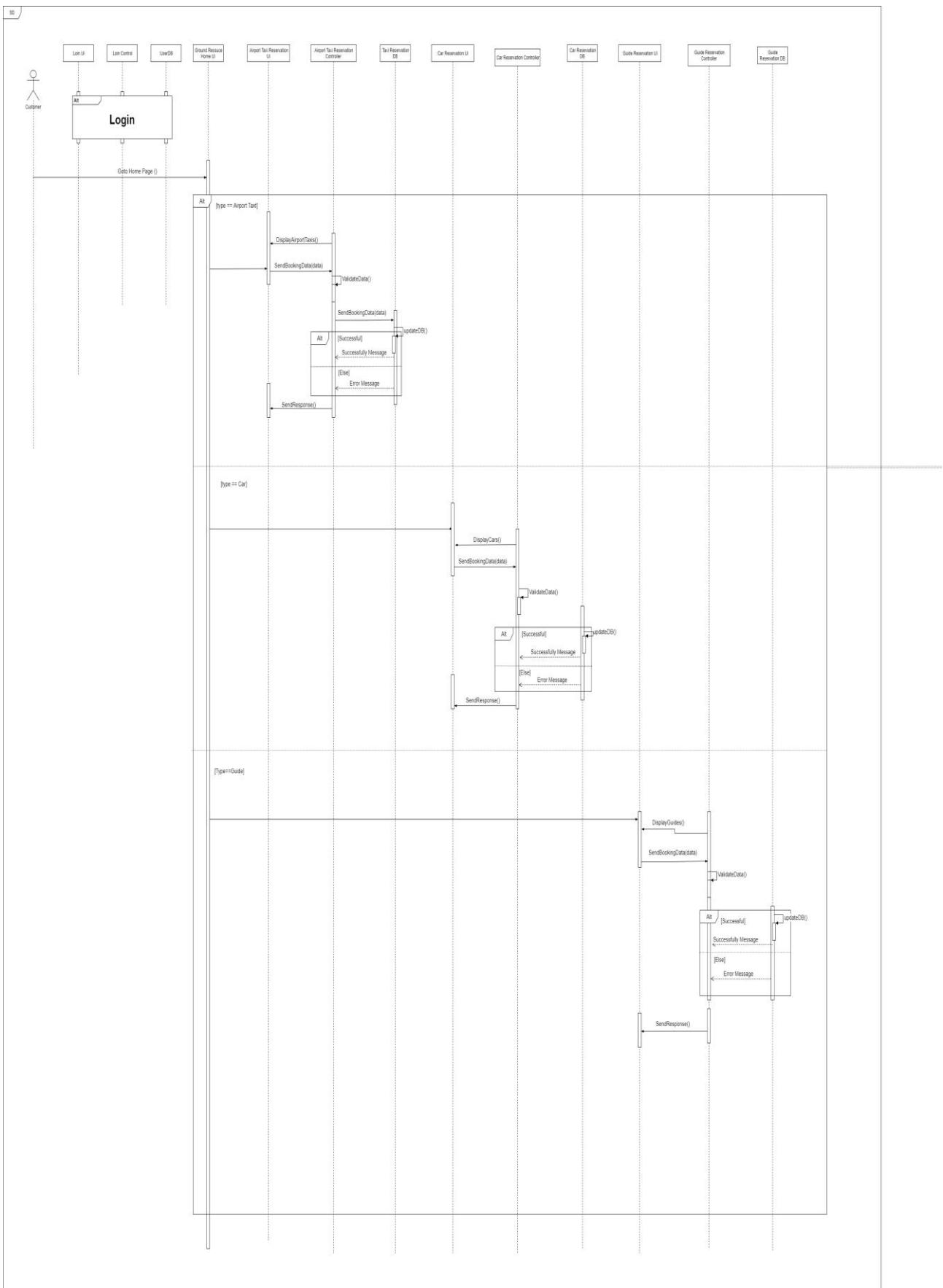
Reservation of Tickets Management (User side)

Figure 26



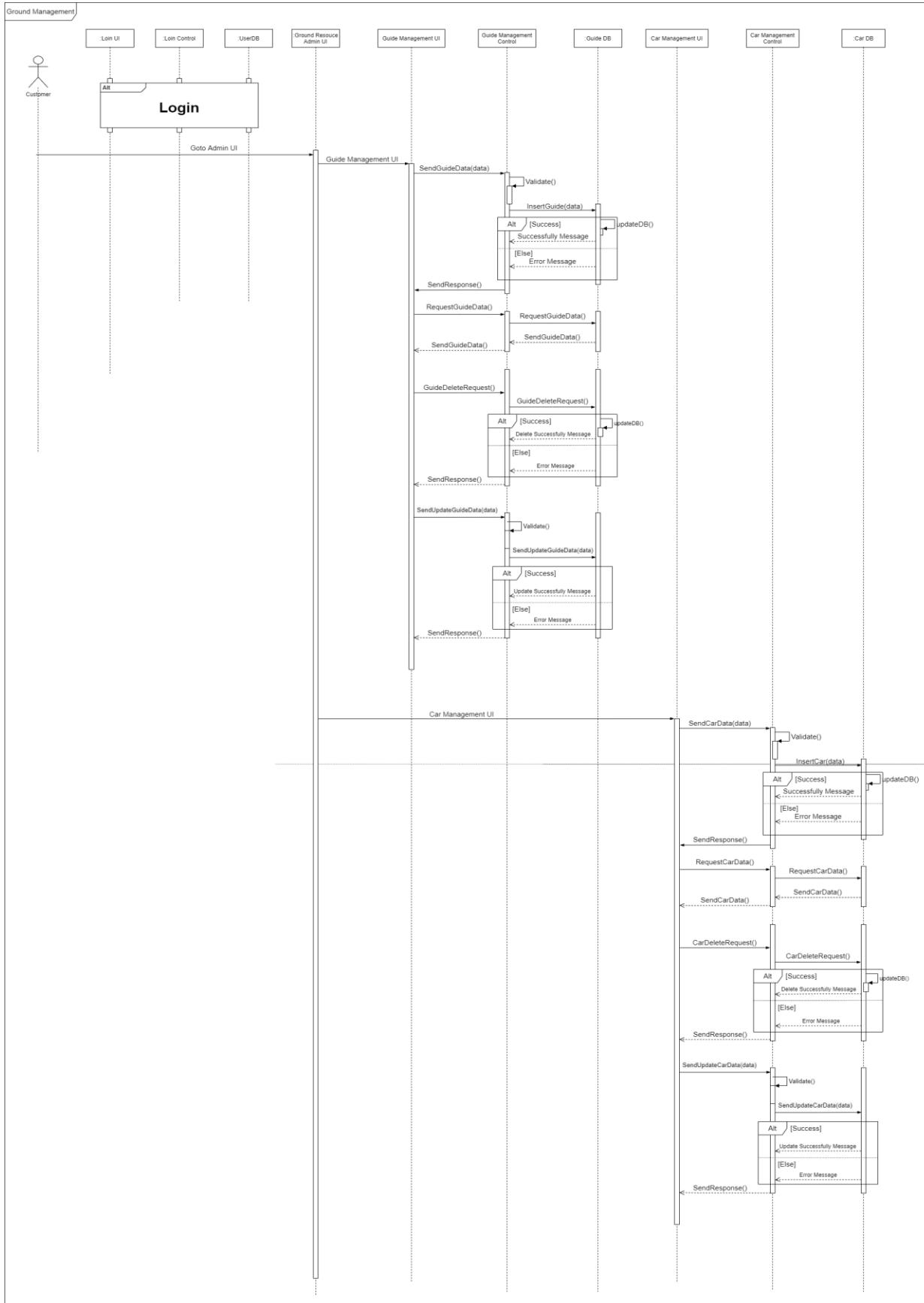
Reservation of Tickets Management (Admin side)

Figure 27



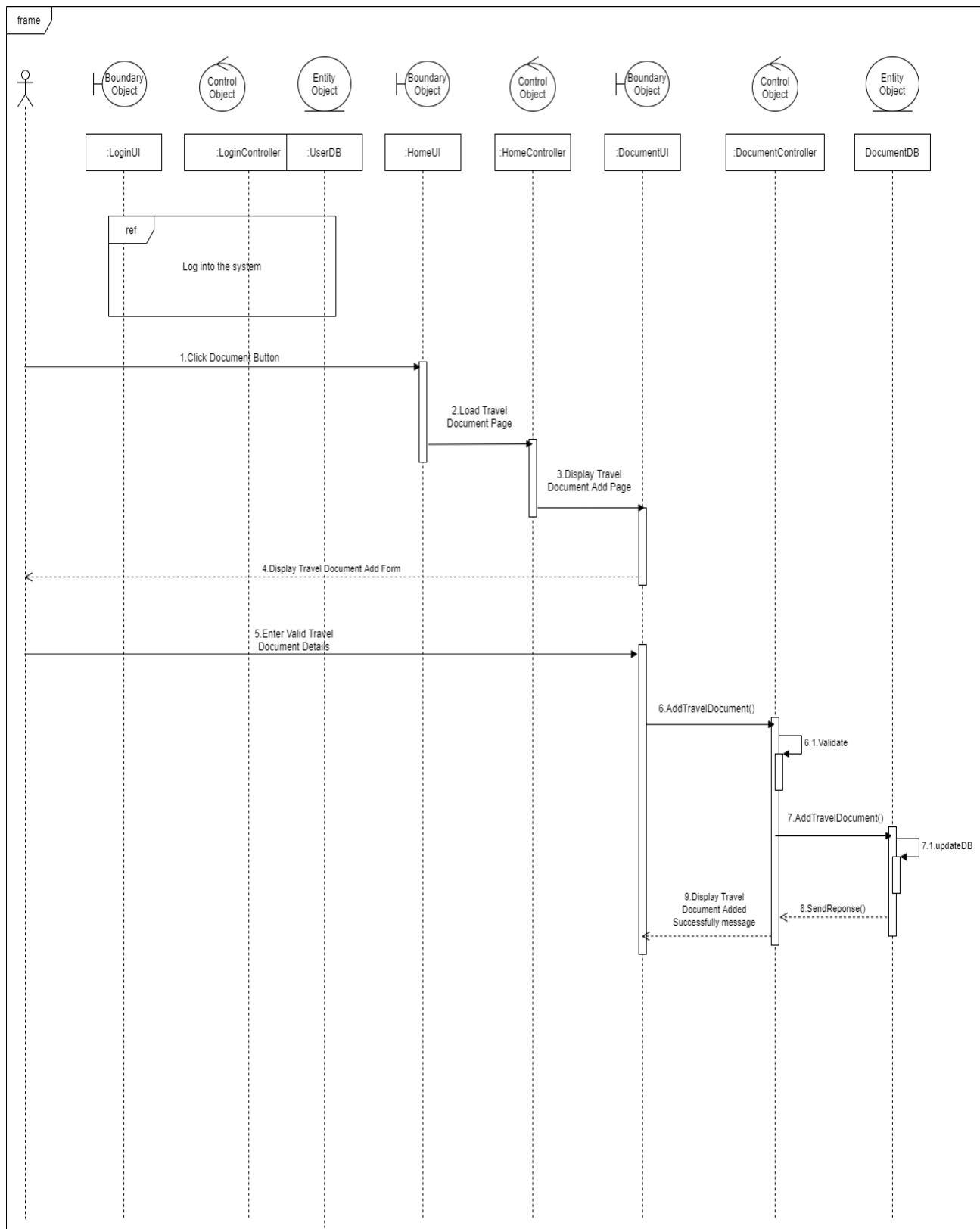
Reservation of Ground Resource Management

Figure 28



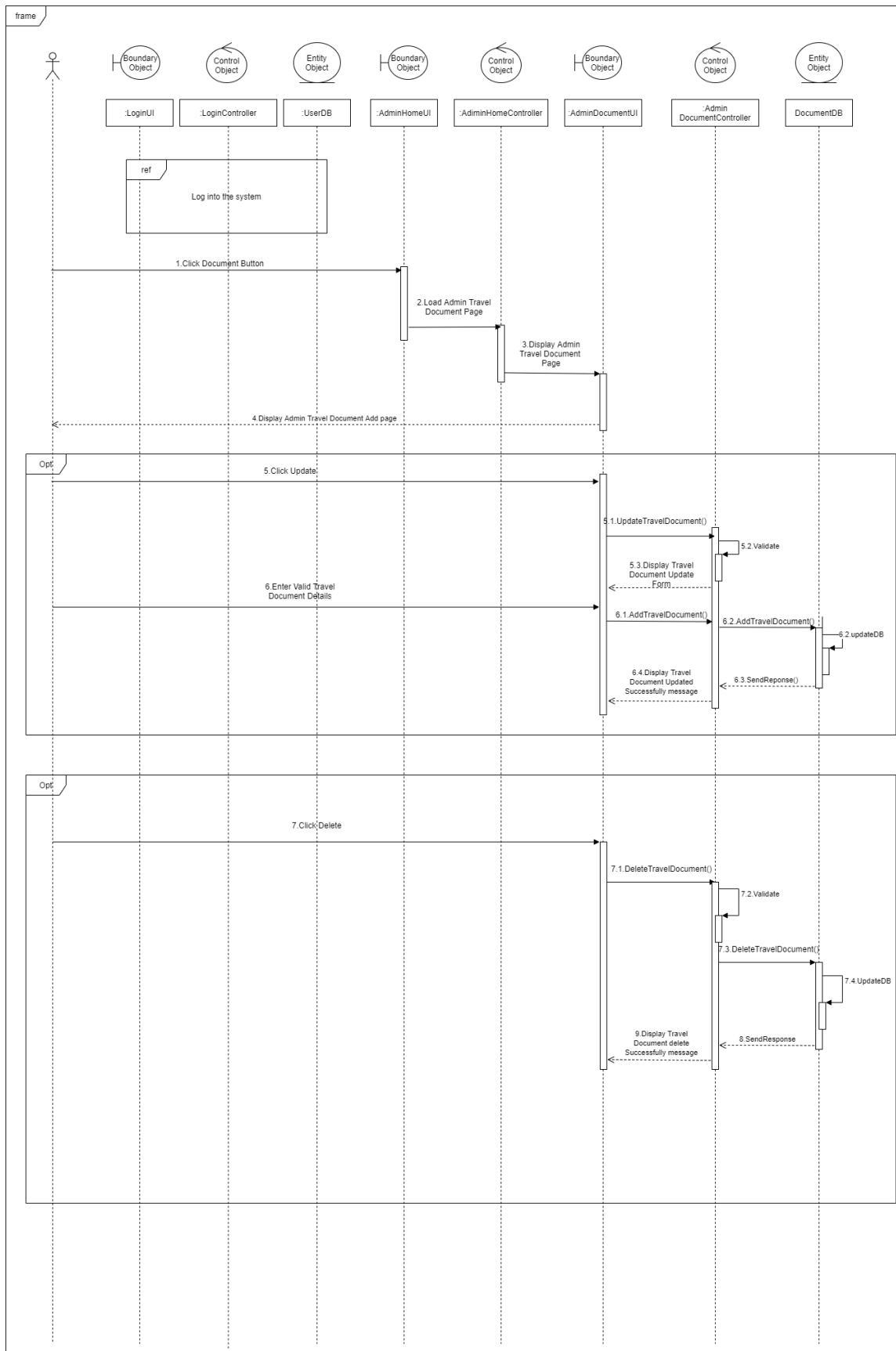
Reservation of Ground Resource Management

Figure 29



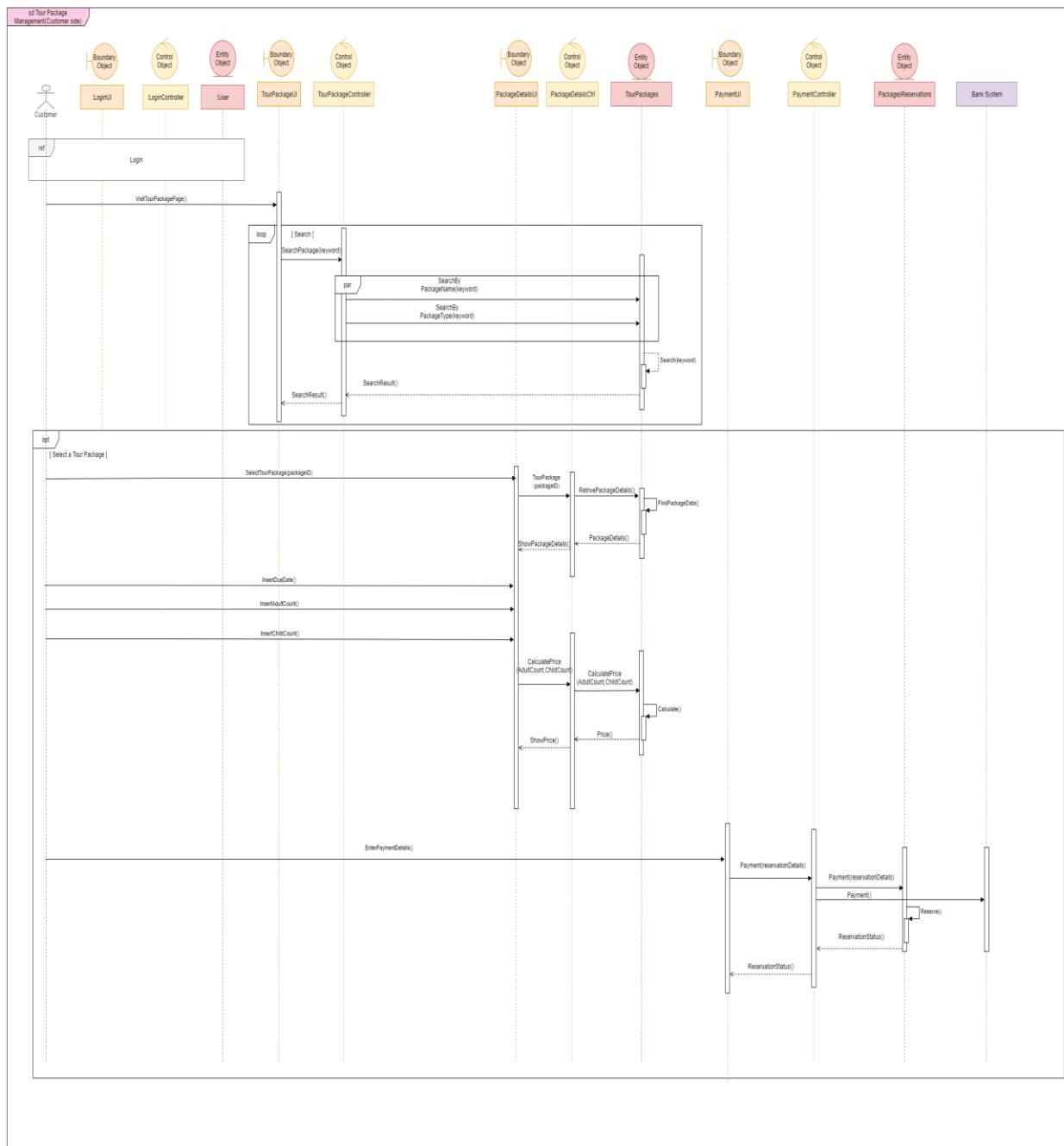
Travel Document Management (User side)

Figure 30



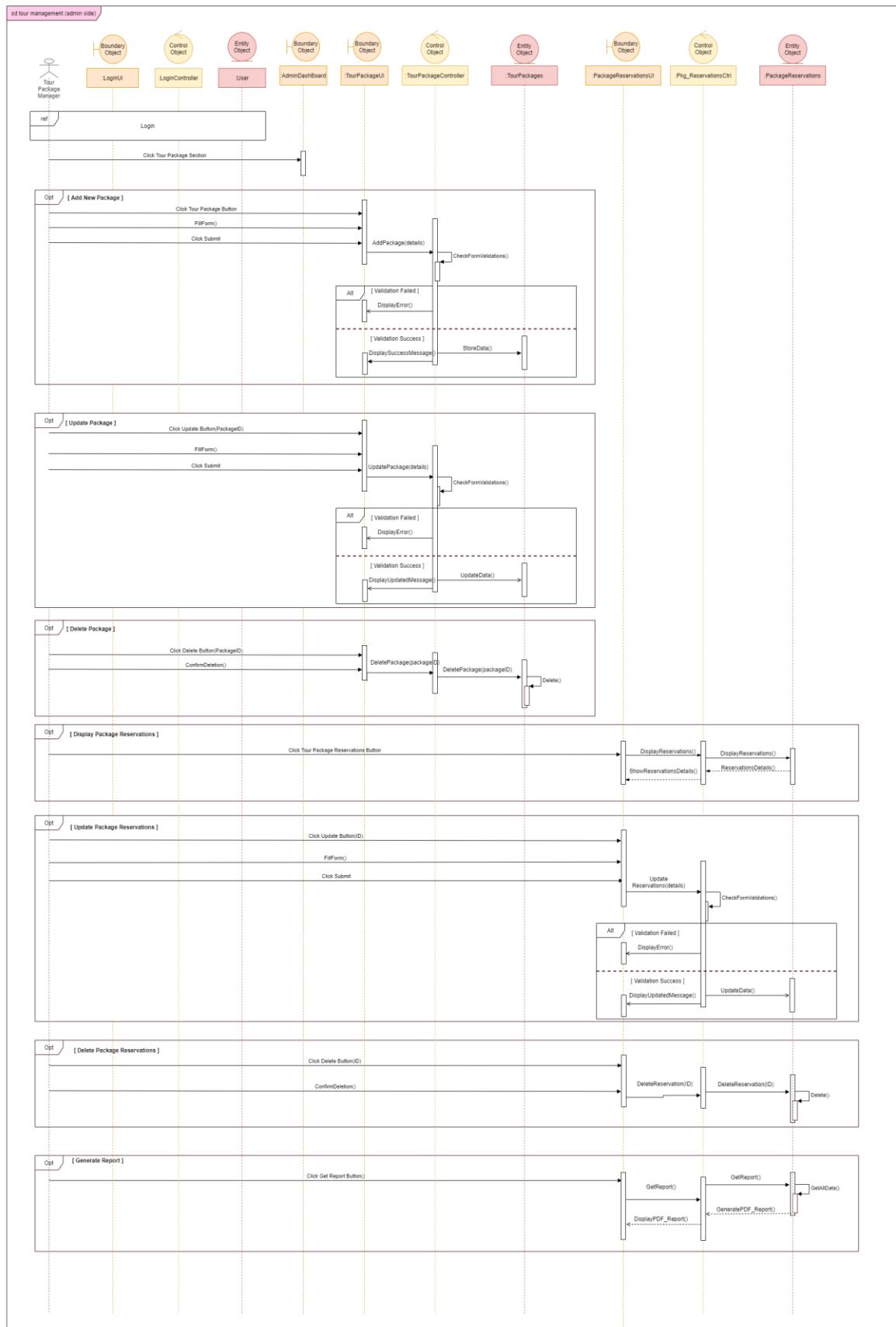
Travel Document Management (Admin side)

Figure 31



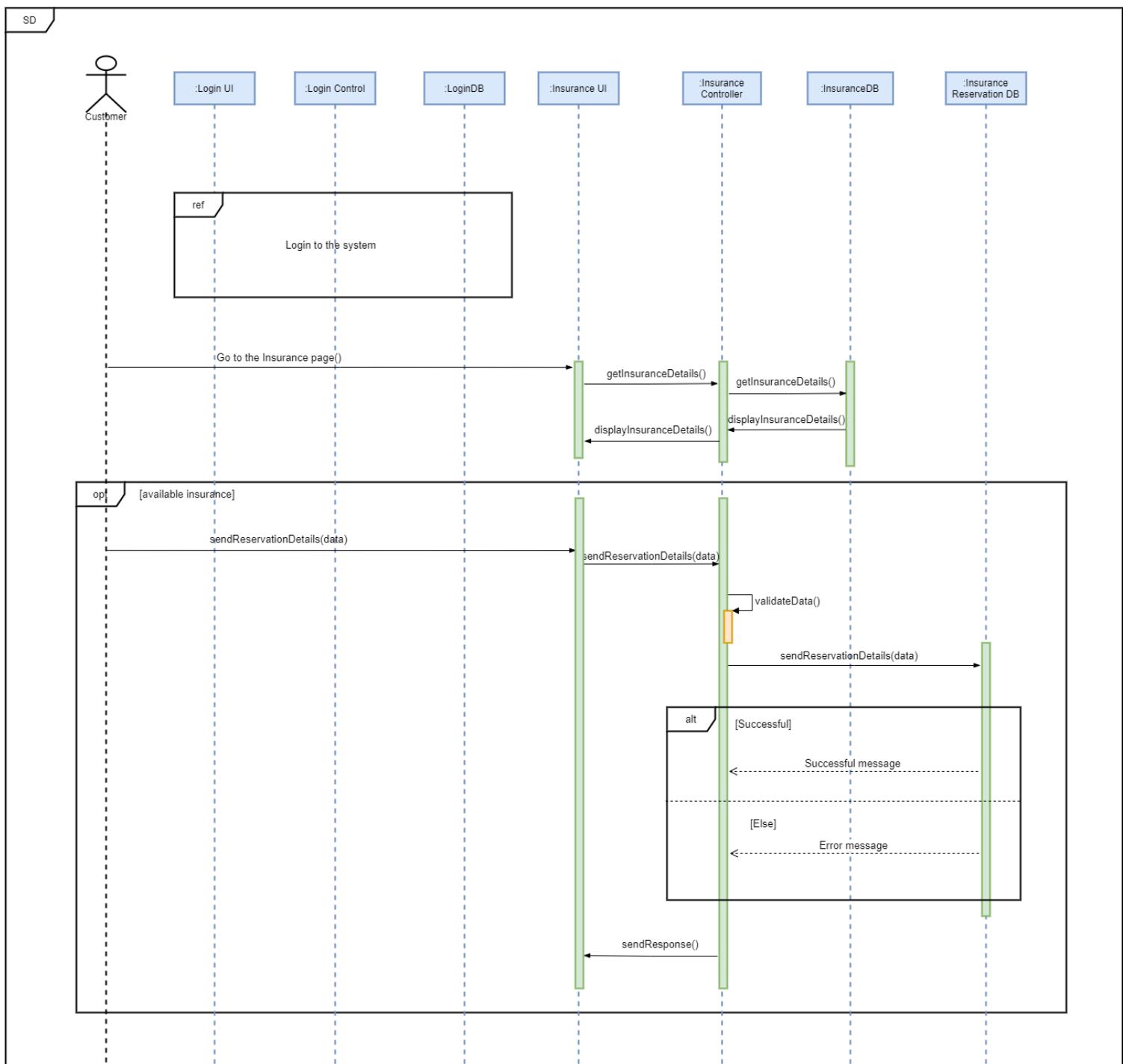
Tour Package (user side)

Figure 32



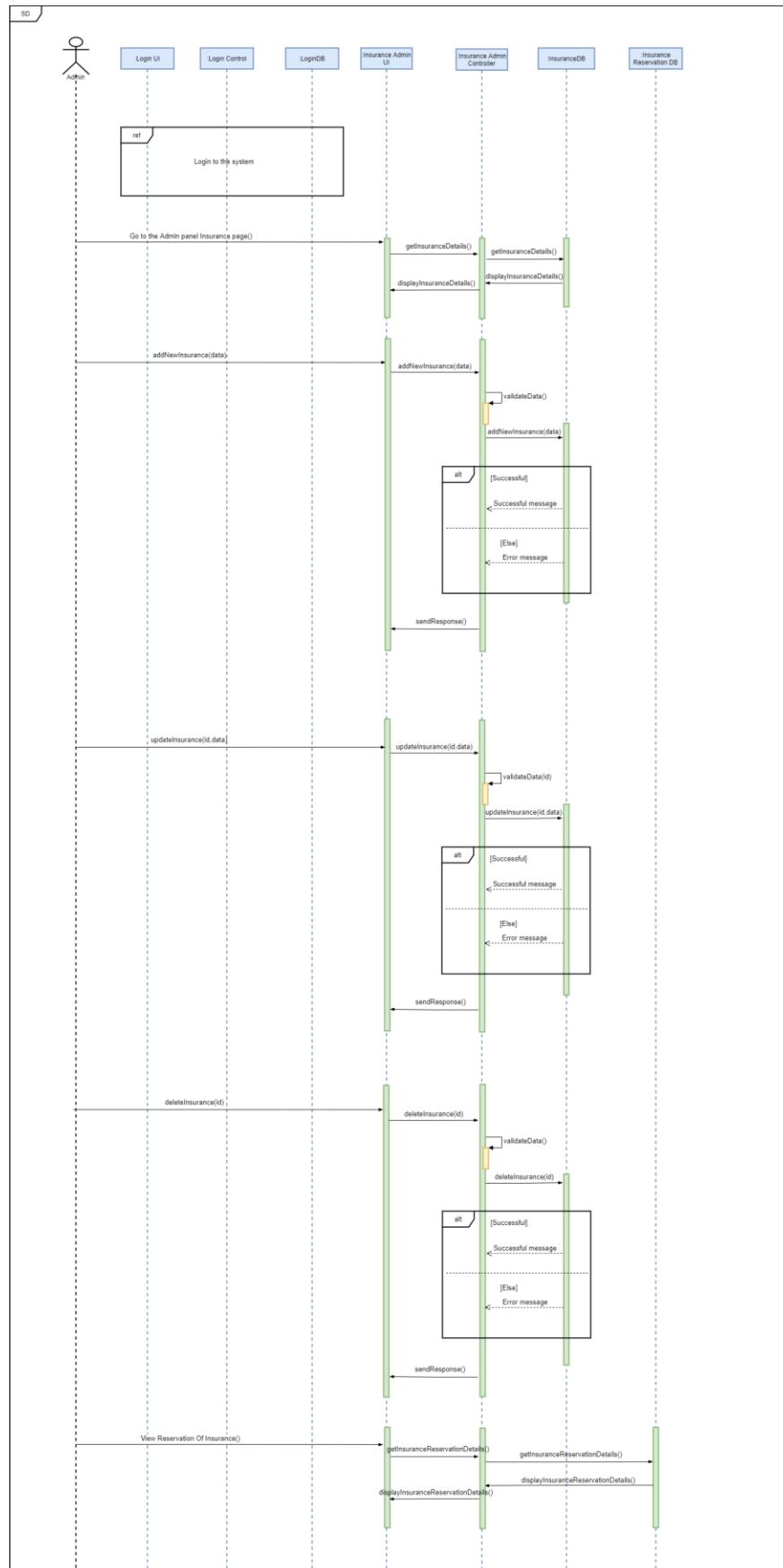
Tour Package Management (Admin side)

Figure 33



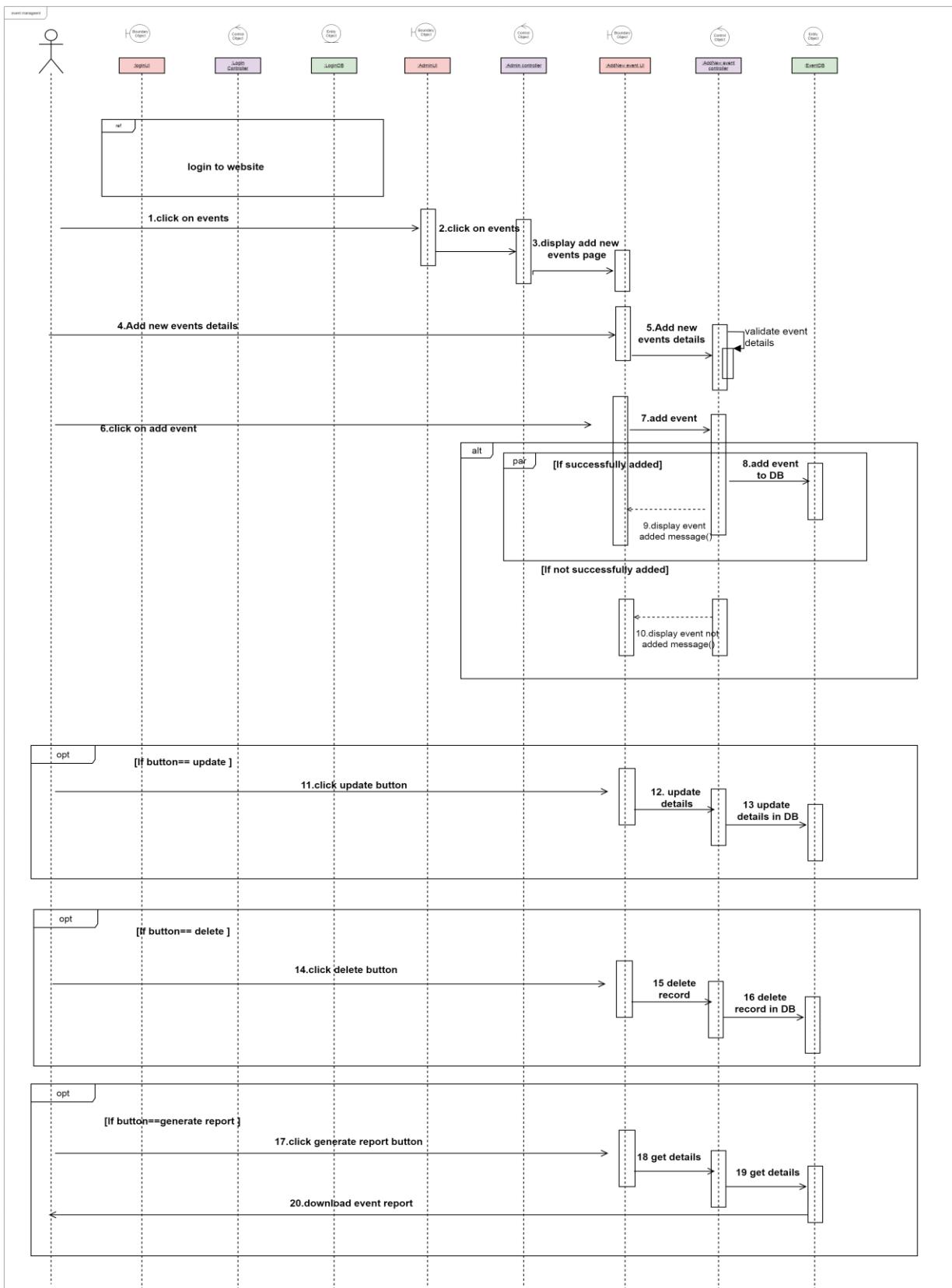
Travel Insurance Management (Admin side)

Figure 34



Travel Insurance Management (Admin side)

Figure 35



Event Management

Figure 36

2.2.3 User Interfaces

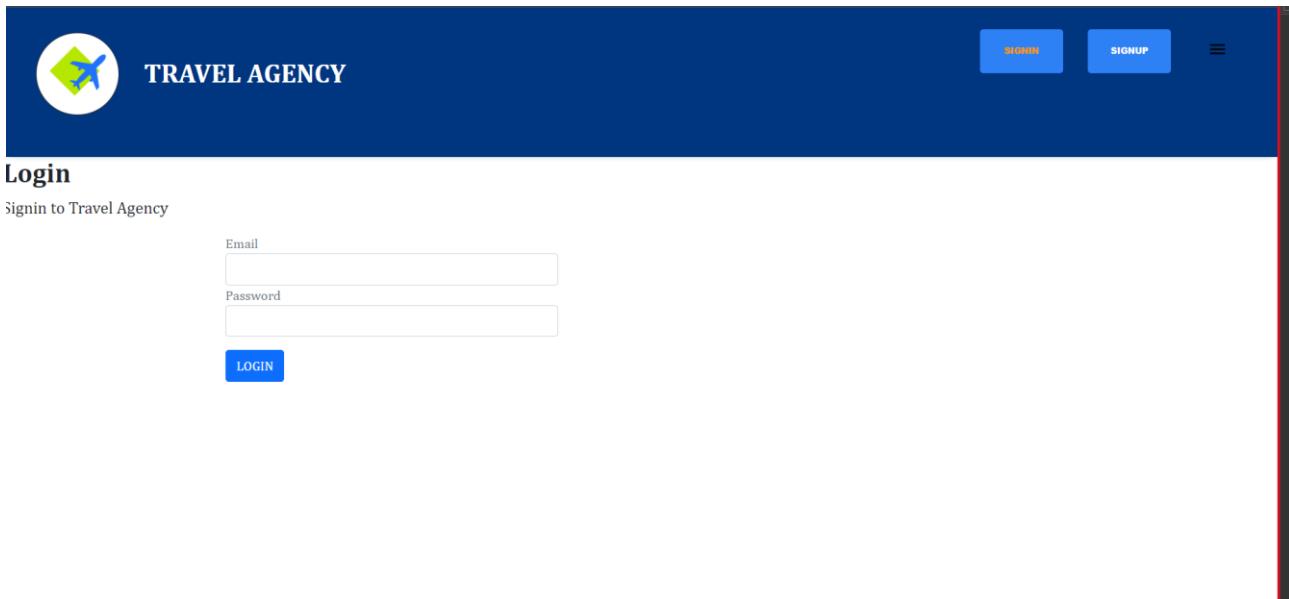


Figure 37 – Login page

A screenshot of a mobile application interface showing user details. The top navigation bar is dark blue with a white logo icon on the left, the text "TRAVEL AGENCY" in the center, and three buttons on the right labeled "SIGNIN", "SIGNUP", and a menu icon. Below the navigation is a white header section with the word "Dashboard" in bold black font. Underneath it is a sub-header "Good day nuwanika gunawardhena r!". The main content area has a light gray background. On the left, there is a sidebar with a "User Links" section containing a "EDIT PROFILE" button. To the right is a green "YOUR ACCOUNT" button. Below these are several user profile details listed in a table-like format:

Full Name	nuwanika gunawardhena r
Phone Number	706565333
Address	N02/3, Hakmana,Sri Lanka
Zip/Postal code	3444
Date of Birth	1999-04-04T00:00:00.000Z
Email	nuwanika@gmail.com
Country	Sri Lanka

A message "You are a Registered User" is displayed in a white box at the bottom of the screen.

Figure 38 – User Details page

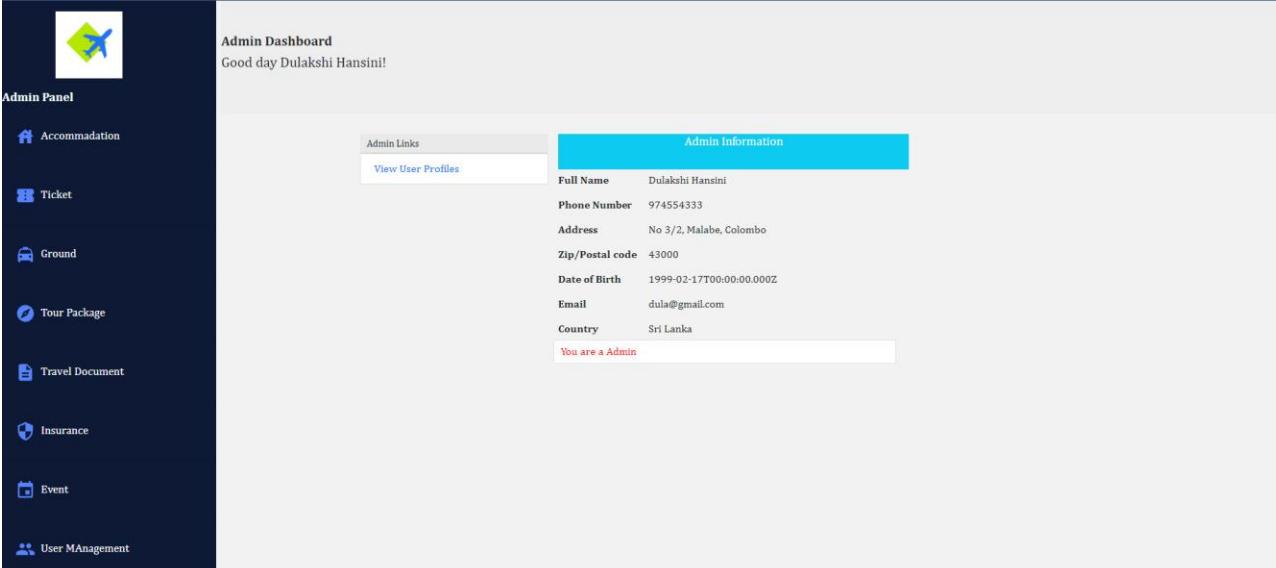
Profile

EDIT PROFILE

Full Name	nuwanika gunawardena r
Email	nuwanika@gmail.com
Date of Birth	1999-04-04T00:00:00.000Z
Address	N02/3, Hakmana,Sri Lanka
Zip/Postal Code	3444
Country	Sri Lanka
Phone Number	706565333

UPDATE

Figure 39 – User Details Edit page



The screenshot shows the Admin Dashboard interface. On the left is a sidebar titled "Admin Panel" with icons for Accommodation, Ticket, Ground, Tour Package, Travel Document, Insurance, Event, and User Management. The main area has a header "Admin Dashboard" and a greeting "Good day Dulakshi Hansini!". Below this is a "View User Profiles" link. To the right is a "Admin Information" section containing the following data:

Full Name	Dulakshi Hansini
Phone Number	974554333
Address	No 3/2, Malabe, Colombo
Zip/Postal code	43000
Date of Birth	1999-02-17T00:00:00.000Z
Email	dula@gmail.com
Country	Sri Lanka

A message at the bottom states "You are a Admin".

Figure 40— Admin Dashboard page



Manage Users
User Profile List

All Users

Full Name	Date of Birth	Zip/Postal Code	Phone	Email	Country	ACTION
nuwanika gunawardena r	1999-04-04T00:00:00.000Z	3444	706655333	nuwanika@gmail.com	Sri Lanka	
Dulakshi Hanini	1999-02-17T00:00:00.000Z	43000	774554333	dula@gmail.com	Sri Lanka	
Tharuka Goyatharan	1999-09-22T00:00:00.000Z	82000	714550733	tharuka.goyatharan.5@gmail.com	Sri Lanka	
yasiru navoda	1999-09-30T00:00:00.000Z	81000	783543103	yasirunavoda510@gmail.com	Sri Lanka	
Oshan gomes	1999-04-26T00:00:00.000Z	81020	715282784	oshangomes@gmail.com	Sri Lanka	
sachini abeywardhana	2021-10-12T00:00:00.000Z	2345	71992026	sachiniach983@gmail.com	Sri Lanka	
Sachini Razanga	1998-02-16T00:00:00.000Z	82270	71992026	sachiniach@gmail.com	Sri Lanka	
nethu sipuna	1999-01-13T00:00:00.000Z	81700	775444386	nethusipuna7@gmail.com	Sri Lanka	

[Generate Report PDF](#)

Figure 41 – Admin User Management page



TRAVEL AGENCY

[DASHBOARD](#) [SIGNOUT](#)

Book Your Accommodation here

[your reservations](#)



Name: Beach side hotel mirissa
Location: Mirissa
Type: hotel rooms
Phone: 412256942
Price: Rs5500/=



Name: Mandara hotel
Location: welligama
Type: hotel rooms
Phone: 415565954
Price: Rs7500/=



Name: Bay view hotel
Location: welligama
Type: hotel rooms
Phone: 415057942



Name: warwick gardens
Location: Nuwara Eliya
Type: hotel rooms

Figure 42 – Accommodation view page

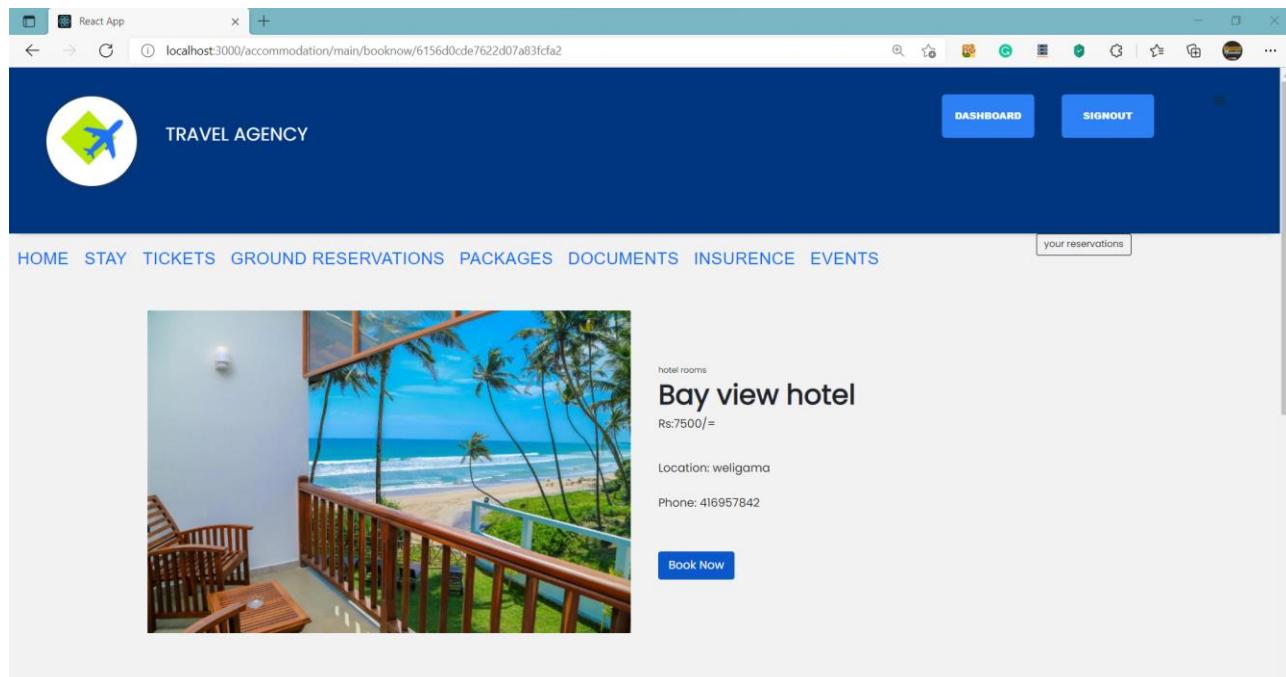


Figure 43 – Selected Accommodation details view page

A screenshot of a web browser displaying a travel agency's website. The header features a logo with a blue and green airplane icon, the text 'TRAVEL AGENCY', and navigation links for 'DASHBOARD' and 'SIGNOUT'. Below the header is a navigation bar with links: 'HOME', 'STAY', 'TICKETS', 'GROUND RESERVATIONS', 'PACKAGES', 'DOCUMENTS', 'INSURANCE', 'EVENTS', and a button labeled 'your reservations'. The main content area has a heading 'collect information' and several input fields for user details: 'User' (nethu nipuna), 'Email' (nethunipuna7@gmail.com), 'Booking Date' (mm/dd/yyyy), 'Due Date' (mm/dd/yyyy), 'No of Rooms' (empty), and 'no of adults' (empty).

Figure 44 – User details for Accommodation page

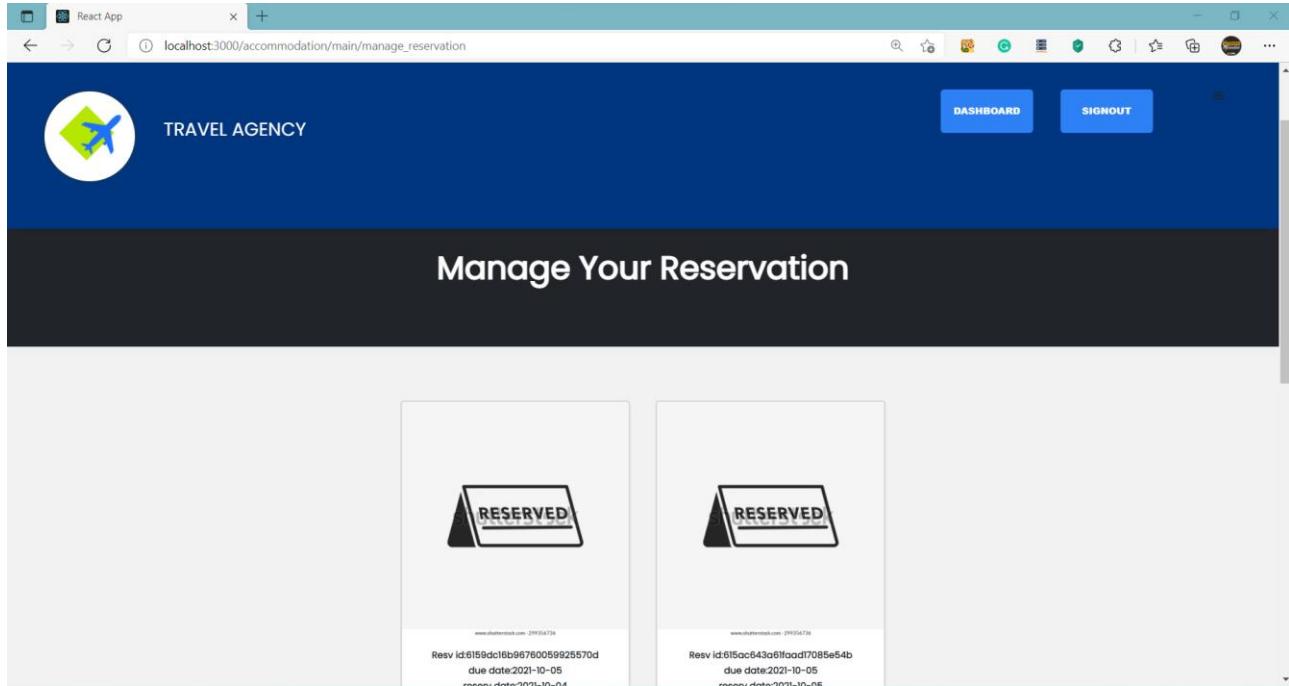


Figure 45 – User’s Reservation of Accommodation page

Accommodation Record List							
Reamrk Id	name	location	type	price	owner	phone	Action
1	Beach side hotel mirissa	Mirissa	hotel rooms	5500	mr nihal	412256942	Edit Delete
2	Mandara hotel	weligama	hotel rooms	7500	mandara pvt ltd	415565954	Edit Delete
3	Bay view hotel	weligama	hotel rooms	7500	mr sunil	416957842	Edit Delete
4	warwick gardens	Nuwara Eliya	hotel rooms	10000	Mr nihal gamgoda	775422269	Edit Delete
5	Nature cottage	Ella	Home stay	5000	Mr Sunil	776359841	Edit Delete
6	Hotel J Unawatuna	Unawatuna, Galle	hotel rooms	7000	hotel j pvt ltd	774222357	Edit Delete
8	White Villa	Galle	hotel rooms	8500	mr nihal	412212340	Edit Delete

Figure 46 – List of Accommodation page

Accommodation Management

All Accommodations Add new Accommodation

Add new Accommodation here

remark_ID

Name

Type

price

owner

City

Phone pattern: (xxxxxxxx)

image link

submit form

Figure 47 – Insert new Accommodation page

Accommodation Management

All Accommodations Add new Accommodation

Update Accommodation details

Remark ID:

1

Name:

Beach side hotel mirissa

Location:

Mirissa

Type

hotel rooms

Price:

5500

Owner:

mr nihal

Phone:

412256942

img link:

https://dynamic-media-cdn.tripadvisor.com/media/photo-o/16/a9/c0/52/mandara-resort.jpg?w=1200&h=-&s=1

Update Record

Figure 48 – Update Accommodation page

React App x +

localhost:3000/admin_panel/accommodation_admin/accommodation_servations

Accommodation Management All Accommodations Add new Accommodation

Accommodation Reservation Record List

Reservation_ID	User	email	Accommodation_ID	Due date	reserv date	no of accommodations	no fo adults	no of child	price	Action
6159dc16b96760059925570d	nethu nipuna	nethunipuna7@gmail.com	6156aa4de7e22d07a83fce93	2021-10-05	2021-10-04	2	3	2	15000	Edit Delete
615ac643a6f1faad17085e54b	nethu nipuna	nethunipuna7@gmail.com	615ac5c9a6f1faad17085e53e	2021-10-05	2021-10-05	2	3	2	17000	Edit Delete

Generate PDF

Figure 49 – Reservation List of Accommodation page

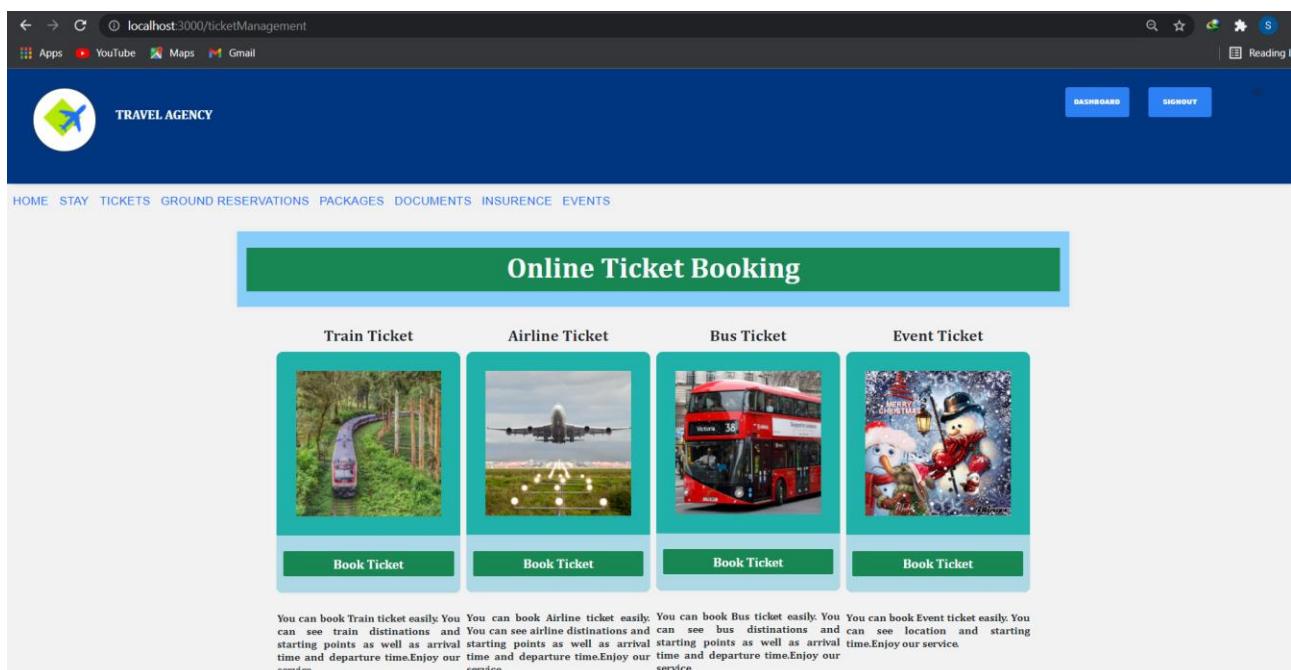


Figure 50 – Ticket Category page

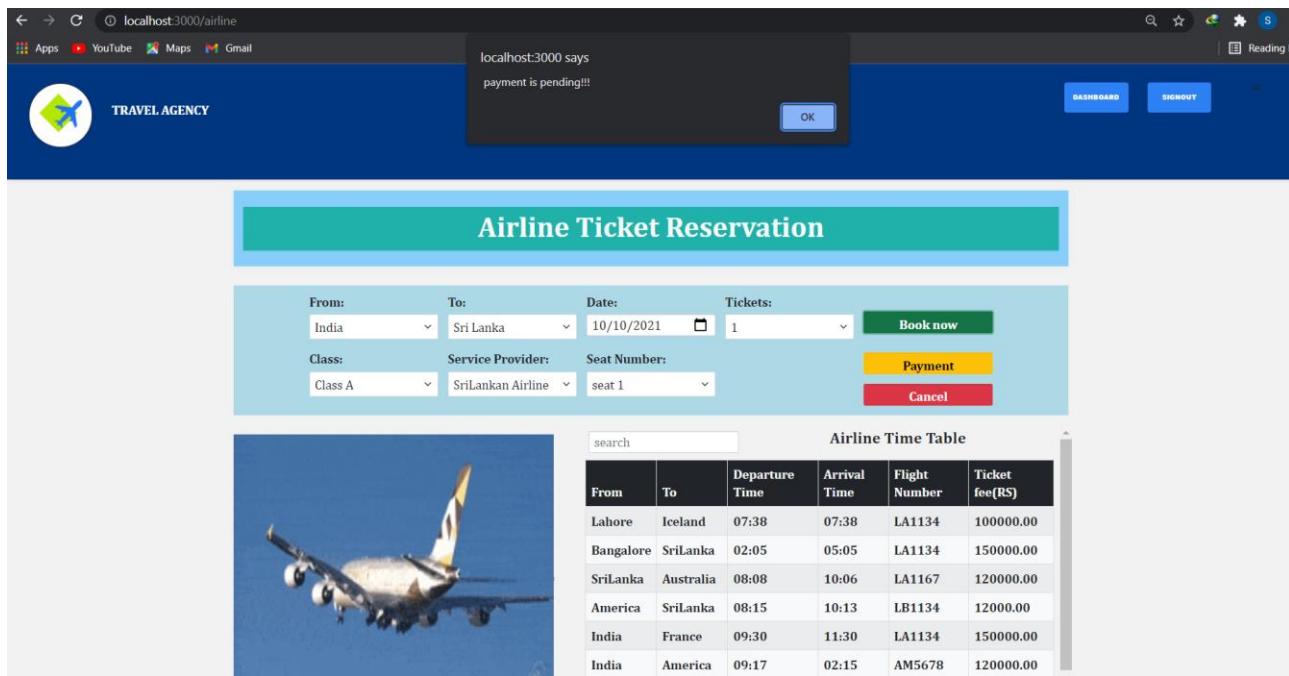


Figure 51 – Airplane Ticket Reservation page

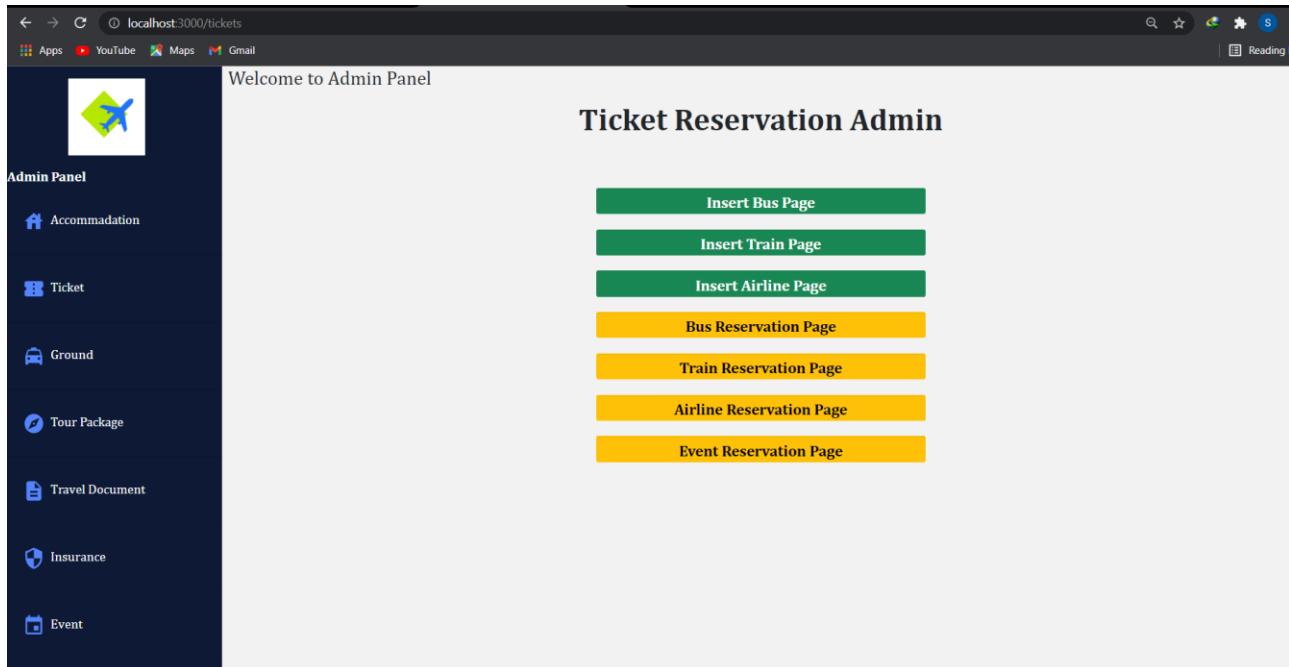


Figure 52 – Ticket Reservation Admin page

localhost:3000/airline

Admin Panel

- Accommodation
- Ticket
- Ground
- Tour Package
- Travel Document
- Insurance
- Event
- User Management

INSERT PLANE

Ticket Type: plane

From: London

To: Bangalore

Flight Number: LB1134

Service Provider: Air Lanka

Departure Time: 01:56 PM

Arrival Time: 01:56 PM

Date: 10/17/2021

No Of Seats: 100

Ticket Fee: 150000.00

Add now

Insert Plane Details

From	To	Flight Number	Service Provider	Departure Time	Arrival Time	Date	No of Seats	Ticket Fee(Rs)	Update	Delete
Lahore	Iceland	LA1134	SriLankanAirline	07:38	07:38	2021-10-24	120	100000.00	update	Delete

Figure 53 – Airplane Ticket insert Admin page

localhost:3000 says

Airplane Sucessfully added

OK

Insert Plane Details

From	To	Flight Number	Service Provider	Departure Time	Arrival Time	Date	No of Seats	Ticket Fee(Rs)	Update	Delete
London	Iceland	LA1134	SriLankanAirline	07:38	07:38	2021-10-24	120	100000.00	update	Delete
Lahore	SriLanka	LA1134	AirLanka	02:05	05:05	2021-10-17	100	150000.00	update	Delete
Bangalore	Australia	LA1167	AirLanka	08:08	10:06	2021-10-23	120	120000.00	update	Delete
India	SriLanka	LB1134	AirLanka	08:15	10:13	2021-10-31	120	12000.00	update	Delete
China	France	LA1134	SriLankanAirline	09:30	11:30	2021-10-17	120	150000.00	update	Delete
Japan	America	AM5678	SriLankanAirline	09:17	02:15	2021-10-22	80	120000.00	update	Delete
South Korea	Australia	LA1134	SriLankanAirline	00:56	11:56	2021-10-08	120	120000.00	update	Delete

Figure 54 – Airplane Tickets view Admin page

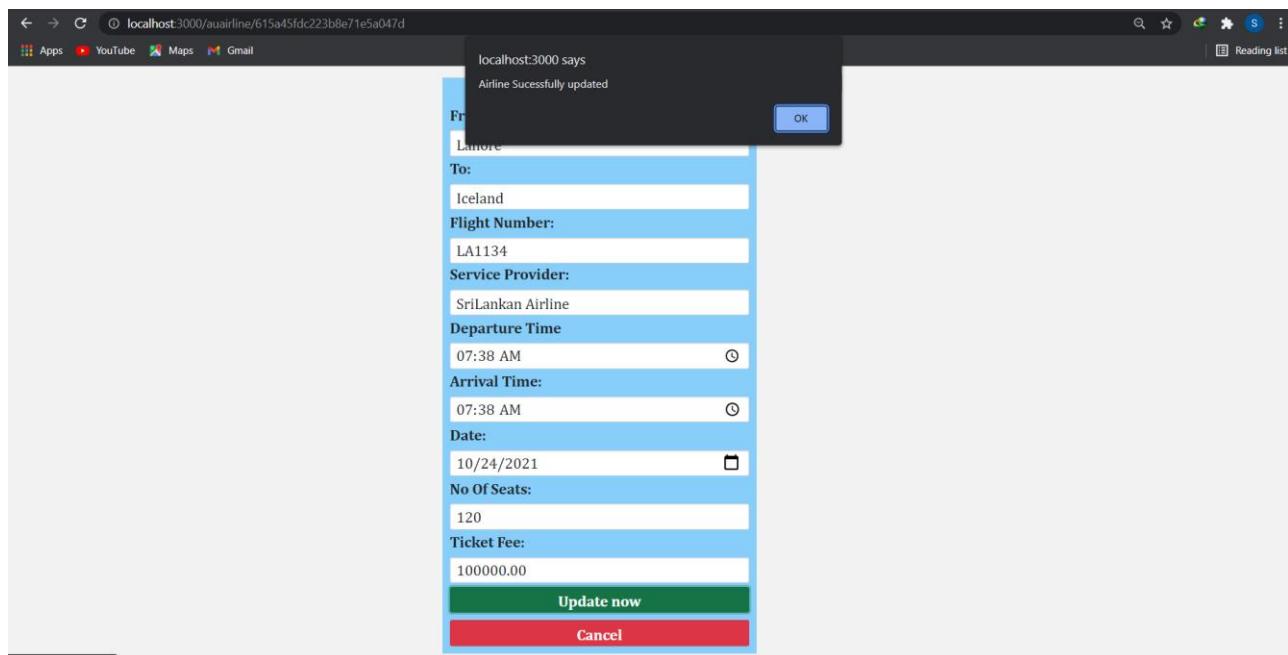


Figure 55 – Airplane Tickets Update Admin page

To	Flight Number	Service Provider	Departure Time	Arrival Time	Date	No of Seats	Ticket Fee(Rs)	Update	Delete
Iceland	LA1134	SriLankanAirline	07:38	07:38	2021-10-24	120	100000.00	<button>update</button>	<button>Delete</button>
Sri Lanka	LA1134	AirLanka	02:05	05:05	2021-10-17	100	150000.00	<button>update</button>	<button>Delete</button>
Australia	LA1167	AirLanka	08:08	10:06	2021-10-23	120	120000.00	<button>update</button>	<button>Delete</button>
Sri Lanka	LB1134	AirLanka	08:15	10:13	2021-10-31	120	12000.00	<button>update</button>	<button>Delete</button>
France	LA1134	SriLankanAirline	09:30	11:30	2021-10-17	120	150000.00	<button>update</button>	<button>Delete</button>
America	AM5678	SriLankanAirline	09:17	02:15	2021-10-22	80	120000.00	<button>update</button>	<button>Delete</button>
America	LA1134	SriLankanAirline	10:20	01:20	2021-10-16	120	100000.00	<button>update</button>	<button>Delete</button>

Figure 56 – Airplane Tickets Delete Admin page

The screenshot shows a travel agency website with a dark blue header. In the top left is a logo of a green and blue airplane icon inside a circle. To its right, the words "TRAVEL AGENCY" are written in white capital letters. On the far right of the header are three buttons: "SIGN IN", "SIGN UP", and a menu icon represented by three horizontal lines. Below the header, a navigation bar contains links: HOME, STAY, TICKETS, GROUND RESERVATIONS, PACKAGES, DOCUMENTS, INSURANCE, and EVENTS. A red banner across the middle of the page says "Online Ground Resources Booking". Below this, there are three categories: "Airport Taxi" (with an icon of people at an airport), "Car" (with an icon of a car and a person), and "Guide" (with an icon of a van). Each category has a "CLICK TO BOOK" button and a subtitle: "Reservation of your Airport Taxi", "Reservation of your Car", and "Reservation of your Travel Guide".

Figure 57 – Ground Resource category page

This screenshot shows the "Car Reservation" section of the website. At the top, it features the same header and navigation bar as Figure 57. Below the header, a black bar contains the text "Car Reservation". Underneath this, there is a search bar with the word "SEARCH" and a blank input field. The main content area displays four different car models in separate boxes: a silver sedan, a red sports car, a gold SUV, and a blue hatchback with its hood open. Each car box includes the word "Car", the brand and model information (e.g., "Brand : BMW Model : i8"), and a "BOOK NOW" button.

Figure 58 – (Ground Resource) Car reservation category page

The screenshot shows a travel agency interface. At the top, there's a logo with a blue and green airplane icon and the text "TRAVEL AGENCY". Below the header, a message box displays "localhost:3000 says Car Reservation successfully" with an "OK" button. To the right, there are "SIGN IN" and "SIGN UP" buttons and a menu icon. The main content area has a yellow header bar with the text "Car Profile". On the left, there's a large image of a red Audi R8 sports car parked in front of a snowy mountain range. To the right of the image, the car's details are listed: **AUDI**, Model : A6, Car ID : 004, Location : Hambantota, and Cost Per KM : 35. On the far right, there's a booking form with fields for Car ID (004), Type (Car), Date (10/16/2021), Time (04:50 PM), From (Matara), and To (Galle). It also features a "BOOK NOW" button and a "CANCEL" button.

Figure 59 – (Ground Resource) Car reservation page

The screenshot shows the "Welcome to Admin Panel" page. On the left, there's a sidebar titled "Admin Panel" with icons and links for Accommodation, Ticket, Ground, Tour Package, Travel Document, Insurance, and Event. The main content area has a title "Welcome to Admin Panel" and several buttons: "GUIDE MANAGEMENT" (green), "CAR MANAGEMENT" (green), "CAR RESERVATIONS" (red), and "GUIDE RESERVATION" (red).

Figure 60 – Ground Resource Admin page

Date	Time	From	To	Type	Car ID	User	Update	Delete
2021-10-21	01:13	matara	galle	Car	001	Dulakshi Hansini	UPDATE	DELETE
2021-10-13	01:18	Matara	Kataragama	Car	003	Tharuka Gayashan	UPDATE	DELETE
2021-10-06	18:25	matara	colombo	Airport Taxi	002	Dulakshi Hansini	UPDATE	DELETE
2021-10-15	11:38	Matara	Hambantota	Car	001	Tharuka Gayashan	UPDATE	DELETE
2021-10-21	07:47	Galle	Hmbantota	Car	004	Tharuka Gayashan	UPDATE	DELETE

**GENERATE
REPORT
PDF**

Figure 61 – (Ground Resource) Car reservation list view Admin page

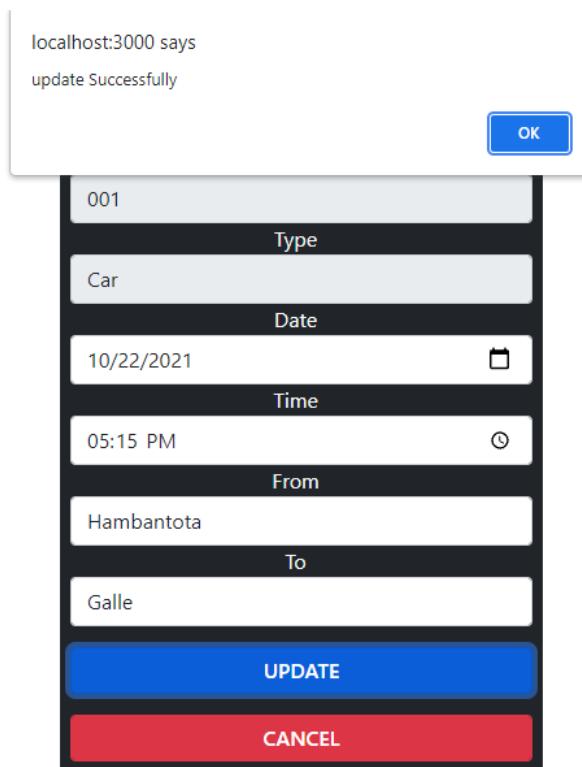


Figure 62 – (Ground Resource) Car reservation Update Admin page

localhost:3000 says
Deleted Successfully

OK

Date	Time	From						Update	Delete
2021-10-22	17:15	Hambantota	Galle	Car	001	Dulakshi Hansini	UPDATE	DELETE	
2021-10-13	01:18	Matara	Kataragama	Car	003	Tharuka Gayashan	UPDATE	DELETE	
2021-10-06	18:25	matara	colombo	Airport Taxi	002	Dulakshi Hansini	UPDATE	DELETE	
2021-10-15	11:38	Matara	Hambantota	Car	001	Tharuka Gayashan	UPDATE	DELETE	
2021-10-21	07:47	Galle	Hmbantota	Car	004	Tharuka Gayashan	UPDATE	DELETE	

GENERATE REPORT PDF

Waiting for localhost...

Figure 63 – (Ground Resource) Car reservation Delete Admin page

Information

Applications

Inquiry

Instruction of Add Document

Add Document

Add Document

Country name
Enter country name

Document Path
Enter URL Link

Document name
Enter Document name

Submit Date
Enter the date

Send

Figure 64 –Travel Document add page

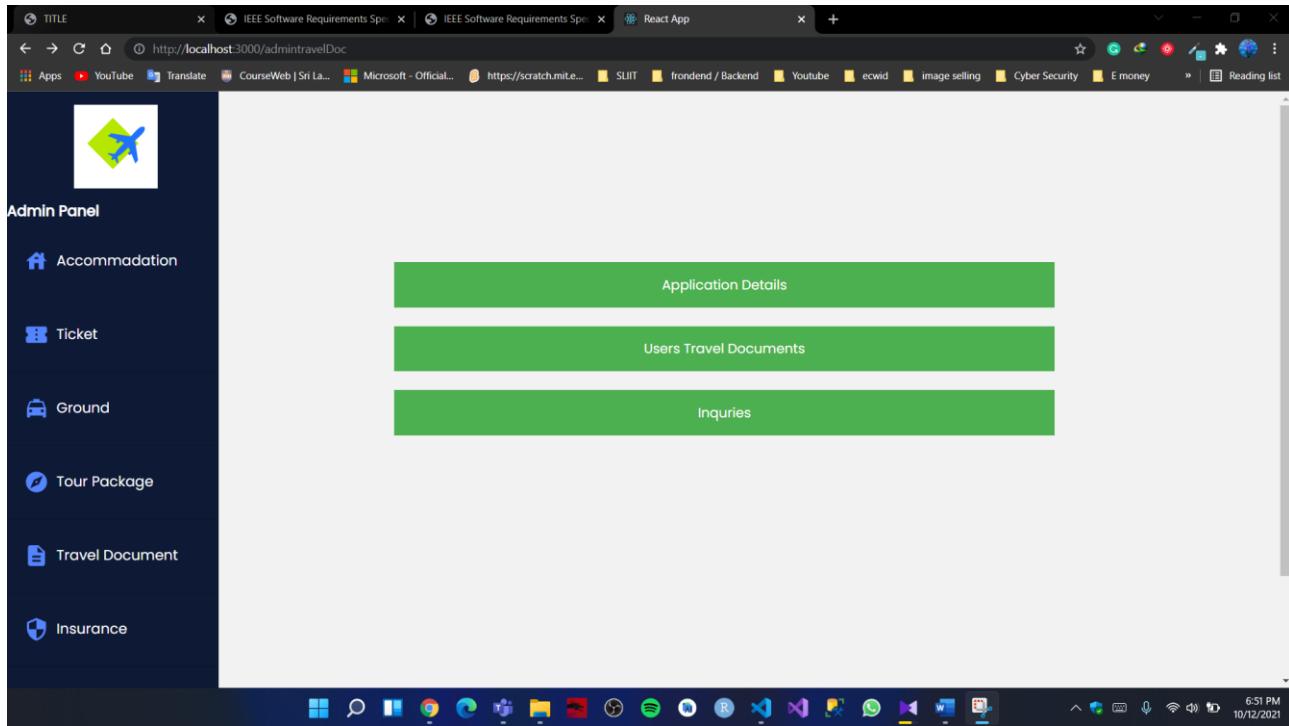


Figure 65 –Travel Document Admin page

#	Document name	Document path	Country name	Submitted date	Action
1	test	dsds	England	11/09/2021	<button>Edit</button> <button>Delete</button>
2	991171380V	https://drive.google.com/drive/folders/1f5BsVdq..._5sZMOLIBtCEloUnmitKGUR	England	10/4/2021	<button>Edit</button> <button>Delete</button>

Figure 66 – Customer's Travel Document added list view page

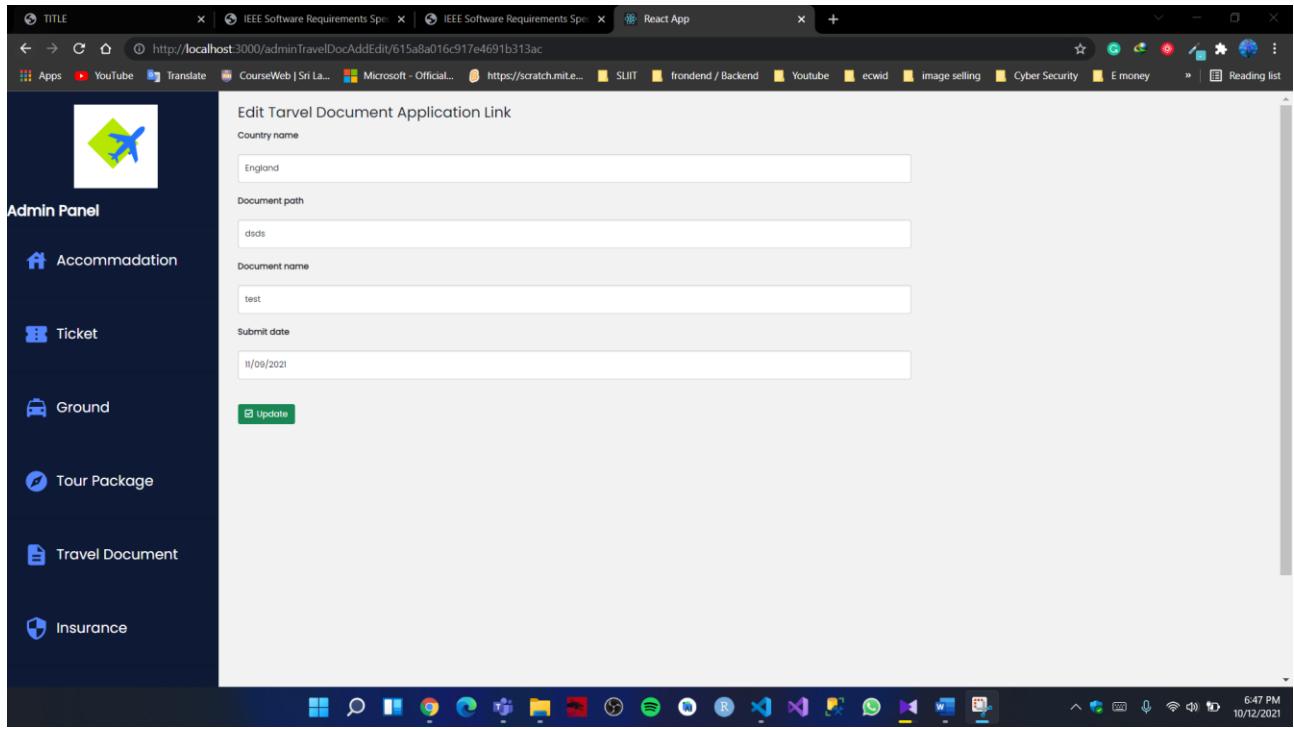


Figure 67 – Customer’s Travel Document update page

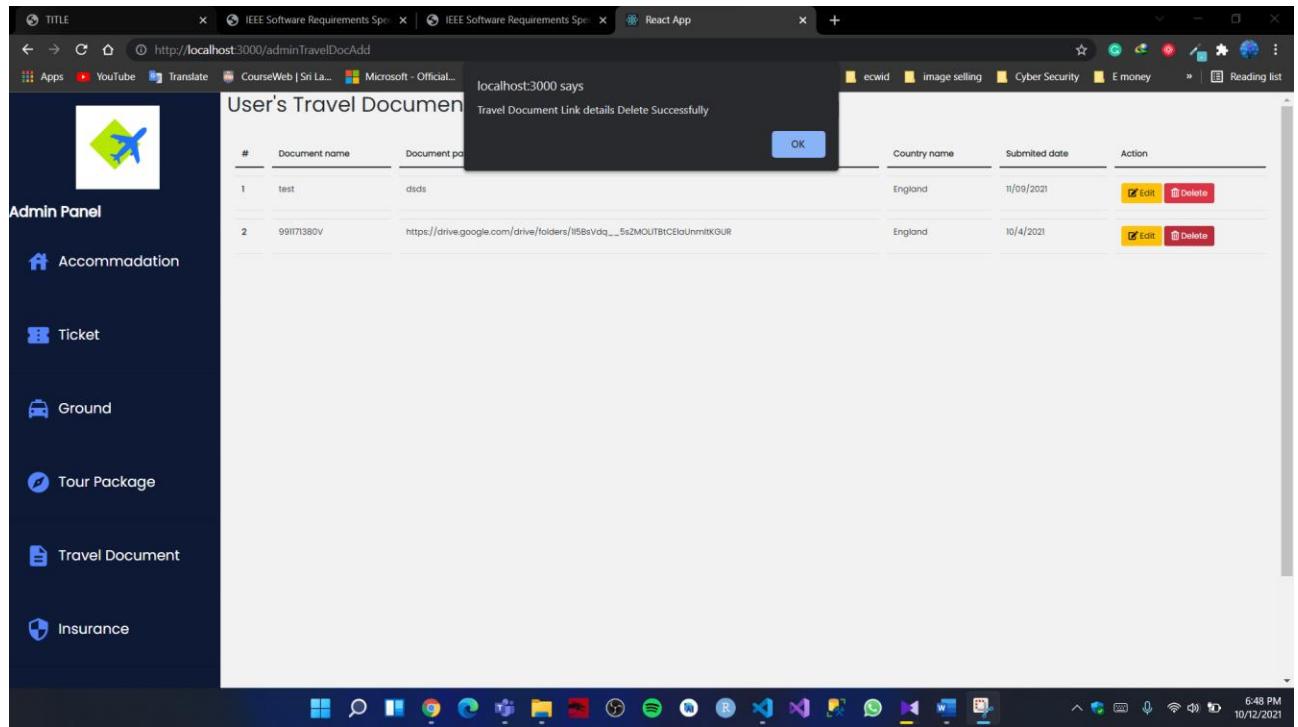


Figure 68 – Customer’s Travel Document Delete page

The screenshot displays a travel agency website interface. At the top, there is a navigation bar with links for Home, Stay, Tickets, Ground Reservations, Packages, Documents, Insurance, and Events. Below the navigation bar, a breadcrumb trail shows the user has navigated from Home to Tour Packages. A sidebar on the left contains a vertical flow diagram with four main nodes: Eat (represented by a plate icon), Travel (represented by a map icon), Sleep (represented by a bed icon), and Repeat (represented by a circular arrow icon). To the right of the sidebar, a section titled "Latest Packages" features a grid of six travel packages, each with a thumbnail image, title, type, and a "View" button. The packages are: "Universal Studios Singapore update" (Family & Holiday), "Adaaran Club Rannali – South Male' Atoll" (Family & Holiday), "Kuala Lumpur & Full Day Sunway Lagoon" (Adventure), "Athens" (Adventure), and "Dambadiva Pilgrimage" (Pilgrimage). At the bottom of the page, there is a footer section with a logo, contact information (Jaipur India, 5353 Road Avenue, phone number 95 711 9 5353, email "Give us a call"), useful links (About us, Services, Projects, Our Team, Contact us, Blog, Testimonials, Faq), a subscribe form, and social media links for Facebook, Twitter, Google+, LinkedIn, and Instagram.

Figure 69 – Tour Package Category view page

The screenshot shows a travel agency website interface. At the top, there's a navigation bar with links like 'HOME', 'STAY', 'TICKETS', 'GROUND RESERVATIONS', 'PACKAGES', 'DOCUMENTS', 'INSURANCE', and 'EVENTS'. A sidebar on the left lists categories such as 'NO OF DAYS' (4), 'TRANSPORTATION' (Speedboat), and 'MEAL PLAN' (Full Board). On the right, a 'Tickets and Prices' section shows a total of LKR 1,200,000 for two adults and one child. A green 'Proceed' button is at the bottom right of this section.

Figure 70 – Selected Tour Package Detail view page

The screenshot shows a payment page for the tour package. It features a 'Check and Pay' header and a central 'Details' section. The details include the package name ('Adaaran Club Rannalhi – South Male' Atoll'), date ('Sat Oct 09 2021 12:36:43 GMT+0530 (India Standard Time)'), and total price ('LKR : 1200000 /='). Below this, there's a 'Payment Details' section with fields for card number, expiration date, CV code, and card owner, along with logos for Visa, American Express, MasterCard, and PayPal. A large yellow 'Submit' button is at the bottom.

Figure 71 – Selected Tour Package Payment page

Tour Packages

Add Project **✉️** **3**

ID	Name	Type	# Of Days	Meal-Plan	Transportation	Activity	Location	Price (Adult)	Price (Child)	Action
TP001	Universal Studios Singapore update	Family & Holiday updated	5	Breakfast and Dinner updated	Airport Transfers updated	Ride the roller coasters updated	Sentosa Gateway, Sentosa Island, Singapore 098269, Singapore updated	100000	50000	>Edit Del
TP002	Adaaran Club Rannathi – South Male' Atoll	Family & Holiday	4	Full Board	Speedboat	scuba diving	Adaaran Club Rannathi – South Male' Atoll	500000	200000	Edit Del
TP003	Kuala Lumpur & Full Day Sunway Lagoon	Adventure	4	Breakfast	Airport Transfers	Experience Full Day Sunway Lagoon & City tour	Kuala Lumpur	120000	60000	Edit Del
TP004	Athens	Adventure	6	Bed & Breakfast	Airport – Hotel – Airport PVT Basis	Visit Corinth Canal, the Lions Gate, the Cyclopean Walls	Delphi & Argolis	500000	250000	Edit Del
TP005	Dombadiwa Pilgrimage	Pilgrimage	7	Bed & Breakfast	Airport transfers	visit of ancient places	Dhabadiwa	120000	60000	Edit Del

Figure 72 – Tour Package Admin page

Update Tour Package

Add Project **✉️** **3**

ID	Name	Type
TP001	Universal Studios Singapore update	Family & Holiday updated
# Of Days	Meal-Plan	Transportation
5	Breakfast and Dinner updated	Airport Transfers updated
Price (Adult)	Price (Child)	Activity
100000	50000	Ride the roller coasters updated
Location	Image Link	
8 Sentosa Gateway, Sentosa Island, Singapore 098269, Singapore updated	https://a.cdn-hotels.com/gdcs/production27/d1203/fdc9472c-0a07-4eb7-88e	

Update

Figure 73 – Tour Package Update Admin page

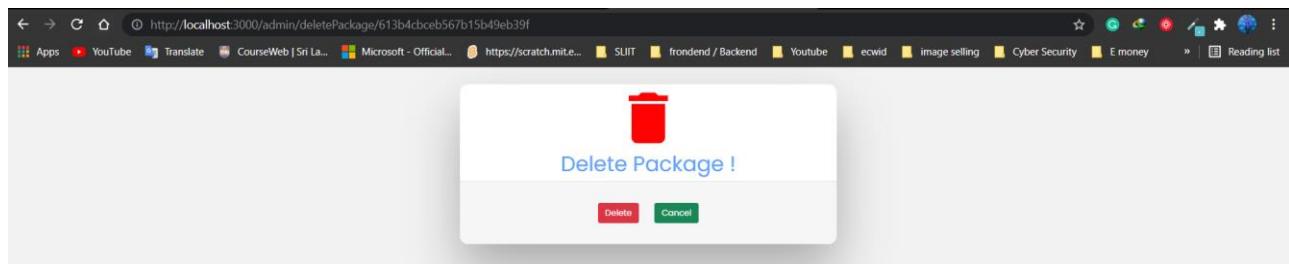


Figure 74 – Tour Package Delete Admin page

Name	E-mail	PackageID	packageStatus	dueDate	reservationDate	noOfAdults	noOfChilds	paidAmount	Action
yasiru navoda	yasirunavoda518@gmail.com	TP001	completed	Tue Oct 12 2021 06:44:34 GMT+0530 (India Standard Time)	Mon Oct 04 2021 08:58:48 GMT+0530 (India Standard Time)	2	1	250000	<button>Edit</button> <button>Del</button>
Oshen gomes	oshengomes@gmail.com	TP003	completed	Fri Oct 22 2021 11:57:05 GMT+0530 (India Standard Time)	Mon Oct 04 2021 12:00:29 GMT+0530 (India Standard Time)	4	2	600000	<button>Edit</button> <button>Del</button>
nuwanika gunawardhena r	nuwanika@gmail.com	TP005	completed	Wed Oct 13 2021 12:34:11 GMT+0530 (India Standard Time)	Mon Oct 04 2021 12:34:49 GMT+0530 (India Standard Time)	3	1	420000	<button>Edit</button> <button>Del</button>

Figure 75 – Tour Package reservation list view Admin page

Update Reservation

Package ID	Status	Due Date
TP001	completed	Tue Oct 12 2021 06:44:34 GMT+0530 (India Standard Time)
Reservation Date	No Of Adults	No Of Childs
Mon Oct 04 2021 06:58:48 GMT+0530 (India Standard Time)	2	1
Paid Amount		
250000		

Update

Figure 76 – Tour Package reservation Update Admin page

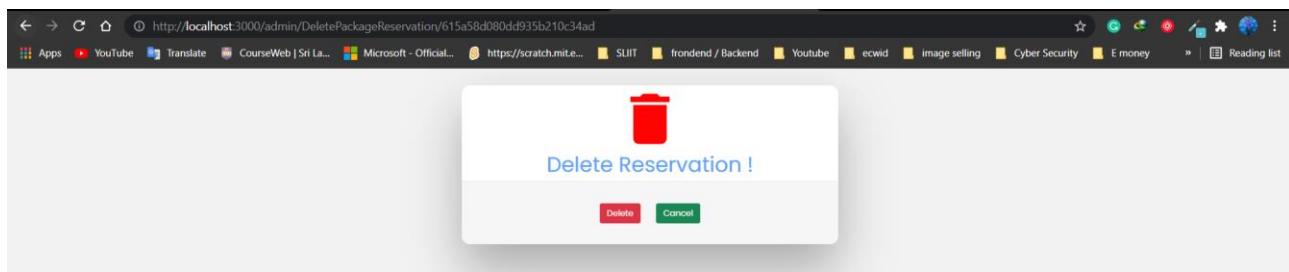


Figure 77 – Tour Package reservation Delete Admin page

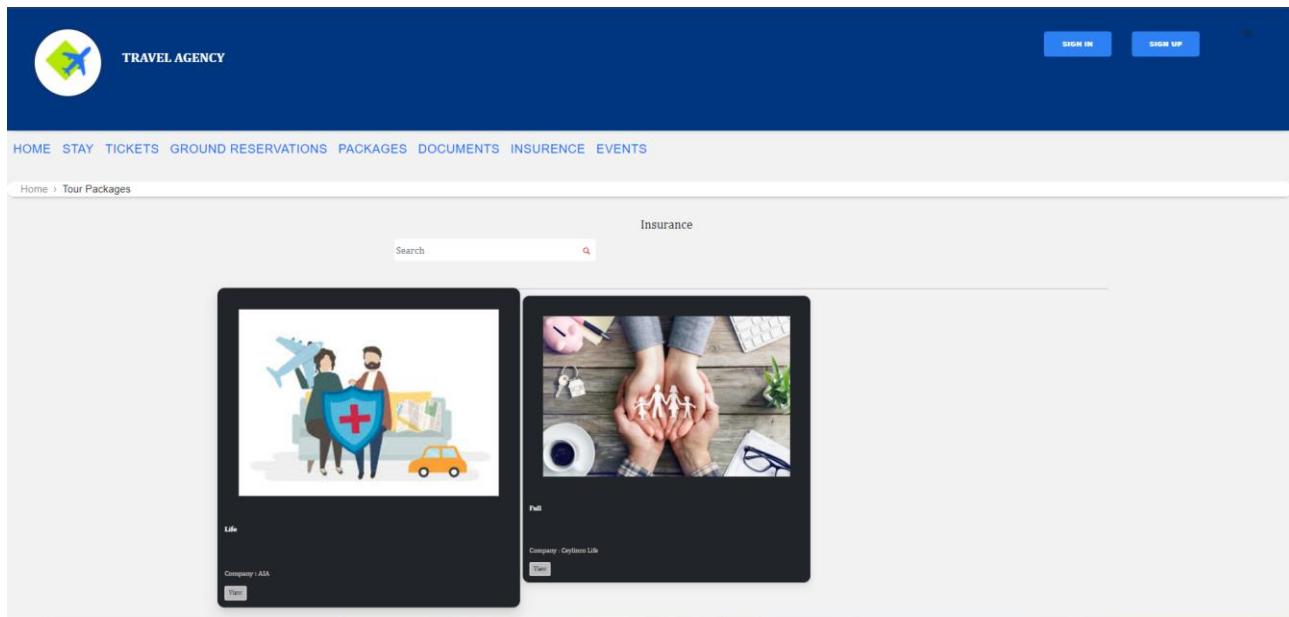


Figure 78 – Insurance Category view page

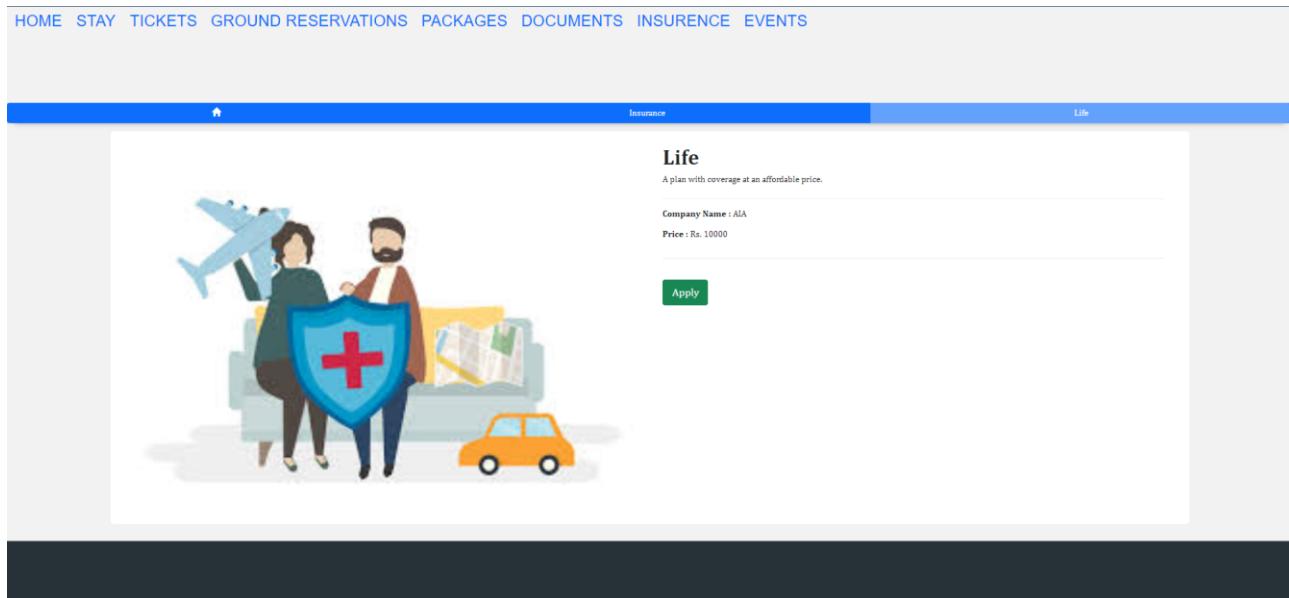
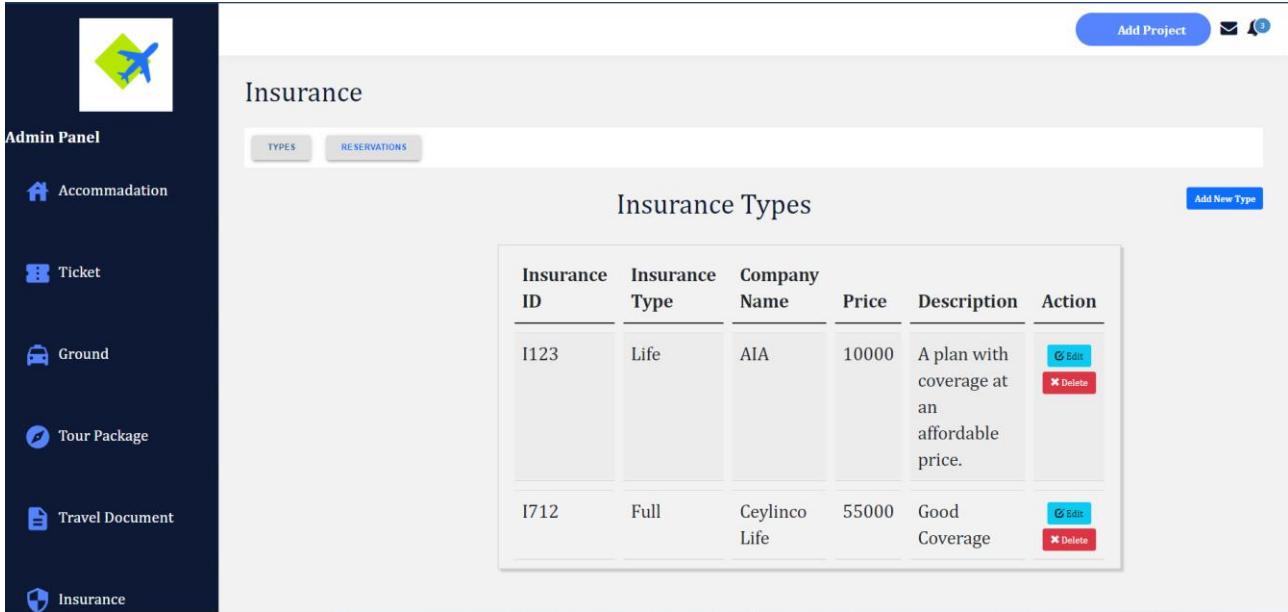


Figure 79 – Selected Insurance Details view page

Figure 80 – Selected Insurance Details Resevarition page

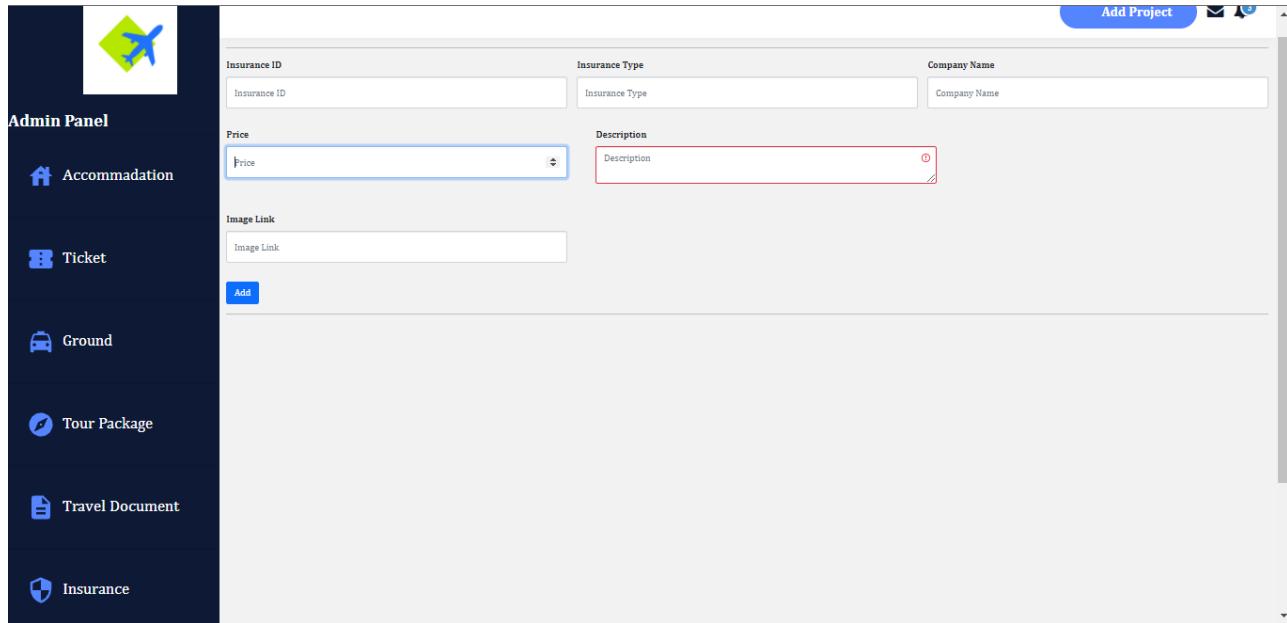
Figure 81 – Insurance Admin page



The screenshot shows the Admin Panel interface for managing insurance types. The left sidebar contains navigation links for Accommodation, Ticket, Ground, Tour Package, Travel Document, and Insurance. The main content area is titled "Insurance" and has tabs for "TYPES" and "RESERVATIONS". The "TYPES" tab is selected, displaying a table titled "Insurance Types" with two rows of data. The columns are: Insurance ID, Insurance Type, Company Name, Price, Description, and Action. The first row (I123) has values: Life, AIA, 10000, "A plan with coverage at an affordable price.", and edit/delete buttons. The second row (I712) has values: Full, Ceylinco Life, 55000, "Good Coverage", and edit/delete buttons.

Insurance ID	Insurance Type	Company Name	Price	Description	Action
I123	Life	AIA	10000	A plan with coverage at an affordable price.	<button>Edit</button> <button>Delete</button>
I712	Full	Ceylinco Life	55000	Good Coverage	<button>Edit</button> <button>Delete</button>

Figure 82 – Insurance type list view Admin page



The screenshot shows the Admin Panel interface for adding a new insurance type. The left sidebar contains navigation links for Accommodation, Ticket, Ground, Tour Package, Travel Document, and Insurance. The main content area has fields for Insurance ID, Insurance Type, Company Name, Price, Description, Image Link, and an Add button. The "Description" field is highlighted with a red border, indicating it is a required field.

Figure 83 – Insurance type Add Admin page

Add Project ✉️ 🔔

Insurance ID	Insurance Type	Company Name
1123	Life	AIA
Price	Description	
10000	A plan with coverage at an affordable price.	✖️

Image Link: <https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTzrm8Tw00Kmg7B...>

Update

Accommodation

Ticket

Ground

Tour Package

Travel Document

Insurance

Figure 84 –Insurance type Update Admin page

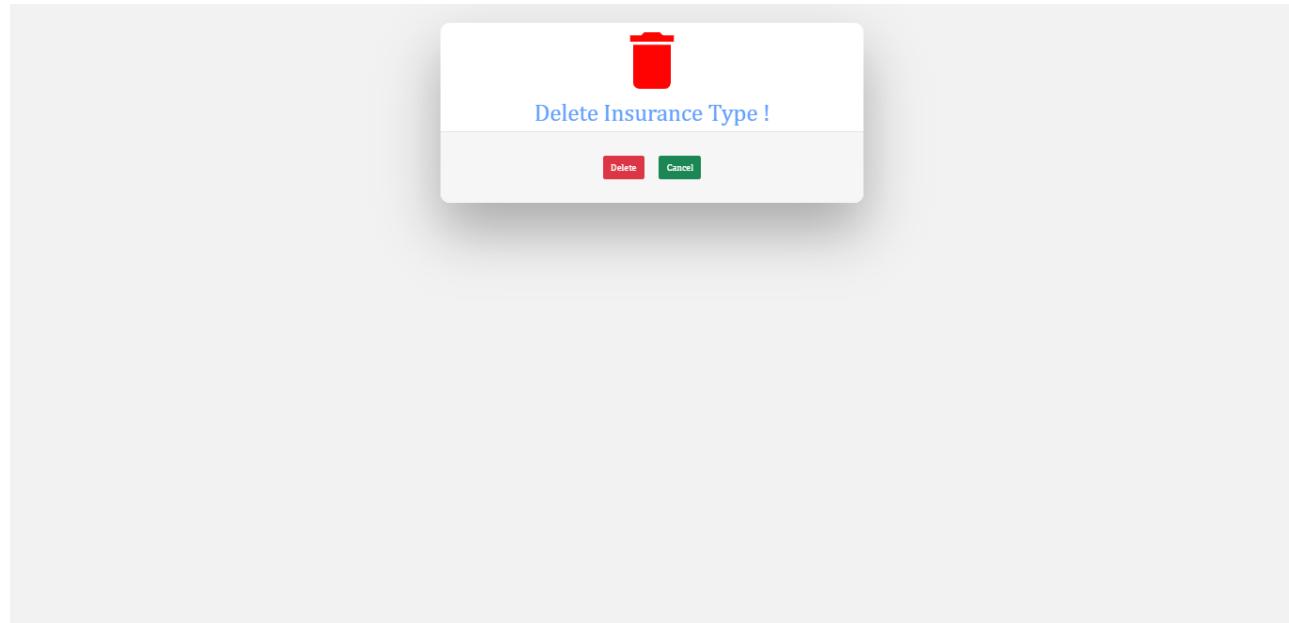


Figure 85 –Insurance type Delete Admin page

User ID	User name	Insurance ID	Reservation Date	Due Date	Price
613e0325834846ff7a67c831	Dulakshi Hansini	I712	2021-10-06	2021-10-04	55000
6158ab6d8c84ccdea424a0ae	Oshen gomes	I123	2021-10-21	2021-10-04	10000

Figure 86 –Insurance Resevarition list Admin page

Figure 87 – Event Category view page



TRAVEL AGENCY

SIGN IN SIGN UP

HOME STAY TICKETS GROUND RESERVATIONS PACKAGES DOCUMENTS INSURANCE EVENTS

ADD EVENT

Event type

Location

Date
 mm/dd/yyyy

Price

Submit

Figure 88 – Event Add page



TRAVEL AGENCY

SIGN IN SIGN UP

team	matara	add	200	update	Delete
wwes	colombo	2021-02-02	123	update	Delete
ent	matara	2021-10-05	130	update	Delete
ent1	matara	2021-10-05	23	update	Delete
tre	galle	2021-10-29	245	update	Delete
ecw	colombo	2021-10-07	211	update	Delete
ghh	galle	2021-10-20	345	update	Delete
dddr	matara	2021-10-19	54	update	Delete

Generate monthly report

Figure 89 – Event Add list view page

2.3 Implementation

2.3.1 Software

- Mongo DB
- Express
- React js
- Node js
- Postman

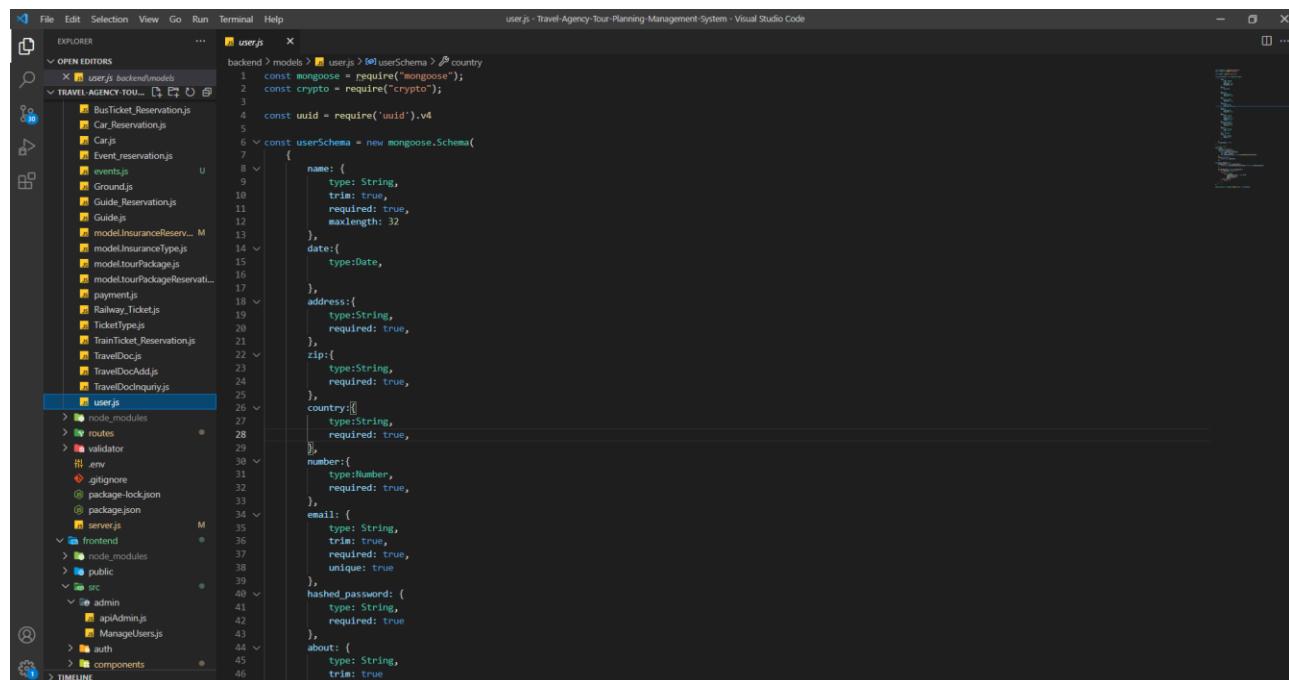
2.3.2 Technology

- Java Script
- CSS
- HTML
- Bootstrap

2.3.3 Codes

2.3.3.1 User Management

Models



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure. The `backend` folder contains several files like `BusTicket_Reservation.js`, `Car_Reservation.js`, `Car.js`, `Event_reservation.js`, `events.js`, `Ground.js`, `Guide_Reservations.js`, `Guide.js`, `modelInsuranceReserv...`, `modelInsuranceType.js`, `modelTourPackage.js`, `modelTourPackageReservati...`, `payment.js`, `Railway_Ticket.js`, `TicketType.js`, `TrainTicket_Reservation.js`, `TravelDoc.js`, `TravelDocAdd.js`, and `TravelDocInquiry.js`. The `user.js` file is selected and open in the editor.
- Editor:** The code for `user.js` is displayed. It imports `mongoose` and `crypto`, and uses `uuid` to generate unique IDs. It defines a schema for a `User` document with fields: `name` (String, trim, required), `date` (Date), `address` (String, required), `zip` (String, required), `country` (Array of String, required), `number` (Number, required), and `email` (String, trim, required, unique). It also includes a hashed password field and an `about` field.

```
user.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

backend > models > user.js > userSchema > country
1 const mongoose = require("mongoose");
2 const crypto = require("crypto");
3
4 const uuid = require('uuid').v4
5
6 const userSchema = new mongoose.Schema(
7   {
8     name: {
9       type: String,
10      trim: true,
11      required: true,
12      maxlength: 32
13    },
14     date: {
15       type: Date,
16     },
17     address: {
18       type: String,
19       required: true,
20     },
21     zip: {
22       type: String,
23       required: true,
24     },
25     country: [
26       type: String,
27       required: true,
28     ],
29     number: {
30       type: Number,
31       required: true,
32     },
33     email: {
34       type: String,
35       trim: true,
36       required: true,
37       unique: true
38     },
39     hashed_password: {
40       type: String,
41       required: true
42     },
43     about: {
44       type: String,
45       trim: true
46     }
47   }
48 )
49
50 module.exports = mongoose.model('User', userSchema)
```

```

    router.put('/post/update/:id', (req, res) => {
      Posts.findByIdAndUpdate(req.params.id,
        {
          $set:req.body
        },
        (err,post)=>{
          if(err){
            return res.status(400).json({error:err});
          }
          return res.status(200).json({
            success:"updated SuccessFully"
          });
        }
      );
    });

    router.delete('/post/delete/:id', (req, res) =>{
      Posts.findByIdAndDelete(req.params.id,exec((err,deletedPost) =>{
        if(err) return res.status(400).json({
          message:"Delete unsuccessful",err
        })
        return res.json({
          message:"Deleted Sucessful",deletedPost
        });
      }));
    });

    //get specific post
    router.get('/post/:id', (req, res) =>{
      let postId = req.params.id;
      Posts.findById(postId,(err,post)>{
        if(err){
          return res.status(400).json({success:false,err});
        }
        return res.status(200).json({
          success:true,
          post
        });
      });
    });
  }
}

```

Figure 90

Routes

```

    const express = require('express');
    const Posts = require('../models/user');

    const router = express.Router();

    //save post
    router.post('/post/save',(req,res)=>{
      let newPost = new Posts(req.body);
      newPost.save((err)=>{
        if(err){
          return res.status(400).json({
            error:err
          });
        }
        return res.status(200).json({
          success:"Post saved successfully"
        });
      });
    });

    //get posts
    router.get('/admin/posts',(req,res)=>{
      Posts.find().exec((err,posts)=>{
        if(err){
          return res.status(400).json({
            error:err
          });
        }
        return res.status(200).json({
          success:true,
          existingPosts:posts
        });
      });
    });

    //update posts
    router.put('/post/update/:id',(req,res)=>{
      Posts.findByIdAndUpdate(req.params.id,
        {
          $set:req.body
        },
        (err,post)=>{
          if(err){
            return res.status(400).json({error:err});
          }
          return res.status(200).json({
            success:"updated SuccessFully"
          });
        }
      );
    });
  }
}

```

```

File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
frontend > src > admin > ManageUsers.js
Profile.js ManageUsers.js M
frontend > src > admin > ManageUsers.js > ManageUsers
44 if (data.error) {
45   console.log(data.error);
46 } else {
47   loadUsers();
48 }
49 );
50 };
51 );
52 };
53 onDelete = (userId) =>{
54   axios.delete(`post/delete/${user._id}`).then((res)>{
55     alert("Delete Successfully");
56     this.loadUsers();
57   })
58 }
59 useEffect(() => {
60   loadUsers();
61 }, []);
62 );
63 return(
64   <div className="display-table">
65     <div className="row display-table-row">
66       <div className="col-md-2 col-sm-1 hidden-xs display-table-cell v-align box" id="navigation">
67         <div className="logo">
68           <a href="home.html"><img src={ImgLogo} alt="merkery_logo" className="hidden-xs hidden-sm" />
69         </div>
70       </div>
71     </div>
72   </div>
73 <h3 style={{ color: "white" }}>Admin Panel</h3>
74 <div className="navi">
75   <ul>
76     <li><a href="#"><HouseIcon style={{ fontSize: 30, color: "#5584FF" }} /><span className="hidden-xs hidden-sm">&ampnbsp&ampnbspAccommodation</span></a></li>
77     <li><a href="#"><ConfirmationNumberIcon style={{ fontSize: 30, color: "#5584FF" }} /><span className="hidden-xs hidden-sm">&ampnbsp&ampnbspTicket</span></a></li>
78     <li><a href="#"><LocalTaxiIcon style={{ fontSize: 30, color: "#5584FF" }} /><span className="hidden-xs hidden-sm">&ampnbsp&ampnbspGround</span></a></li>
79     <li><a href="#"><AdminIcon>ExploreIcon</a><span className="hidden-xs hidden-sm">&ampnbsp&ampnbspTour Package</span></a></li>
80     <li><a href="#"><DescriptionIcon style={{ fontSize: 30, color: "#5584FF" }} /><span className="hidden-xs hidden-sm">&ampnbsp&ampnbspTravel Doc</span></a></li>
81     <li><a href="#"><SecurityIcon style={{ fontSize: 30, color: "#5584FF" }} /><span className="hidden-xs hidden-sm">&ampnbsp&ampnbspInsurance</span></a></li>
82     <li><a href="#"><EventIcon style={{ fontSize: 30, color: "#5584FF" }} /><span className="hidden-xs hidden-sm">&ampnbsp&ampnbspEvent</span></a></li>
83     <li><a href="#"><AdminUserManagement>GroupIcon</a><span className="hidden-xs hidden-sm">&ampnbsp&ampnbspUser Management</span></a></li>
84   </ul>
85 </div>
86 <App.js>
87 <config.js>
88 <index.js>
89

```

Figure 91

Frontend

```

File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
Signinjs Profile.js ManageUsers.js
frontend > src > admin > ManageUsers.js > ManageUsers
1 import React, { useState, useEffect } from 'react';
2 import Layout from '../components/Layout';
3 import { getUsers, deleteUser } from './apiAdmin';
4 import 'bootstrap/dist/css/bootstrap.min.css';
5 import './style/admin.css';
6 import ImgLogo from '../images/logo.png';
7 import HouseIcon from '@material-ui/icons/House';
8 import ConfirmationNumberIcon from '@material-ui/icons/ConfirmationNumber';
9 import LocalTaxiIcon from '@material-ui/icons/localTaxi';
10 import ExploreIcon from '@material-ui/icons/explore';
11 import DescriptionIcon from '@material-ui/icons/Description';
12 import SecurityIcon from '@material-ui/icons/Security';
13 import EventIcon from '@material-ui/icons/Event';
14 import GroupIcon from '@material-ui/icons/Group';
15 import GeneratePdf from '../components/UserReport';
16
17
18 const ManageUsers = () => {
19
20
21   const [users, setUsers] = useState([]);
22
23   //const { user, token } = isAuthenticated();
24
25   //search users
26
27   const [searchTerm, setSearchTerm] = useState("");
28
29   const loadUsers = () => {
30     getUsers().then(data => {
31       if (data.error) {
32         console.log(data.error);
33
34       } else {
35         setUsers(data);
36       }
37     });
38   };
39
40   const destroy = userId => {
41     deleteUser(userId).then(data => {
42       if (data.error) {
43         console.log(data.error);
44
45       }
46     });
47   };
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89

```

```
File Edit Selection View Go Run Terminal Help
ManagementSystem - Travel Agency Tour Planning Management System - Visual Studio Code

EXPLORER
OPEN EDITORS
frontend/src/admin> ManageUsers.js M
frontend/src/admin> ManageUsers.js > ManageUsers
  134
  135
  136
  137
  138
  139
  140
  141
  142
  143
  144
  145
  146
  147
  148
  149
  150
  151
  152
  153
  154
  155
  156
  157
  158
  159
  160
  161
  162
  163
  164
  165
  166
  167
  168
  169
  170
  171
  172
  173
  174
  175
  176
  177
  178
  179

  users.filter(p=>{
    if(searchTerm===''){
      return p;
    }
    else if(
      p.zip.toLowerCase().includes(searchTerm.toLowerCase()) ||
      p.email.toLowerCase().includes(searchTerm.toLowerCase()) ||
      p.name.toLowerCase().includes(searchTerm.toLowerCase())
    ){
      return p;
    }
  })

  .map((p, i) => (
    <tr key={i}>
      <td>p.name</td>
      <td>p.date</td>
      <td>p.zip</td>
      <td>p.number</td>
      <td>p.email</td>
      <td>p.country</td>
      <td><a onClick={() => destroy(p._id)} className="btn btn-danger mt-2" style={{backgroundColor: "#ff4d4d" }}>
        <i className="far fa-trash-alt" /></a></td>
    </tr>
  )),>
  </tbody>
</table>

<button className="btn btn-primary col-lg-2" onClick={()=>GeneratePdf(users)}>
  Generate Report PDF
</button>
```

File Edit Selection View Go Run Terminal Help

ManageUsers.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

EXPLORER

- OPEN EDITORS
- Signinjs
- Profile.js
- ManageUsers.js M

frontend > src > admin > ManageUsers

```
<a href="#" style={{ fontSize: 30, color: "#5584FF" }}><span className="hidden-xs hidden-sm">&nbsp;&nbsp;</span></a></li>
<li><a href="/AdminUserManagement"><GroupIcon style={{ fontSize: 30, color: "#5584FF" }}></span><span className="hidden-xs hidden-sm">&nbsp;&nbsp;</span></a></li>
```

frontend > src > admin > ManageUsers

```
</ul>
</div>
<div style={{marginLeft:"15px"}}>
<Layout title="Manage Users" description="User Profile List" className="container-fluid">
```

backend

- controllers
- helpers
- models
- node_modules
- routes

validator

- .env
- .gitignore
- package-lock.json
- package.json
- server.js M

frontend

- node_modules
- public
- src
 - admin
 - apidashboard.js
 - ManageUsers.js M
 - auth
 - components
 - css
 - images
 - img
 - style

user

- AdminDashboard.js
- apisusers.js
- Profile.js
- Signinjs
- Signup.js
- UserDashboard.js
- Apps
- config.js
- index.js

TIMELINE

frontend > src > admin > ManageUsers

```
<div className="container">
<div className="row">
<div className="col-lg-9 mt-2 mb-2">
<h2>All Users</h2>
</div>
<div className="col-lg-3 mt-2 mb-2">
<input className="form-control" style={{fontSize:"16px"}} type="search" placeholder="Search" name="searchTerm" onChange={(e)>(setSearchTerm(e.target.value))}>
```

frontend > src > admin > ManageUsers

```
<table className="table table-striped table-dark">
<thead >
```

Full Name	Date of Birth	Zip/Postal Code	Phone	Email	Country	ACTION
-----------	---------------	-----------------	-------	-------	---------	--------

```
</thead>
<tbody>
```

frontend > src > admin > ManageUsers

```
</tbody>
```

Figure 92

2.3.3.2 Accommodation reservation Management

Models

```
1 const mongoose = require('mongoose');
2
3 const Schema = mongoose.Schema;
4
5 const accommodationSchema = new Schema({
6     accommodation_ID : {
7         type : Number,
8         required: true
9     },
10    name : {
11        type : String,
12        required: true
13    },
14
15    location : {
16        type : String,
17        required: true
18
19    },
20    type : {
21        type : String,
22        required: true
23    },
24    price : {
25        type : Number,
26        required: true
27    },
28    owner : {
29        type : String,
30        required: true
31    },
32    phone : {
33        type : Number,
34        required: true
35    },
36    link : {
37        type : String,
```

```

    },
    type : {
      type : String,
      required: true
    },
    price : {
      type : Number,
      required: true
    },
    owner : {
      type : String,
      required: true
    },
    phone : {
      type : Number,
      required: true
    },
    link : {
      type : String,
      required: true
    }
  })
const Accommodation = mongoose.model("Accommodation",accommodationSchema);
module.exports=Accommodation;

```

Figure 93

Routes

```

1 const router= require("express").Router();
2 let Accommodation =require("../models/accommodation");
3
4 router.route("/add").post((req,res)=>[
5   const accommodation_ID= Number(req.body.accommodation_ID);
6   const name= req.body.name;
7   const location= req.body.location;
8   const type= req.body.type;
9   const phone= Number(req.body.phone);
10  const price= Number(req.body.price);
11  const owner = req.body.owner;
12  const link = req.body.link;
13
14  const newAccommodation = new Accommodation({
15    accommodation_ID,
16    name,
17    location,
18    type,
19    price,
20    owner,
21    phone,
22    link
23  })
24  newAccommodation.save().then(()=>{
25    res.json("Accommodation Added")
26  }).catch((err)=>{
27    console.log(err);
28  })
29 ])
30
31 router.route("/").get((req,res)=>{
32   Accommodation.find().then((accommodations)=>{
33     res.json(accommodations)
34   }).catch((err)=>{
35     console.log(err)
36   })
37 })

```

```

38   router.route('/update/:id').put((req, res, next) => {
39     Accommodation.findByIdAndUpdate(req.params.id, {
40       $set: req.body
41     }, (error, data) => {
42       if (error) {
43         return next(error);
44         console.log(error)
45       } else {
46         res.json(data)
47         console.log('Accommodation updated successfully !')
48       }
49     })
50   })
51
52 router.route("/delete/:id").delete(async(req, res)=>{
53   let accId=req.params.id;
54
55   await Accommodation.findByIdAndDelete(accId)
56   .then(()=>{
57     res.status(200).send({status:" accommodation deleted"});
58   }).catch((err)=>{
59     console.log(err.message);
60     res.status(500).send({status:"Error with delete",error:err.message});
61   })
62 })
63
64
65 router.route('/get/:id').get((req, res) => {
66   Accommodation.findById(req.params.id, (error, data) => {
67     if (error) {
68       return next(error)
69     } else {
70       res.json(data)
71     }
72   })
73 })
74 module.exports=router;

```

Figure 94

Frontend

```

1 import React,{useState} from "react"
2 import axios from "axios";
3
4
5 export default function AddAccommodation(){
6
7   const[accommodation_ID, setAccommodation_ID]= useState("");
8   const[name, setname]= useState("");
9   const[location, setlocation]= useState("");
10  const[type, setType]= useState("");
11  const[price, setprice]= useState("");
12  const[owner, setowner]= useState("");
13  const[phone, setphone]= useState("");
14  const[link, setlink]= useState("");
15
16  function sendData(e){
17    e.preventDefault();
18    const newAccommodation={
19      accommodation_ID,
20      name,
21      location,
22      type,
23      price,
24      owner,
25      phone,
26      link
27    }
28    //console.log(newAccommodation);
29    axios.post("http://localhost:8000/accommodation/add",newAccommodation).then(()=>{
30      alert("Accommodation added")
31    }).catch((err)=>{
32      alert(err)
33    })
34  }
35
36  return(
37

```

```

39     <div>
40       <nav className="navbar navbar-expand-lg navbar-light bg-light">
41         <a className="navbar-brand" href="/admin_panel/home" style={{color: "red "}}>Accommodation Management</a>
42         <button className="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
43           <span className="navbar-toggler-icon"></span>
44         </button>
45       </div>
46       <div className="collapse navbar-collapse" id="navbarSupportedContent">
47         <ul className="navbar-nav mr-auto">
48           <li className="nav-item active">
49             <a className="nav-link" href="/admin_panel/accommodation_admin">All Accommodations <span className="sr-only" style={{color: "red "}}>(current)</span></a>
50           </li>
51           <li className="nav-item">
52             <a className="nav-link" href="/admin_panel/accommodation_admin/addnew">Add new Accommodation</a>
53           </li>
54         </ul>
55       </div>
56     </div>
57   </nav>
58
59   <header className="bg-dark py-5">
60     <div className="container px-4 px-lg-5 my-5">
61       <div className="text-center text-white">
62         <h1 className="display-4 fw-bolder">Add new Accommodation here</h1>
63         <p className="lead fw-normal text-white-50 mb-0"></p>
64       </div>
65     </div>
66   </header>
67   <br/><br/>
68   <div className="container">
69     <form className="row g-3 needs-validation" novalidate onSubmit={sendData}>
70       <div className="col-md-4">
71
72         <br/><br/>
73         <div className="form-group">
74           <form className="row g-3 needs-validation" novalidate onSubmit={sendData}>
75             <div className="col-md-4">
76               <label for="accommodation_ID" className="form-label">remark_ID</label>
77               <input type="number" className="form-control" id="validationServer01" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
78               onChange={(e)=>{
79                 setaccommodation_ID(e.target.value)
80               }}/>
81
82             <div className="valid-feedback">
83               Looks good!
84             </div>
85           </div>
86           <div className="col-md-4">
87             <label for="name" className="form-label">Name</label>
88             <input type="text" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
89             onChange={(e)=>{
90               setname(e.target.value)
91             }}/>
92
93             <div className="valid-feedback">
94               Looks good!
95             </div>
96           </div>
97         </div>
98       </div>
99     </form>
100    <div className="col-md-4">
101      <label for="type_ID" className="form-label">Type</label>
102      <input type="text" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
103      onChange={(e)=>{
104        settype(e.target.value)
105      }}/>
106
107    </div>
108  </div>

```

```

72         <br/><br/>
73         <div className="form-group">
74           <form className="row g-3 needs-validation" novalidate onSubmit={sendData}>
75             <div className="col-md-4">
76               <label for="accommodation_ID" className="form-label">remark_ID</label>
77               <input type="number" className="form-control" id="validationServer01" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
78               onChange={(e)=>{
79                 setaccommodation_ID(e.target.value)
80               }}/>
81
82             <div className="valid-feedback">
83               Looks good!
84             </div>
85           </div>
86           <div className="col-md-4">
87             <label for="name" className="form-label">Name</label>
88             <input type="text" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
89             onChange={(e)=>{
90               setname(e.target.value)
91             }}/>
92
93             <div className="valid-feedback">
94               Looks good!
95             </div>
96           </div>
97         </div>
98       </div>
99     </form>
100    <div className="col-md-4">
101      <label for="type_ID" className="form-label">Type</label>
102      <input type="text" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
103      onChange={(e)=>{
104        settype(e.target.value)
105      }}/>
106
107    </div>
108  </div>

```

```

101      <div className="col-md-4">
102        <label for="type_ID" className="form-label">Type</label>
103        <input type="text" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
104        onChange={(e)=>{
105          setType(e.target.value)
106        }}/>
107
108      <div className="valid-feedback">
109
110      </div>
111
112    </div>
113    <div className="col-md-4">
114      <label for="price" className="form-label">price</label>
115      <input type="number" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
116      onChange={(e)=>{
117        setprice(e.target.value)
118      }}/>
119
120      <div className="valid-feedback">
121
122      </div>
123
124    </div>
125
126  </div>
127  <div className="col-md-4">
128    <label for="owner" className="form-label">owner</label>
129    <input type="text" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
130    onChange={(e)=>{
131      setwowner(e.target.value)
132    }}/>
133    <div id="validationServerUsernameFeedback" className="invalid-feedback">
134      Enter owner's name
135    </div>
136  </div>
137
```

```

134      <div id="validationServer03Feedback" className="invalid-feedback">
135        Please provide a valid city.
136      </div>
137
138  </div>
139
140  <div className="col-md-3">
141    <label for="phone" className="form-label">Phone pattern:(xxxxxxxxx)</label>
142    <input type="tel" className="form-control" id="validationServer05" pattern="^\d{3}\.\d{7}$" aria-describedby="validationServer05Feedback" required
143    onChange={(e)=>{
144      setphone(e.target.value)
145    }}/>
146    <div id="validationServer05Feedback" className="invalid-feedback">
147      Please provide a valid phone number.
148    </div>
149
150  </div>
151
152  <div className="col-md-4">
153    <label for="owner" className="form-label">image link</label>
154    <input type="text" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend3 validationServerUsernameFeedback" required
155    onChange={(e)=>{
156      setlink(e.target.value)
157    }}/>
158
```

```

159         <div id="validationServer05Feedback" className="invalid-feedback">
160             Please provide a valid phone number.
161         </div>
162     </div>
163     </div>
164     <div className="col-md-4">
165         <label for="owner" className="form-label">image link</label>
166         <input type="text" className="form-control" id="validationServer02" aria-describedby="inputGroupPrepend validationServerUsernameFeedback" required
167         onChange={(e)}>{
168             | setlink(e.target.value)
169         }>
170         <div id="validationServerUsernameFeedback" className="invalid-feedback">
171             | Enter img link name
172         </div>
173     </div>
174     <br/>
175     <div className="col-12">
176         <button className="btn btn-primary" type="submit">submit form</button>
177     </div>
178     </form>
179   </div>
180 </div>
181 </div>
182 )
183 }

```

```

1 import React, { Component } from "react";
2 // This will require to npm install axios
3 import axios from 'axios';
4 import { Link } from "react-router-dom";
5
6 const Record = (props) => (
7   <tr>
8     <td>{props.record.accommodation_ID}</td>
9     <td>{props.record.name}</td>
10    <td>{props.record.location}</td>
11    <td>{props.record.type}</td>
12    <td>{props.record.price}</td>
13    <td>{props.record.owner}</td>
14    <td>{props.record.phone}</td>
15    <td>
16      <Link to={"/update/" + props.record._id}>Edit</Link> |
17      <a
18        href="/admin_panel/accommodation_admin"
19        onClick={() => {
20          props.deleteRecord(props.record._id);
21          this.props.history.push("/admin_panel/accommodation_admin");
22        }}
23      >
24        Delete
25      </a>
26    </td>
27  </tr>
28);
29
30
31 export default class RecordList extends Component {
32   // This is the constructor that shall store our data retrieved from the database
33   constructor(props) {
34     super(props);
35     this.deleteRecord = this.deleteRecord.bind(this);
36     this.state = { records: [] };
37   }

```

```

39 // This method will get the data from the database.
40 componentDidMount() {
41   axios
42     .get("http://localhost:8000/accommodation/")
43     .then((response) => {
44       this.setState({ records: response.data });
45     })
46     .catch(function (error) {
47       console.log(error);
48     });
49 }
50
51 // This method will delete a record based on the method
52 deleteRecord(id) {
53   axios.delete("http://localhost:8000/accommodation/delete/" + id).then((response) => {
54     console.log(response.data);
55   });
56
57   this.setState({
58     record: this.state.records.filter((el) => el._id !== id),
59   });
60 }
61
62 // This method will map out the users on the table
63 recordList() {
64   return this.state.records.map((currentrecord) => {
65     return (
66       <Record
67         record={currentrecord}
68         deleteRecord={this.deleteRecord}
69         key={currentrecord._id}
70       />
71     );
72   });
73 }
74
75 filterData(records,searchKey){
```

```

75   filterData(records,searchKey){
76     const result=records.filter((accommodations)=>
77       accommodations.name.toLowerCase().includes(searchKey)|||
78       accommodations.location.toLowerCase().includes(searchKey)|||
79       accommodations.type.toLowerCase().includes(searchKey)
80
81     )
82     this.setState({records:result})
83   }
84
85   handleSerchArea=(e) =>{
86
87     const searchKey=e.currentTarget.value;
88     console.log(searchKey);
89     axios
90       .get("http://localhost:8000/accommodation/")
91       .then((response) => {
92         this.filterData(response.data,searchKey);
93         console.log("successfull");
94       });
95
96
97   }
98
99   // This following section will display the table with the records of individuals.
100  render() {
101    return [
102      <div>
103        <div><nav className="navbar navbar-expand-lg navbar-light bg-light">
104          <div className= "container-fluid">
105            <a className="navbar-brand" href="/admin_panel/home" style={{color: "red "}}>Accommodation Management</a>
106            <button className="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
107              <span className="navbar-toggler-icon"></span>
108            </button>
109          <div className="collapse navbar-collapse" id="navbarSupportedContent">
```

```

110 <div className="collapse navbar-collapse" id="navbarSupportedContent">
111   <ul className="navbar-nav mr-auto">
112     <li className="nav-item active">
113       <a className="nav-link" href="/admin_panel/accommmodation_admin">All Accommodations <span className="sr-only">(current)</span></a>
114     </li>
115     <li className="nav-item">
116       <a className="nav-link" href="/admin_panel/accommmodation_admin/addnew">Add new Accommodation</a>
117     </li>
118     <li className="nav-item">
119       <a className="nav-link" href="/admin_panel/accommmodation_admin/accommmodation_servations">All Accommodation reservations</a>
120     </li>
121   </ul>
122 </div>
123   </div>
124
125 </nav></div>
126 <nav class="navbar navbar-light bg-light">
127   <form class="form-inline">
128     <input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search" onChange={this.handleSearchArea} />
129   </form>
130 </nav>
131
132
133   <br/><br/><br/><br/>
134   <div className="container">
135
136     <div><h3> Accommodation Record List</h3></div>
137     <table className="table table-striped" style={{ marginTop: 20 }}>
138       <thead>
139         <tr>
140           <th>Reamrk Id</th>
141           <th>name</th>
142           <th>location</th>
143           <th>type</th>
144           <th>price</th>
145         </tr>
146       </thead>
147       <tbody>{this.recordList()}</tbody>
148     </table>
149   </div>
150
151 </div>
152
153 </div>
154
155 </div>
156
157 </div>
158
159

```

```

126   </nav></div>
127   <nav class="navbar navbar-light bg-light">
128     <form class="form-inline">
129       <input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search" onChange={this.handleSearchArea} />
130     </form>
131   </nav>
132
133   <br/><br/><br/><br/>
134   <div className="container">
135
136     <div><h3> Accommodation Record List</h3></div>
137     <table className="table table-striped" style={{ marginTop: 20 }}>
138       <thead>
139         <tr>
140           <th>Reamrk Id</th>
141           <th>name</th>
142           <th>location</th>
143           <th>type</th>
144           <th>price</th>
145           <th>owner</th>
146           <th>phone</th>
147           <th>Action</th>
148         </tr>
149       </thead>
150       <tbody>{this.recordList()}</tbody>
151     </table>
152   </div>
153
154 </div>
155
156 </div>
157
158 </div>
159

```

Figure 95

2.3.3.3 Reservation of Ticket Management

Models

A screenshot of Visual Studio Code showing the file structure and code for `Airline_Ticket.js`. The code defines a Mongoose schema for an `Airline_Ticket` document. It includes fields for `Ticket_Type`, `From`, `To`, `Flight_Number`, `Service_Provider`, `Departure_Time`, `Arrival_Time`, `Date`, and `No_of_Seats`. It also includes a `Ticket_Fee` calculation and a `save` method. The code is part of the `TRAVEL-AGENCY-TOUR-PLANNING-MANAGEMENT-SYSTEM` project.

```

Airline.js U Airline_Ticket.js
backend> models > Airline_Ticket.js > Mairline
1 const mongoose = require("mongoose");
2
3 const airline=new mongoose.Schema({
4
5   Ticket_Type:{
6     type:String,
7     required:true
8   },
9   From:{
10     type:String,
11     required:true
12   },
13   To:{
14     type:String,
15     required:true
16   },
17   Flight_Number:{
18     type:String,
19     required:true
20   },
21   Service_Provider:{
22     type:String,
23     required:true
24   },
25   Departure_Time:{
26     type:String,
27     required:true
28   },
29   Arrival_Time:{
30     type:String,
31     required:true
32   },
33   Date:{
34     type:String,
35     required:true
36   },
37   No_of_Seats:{
38     type:String,
39     required:true
40   },
41   Ticket_Fee:{
42     type:String,
43     required:true
44   }
45 },
46 );
47
48 const ticket=mongoose.model("Airline_Ticket",airline);
49 module.exports=ticket;

```

Figure 96

Routes

A screenshot of Visual Studio Code showing the file structure and code for `airline_tickets.js`. The code defines routes for inserting and viewing airline tickets. It uses an `express` router and the `airlineticket` model. The `POST /add` route handles ticket insertion with validation and saving to the database. The `GET /` route handles ticket retrieval.

```

airline_tickets.js U x
backend> routes > airline_tickets.js > ...
1 const router=require("express").Router();
2 const airlineticket=require("../models/Airline_Ticket");
3
4 //insert
5 router.route("/add").post((req,res)=>{
6
7   let Ticket_Type=req.body.Ticket_Type;
8   let From=req.body.From;
9   let To=req.body.To;
10  let Flight_Number=req.body.Flight_Number;
11  let Service_Provider=req.body.Service_Provider;
12  let Departure_Time=req.body.Departure_Time;
13  let Arrival_Time=req.body.Arrival_Time;
14  let Date=req.body.Date;
15  let No_of_Seats=req.body.No_of_Seats;
16  let Ticket_Fee=req.body.Ticket_Fee;
17
18  const airlineticketobj=new airlineticket({
19    Ticket_Type,Service_Provider,Flight_Number,Departure_Time,Arrival_Time,Date,From,To,Ticket_Fee,No_of_Seats
20 });
21
22
23  airlineticketobj.save().then(()=>{
24    res.json("Insert successfully");
25  }).catch((err)=>{
26    console.log(err);
27  });
28
29 });
30
31 //view
32 router.route("/").get((req,res)=>{
33
34   airlineticket.find().then((aticket)=>{
35     res.json(aticket);
36   }).catch((err)=>{
37     console.log(err);
38   });
39
40 });

```

```
airline_tickets.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

File Edit Selection View Go Run Terminal Help
airline_tickets.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

EXPLORER
OPEN EDITORS
backend > routes > airline_tickets.js ...
airlineticket_reservation.js U
Airline.js U
airline_tickets.js U

238
TRAVEL...
backend
controllers
helpers
models
node_modules
routes
accommo... U
accommo... U
airline_tic... U
airlinetick... U
auth.js U
bus_ticket... U
busticket... U
car_reserv... U
cars.js U
event_res... U
ground.js U
guide_res... U
guides.js U
payments.js U
railway_tic... U
routes.tou... U
routestou... U
tickettype... U
trainTicket... U
TravelDocjs U

74    })
75
76    airlineticket.findByIdAndUpdate(uid,airlineticketobj).then((udata)=>{
77      res.json(udata);
78    }).catch((err)>{
79      console.log(err);
80    });
81  });
82})
83//Find one
84router.route("/:id").get((req,res)=>{
85
86  let id = req.params.id;
87
88  airlineticket.findById(id).then((data)=>{
89    res.json(data);
90  }).catch((err)>{
91    console.log(err);
92  })
93})
94})
95
96module.exports=router;
```

Figure 97

Frontend

File Edit Selection View Go Run Terminal Help • Airline.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

frontend > src > components > **Airline.js** > **AirlinePage**

```
38
39     //view
40     const [AdminAirlineData, setAdminAirlineData] = useState([]);
41
42     useEffect(() => {
43         axios.get("http://localhost:8000/admin/airticket/").then((res) => {
44             setAdminAirlineData(res.data);
45         }).catch((err) => {
46             console.log(err);
47         })
48     }, [])
49
50
51 //search part
52 const [searchTerm, setSearchTerm] = useState("");
53
54
55 return (
56
57     <div className="container">
58
59         <div className="mt-5"></div>
60         <form onSubmit={sendData} className="text-dark tc-light p-3 pb-4 fw-bold" style={{ fontSize: "20px", backgroundColor: "lightblue" }}>
61             <div className="row">
62                 <div className="col-lg-1"></div>
63                 <div className="col-lg-2" >
64
65                     <label htmlFor="from">From:</label><br />
66                     <select className="form-select" aria-label="Default select example" style={{ fontSize: "20px" }} required onChange={(e) => {setFrom(e.target.value);}}>
67                         <option selected>Select</option>
68                         <option value="London">London</option>
69                         <option value="India">India</option>
70                         <option value="Srilanka">Sri Lanka</option>
71                     </select>
72
73                 </div>
74             </div>
75         </form>
76     </div>
77 
```

A screenshot of Visual Studio Code showing the `Airline.js` file. The code defines a component with two dropdown menus. The first dropdown menu has options for London, India, Sri Lanka, Australia, Bangalore, Lahore, Melbourne, America, Bahrain, Canada, France, Iceland, and Japan. The second dropdown menu has options for London, India, Sri Lanka, Australia, Bangalore, Lahore, Melbourne, America, and Bahrain. The code uses the `useState` hook to manage the selected values.

```
70      <option selected>Select</option>
71      <option value="London">London</option>
72      <option value="India">India</option>
73      <option value="SriLanka">Sri Lanka</option>
74      <option value="Australia">Australia</option>
75      <option value="Bangalore">Bangalore</option>
76      <option value="Lahore">Lahore</option>
77      <option value="Melbourne">Melbourne</option>
78      <option value="America">America</option>
79      <option value="Bahrain">Bahrain</option>
80      <option value="Canada">Canada</option>
81      <option value="France">France</option>
82      <option value="Iceland">Iceland</option>
83      <option value="Japan">Japan</option>
84    </select>
85  </div>
86
87
88  <div className="col-lg-2">
89    <label for="to">To:</label><br />
90
91    <select class="form-select" aria-label="Default select example" style={{ fontSize: "20px" }} required onChange={(e) => {
92      setTo(e.target.value);
93    }}>
94      <option selected>Select</option>
95      <option value="London">London</option>
96      <option value="India">India</option>
97      <option value="SriLanka">Sri Lanka</option>
98      <option value="Australia">Australia</option>
99      <option value="Bangalore">Bangalore</option>
100     <option value="Lahore">Lahore</option>
101     <option value="Melbourne">Melbourne</option>
102     <option value="America">America</option>
103     <option value="Bahrain">Bahrain</option>
104   </select>
105 </div>
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
```

A screenshot of Visual Studio Code showing the `Airline.js` file. The code defines a component with three dropdown menus. The first dropdown menu has options for Japan, London, India, Sri Lanka, Australia, Bangalore, Lahore, Melbourne, America, and Bahrain. The second dropdown menu has options for Japan, London, India, Sri Lanka, Australia, Bangalore, Lahore, Melbourne, America, and Bahrain. The third dropdown menu has options for No of Tickets, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, and 12. The code uses the `useState` hook to manage the selected values.

```
110      <option value="Japan">Japan</option>
111    </select>
112  </div>
113  <div className="col-lg-2">
114    <label for="date">Date:</label><br />
115    <input type="Date" name='date' id='date' className='form-control' style={{ fontSize: "20px" }} required onChange={(e) => {
116      setDate(e.target.value);
117    }}>
118  </div>
119
120
121
122
123
124
125
126
127
128
129
130
131      <option selected>No of Tickets:</option>
132      <option value="1">1</option>
133      <option value="2">2</option>
134      <option value="3">3</option>
135      <option value="4">4</option>
136      <option value="5">5</option>
137      <option value="6">6</option>
138      <option value="7">7</option>
139      <option value="8">8</option>
140      <option value="9">9</option>
141      <option value="10">10</option>
142      <option value="11">11</option>
143      <option value="12">12</option>
144    </select>
145  </div>
146  <div className="col-lg-2 mt-5">
147    <button type="submit" class="btn btn-success form-control fw-bold" style={{ fontSize: "20px" }}>Book now</button>
148  </div>
```

A screenshot of Visual Studio Code showing the code for `Airline.js`. The code is a React component named `AirlinePage` located in the `frontend/src/components` directory. The component contains several form fields: a submit button, two dropdown menus for selecting a class and service provider, and a dropdown menu for seat number. The code uses functional state management with `useState` and `useEffect` hooks.

```
frontend > src > components > Airline.js > AirlinePage
  146   <div className="col-lg-2 mt-5">
  147     <button type="submit" class="btn btn-success form-control fw-bold" style={{ fontSize: "20px" }}>Book now</button>
  148   </div>
  149   </div>
  150   </div>
  151   <div className="row">
  152     <div className="col-lg-1"></div>
  153     <div className="col-lg-2 mt-4">
  154       <label for="class"></label><br />
  155       <select class="form-select" aria-label="Default select example" style={{ fontSize: "20px" }} required onChange={(e) => {
  156         setClass(e.target.value);
  157       }}>
  158         <option selected>Select</option>
  159         <option value="classA">Class A</option>
  160         <option value="classB">Class B</option>
  161         <option value="classC">Class C</option>
  162       </select>
  163     </div>
  164     <div className="col-lg-2 mt-4">
  165       <label for="class">Service Provider:</label><br />
  166       <select class="form-select" aria-label="Default select example" style={{ fontSize: "20px" }} required onChange={(e) => {
  167         setService_Provider(e.target.value);
  168       }}>
  169         <option selected>Select</option>
  170         <option value="SriLankanAirline">SriLankan Airline</option>
  171         <option value="AirLanka">Air Lanka</option>
  172       </select>
  173     </div>
  174     <div className="col-lg-2 mt-4">
  175       <label for="class">Seat Number:</label><br />
  176       <select class="form-select" aria-label="Default select example" style={{ fontSize: "20px" }} required onChange={(e) => {
  177         setSeatNumber(e.target.value);
  178       }}>
  179         <option selected>seat Number</option>
  180         <option value="s1">seat 1</option>
  181         <option value="s2">seat 2</option>
  182         <option value="s3">seat 3</option>
  183       </select>
  184     </div>
  185   </div>
  186   <div className="col-lg-2 mt-4">
  187     <a href="/ticketPayment"> <button type="button" class="btn btn-warning form-control mt-2 fw-bold" style={{ fontSize: "20px" }}>Payment</button></a>
  188     <a href="/ticketManagement"> <button type="button" class="btn btn-danger form-control mt-4 bg-danger fw-bold" style={{ fontSize: "20px" }}>Cancel</button>
  189   </div>
  190 </div>
  191 </div>
  192 </div>
  193 </div>
  194 </div>
  195 </div>
  196 </div>
  197 </div>
  198 </div>
  199 </div>
  200 </div>
  201 </div>
  202 </div>
  203 </div>
  204 </div>
  205 </div>
  206 </div>
  207 </div>
  208 </div>
  209 
```

A screenshot of Visual Studio Code showing the same `Airline.js` file after some changes have been made. The `setService_Provider` function has been moved to the `useEffect` hook instead of being placed directly under the `setClass` call. The `setSeatNumber` function has also been moved to the `useEffect` hook. The rest of the code remains the same as in the previous screenshot.

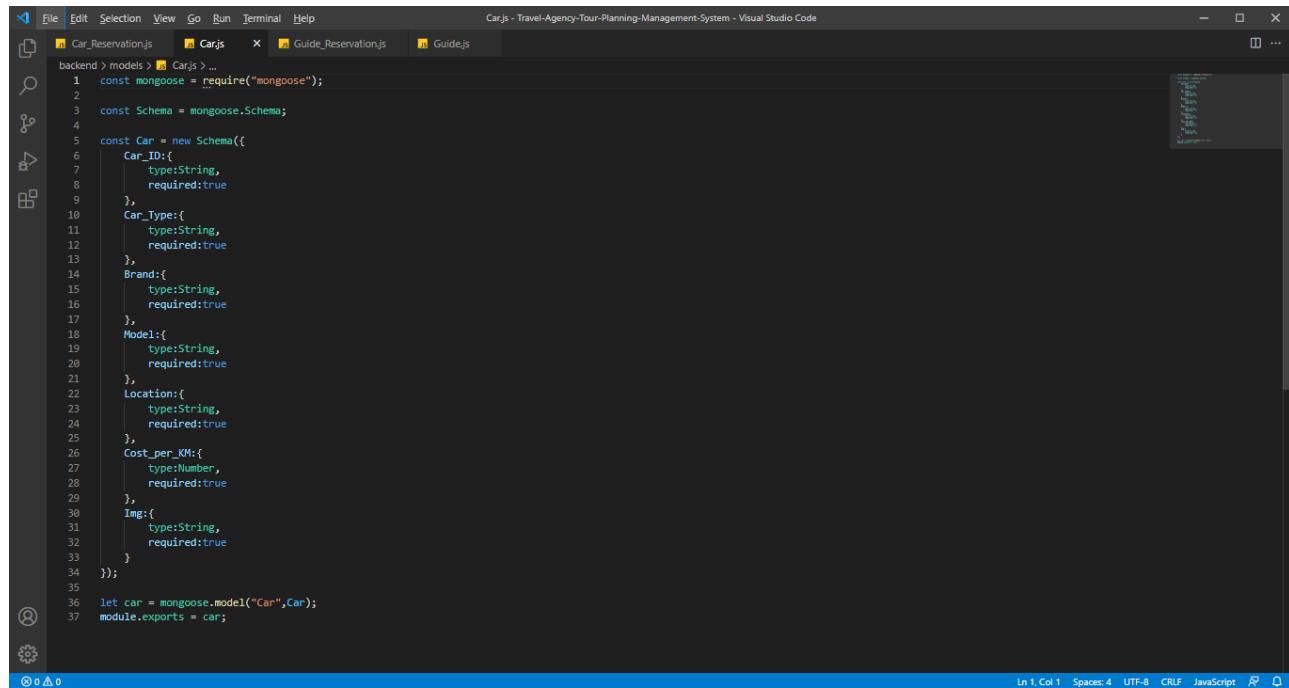
```
frontend > src > components > Airline.js > AirlinePage
  173   setService_Provider(e.target.value);
  174 }
  175 }>
  176   <option selected>Select</option>
  177   <option value="SriLankanAirline">SriLankan Airline</option>
  178   <option value="AirLanka">Air Lanka</option>
  179 </select>
  180 </div>
  181 <div className="col-lg-2 mt-4">
  182   <label for="class">Seat Number:</label><br />
  183   <select class="form-select" aria-label="Default select example" style={{ fontSize: "20px" }} required onChange={(e) => {
  184     setSeatNumber(e.target.value);
  185   }}>
  186     <option selected>seat Number</option>
  187     <option value="s1">seat 1</option>
  188     <option value="s2">seat 2</option>
  189     <option value="s3">seat 3</option>
  190   </select>
  191 </div>
  192 <div className="col-lg-2 mt-4">
  193   <a href="/ticketPayment"> <button type="button" class="btn btn-warning form-control mt-2 fw-bold" style={{ fontSize: "20px" }}>Payment</button></a>
  194   <a href="/ticketManagement"> <button type="button" class="btn btn-danger form-control mt-4 bg-danger fw-bold" style={{ fontSize: "20px" }}>Cancel</button>
  195 </div>
  196 </div>
  197 </div>
  198 </div>
  199 </div>
  200 </div>
  201 </div>
  202 </div>
  203 </div>
  204 </div>
  205 </div>
  206 </div>
  207 </div>
  208 </div>
  209 
```

A screenshot of Visual Studio Code showing the code for `Airline.js`. The code is a React component for an airline page. It includes a search bar, a table for flight information, and a footer. The code uses CSS-in-JS and various React components like `<div>`, `<input>`, and `<table>`.

```
frontend > src > components > Airline.js > AirlinePage
  201   </div>
  202   </div>
  203 </div>
  204 </form>
  205
  206 <div className="row">
  207   <div className="col-lg-5 mt-5 ">
  208     <img src=(airplane) width="500px" height="450px" />
  209   </div>
  210   <div className="mt-5 "></div>
  211 </div>
  212
  213 <div className="col-lg-7 mt-4 ml-5 mb-4 fw-bold " style={{ height: "500px", overflow: "scroll", overflow: "auto" }}>
  214
  215   <div className="col-lg-4 p-2 mt-2 mb-2">
  216     <input type="search" placeholder="search" name="search" className="form-control" style={{ fontSize: "20px" }}>
  217     onChange={(e) => {
  218       setSearchTerm(e.target.value)
  219     }} />
  220   </div>
  221   <center> <h1>Airline Time Table</h1></center>
  222   <table class="table table-striped table-bordered table-hover bg-light" style={{ fontSize: "20px" }} >
  223     <thead class="bg-dark text-light">
  224       <tr>
  225         <th>From</th>
  226         <th>To</th>
  227         <th>Departure Time</th>
  228         <th>Arrival Time</th>
  229         <th>Flight Number</th>
  230         <th>Ticket fee(Rs)</th>
  231       </tr>
  232     </thead>
  233     <tbody>
  234       <tr>
  235         <td></td>
  236         <td></td>
  237         <td></td>
  238         <td></td>
  239         <td></td>
  240       </tr>
  241     </tbody>
  242   </table>
  243
  244   <AdminAirlineData filter={val} />
  245 </div>
  246
  247 <div>
  248   <img alt="Footer logo" />
  249   <p>Travel Agency Tour Planning Management System</p>
  250 </div>
  251
  252 <Footer />
  253
  254 <Footer />
  255
  256 <Footer />
  257
  258 <Footer />
  259
  260 <Footer />
  261
  262 <Footer />
  263
  264 <Footer />
  265
  266 <Footer />
  267
  268 <Footer />
  269
  270 <Footer />
  271
  272
  273
  274
  275
  276
  277
  278
  279
  280
  281
  282
  283
  284
  285
  286
  287
  288
  289
  290
  291
  292
  293
  294
  295
  296
  297
  298
  299
  300
  301
  302
  303
  304
  305
  306
  307
  308
  309
  310
  311
  312
  313
  314
  315
  316
  317
  318
  319
  320
  321
  322
  323
  324
  325
  326
  327
  328
  329
  330
  331
  332
  333
  334
  335
  336
  337
  338
  339
  340
  341
  342
  343
  344
  345
  346
  347
  348
  349
  350
  351
  352
  353
  354
  355
  356
  357
  358
  359
  360
  361
  362
  363
  364
  365
  366
  367
  368
  369
  370
  371
  372
  373
  374
  375
  376
  377
  378
  379
  380
  381
  382
  383
  384
  385
  386
  387
  388
  389
  390
  391
  392
  393
  394
  395
  396
  397
  398
  399
  400
  401
  402
  403
  404
  405
  406
  407
  408
  409
  410
  411
  412
  413
  414
  415
  416
  417
  418
  419
  420
  421
  422
  423
  424
  425
  426
  427
  428
  429
  430
  431
  432
  433
  434
  435
  436
  437
  438
  439
  440
  441
  442
  443
  444
  445
  446
  447
  448
  449
  450
  451
  452
  453
  454
  455
  456
  457
  458
  459
  460
  461
  462
  463
  464
  465
  466
  467
  468
  469
  470
  471
  472
  473
  474
  475
  476
  477
  478
  479
  480
  481
  482
  483
  484
  485
  486
  487
  488
  489
  490
  491
  492
  493
  494
  495
  496
  497
  498
  499
  500
  501
  502
  503
  504
  505
  506
  507
  508
  509
  510
  511
  512
  513
  514
  515
  516
  517
  518
  519
  520
  521
  522
  523
  524
  525
  526
  527
  528
  529
  530
  531
  532
  533
  534
  535
  536
  537
  538
  539
  540
  541
  542
  543
  544
  545
  546
  547
  548
  549
  550
  551
  552
  553
  554
  555
  556
  557
  558
  559
  560
  561
  562
  563
  564
  565
  566
  567
  568
  569
  570
  571
  572
  573
  574
  575
  576
  577
  578
  579
  580
  581
  582
  583
  584
  585
  586
  587
  588
  589
  590
  591
  592
  593
  594
  595
  596
  597
  598
  599
  600
  601
  602
  603
  604
  605
  606
  607
  608
  609
  610
  611
  612
  613
  614
  615
  616
  617
  618
  619
  620
  621
  622
  623
  624
  625
  626
  627
  628
  629
  630
  631
  632
  633
  634
  635
  636
  637
  638
  639
  640
  641
  642
  643
  644
  645
  646
  647
  648
  649
  650
  651
  652
  653
  654
  655
  656
  657
  658
  659
  660
  661
  662
  663
  664
  665
  666
  667
  668
  669
  670
  671
  672
  673
  674
  675
  676
  677
  678
  679
  680
  681
  682
  683
  684
  685
  686
  687
  688
  689
  690
  691
  692
  693
  694
  695
  696
  697
  698
  699
  700
  701
  702
  703
  704
  705
  706
  707
  708
  709
  710
  711
  712
  713
  714
  715
  716
  717
  718
  719
  720
  721
  722
  723
  724
  725
  726
  727
  728
  729
  730
  731
  732
  733
  734
  735
  736
  737
  738
  739
  740
  741
  742
  743
  744
  745
  746
  747
  748
  749
  750
  751
  752
  753
  754
  755
  756
  757
  758
  759
  760
  761
  762
  763
  764
  765
  766
  767
  768
  769
  770
  771
  772
  773
  774
  775
  776
  777
  778
  779
  780
  781
  782
  783
  784
  785
  786
  787
  788
  789
  790
  791
  792
  793
  794
  795
  796
  797
  798
  799
  800
  801
  802
  803
  804
  805
  806
  807
  808
  809
  810
  811
  812
  813
  814
  815
  816
  817
  818
  819
  820
  821
  822
  823
  824
  825
  826
  827
  828
  829
  830
  831
  832
  833
  834
  835
  836
  837
  838
  839
  840
  841
  842
  843
  844
  845
  846
  847
  848
  849
  850
  851
  852
  853
  854
  855
  856
  857
  858
  859
  860
  861
  862
  863
  864
  865
  866
  867
  868
  869
  870
  871
  872
  873
  874
  875
  876
  877
  878
  879
  880
  881
  882
  883
  884
  885
  886
  887
  888
  889
  890
  891
  892
  893
  894
  895
  896
  897
  898
  899
  900
  901
  902
  903
  904
  905
  906
  907
  908
  909
  910
  911
  912
  913
  914
  915
  916
  917
  918
  919
  920
  921
  922
  923
  924
  925
  926
  927
  928
  929
  930
  931
  932
  933
  934
  935
  936
  937
  938
  939
  940
  941
  942
  943
  944
  945
  946
  947
  948
  949
  950
  951
  952
  953
  954
  955
  956
  957
  958
  959
  960
  961
  962
  963
  964
  965
  966
  967
  968
  969
  970
  971
  972
  973
  974
  975
  976
  977
  978
  979
  980
  981
  982
  983
  984
  985
  986
  987
  988
  989
  990
  991
  992
  993
  994
  995
  996
  997
  998
  999
  1000
  1001
  1002
  1003
  1004
  1005
  1006
  1007
  1008
  1009
  1010
  1011
  1012
  1013
  1014
  1015
  1016
  1017
  1018
  1019
  1020
  1021
  1022
  1023
  1024
  1025
  1026
  1027
  1028
  1029
  1030
  1031
  1032
  1033
  1034
  1035
  1036
  1037
  1038
  1039
  1040
  1041
  1042
  1043
  1044
  1045
  1046
  1047
  1048
  1049
  1050
  1051
  1052
  1053
  1054
  1055
  1056
  1057
  1058
  1059
  1060
  1061
  1062
  1063
  1064
  1065
  1066
  1067
  1068
  1069
  1070
  1071
  1072
  1073
  1074
  1075
  1076
  1077
  1078
  1079
  1080
  1081
  1082
  1083
  1084
  1085
  1086
  1087
  1088
  1089
  1090
  1091
  1092
  1093
  1094
  1095
  1096
  1097
  1098
  1099
  1100
  1101
  1102
  1103
  1104
  1105
  1106
  1107
  1108
  1109
  1110
  1111
  1112
  1113
  1114
  1115
  1116
  1117
  1118
  1119
  1120
  1121
  1122
  1123
  1124
  1125
  1126
  1127
  1128
  1129
  1130
  1131
  1132
  1133
  1134
  1135
  1136
  1137
  1138
  1139
  1140
  1141
  1142
  1143
  1144
  1145
  1146
  1147
  1148
  1149
  1150
  1151
  1152
  1153
  1154
  1155
  1156
  1157
  1158
  1159
  1160
  1161
  1162
  1163
  1164
  1165
  1166
  1167
  1168
  1169
  1170
  1171
  1172
  1173
  1174
  1175
  1176
  1177
  1178
  1179
  1180
  1181
  1182
  1183
  1184
  1185
  1186
  1187
  1188
  1189
  1190
  1191
  1192
  1193
  1194
  1195
  1196
  1197
  1198
  1199
  1200
  1201
  1202
  1203
  1204
  1205
  1206
  1207
  1208
  1209
  1210
  1211
  1212
  1213
  1214
  1215
  1216
  1217
  1218
  1219
  1220
  1221
  1222
  1223
  1224
  1225
  1226
  1227
  1228
  1229
  1230
  1231
  1232
  1233
  1234
  1235
  1236
  1237
  1238
  1239
  1240
  1241
  1242
  1243
  1244
  1245
  1246
  1247
  1248
  1249
  1250
  1251
  1252
  1253
  1254
  1255
  1256
  1257
  1258
  1259
  1260
  1261
  1262
  1263
  1264
  1265
  1266
  1267
  1268
  1269
  1270
  1271
  1272
  1273
  1274
  1275
  1276
  1277
  1278
  1279
  1280
  1281
  1282
  1283
  1284
  1285
  1286
  1287
  1288
  1289
  1290
  1291
  1292
  1293
  1294
  1295
  1296
  1297
  1298
  1299
  1300
  1301
  1302
  1303
  1304
  1305
  1306
  1307
  1308
  1309
  1310
  1311
  1312
  1313
  1314
  1315
  1316
  1317
  1318
  1319
  1320
  1321
  1322
  1323
  1324
  1325
  1326
  1327
  1328
  1329
  1330
  1331
  1332
  1333
  1334
  1335
  1336
  1337
  1338
  1339
  1340
  1341
  1342
  1343
  1344
  1345
  1346
  1347
  1348
  1349
  1350
  1351
  1352
  1353
  1354
  1355
  1356
  1357
  1358
  1359
  1360
  1361
  1362
  1363
  1364
  1365
  1366
  1367
  1368
  1369
  1370
  1371
  1372
  1373
  1374
  1375
  1376
  1377
  1378
  1379
  1380
  1381
  1382
  1383
  1384
  1385
  1386
  1387
  1388
  1389
  1390
  1391
  1392
  1393
  1394
  1395
  1396
  1397
  1398
  1399
  1400
  1401
  1402
  1403
  1404
  1405
  1406
  1407
  1408
  1409
  1410
  1411
  1412
  1413
  1414
  1415
  1416
  1417
  1418
  1419
  1420
  1421
  1422
  1423
  1424
  1425
  1426
  1427
  1428
  1429
  1430
  1431
  1432
  1433
  1434
  1435
  1436
  1437
  1438
  1439
  1440
  1441
  1442
  1443
  1444
  1445
  1446
  1447
  1448
  1449
  1450
  1451
  1452
  1453
  1454
  1455
  1456
  1457
  1458
  1459
  1460
  1461
  1462
  1463
  1464
  1465
  1466
  1467
  1468
  1469
  1470
  1471
  1472
  1473
  1474
  1475
  1476
  1477
  1478
  1479
  1480
  1481
  1482
  1483
  1484
  1485
  1486
  1487
  1488
  1489
  1490
  1491
  1492
  1493
  1494
  1495
  1496
  1497
  1498
  1499
  1500
  1501
  1502
  1503
  1504
  1505
  1506
  1507
  1508
  1509
  1510
  1511
  1512
  1513
  1514
  1515
  1516
  1517
  1518
  1519
  1520
  1521
  1522
  1523
  1524
  1525
  1526
  1527
  1528
  1529
  1530
  1531
  1532
  1533
  1534
  1535
  1536
  1537
  1538
  1539
  1540
  1541
  1542
  1543
  1544
  1545
  1546
  1547
  1548
  1549
  1550
  1551
  1552
  1553
  1554
  1555
  1556
  1557
  1558
  1559
  1560
  1561
  1562
  1563
  1564
  1565
  1566
  1567
  1568
  1569
  1570
  1571
  1572
  1573
  1574
  1575
  1576
  1577
  1578
  1579
  1580
  1581
  1582
  1583
  1584
  1585
  1586
  1587
  1588
  1589
  1590
  1591
  1592
  1593
  1594
  1595
  1596
  1597
  1598
  1599
  1600
  1601
  1602
  1603
  1604
  1605
  1606
  1607
  1608
  1609
  1610
  1611
  1612
  1613
  1614
  1615
  1616
  1617
  1618
  1619
  1620
  1621
  1622
  1623
  1624
  1625
  1626
  1627
  1628
  1629
  1630
  1631
  1632
  1633
  1634
  1635
  1636
  1637
  1638
  1639
  1640
  1641
  1642
  1643
  1644
  1645
  1646
  1647
  1648
  1649
  1650
  1651
  1652
  1653
  1654
  1655
  1656
  1657
  1658
  1659
  1660
  1661
  1662
  1663
  1664
  1665
  1666
  1667
  1668
  1669
  1670
  1671
  1672
  1673
  1674
  1675
  1676
  1677
  1678
  1679
  1680
  1681
  1682
  1683
  1684
  1685
  1686
  1687
  1688
  1689
  1690
  1691
  1692
  1693
  1694
  1695
  1696
  1697
  1698
  1699
  1700
  1701
  1702
  1703
  1704
  1705
  1706
  1707
  1708
  1709
  1710
  1711
  1712
  1713
  1714
  1715
  1716
  1717
  1718
  1719
  1720
  1721
  1722
  1723
  1724
  1725
  1726
  1727
  1728
  1729
  1730
  1731
  1732
  1733
  1734
  1735
  1736
  1737
  1738
  1739
  1740
  1741
  1742
  1743
  1744
  1745
  1746
  1747
  1748
  1749
  1750
  1751
  1752
  1753
  1754
  1755
  1756
  1757
  1758
  1759
  1760
  1761
  1762
  1763
  1764
  1765
  1766
  1767
  1768
  1769
  1770
  1771
  1772
  1773
  1774
  1775
  1776
  1777
  1778
  1779
  1780
  1781
  1782
  1783
  1784
  1785
  1786
  1787
  1788
  1789
  1790
  1791
  1792
  1793
  1794
  1795
  1796
  1797
  1798
  1799
  1800
  1801
  1802
  1803
  1804
  1805
  1806
  1807
  1808
  1809
  1810
  1811
  1812
  1813
  1814
  1815
  1816
  1817
  1818
  1819
  1820
  1821
  1822
  1823
  1824
  1825
  1826
  1827
  1828
  1829
  1830
  1831
  1
```

2.3.3.4 Reservation of Ground resource Management

Models

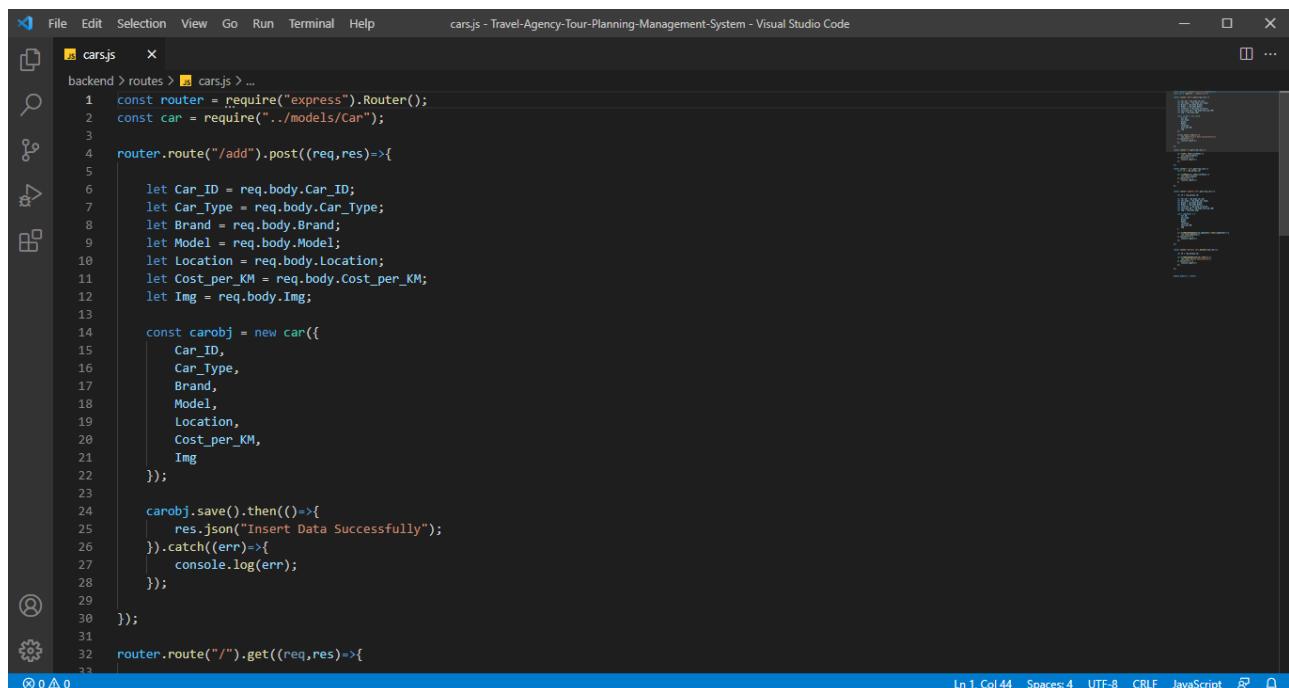


```
Car.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code
File Edit Selection View Go Run Terminal Help
Car.Reservation.js Car.js ✘ Guide.Reservation.js Guide.js
backend > models > Car.js > ...
1 const mongoose = require("mongoose");
2
3 const Schema = mongoose.Schema;
4
5 const Car = new Schema({
6   Car_ID:{
7     type:String,
8     required:true
9   },
10  Car_Type:{
11    type:String,
12    required:true
13  },
14  Brand:{
15    type:String,
16    required:true
17  },
18  Model:{
19    type:String,
20    required:true
21  },
22  Location:{
23    type:String,
24    required:true
25  },
26  Cost_per_KM:{
27    type:Number,
28    required:true
29  },
30  Img:{
31    type:String,
32    required:true
33  }
34 });
35
36 let car = mongoose.model("Car",Car);
37 module.exports = car;
```

The screenshot shows the Visual Studio Code interface with the 'Car.js' file open. The code defines a MongoDB schema for a 'Car' document. It includes fields for Car_ID (String, required), Car_Type (String, required), Brand (String, required), Model (String, required), Location (String, required), Cost_per_KM (Number, required), and Img (String, required). The file concludes by creating a Mongoose model named 'car' based on this schema and exporting it.

Figure 99

Routes



```
cars.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code
File Edit Selection View Go Run Terminal Help
cars.js ✘
backend > routes > cars.js > ...
1 const router = require("express").Router();
2 const car = require("../models/Car");
3
4 router.route("/add").post((req,res)=>{
5
6   let Car_ID = req.body.Car_ID;
7   let Car_Type = req.body.Car_Type;
8   let Brand = req.body.Brand;
9   let Model = req.body.Model;
10  let Location = req.body.Location;
11  let Cost_per_KM = req.body.Cost_per_KM;
12  let Img = req.body.Img;
13
14  const carobj = new car({
15    Car_ID,
16    Car_Type,
17    Brand,
18    Model,
19    Location,
20    Cost_per_KM,
21    Img
22  });
23
24  carobj.save().then(()=>{
25    res.json("Insert Data Successfully");
26  }).catch((err)=>{
27    console.log(err);
28  });
29
30 });
31
32 router.route("/").get((req,res)=>{
```

The screenshot shows the Visual Studio Code interface with the 'cars.js' file open. The code sets up two Express.js routes: a POST endpoint at '/add' to insert data into the 'Car' collection, and a GET endpoint at '/' to handle requests. The POST route extracts data from the request body (Car_ID, Car_Type, Brand, Model, Location, Cost_per_KM, Img) and creates a new 'carobj' document using the 'car' model. It then saves the document and returns a success message ('Insert Data Successfully'). The GET route is currently empty.

The screenshot shows the Visual Studio Code interface with the file 'cars.js' open. The code defines a router for handling car data. It includes methods for finding all cars, getting a car by ID, updating a car, and deleting a car.

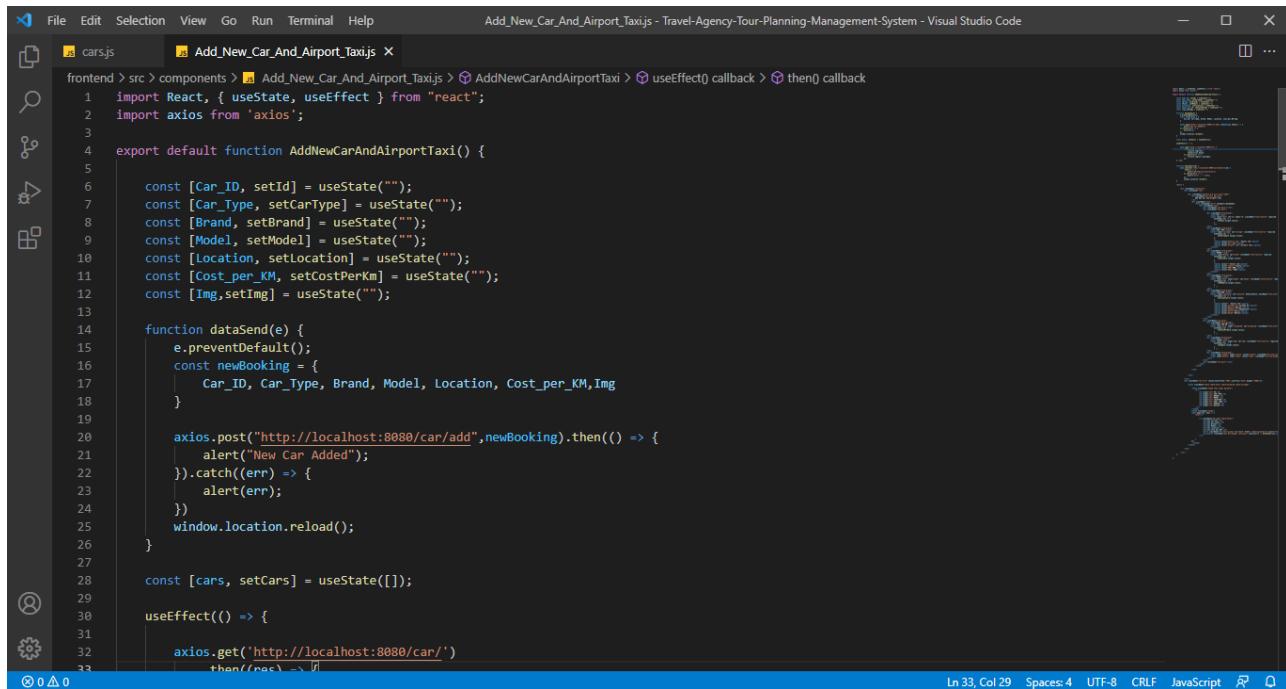
```
File Edit Selection View Go Run Terminal Help carsjs - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code
backend > routes > carsjs > ...
34     car.find().then((carData)=>{
35         res.json(carData);
36     }).catch((err)=>{
37         console.log(err);
38     });
39 );
40 );
41 router.route("/:id").get((req,res)=>{
42     const id = req.params.id;
43
44     car.findById(id).then((carData)=>{
45         res.json(carData)
46     }).catch((err)=>{
47         console.log(err);
48     })
49 );
50 );
51 );
52 );
53 router.route("/update/:id").put((req,res)=>{
54
55     let id = req.params.id;
56
57     let Car_ID = req.body.Car_ID;
58     let Car_Type = req.body.Car_Type;
59     let Brand = req.body.Brand;
60     let Model = req.body.Model;
61     let Location = req.body.Location;
62     let Cost_per_KM = req.body.Cost_per_KM;
63     let Img = req.body.Img;
64
65 );
66 );
67 );
68 );
69 );
70 );
71 );
72 );
73 );
74 );
75 );
76 );
77 );
78 );
79 );
80 );
81 );
82 );
83 );
84 );
85 );
86 );
87 );
88 );
89 );
90 );
91 );
92 );
93 );
94 );
95 );
96 );
97 );
98 );
99 );
module.exports = router;
```

The screenshot shows the Visual Studio Code interface with the file 'cars.js' open. The code defines a router for handling car data. It includes methods for finding all cars, getting a car by ID, updating a car, and deleting a car. A cursor is positioned over the 'delete' method in the router definition.

```
File Edit Selection View Go Run Terminal Help carsjs - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code
backend > routes > carsjs > delete() callback > then() callback
0
1     Car_ID,
2     Car_Type,
3     Brand,
4     Model,
5     Location,
6     Cost_per_KM,
7     Img
8 )
9
10    car.findByIdAndUpdate(id,updateCar).then((updateCar)=>{
11        res.json(updateCar);
12    }).catch((err)=>{
13        console.log(err);
14    });
15 );
16 );
17 );
18 );
19 );
20 );
21 );
22 );
23 );
24 );
25 );
26 );
27 );
28 );
29 );
30 );
31 );
32 );
33 );
34 );
35 );
36 );
37 );
38 );
39 );
40 );
41 );
42 );
43 );
44 );
45 );
46 );
47 );
48 );
49 );
50 );
51 );
52 );
53 );
54 );
55 );
56 );
57 );
58 );
59 );
60 );
61 );
62 );
63 );
64 );
65 );
66 );
67 );
68 );
69 );
69 );
70 );
71 );
72 );
73 );
74 );
75 );
76 );
77 );
78 );
79 );
80 );
81 );
82 );
83 );
84 );
85 );
86 );
87 );
88 );
89 );
90 );
91 );
92 );
93 );
94 );
95 );
96 );
97 );
98 );
99 );
module.exports = router;
```

Figure 100

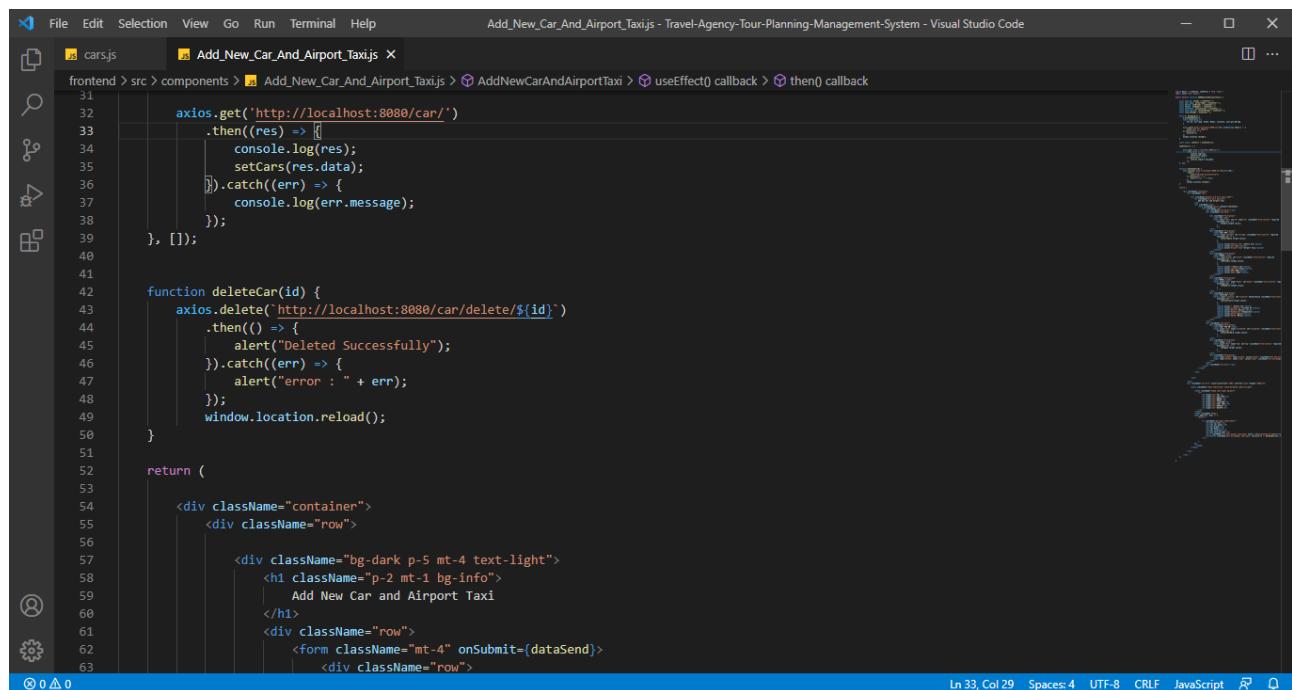
Frontend



frontend > src > components > Add_New_Car_And_Airport_Taxi.js > useEffect() callback > then() callback

```
1 import React, { useState, useEffect } from "react";
2 import axios from 'axios';
3
4 export default function AddNewCarAndAirportTaxi() {
5
6     const [Car_ID, setId] = useState("");
7     const [Car_Type, setCarType] = useState("");
8     const [Brand, setBrand] = useState("");
9     const [Model, setModel] = useState("");
10    const [Location, setLocation] = useState("");
11    const [Cost_per_KM, setCostPerKM] = useState("");
12    const [Img, setImg] = useState("");
13
14    function dataSend(e) {
15        e.preventDefault();
16        const newBooking = {
17            Car_ID, Car_Type, Brand, Model, Location, Cost_per_KM, Img
18        }
19
20        axios.post("http://localhost:8080/car/add", newBooking).then(() => {
21            alert("New Car Added");
22        }).catch((err) => {
23            alert(err);
24        })
25        window.location.reload();
26    }
27
28    const [cars, setCars] = useState([]);
29
30    useEffect(() => {
31
32        axios.get('http://localhost:8080/car/')
33            .then((res) => {
34                console.log(res);
35                setCars(res.data);
36            })
37            .catch((err) => {
38                console.log(err.message);
39            });
40    }, []);
41
42    function deleteCar(id) {
43        axios.delete(`http://localhost:8080/car/delete/${id}`)
44            .then(() => {
45                alert("Deleted Successfully");
46            })
47            .catch((err) => {
48                alert("Error : " + err);
49            });
50        window.location.reload();
51    }
52
53    return (
54        <div className="container">
55            <div className="row">
56
57                <div className="bg-dark p-5 mt-4 text-light">
58                    <h1 className="p-2 mt-1 bg-info">
59                        Add New Car and Airport Taxi
60                    </h1>
61                    <div className="row">
62                        <form className="mt-4" onSubmit={dataSend}>
63                            <div className="row">
```

Ln 33, Col 29 Spaces:4 UTF-8 CRLF JavaScript ⚡



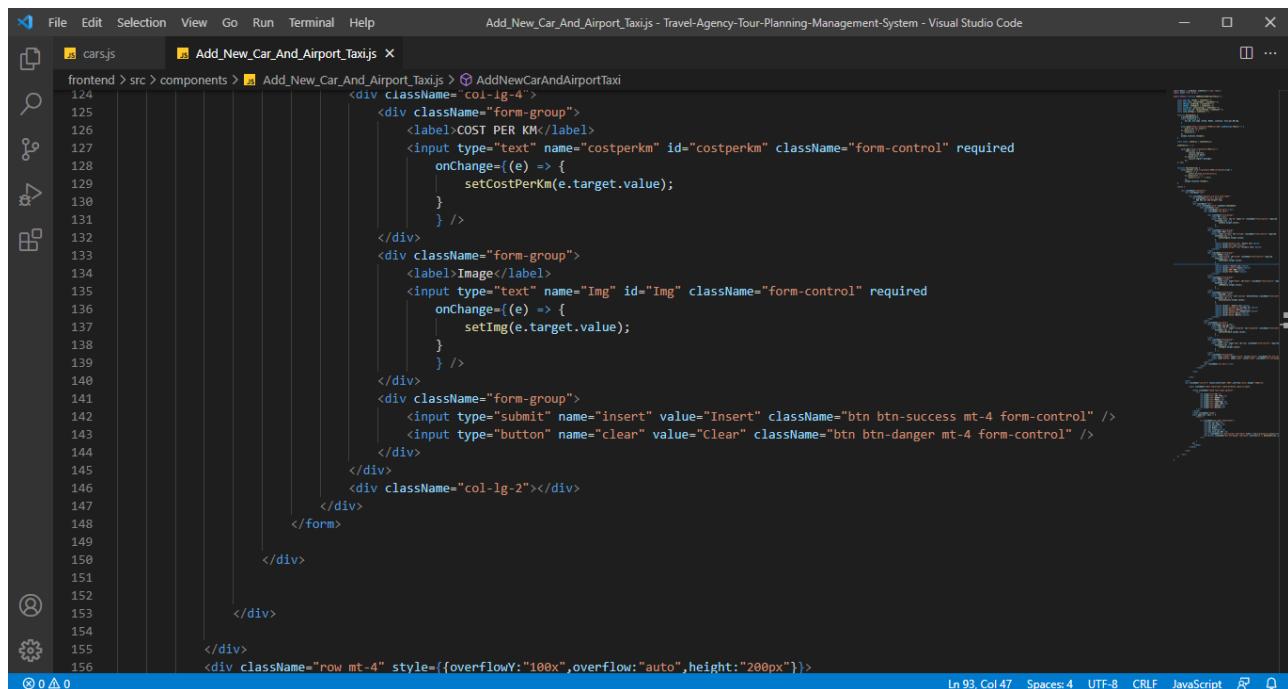
frontend > src > components > Add_New_Car_And_Airport_Taxi.js > useEffect() callback > then() callback

```
31
32    axios.get('http://localhost:8080/car/')
33        .then((res) => [
34            console.log(res);
35            setCars(res.data);
36        ])
37        .catch((err) => [
38            console.log(err.message);
39        ]);
40    [], [];
41
42    function deleteCar(id) {
43        axios.delete(`http://localhost:8080/car/delete/${id}`)
44            .then(() => {
45                alert("Deleted Successfully");
46            })
47            .catch((err) => {
48                alert("Error : " + err);
49            });
50        window.location.reload();
51    }
52
53    return (
54        <div className="container">
55            <div className="row">
56
57                <div className="bg-dark p-5 mt-4 text-light">
58                    <h1 className="p-2 mt-1 bg-info">
59                        Add New Car and Airport Taxi
60                    </h1>
61                    <div className="row">
62                        <form className="mt-4" onSubmit={dataSend}>
63                            <div className="row">
```

Ln 33, Col 29 Spaces:4 UTF-8 CRLF JavaScript ⚡

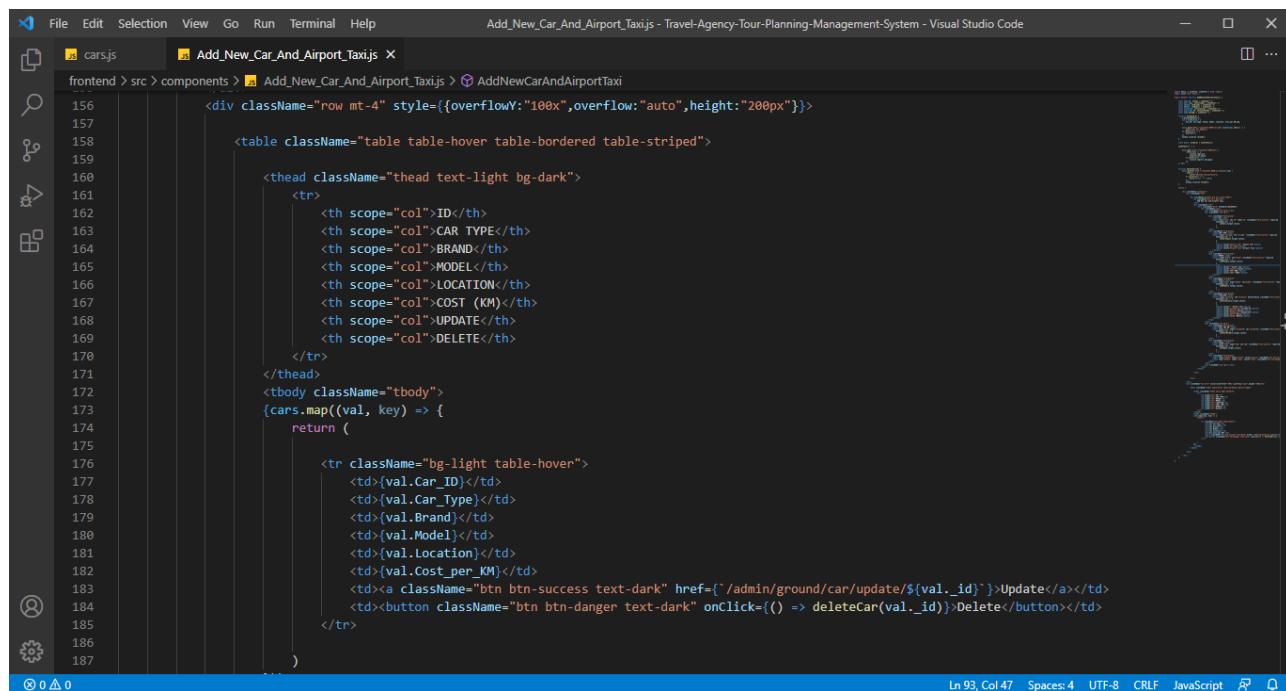
```
frontend > src > components > Add_New_Car_And_Airport_Taxi.js > AddNewCarAndAirportTaxi
  <form className="m-4" onSubmit={onSubmit}>
    <div className="row">
      <div className="col-lg-2"></div>
      <div className="col-lg-4">
        <div className="form-group">
          <label>ID</label>
          <input type="text" id="id" name="id" className="form-control" required
            onChange={(e) => {
              setId(e.target.value);
            }}>
        </div>
        <div className="form-group">
          <label>CAR TYPE</label>
          <select name="cartype" id="cartype" className="form-control" required
            onChange={(e) => {
              setCartype(e.target.value);
            }}>
            <option value="">Select one</option>
            <option value="Car">Car</option>
            <option value="Airport Taxi">Airport Taxi</option>
          </select>
        </div>
        <div className="form-group">
          <label>BRAND</label>
          <select name="brand" id="brand" className="form-control" required
            onChange={(e) => {
              setBrand(e.target.value);
            }}>
            <option value="">Select one</option>
```

```
  <option value="Toyota">Toyota</option>
  <option value="BMW">BMW</option>
  <option value="AUDI">AUDI</option>
</select>
</div>
<div className="form-group">
  <label>MODEL</label>
  <input type="text" id="model" name="model" className="form-control" required
    onChange={(e) => {
      setModel(e.target.value);
    }} />
</div>
<div className="form-group">
  <label>LOCATION:</label>
  <select name="location" id="location" defaultValue className="form-control" required
    onChange={(e) => {
      setLocation(e.target.value);
    }}>
    <option value="">Select one</option>
    <option value="Colombo 07">Colombo 07</option>
    <option value="Matara">Matara</option>
    <option value="Hambantota">Hambantota</option>
    <option value="Galle">Galle</option>
    <option value="Kandy">Matara</option>
```



```
124 <div className="col-lg-4">
125   <div className="form-group">
126     <label>COST PER KM</label>
127     <input type="text" name="costperkm" id="costperkm" className="form-control" required
128       onChange={(e) => {
129         setCostPerKm(e.target.value);
130       }} />
131   </div>
132   <div className="form-group">
133     <label>Image</label>
134     <input type="text" name="Img" id="Img" className="form-control" required
135       onChange={(e) => {
136         setImg(e.target.value);
137       }} />
138   </div>
139   <div className="form-group">
140     <input type="submit" name="insert" value="Insert" className="btn btn-success mt-4 form-control" />
141     <input type="button" name="clear" value="Clear" className="btn btn-danger mt-4 form-control" />
142   </div>
143 </div>
144 </div>
145 </div>
146 </div>
147 </div>
148 </div>
149 </div>
150 </div>
151 </div>
152 </div>
153 </div>
154 </div>
155 </div>
156 </div>
```

Ln 93, Col 47 Spaces: 4 UTF-8 CRLF JavaScript



```
156 <div className="row mt-4" style={{overflowY:"100x",overflow:"auto",height:"200px"}}>
157
158   <table className="table table-hover table-bordered table-striped">
159
160     <thead className="thead text-light bg-dark">
161       <tr>
162         <th scope="col">ID</th>
163         <th scope="col">CAR TYPE</th>
164         <th scope="col">BRAND</th>
165         <th scope="col">MODEL</th>
166         <th scope="col">LOCATION</th>
167         <th scope="col">COST (KM)</th>
168         <th scope="col">UPDATE</th>
169         <th scope="col">DELETE</th>
170       </th>
171     </thead>
172     <tbody className="tbody">
173       {cars.map((val, key) => {
174         return (
175
176           <tr className="bg-light table-hover">
177             <td>{val.Car_ID}</td>
178             <td>{val.Car_Type}</td>
179             <td>{val.Brand}</td>
180             <td>{val.Model}</td>
181             <td>{val.Location}</td>
182             <td>{val.Cost_per_KM}</td>
183             <td><a className="btn btn-success text-dark" href={`/admin/ground/car/update/${val._id}}>Update</a></td>
184             <td><button className="btn btn-danger text-dark" onClick={() => deleteCar(val._id)}>Delete</button></td>
185           </tr>
186         );
187       })
188     </tbody>
189   </table>
190 </div>
```

Ln 93, Col 47 Spaces: 4 UTF-8 CRLF JavaScript

The screenshot shows a dark-themed instance of Visual Studio Code. The title bar reads "Add_New_Car_And_Airport_Taxi.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code". The main editor area displays code for "Add_New_Car_And_Airport_Taxi.js", specifically the "Cars" component. The code includes HTML-like syntax with line numbers from 187 to 197. A vertical sidebar on the right shows a tree view of the project structure, including "src", "components", and "Add_New_Car_And_Airport_Taxi.js". The status bar at the bottom indicates "Ln 93, Col 47" and "JavaScript".

Figure 101

2.3.3.5 Travel Document Management

Models

The screenshot shows a dark-themed instance of Visual Studio Code. The title bar reads "TravelDocAdd.js - Travel update with oshe - Visual Studio Code". The main editor area displays code for "TravelDocAdd.js", which uses Mongoose to define a schema for travel documents. The code includes imports for mongoose and defines a schema with fields like "Document_name", "Document_link", "Country_name", and "Submit_date". The status bar at the bottom indicates "Ln 1, Col 1" and "JavaScript". The left sidebar shows the project structure under "TRAVEL UPDATE WITH OSHE", including "Travel", "backend", "models", and "TravelDocAdd.js".

Figure 102

Routes

```

File Edit Selection View Go Run Terminal Help
Travel > backend > routes > TravelDocAdd.js - Travel update with oshe - Visual Studio Code
OPEN EDITORS .env server.js M App.js M navbar.js M admin.js M TravelDocAdd.js ...models TravelDocAdd.js ...routes AdminButton.js U ...
Travel > backend > routes > TravelDocAdd.js ...
1 const express = require('express');
2 const TravelDocAdd = require('../models/TravelDocAdd');
3
4 const router = express.Router();
5
6 //save Inquiry
7
8 router.post('/TravelDocAdd/save',(req,res) =>{
9
10     let NewTravelDocAdd = new TravelDocAdd(req.body);
11
12     NewTravelDocAdd.save((err) =>{
13         if(err){
14             return res.status(400).json({
15                 error:err
16             });
17         }
18         return res.status(200).json({
19             success:"Travel document sent successfully"
20         });
21     });
22 });
23
24
25 router.get('/TravelDocAdd',(req,res) =>{
26     TravelDocAdd.find().exec((err,TravelDocAdd) =>{
27         if(err){
28             return res.status(400).json({
29                 error:err
30             });
31         }
32         return res.status(200).json({
33             success:true,
34             existingTravelDocAdd:TravelDocAdd
35         });
36     });
37 });

```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF {} JavaScript

```

File Edit Selection View Go Run Terminal Help
Travel > backend > routes > TravelDocAdd.js - Travel update with oshe - Visual Studio Code
OPEN EDITORS .env server.js M App.js M navbar.js M admin.js M TravelDocAdd.js ...models TravelDocAdd.js ...routes AdminButton.js U ...
Travel > backend > routes > TravelDocAdd.js ...
38 },
39 //get specific Traveldoc
40
41 router.get("/TravelDocAdd/:id",(req,res) => {
42
43     let TravelDocAddId = req.params.id;
44
45     TravelDocAdd.findById(TravelDocAddId,(err,post) =>{
46         if(err){
47             return res.status(400).json({success:false, err});
48         }
49
50         return res.status(200).json({
51             success:true,
52             post
53         });
54     });
55 });
56
57
58 router.put('/TravelDocAdd/update/:id',(req,res)->{
59     TravelDocAdd.findByIdAndUpdateAndUpdate(
60         req.params.id,
61         {
62             $set:req.body
63         },
64         (err,post)->{
65             if(err){
66                 return res.status(400).json({error:err});
67             }
68             return res.status(200).json({
69                 success:"Updated Travel Document succesfully"
70             });
71         }
72     );
73 });
74

```

Ln 5, Col 1 Spaces: 4 UTF-8 CRLF {} JavaScript

The screenshot shows the Visual Studio Code interface with the 'Travel' project open. The Explorer sidebar on the left lists files and folders under 'OPEN EDITORS' and 'TRAVEL UPDATE WITH OSHE'. The main editor area displays the code for `TravelDocAdd.js`, which handles routes for adding travel documents. The status bar at the bottom shows 'Ln 42, Col 1' and other settings.

```

Travel > backend > routes > TravelDocAdd.js > router.get('/TravelDocAdd/:id') callback
74
75 router.delete('/TravelDocAdd/delete/:id',(req,res) =>
76   TravelDocAdd.findByIdAndDelete(req.params.id).exec((err,deleteTravelDocAdd) =>
77     if(err) return res.status(400).json({
78       message:"Delete unsuccesfull",err
79     });
80
81     return res.json({
82       message:"Delete succesfull",deleteTravelDocAdd
83     });
84   );
85
86 module.exports = router;

```

Figure 103

Frontend

The screenshot shows the Visual Studio Code interface with the 'Travel' project open. The Explorer sidebar on the left lists files and folders under 'OPEN EDITORS' and 'TRAVEL UPDATE WITH OSHE'. The main editor area displays the code for `AddDocument.js`, which is a React component for adding documents. The status bar at the bottom shows 'Ln 1, Col 1' and other settings.

```

1 import React, { Component } from 'react'
2 import Header from './header';
3 import Navbar from './navbar';
4 import PageImage from './pageimage';
5 import Sidenavavigation from './sidenavigation';
6 import Footer from './footer';
7 import Layout from './Layout';
8 import axios from 'axios';
9
10 import "../style/adddocument.css"
11
12 export default class AddDocument extends Component {
13   constructor(props){
14     super(props);
15     this.state={
16       Document_name:"",
17       Document_Link:"",
18       Country_name:"",
19       Submit_date:""
20     }
21   }
22
23   handleInputChange = (e) =>{
24     const {name,value} = e.target;
25
26     this.setState({
27       ...this.state,
28       [name]:value
29     })
30   }
31
32   onSubmit = (e) =>{
33     e.preventDefault();
34
35     const {Document_name,Document_Link,Country_name,Submit_date} = this.state;
36   }
37

```

```
File Edit Selection View Go Run Terminal Help AddDocument.js - Travel update with osn - Visual Studio Code

EXPLORER
OPEN EDITORS
Travel > frontend > src > components > AddDocument.js > AddDocument > constructor
... .env server.js M App.js M navbar.js M admin.js M TravelDocAdd.js .../models TravelDocAdd.js .../routes AddDocument.js x

28     })
29   }
30 }
31
32 onSubmit = (e) => {
33   e.preventDefault();
34
35   const {Document_name,Document_Link,country_name,Submit_date} = this.state;
36
37   const data = {
38     Document_name:Document_name,
39     Document_Link:Document_Link,
40     Country_name:Country_name,
41     Submit_date:Submit_date
42   }
43
44   console.log(data)
45
46   axios.post('http://localhost:8080/TravelDocAdd/save',data).then((res) =>{
47     if(res.data.success){
48       this.setState(
49         {
50           Document_name:"",
51           Document_Link:"",
52           Country_name:"",
53           Submit_date:""
54         }
55       )
56     }
57   })
58 }
59
60 render() {
61   return (
62     ...
63   )
64 }
```

```
File Edit Selection View Go Run Terminal Help
AddDocument.js - Travel update with oshei - Visual Studio Code

EXPLORER .env server.js M App.js M navbar.js M admin.js M TravelDocAdd.js ...models TravelDocAdd.js ...routes AddDocument.js
OPEN EDITORS
Travel front end > src > components > AddDocument.js > AddDocument > constructor
render() {
  return (
    <div>
      <Layout />
      <Navbar />
      <PageImage />
      <SideNavigation />
      <div className="inquiryform">
        <div className="col-md-8 mt-4 mx-auto">
          <h1 className="h3 mb-3 font-weight-normal">Add Document</h1>
          <form className="needs-validation" noValidation>
            <div className="form-group" style={{marginBottom:'15px'}}>
              <label style={{marginBottom:'15px'}}>Country name</label>
              <input type="text" className="form-control" name="country_name" placeholder="Enter country name" value={this.state.country_name} onChange={this.handleInputChange}/>
            </div>
            <div className="form-group" style={{marginBottom:'15px'}}>
              <label style={{marginBottom:'15px'}}>Document Path</label>
              <input type="text" className="form-control" name="Document_link" placeholder="Enter URL Link" value={this.state.Document_link} onChange={this.handleInputChange}/>
            </div>
            <div className="form-group" style={{marginBottom:'15px'}}>
              <label style={{marginBottom:'15px'}}>Document name</label>
              <input type="text" />
            </div>
          </form>
        </div>
      </div>
    </div>
  )
}

</div>
```

File Edit Selection View Go Run Terminal Help

AddDocument.js - Travel update with oshe - Visual Studio Code

OPEN EDITORS

- Travel\backend \.env
- Travel\backend server.js M
- Travel\frontend\src App.js M
- Travel\frontend\src navbar.js M
- Travel\backend\src admin.js M
- TravelDocAdd.js Travel\backend\...\models
- TravelDocAdd.js Travel\backend\...\routes
- AddDocument.js Travel\frontend\...\components**
- AdminButton.js Travel\frontend\...\components

TRAVEL UPDATE WITH OSHE

- src
- admin
- auth
- components
- AddDocument.js
- Admin_Airline_page.js U
- Admin_Bus_page.js U
- Admin_Railway_page.js U
- admin.js M
- adminDeletePackage.js
- adminEditPackage.js
- adminReservations.js
- adminSidebar.js M
- adminTour.js
- adminTravelDoc.js
- adminTravelDocAdd.js
- adminTravelDocAddEdit.js
- adminTravelDocApplication.js

```

84      <div className="form-group" style={{marginBottom:'15px'}}>
85        <label style={{marginBottom:'15px'}}>Document Path</label>
86        <input type="text" className="form-control" name="document_link" placeholder="Enter URL Link" value={this.state.Document_Link} onChange={this.handleInputChange}/>
87      </div>
88
89      <div className="form-group" style={{marginBottom:'15px'}}>
90        <label style={{marginBottom:'15px'}}>Document name</label>
91        <input type="text" className="form-control" name="Document_name" placeholder="Enter Document name" value={this.state.Document_name} onChange={this.handleInputChange}/>
92      </div>
93
94      <div className="form-group" style={{marginBottom:'15px'}}>
95        <label style={{marginBottom:'15px'}}>Submit Date</label>
96        <input type="text" className="form-control" name="Submit_date" placeholder="Enter the date" value={this.state.Submit_date} onChange={this.handleInputChange}/>
97      </div>
98
99      <button className="btn btn-success" type="submit" style={{marginTop:'15px'}} onClick={this.onSubmit}>
100        <i className="far fa-check-square"></i>
101        &ampnbsp Send
102      </button>
103      &ampnbsp
104      <a href="/Infomation">
105    
```

Line 69, Col 31 Spaces: 4 UTF-8 CRLF {} JavaScript

File Edit Selection View Go Run Terminal Help

Presenting... Give control Stop presenting

AddDocument.js - Travel update with oshe - Visual Studio Code

OPEN EDITORS

- Travel\backend \.env
- Travel\backend server.js M
- Travel\frontend\src App.js M
- Travel\frontend\src navbar.js M
- Travel\backend\src admin.js M
- TravelDocAdd.js Travel\backend\...\models
- TravelDocAdd.js Travel\backend\...\routes
- AddDocument.js Travel\frontend\...\components**
- AdminButton.js Travel\frontend\...\components

TRAVEL UPDATE WITH OSHE

- src
- admin
- auth
- components
- AddDocument.js
- Admin_Airline_page.js U
- Admin_Bus_page.js U
- Admin_Railway_page.js U
- admin.js M
- adminDeletePackage.js
- adminEditPackage.js
- adminReservations.js
- adminSidebar.js M
- adminTour.js
- adminTravelDoc.js
- adminTravelDocAdd.js
- adminTravelDocAddEdit.js
- adminTravelDocApplication.js

```

107      <input type="text" className="form-control" name="Submit_date" placeholder="Enter the date" value={this.state.Submit_date} onChange={this.handleInputChange}/>
108
109      <button className="btn btn-success" type="submit" style={{marginTop:'15px'}} onClick={this.onSubmit}>
110        <i className="far fa-check-square"></i>
111        &ampnbsp Send
112      </button>
113      &ampnbsp
114      <a href="/Infomation">
115        <button className="btn btn-success" style={{marginTop:'15px'}} onClick={this.onSubmit}>
116          <i className="far fa-cross"></i>
117          &ampnbsp Cancel
118        </button>
119      </a>
120
121      </div>
122
123      <div className="footerpartAd">
124        <Footer />
125      </div>
126
127    }
128
129  }
130
131  </div>
132
133  </div>
134
135  </div>
136
137  </div>
138
139  </div>
140
141  </div>
142
143  </div>
144
145  </div>
146
147  </div>
148
149  </div>
150
151  </div>
152
153  </div>
154
155  </div>
156
157  </div>
158
159  </div>
160
161  </div>
162
163  </div>
164
165  </div>
166
167  </div>
168
169  </div>
170
171  </div>
172
173  </div>
174
175  </div>
176
177  </div>
178
179  </div>
180
181  </div>
182
183  </div>
184
185  </div>
186
187  </div>
188
189  </div>
190
191  </div>
192
193  </div>
194
195  </div>
196
197  </div>
198
199  </div>
200
201  </div>
202
203  </div>
204
205  </div>
206
207  </div>
208
209  </div>
210
211  </div>
212
213  </div>
214
215  </div>
216
217  </div>
218
219  </div>
220
221  </div>
222
223  </div>
224
225  </div>
226
227  </div>
228
229  </div>
230
231  </div>
232
233  </div>
234
235  </div>
236
237  </div>
238
239  </div>
240
241  </div>
242
243  </div>
244
245  </div>
246
247  </div>
248
249  </div>
250
251  </div>
252
253  </div>
254
255  </div>
256
257  </div>
258
259  </div>
260
261  </div>
262
263  </div>
264
265  </div>
266
267  </div>
268
269  </div>
270
271  </div>
272
273  </div>
274
275  </div>
276
277  </div>
278
279  </div>
280
281  </div>
282
283  </div>
284
285  </div>
286
287  </div>
288
289  </div>
290
291  </div>
292
293  </div>
294
295  </div>
296
297  </div>
298
299  </div>
300
301  </div>
302
303  </div>
304
305  </div>
306
307  </div>
308
309  </div>
310
311  </div>
312
313  </div>
314
315  </div>
316
317  </div>
318
319  </div>
320
321  </div>
322
323  </div>
324
325  </div>
326
327  </div>
328
329  </div>
330
331  </div>
332
333  </div>
334
335  </div>
336
337  </div>
338
339  </div>
340
341  </div>
342
343  </div>
344
345  </div>
346
347  </div>
348
349  </div>
350
351  </div>
352
353  </div>
354
355  </div>
356
357  </div>
358
359  </div>
360
361  </div>
362
363  </div>
364
365  </div>
366
367  </div>
368
369  </div>
370
371  </div>
372
373  </div>
374
375  </div>
376
377  </div>
378
379  </div>
380
381  </div>
382
383  </div>
384
385  </div>
386
387  </div>
388
389  </div>
390
391  </div>
392
393  </div>
394
395  </div>
396
397  </div>
398
399  </div>
400
401  </div>
402
403  </div>
404
405  </div>
406
407  </div>
408
409  </div>
410
411  </div>
412
413  </div>
414
415  </div>
416
417  </div>
418
419  </div>
420
421  </div>
422
423  </div>
424
425  </div>
426
427  </div>
428
429  </div>
430
431  </div>
432
433  </div>
434
435  </div>
436
437  </div>
438
439  </div>
440
441  </div>
442
443  </div>
444
445  </div>
446
447  </div>
448
449  </div>
450
451  </div>
452
453  </div>
454
455  </div>
456
457  </div>
458
459  </div>
460
461  </div>
462
463  </div>
464
465  </div>
466
467  </div>
468
469  </div>
470
471  </div>
472
473  </div>
474
475  </div>
476
477  </div>
478
479  </div>
480
481  </div>
482
483  </div>
484
485  </div>
486
487  </div>
488
489  </div>
490
491  </div>
492
493  </div>
494
495  </div>
496
497  </div>
498
499  </div>
500
501  </div>
502
503  </div>
504
505  </div>
506
507  </div>
508
509  </div>
510
511  </div>
512
513  </div>
514
515  </div>
516
517  </div>
518
519  </div>
520
521  </div>
522
523  </div>
524
525  </div>
526
527  </div>
528
529  </div>
530
531  </div>
532
533  </div>
534
535  </div>
536
537  </div>
538
539  </div>
540
541  </div>
542
543  </div>
544
545  </div>
546
547  </div>
548
549  </div>
550
551  </div>
552
553  </div>
554
555  </div>
556
557  </div>
558
559  </div>
560
561  </div>
562
563  </div>
564
565  </div>
566
567  </div>
568
569  </div>
570
571  </div>
572
573  </div>
574
575  </div>
576
577  </div>
578
579  </div>
580
581  </div>
582
583  </div>
584
585  </div>
586
587  </div>
588
589  </div>
590
591  </div>
592
593  </div>
594
595  </div>
596
597  </div>
598
599  </div>
599  </div>
600
601  </div>
602
603  </div>
604
605  </div>
606
607  </div>
608
609  </div>
610
611  </div>
612
613  </div>
614
615  </div>
616
617  </div>
618
619  </div>
620
621  </div>
622
623  </div>
624
625  </div>
626
627  </div>
628
629  </div>
630
631  </div>
632
633  </div>
634
635  </div>
636
637  </div>
638
639  </div>
640
641  </div>
642
643  </div>
644
645  </div>
646
647  </div>
648
649  </div>
650
651  </div>
652
653  </div>
654
655  </div>
656
657  </div>
658
659  </div>
659  </div>
660
661  </div>
662
663  </div>
664
665  </div>
666
667  </div>
668
669  </div>
670
671  </div>
672
673  </div>
674
675  </div>
676
677  </div>
678
679  </div>
679  </div>
680
681  </div>
682
683  </div>
684
685  </div>
686
687  </div>
688
689  </div>
689  </div>
690
691  </div>
692
693  </div>
694
695  </div>
696
697  </div>
698
699  </div>
699  </div>
700
701  </div>
702
703  </div>
704
705  </div>
706
707  </div>
708
709  </div>
710
711  </div>
712
713  </div>
714
715  </div>
716
717  </div>
718
719  </div>
719  </div>
720
721  </div>
722
723  </div>
724
725  </div>
726
727  </div>
728
729  </div>
729  </div>
730
731  </div>
732
733  </div>
734
735  </div>
736
737  </div>
737  </div>
738
739  </div>
740
741  </div>
742
743  </div>
744
745  </div>
745  </div>
746
747  </div>
748
749  </div>
749  </div>
750
751  </div>
752
753  </div>
754
755  </div>
755  </div>
756
757  </div>
758
759  </div>
759  </div>
760
761  </div>
762
763  </div>
764
765  </div>
765  </div>
766
767  </div>
768
769  </div>
769  </div>
770
771  </div>
772
773  </div>
773  </div>
774
775  </div>
776
777  </div>
777  </div>
778
779  </div>
779  </div>
780
781  </div>
782
783  </div>
783  </div>
784
785  </div>
786
787  </div>
787  </div>
788
789  </div>
789  </div>
790
791  </div>
792
793  </div>
793  </div>
794
795  </div>
795  </div>
796
797  </div>
797  </div>
798
799  </div>
799  </div>
800
801  </div>
802
803  </div>
803  </div>
804
805  </div>
805  </div>
806
807  </div>
807  </div>
808
809  </div>
809  </div>
810
811  </div>
811  </div>
812
813  </div>
813  </div>
814
815  </div>
815  </div>
816
817  </div>
817  </div>
818
819  </div>
819  </div>
820
821  </div>
821  </div>
822
823  </div>
823  </div>
824
825  </div>
825  </div>
826
827  </div>
827  </div>
828
829  </div>
829  </div>
830
831  </div>
831  </div>
832
833  </div>
833  </div>
834
835  </div>
835  </div>
836
837  </div>
837  </div>
838
839  </div>
839  </div>
840
841  </div>
841  </div>
842
843  </div>
843  </div>
844
845  </div>
845  </div>
846
847  </div>
847  </div>
848
849  </div>
849  </div>
850
851  </div>
851  </div>
852
853  </div>
853  </div>
854
855  </div>
855  </div>
856
857  </div>
857  </div>
858
859  </div>
859  </div>
860
861  </div>
861  </div>
862
863  </div>
863  </div>
864
865  </div>
865  </div>
866
867  </div>
867  </div>
868
869  </div>
869  </div>
870
871  </div>
871  </div>
872
873  </div>
873  </div>
874
875  </div>
875  </div>
876
877  </div>
877  </div>
878
879  </div>
879  </div>
880
881  </div>
881  </div>
882
883  </div>
883  </div>
884
885  </div>
885  </div>
886
887  </div>
887  </div>
888
889  </div>
889  </div>
890
891  </div>
891  </div>
892
893  </div>
893  </div>
894
895  </div>
895  </div>
896
897  </div>
897  </div>
898
899  </div>
899  </div>
900
901  </div>
901  </div>
902
903  </div>
903  </div>
904
905  </div>
905  </div>
906
907  </div>
907  </div>
908
909  </div>
909  </div>
910
911  </div>
911  </div>
912
913  </div>
913  </div>
914
915  </div>
915  </div>
916
917  </div>
917  </div>
918
919  </div>
919  </div>
920
921  </div>
921  </div>
922
923  </div>
923  </div>
924
925  </div>
925  </div>
926
927  </div>
927  </div>
928
929  </div>
929  </div>
930
931  </div>
931  </div>
932
933  </div>
933  </div>
934
935  </div>
935  </div>
936
937  </div>
937  </div>
938
939  </div>
939  </div>
940
941  </div>
941  </div>
942
943  </div>
943  </div>
944
945  </div>
945  </div>
946
947  </div>
947  </div>
948
949  </div>
949  </div>
950
951  </div>
951  </div>
952
953  </div>
953  </div>
954
955  </div>
955  </div>
956
957  </div>
957  </div>
958
959  </div>
959  </div>
960
961  </div>
961  </div>
962
963  </div>
963  </div>
964
965  </div>
965  </div>
966
967  </div>
967  </div>
968
969  </div>
969  </div>
970
971  </div>
971  </div>
972
973  </div>
973  </div>
974
975  </div>
975  </div>
976
977  </div>
977  </div>
978
979  </div>
979  </div>
980
981  </div>
981  </div>
982
983  </div>
983  </div>
984
985  </div>
985  </div>
986
987  </div>
987  </div>
988
989  </div>
989  </div>
990
991  </div>
991  </div>
992
993  </div>
993  </div>
994
995  </div>
995  </div>
996
997  </div>
997  </div>
998
999  </div>
999  </div>
1000
1001  </div>
1001  </div>
1002
1003  </div>
1003  </div>
1004
1005  </div>
1005  </div>
1006
1007  </div>
1007  </div>
1008
1009  </div>
1009  </div>
1010
1011  </div>
1011  </div>
1012
1013  </div>
1013  </div>
1014
1015  </div>
1015  </div>
1016
1017  </div>
1017  </div>
1018
1019  </div>
1019  </div>
1020
1021  </div>
1021  </div>
1022
1023  </div>
1023  </div>
1024
1025  </div>
1025  </div>
1026
1027  </div>
1027  </div>
1028
1029  </div>
1029  </div>
1030
1031  </div>
1031  </div>
1032
1033  </div>
1033  </div>
1034
1035  </div>
1035  </div>
1036
1037  </div>
1037  </div>
1038
1039  </div>
1039  </div>
1040
1041  </div>
1041  </div>
1042
1043  </div>
1043  </div>
1044
1045  </div>
1045  </div>
1046
1047  </div>
1047  </div>
1048
1049  </div>
1049  </div>
1050
1051  </div>
1051  </div>
1052
1053  </div>
1053  </div>
1054
1055  </div>
1055  </div>
1056
1057  </div>
1057  </div>
1058
1059  </div>
1059  </div>
1060
1061  </div>
1061  </div>
1062
1063  </div>
1063  </div>
1064
1065  </div>
1065  </div>
1066
1067  </div>
1067  </div>
1068
1069  </div>
1069  </div>
1070
1071  </div>
1071  </div>
1072
1073  </div>
1073  </div>
1074
1075  </div>
1075  </div>
1076
1077  </div>
1077  </div>
1078
1079  </div>
1079  </div>
1080
1081  </div>
1081  </div>
1082
1083  </div>
1083  </div>
1084
1085  </div>
1085  </div>
1086
1087  </div>
1087  </div>
1088
1089  </div>
1089  </div>
1090
1091  </div>
1091  </div>
1092
1093  </div>
1093  </div>
1094
1095  </div>
1095  </div>
1096
1097  </div>
1097  </div>
1098
1099  </div>
1099  </div>
1100
1101  </div>
1101  </div>
1102
1103  </div>
1103  </div>
1104
1105  </div>
1105  </div>
1106
1107  </div>
1107  </div>
1108
1109  </div>
1109  </div>
1110
1111  </div>
1111  </div>
1112
1113  </div>
1113  </div>
1114
1115  </div>
1115  </div>
1116
1117  </div>
1117  </div>
1118
1119  </div>
1119  </div>
1120
1121  </div>
1121  </div>
1122
1123  </div>
1123  </div>
1124
1125  </div>
1125  </div>
1126
1127  </div>
1127  </div>
1128
1129  </div>
1129  </div>
1130
1131  </div>
1131  </div>
1132
1133  </div>
1133  </div>
1134
1135  </div>
1135  </div>
1136
1137  </div>
1137  </div>
1138
1139  </div>
1139  </div>
1140
1141  </div>
1141  </div>
1142
1143  </div>
1143  </div>
1144
1145  </div>
1145  </div>
1146
1147  </div>
1147  </div>
1148
1149  </div>
1149  </div>
1150
1151  </div>
1151  </div>
1152
1153  </div>
1153  </div>
1154
1155  </div>
1155  </div>
1156
1157  </div>
1157  </div>
1158
1159  </div>
1159  </div>
1160
1161  </div>
1161  </div>
1162
1163  </div>
1163  </div>
1164
1165  </div>
1165  </div>
1166
1167  </div>
1167  </div>
1168
1169  </div>
1169  </div>
1170
1171  </div>
1171  </div>
1172
1173  </div>
1173  </div>
1174
1175  </div>
1175  </div>
1176
1177  </div>
1177  </div>
1178
1179  </div>
1179  </div>
1180
1181  </div>
1181  </div>
1182
1183  </div>
1183  </div>
1184
1185  </div>
1185  </div>
1186
1187  </div>
1187  </div>
1188
1189  </div>
1189  </div>
1190
1191  &lt
```

2.3.3.6 Tour Package Management

Models

```

modeltourPackage.js
1 const mongoose = require("mongoose");
2 const Schema = mongoose.Schema;
3
4 //create tour packages schema
5 const tourPackageSchema = new Schema({
6   packageID : {
7     type : String,
8     required : true
9   },
10  packageName : {
11    type : String,
12    required : true
13  },
14  PackageType : {
15    type : String,
16    required : true
17  },
18  NoofDays : {
19    type : Number,
20    required : true
21  },
22  MealPlan : {
23    type : String,
24    required : true
25  },
26  Transportation : {
27    type : String,
28    required : true
29  },
30  Activities : {
31    type : String,
32    required : true
33  },
34  locationlist : {
35    type : String,
36    required : true
37  },
38  PriceForAdult : {
39    type : Number,
40    required : true
41  },
42  PriceForChild : {
43    type : Number,
44    required : true
45  },
46  ImageLink : {
47    type : string,
48    required:true
49  }
50 })
51 const TourPackage = mongoose.model("TourPackage",tourPackageSchema);
52 module.exports = TourPackage;

```



```

model.tourPackageReservationsjs
1 const mongoose = require("mongoose");
2 const Schema = mongoose.Schema;
3
4 //create tour package reservation schema
5 const tourPackageReservation = new Schema({
6
7   userName : {
8     type : String,
9     required: true
10  },
11  userEmail : {
12    type : String,
13    required : true
14  },
15  PackageID : {
16    type : String,
17    required : true
18  },
19  packageStatus : {
20    type : String,
21    required : true
22  },
23  dueDate : {
24    type : String,
25  },
26  reservationDate : {
27    type : String,
28    default : Date.now
29  },
30  noofAdults : {
31    type : Number,
32  },
33  noofChilds : {
34    type : Number
35  },
36  paidAmount : {
37    type : Number
38  }
39 })
40
41 const TourPackageReservation = mongoose.model("TourPackageReservation",tourPackageReservation);
42 module.exports = TourPackageReservation;

```

Figure 105

Routes

```
routes.js routes.tourPackage.js routes.tourPackageReservations.js routes.tourPackage.js M

1 const router = require("express").Router();
2 const { json } = require("body-parser");
3 let tourpackage = require("../models/model.tourPackage");
4
5 //create package
6 router.route("/addPackage").post((req,res) => {
7
8     const packageID = req.body.packageID;
9     const packageName = req.body.packageName;
10    const PackageType = req.body.PackageType;
11    const NoDays = Number(req.body.NoofDays);
12    const MealPlan = req.body.MealPlan;
13    const Transportation = req.body.Transportation;
14    const Activities = req.body.Activities;
15    const locationList = req.body.locationList;
16    const PriceForAdult = Number(req.body.PriceForAdult);
17    const PriceForChild = Number(req.body.PriceForChild);
18    const ImageLink = req.body.ImageLink;
19
20
21    const newPackage = new tourpackage({
22        packageID,
23        packageName,
24        PackageType,
25        NoDays,
26        MealPlan,
27        Transportation,
28        Activities,
29        locationList,
30        PriceForAdult,
31        PriceForChild,
32        ImageLink
33    })
34
35    newPackage.save().then(() => {
36        res.json("Package Added")
37    }).catch((err) => {
38        console.log(err);
39    })
40})
41
42
43 //View All
44 router.get('/getAll',(req,res) => {
45     tourpackage.find().exec((err,posts) => {
46         if(err){
47             return res.status(400).json({
48                 error:err
49             });
50         }
51
52         return res.status(200).json({
53             success:true,
54             existingPosts:posts
55         });
56     });
57 });
58
59
60 //view package
61 router.route("/viewPackage").get((req,res) => {
62
63     tourpackage.find().then((packages) => {
64         res.json(packages)
65     }).catch((err)=>{
66         console.log(err);
67     })
68 })
69
70 //view specific package
71 router.get("/get/:id",(req,res) => {
72     let postID = req.params.id;
73
74     tourpackage.findById(postID,(err,post) =>{
75         if(err){
76             return res.status(400).json({success:false, err});
77         }
78         return res.status(200).json({success:true,post});
79     });
80 })
81
82 //update Package
83 router.put('/updatePackage/:id',(req,res) => {
84     tourpackage.findByIdAndUpdate(
85         req.params.id,
86         {
87             $set:req.body
88         },
89         (err,post) => {
90             if(err){
91                 return res.status(400).json({error:err});
92             }
93
94             return res.status(200).json({
95                 success:"Updated Successfully"
96             });
97         }
98     );
99 })
100
101 //delete Package
102 router.delete('/deletePackage/:id',(req,res) => {
103     tourpackage.findByIdAndRemove(req.params.id).exec((err,deletePackage) => {
104         if(err) return res.status(400).json({
105             message:"Delete Unsuccessfully",err
106         });
107
108         return res.json({
109             message:"Delete Successfully",deletePackage
110         });
111     });
112 })
113 module.exports = router;
```

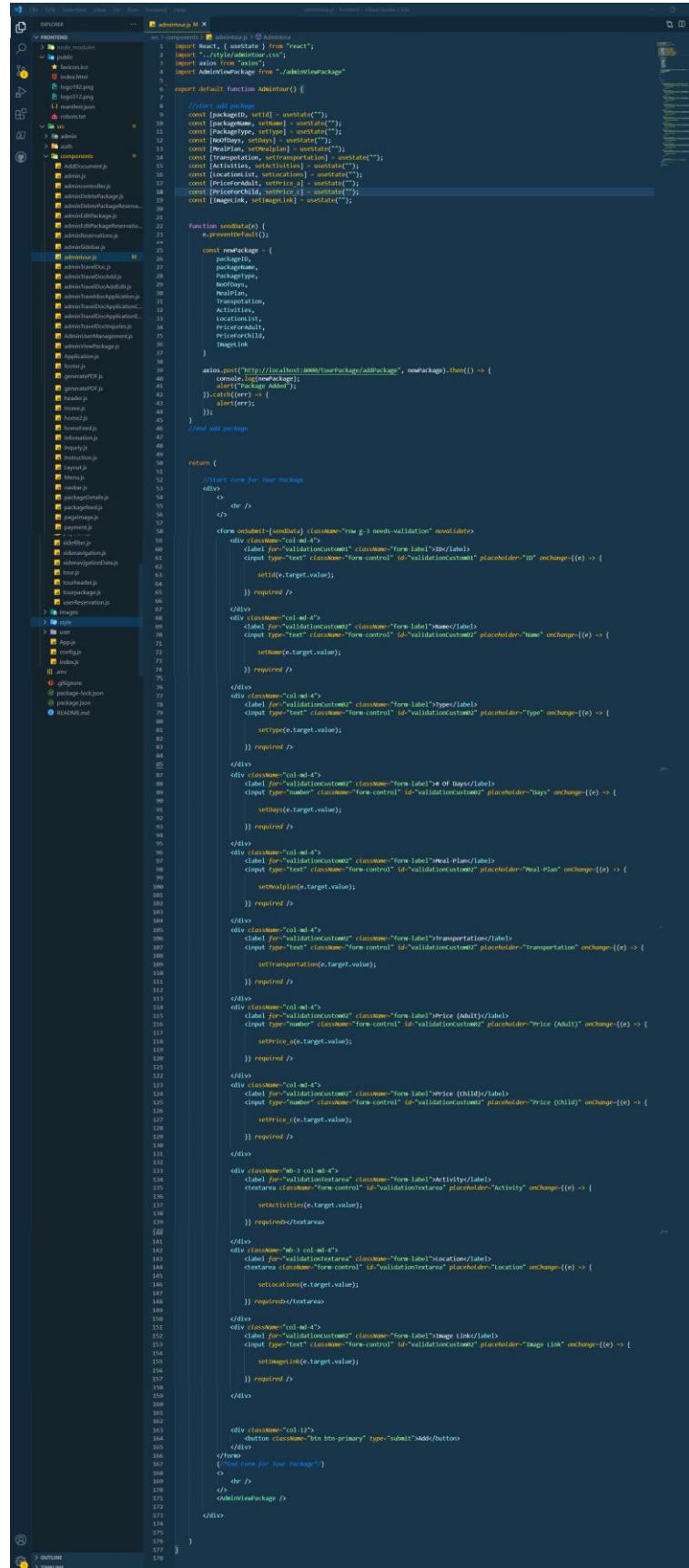
```

File Edit Selection View Go Run Terminal Help routes.tourPackageReservations.js - backend - Visual Studio Code
model.tourPackage.js model.tourPackageReservations.js routes.tourPackage.js M routes.tourPackageReservations.js M X
routes > routes.tourPackageReservations.js > post callback > newReservation
1 const router = require("express").Router();
2 let tourPackageReservation = require("../models/model.tourPackageReservations");
3
4 //create package reservation
5 router.route("/createReservation").post((req,res) => {
6
7     const userName = req.body.userName;
8     const userEmail = req.body.userEmail;
9     const PackageID = req.body.PackageID;
10    const packageStatus = req.body.packageStatus;
11    const dueDate = (req.body.dueDate);
12    const reservationDate = new Date();
13    const noOfAdults = Number(req.body.noOfAdults);
14    const noOfChilds = Number(req.body.noOfChilds);
15    const paidAmount = Number(req.body.paidAmount);
16
17    const newReservation = new tourPackageReservation({
18        userName,
19        userEmail,
20        PackageID,
21        packageStatus,
22        dueDate,
23        reservationDate,
24        noOfAdults,
25        noOfChilds,
26        paidAmount
27    })
28    newReservation.save().then(()=>{
29        res.json("Reservation Added")
30    }).catch((err)>{
31        console.log(err);
32    })
33}
34)
35 //view All
36 router.get('/getAllReservations',(req,res) => {
37     tourPackageReservation.find().exec((err,posts) => {
38         if(err){
39             return res.status(400).json({
40                 error:err
41             });
42         }
43         return res.status(200).json({
44             success:true,
45             existingPosts:posts
46         });
47     });
48 });
49 //view Reservation
50 router.route("/viewReservations").get((req,res)>{
51     tourPackageReservation.find().then((reservations)=>{
52         res.json(reservations)
53     }).catch((err)>{
54         console.log(err);
55     })
56 });
57 //view specific Reservation
58 router.get("/get/:id",(req,res) => {
59     let postID = req.params.id;
60
61     tourPackageReservation.findById(postID,(err,post) =>{
62         if(err) {
63             return res.status(400).json({success:false, err});
64         }
65         return res.status(200).json({success:true,post});
66     });
67 });
68 //update Reservation
69 router.put('/updateReservation/:id',(req,res) => {
70     tourPackageReservation.findByIdAndUpdate(
71         req.params.id,
72         {
73             $set:req.body
74         },
75         (err,post) => {
76             if(err){
77                 return res.status(400).json({error:err});
78             }
79             return res.status(200).json({
80                 success:"Updated Successfully"
81             });
82         }
83     );
84 });
85 });
86 //delete reservation
87 router.delete('/deleteReservation/:id',(req,res) => {
88     tourPackageReservation.findByIdAndRemove(req.params.id).exec((err,deletePackage) => {
89         if(err) return res.status(400).json({
90             message:"Delete Unsuccessfully",err
91         });
92
93         return res.json({
94             message:"Delete Successfully",deletePackage
95         });
96     });
97 });
98 });
99 });
100 module.exports = router;

```

Figure 106

Frontend



```
1 // @flow
2 import React, { useState } from "react";
3 import "./style/administer.css";
4 import axios from "axios";
5 import AdminInterface from "./adminInterface";
6
7 export default function Administer() {
8
9   //state add package
10  const [packageId, setpackageId] = useState("");
11  const [packageName, setpackageName] = useState("");
12  const [packageType, setpackageType] = useState("");
13  const [noOfDays, setnoOfDays] = useState("");
14  const [noOfNights, setnoOfNights] = useState("");
15  const [transpotation, settranspotation] = useState("");
16  const [activities, setactivities] = useState("");
17  const [locations, setlocations] = useState("");
18  const [priceForAdult, setpriceForAdult] = useState("");
19  const [priceForChild, setpriceForChild] = useState("");
20  const [imageLink, setImageLink] = useState("");
21
22  function submitForm() {
23    e.preventDefault();
24
25    const newPackage = {
26      packageId,
27      packageName,
28      packageType,
29      noOfDays,
30      noOfNights,
31      transpotation,
32      activities,
33      locations,
34      priceForAdult,
35      priceForChild,
36      imageLink,
37    };
38
39    axios.get(`http://localhost:5000/admin/package/addPackage`, newPackage).then(() => {
40      console.log("new package");
41      alert("package added");
42      e.preventDefault();
43    });
44  }
45
46  </form>
47
48  return (
49    <div>
50      <h2>Create new package</h2>
51      <div>
52        <div>
53          <label>Name</label>
54          <input type="text" id="name" placeholder="Name" value={packageName} onChange={(e) => {
55            setpackageName(e.target.value);
56          }} required />
57        </div>
58        <div>
59          <label>Type</label>
60          <input type="text" id="type" placeholder="Type" value={packageType} onChange={(e) => {
61            setpackageType(e.target.value);
62          }} required />
63        </div>
64        <div>
65          <label>Days</label>
66          <input type="text" id="days" placeholder="Days" value={noOfDays} onChange={(e) => {
67            setnoOfDays(e.target.value);
68          }} required />
69        </div>
70        <div>
71          <label>Price</label>
72          <input type="text" id="price" placeholder="Price" value={priceForAdult} onChange={(e) => {
73            setpriceForAdult(e.target.value);
74          }} required />
75        </div>
76        <div>
77          <label>Transportation</label>
78          <input type="text" id="transportation" placeholder="Transportation" value={transpotation} onChange={(e) => {
79            settranspotation(e.target.value);
80          }} required />
81        </div>
82        <div>
83          <label>Activities</label>
84          <input type="text" id="activities" placeholder="Activities" value={activities} onChange={(e) => {
85            setactivities(e.target.value);
86          }} required />
87        </div>
88        <div>
89          <label>Locations</label>
90          <input type="text" id="locations" placeholder="Locations" value={locations} onChange={(e) => {
91            setlocations(e.target.value);
92          }} required />
93        </div>
94        <div>
95          <label>Image Link</label>
96          <input type="text" id="imageLink" placeholder="Image Link" value={imageLink} onChange={(e) => {
97            setImageLink(e.target.value);
98          }} required />
99        </div>
100      </div>
101      <div>
102        <button type="submit">Add</button>
103      </div>
104    </div>
105  );
106}
```

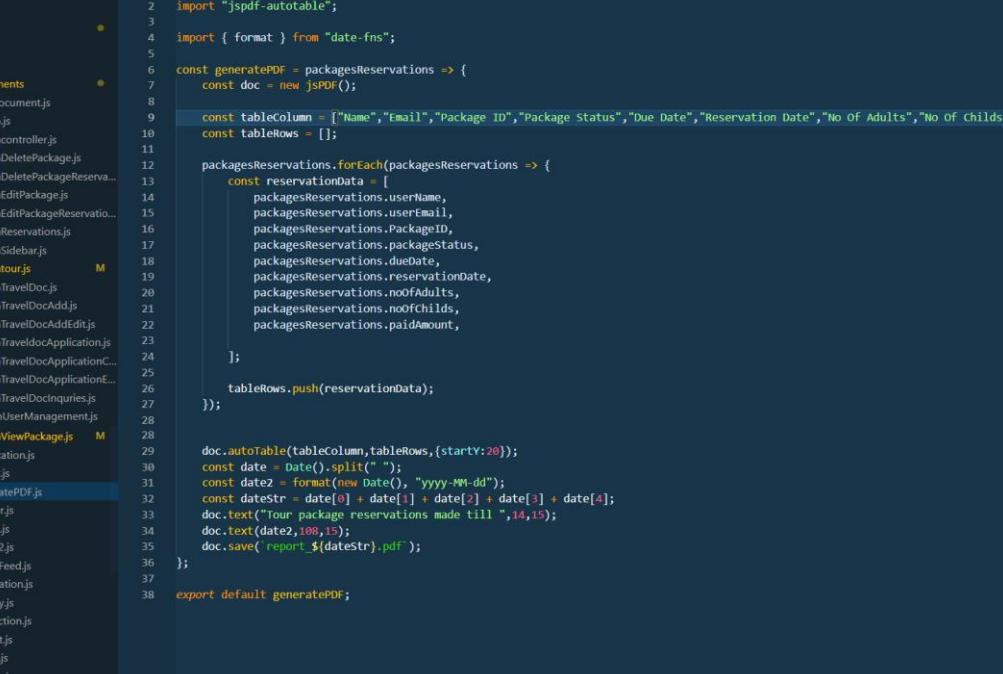
The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows the project structure with a tree view. The "components" folder under "src" is currently selected.
- Editor View:** Displays the code for `adminDeletePackage.js`. The code handles deleting a tour package from a database using Axios and Material-UI's Delete icon.
- Status Bar:** Shows the file path as `adminDeletePackage.js - frontend - Visual Studio Code`.

```
File Edit Selection View Go Run Terminal Help adminDeletePackage.js - frontend - Visual Studio Code

EXPLORER
  ✓ FRONTEND
    > node_modules
    > public
    ✓ src
      > admin
      > auth
      > components
        > images
        > style
        > App.js
        > config.js
        > index.js
      & env
      & .gitignore
      & package.json
      & README.md

src > components > adminDeletePackage.js > adminDeletePackage.js > render
  1 import React, { Component } from 'react'
  2 import './style/adminDeletePackage.css';
  3 import DeleteIcon from '@material-ui/icons/Delete';
  4 import axios from 'axios';
  5
  6 export default class adminDeletePackage extends Component {
  7
  8   componentDidMount() {
  9
  10     const id = this.props.match.params.id;
  11
  12     axios.get(`http://localhost:8000/tourPackage/get/${id}`).then((res) => {
  13       if (res.data.success) {
  14         this.setState({
  15           post: res.data.post
  16         });
  17
  18         console.log(this.state.post);
  19
  20       }
  21     });
  22
  23     onDelete = (id) => {
  24       axios.delete(`http://localhost:8000/tourPackage/deletePackage/${id}`).then((res) => {
  25         alert("Delete Successfully");
  26       })
  27     }
  28   render() {
  29     const id = this.props.match.params.id;
  30     return (
  31       <div className="delete">
  32         <div className="wrapper fadeInDown">
  33           <div id="formContent">
  34             <div style={{ font-size: 80, color: "red" }}>DeleteIcon</div>
  35             <h1>Delete Package !</h1>
  36             </div>
  37             <div id="formFooter">
  38               <form action="/adminTourHome">
  39                 <button type="submit" class="btn btn-danger" onClick={() => this.onDelete(id)}>Delete</button>
  40                 <span>&ampnbsp&ampnbsp&ampnbsp&ampnbsp&ampnbsp&ampnbsp</span>
  41                 <button type="submit" class="btn btn-success">Cancel</button>
  42               </form>
  43             </div>
  44           </div>
  45         </div>
  46       </div>
  47     )
  48   }
  49 }
  50
  51
  52 }
```



```
src > components > generatePDF.js > generatePDF > tableColumn
1 import { autoTable } from 'jspdf';
2 import 'jspdf-autotable';
3
4 import { format } from "date-fns";
5
6 const generatePDF = packagesReservations => {
7   const doc = new jsPDF();
8
9   const tableColumn = ["Name", "Email", "Package ID", "Package Status", "Due Date", "Reservation Date", "No Of Adults", "No Of Childs", "Paid Amount"];
10  const tableRows = [];
11
12  packagesReservations.forEach(packagesReservations => {
13    const reservationData = [
14      packagesReservations.userName,
15      packagesReservations.userEmail,
16      packagesReservations.PackageID,
17      packagesReservations.packageStatus,
18      packagesReservations.dueDate,
19      packagesReservations.reservationDate,
20      packagesReservations.noOfAdults,
21      packagesReservations.noOfChilds,
22      packagesReservations.paidAmount,
23    ];
24
25    tableRows.push(reservationData);
26  });
27
28  doc.autoTable(tableColumn, tableRows, { startY: 20 });
29  const date = Date().split(" ");
30  const date2 = format(new Date(), "yyyy-MM-dd");
31  const dateStr = date[0] + date[1] + date[2] + date[3] + date[4];
32  doc.text("Tour package reservations made till ", 14, 15);
33  doc.text(date2, 108, 15);
34  doc.save(`report_${dateStr}.pdf`);
35
36 };
37
38 export default generatePDF;
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows the project structure under the "FRONTEND" folder. The "packagefeed.js" file is currently selected.
- Code Editor:** The main area displays the code for "packagefeed.js". The code is a class component named "packagefeed" that extends "Component". It includes methods for "componentDidMount" (which calls "retrievePackages"), "filterData" (which filters posts by a search key), and "handleSearchArea" (which handles a search input). The render method contains a search bar and a list of posts. Each post is rendered as a card with a thumbnail image, title, and a link to its detail page.
- Status Bar:** At the bottom, it shows "packagefeed.js - front-end - Visual Studio Code".

```
src > components > packagefeed.js > ...
1 import "./style/packagefeed.css";
2 import axios from "axios";
3 import React, { Component } from 'react';
4
5 export default class packagefeed extends Component {
6
7   constructor(props) {
8     super(props);
9
10    this.state = {
11      posts: []
12    };
13
14  }
15
16  componentDidMount() {
17    this.retrievePackages();
18  }
19
20  retrievePackages() {
21    axios.get("http://localhost:8000/tourPackage/getAll").then(res => {
22      if (res.data.success) {
23        this.setState({
24          posts: res.data.existingPosts
25        });
26      }
27    });
28    console.log(this.state.posts);
29  }
30
31  filterData(posts, searchKey) {
32    const result = posts.filter((post) =>
33      post.packageName.toLowerCase().includes(searchKey) || post.packageName.includes(searchKey)
34    );
35    this.setState({ posts: result });
36  }
37
38  handleSearchArea = (e) => {
39    const searchKey = e.currentTarget.value;
40    console.log(e.currentTarget.value);
41    axios.get("http://localhost:8000/tourPackage/getAll").then(res => {
42      if (res.data.success) {
43        this.filterData(res.data.existingPosts, searchKey)
44      }
45    });
46  }
47
48  render() {
49    return (
50      <div>
51        <div className="feed">
52          <div className="container">
53            <h3 style={{text-align:"center",marginTop:20}}>
54              Latest Packages
55            </h3>
56
57            <br />
58            <div className="searchBar">
59              <div className="searchBar">
60                <div className="row3 searchBar">
61                  <div id="custom-search-input">
62                    <div className="input-group col-md-12 search">
63                      <input
64                        type="text"
65                        className="search-query form-control"
66                        placeholder="Search"
67                        name="searchQueue"
68                        onChange={this.handleSearchArea}
69                      />
70                      <span className="input-group-btn">
71                        <button className="btn btn-danger" type="button" style={{marginLeft:5,marginTop:10}}>
72                          <span className="glyphicon glyphicon-search"></span>
73                        </button>
74                      </span>
75                    </div>
76                  </div>
77                </div>
78              </div>
79            </div>
80            <div className="row">
81              {this.state.posts.map(post => (
82                <div className="col-md-4">
83                  <div className="card mb-4 text-white bg-dark">
84                    <img className="card-img-top" src={post.ImageLink} alt="Card image cap" />
85                    <div className="card-body">
86                      <h5 className="card-title">{post.packageName}</h5>
87                      <p style={{ color: "#E8C0BE" }} className="card-text">Type : {post.PackageType} </p>
88                      <a href={`/package/${post._id}} style={{ background-color: "#BCC8EE", color: "black" }} className="card-link">View Details</a>
89                    </div>
90                  </div>
91                ))
92              )
93            </div>
94          <br />
95        </div>
96      </div>
97    )
98  }
99
100 }
101
102 }
```


Figure 107

2.3.3.7 Insurance Management

Models

```

File Edit Selection View Go Run Terminal Help
OPEN EDITORS
backend > models > modelInsuranceType.js ...
Signin.js Profile.js ManageUsers.js modelInsuranceType.js
backend > models > modelInsuranceType.js ...
1 const mongoose = require('mongoose');
2 const Schema = mongoose.Schema;
3
4
5 const insuranceTypeSchema = new Schema({
6   insuranceID : {
7     type : String,
8     required : true
9   },
10  insuranceType : {
11    type : String,
12    required : true
13  },
14  companyName : {
15    type : String,
16    required : true
17 },
18  price : {
19    type : Number,
20    required : true
21 },
22  description : {
23    type : String,
24    required : true
25 },
26  ImageLink : {
27    type : String,
28    required: true
29 }
30
31 })
32
33
34 const InsuranceType = mongoose.model("InsuranceType",insuranceTypeSchema);
35
36 module.exports = InsuranceType;

```

Figure 108

Routes

```

File Edit Selection View Go Run Terminal Help
OPEN EDITORS
backend > routes > routes.insuranceType.js ...
Signin.js Profile.js ManageUsers.js routes.insuranceType.js
backend > routes > routes.insuranceType.js ...
1 const router = require("express").Router();
2 const { json } = require("body-parser");
3 let insurancetype = require("../models/model.InsuranceType");
4
5 router.route("/addInsuranceType").post((req,res) => {
6
7
8  const insuranceID = req.body.insuranceID;
9  const insuranceType = req.body.insuranceType;
10 const companyName = req.body.companyName;
11 const price = Number(req.body.price);
12 const description = req.body.description;
13 const ImageLink = req.body.ImageLink;
14
15  const newInsuranceType = new insurancetype({
16    insuranceID,
17    insuranceType,
18    companyName,
19    price,
20    description,
21    ImageLink
22  })
23
24  newInsuranceType.save().then(() => {
25    res.json("Insurance Type Added")
26  }).catch((err) => {
27    console.log(err);
28  })
29
30})
31
32
33
34 //View All Insurance
35 router.get('/viewAllInsuranceType',(req,res) => {
36   insurancetype.find().exec((err,posts) => {
37     if(err){
38       if(err){
```

```

File Edit Selection View Go Run Terminal Help
routes/InsuranceType.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

OPEN EDITORS
  Signings frontendUser
  Profilejs frontendUser
  ManageUsers.js frontendUser
  routes/InsuranceType.js
  routes/InsuranceType.js (background)

  TRAVEL-AGENCY-TOUR-PLANNING-MAN...
    backend
      controllers
      helpers
      models
      node_modules
      routes
        accommodation_reservation.js
        accommodations.js
        airline_tickets.js
        airnicket_reservation.js
        auth.js
        bus_tickets.js
        busticket_reservations.js
        car_reservations.js
        cars.js
        event_reservations.js
        events.js
        grounds.js
        guide_reservations.js
        guides.js
        payments.js
        railway_tickets.js
        routes/InsuranceReserv...
        routes/InsuranceType.js
        routes/tourPackage.js
        routes/tourPackageReserv...
        ticketypes.js
        trainTicket_reservation.js
      ...
    ...
  ...

if(err){
  return res.status(400).json({
    error:err
  });
}

return res.status(200).json({
  success:true,
  existingPosts:posts
});

});

//view all Insurance
router.route("/viewInsuranceType").get((req,res) => {
  insurancetype.find().then((packages) => {
    res.json(packages)
  }).catch((err)=>{
    console.log(err);
  })
});

router.get("/get/:id", (req,res) => {
  let postID = req.params.id;

  insurancetype.findById(postID,(err,post) => {
    if(err) {
      return res.status(400).json({success:false, err});
    }
    return res.status(200).json({success:true,post});
  });
});

//update Package
router.put('/UpdateInsuranceType/:id', (req,res) => {
  insurancetype.findByIdAndUpdate(
    req.params.id,
    {
      $set:req.body
    },
    (err,post) => {
      if(err) {
        return res.status(400).json({error:err});
      }

      return res.status(200).json({
        success:"Updated Successfully"
      });
    }
  );
});

//delete Package
router.delete('/deleteInsuranceType/:id', (req,res) => {
  insurancetype.findByIdAndRemove(req.params.id).exec((err,deleteInsuranceType) => {
    if(err) return res.status(400).json({
      message:"Delete Unsuccessfully",err
    });

    return res.json({
      message:"Delete Successfully",deleteInsuranceType
    });
  });
});

module.exports = router;

```

```

File Edit Selection View Go Run Terminal Help
routes/InsuranceType.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

OPEN EDITORS
  Signings frontendUser
  Profilejs frontendUser
  ManageUsers.js frontendUser
  routes/InsuranceType.js
  routes/InsuranceType.js (background)

  TRAVEL-AGENCY-TOUR-PLANNING-MAN...
    backend
      controllers
      helpers
      models
      node_modules
      routes
        accommodation_reservation.js
        accommodations.js
        airline_tickets.js
        airnicket_reservation.js
        auth.js
        bus_tickets.js
        busticket_reservations.js
        car_reservations.js
        cars.js
        event_reservations.js
        events.js
        grounds.js
        guide_reservations.js
        guides.js
        payments.js
        railway_tickets.js
        routes/InsuranceReserv...
        routes/InsuranceType.js
        routes/tourPackage.js
        routes/tourPackageReserv...
        ticketypes.js
        trainTicket_reservation.js
      ...
    ...
  ...

//update Package
router.put('/UpdateInsuranceType/:id', (req,res) => {
  insurancetype.findByIdAndUpdate(
    req.params.id,
    {
      $set:req.body
    },
    (err,post) => {
      if(err) {
        return res.status(400).json({error:err});
      }

      return res.status(200).json({
        success:"Updated Successfully"
      });
    }
  );
});

//delete Package
router.delete('/deleteInsuranceType/:id', (req,res) => {
  insurancetype.findByIdAndRemove(req.params.id).exec((err,deleteInsuranceType) => {
    if(err) return res.status(400).json({
      message:"Delete Unsuccessfully",err
    });

    return res.json({
      message:"Delete Successfully",deleteInsuranceType
    });
  });
});

module.exports = router;

```

Figure 109

Frontend

```
frontend > src > components > JS adminViewInsuranceType.js > AdminViewInsuranceType > insurancetype.map() callback
  1 import React, { useState, useEffect } from 'react';
  2 import './style/adminViewPackage.css';
  3 import axios from "axios";
  4 import SearchIcon from '@material-ui/icons/Search';
  5
  6 function AdminViewInsuranceType() {
  7
  8     const [insurancetype, setInsurancetype] = useState([]);
  9
 10    useEffect(() => {
 11        function getInsurancetype() {
 12            axios.get("http://localhost:8000/InsuranceType/viewInsuranceType").then((res) => {
 13                console.log(res);
 14                setInsurancetype(res.data);
 15            }).catch((err) => {
 16                alert(err.message);
 17            })
 18        }
 19        getInsurancetype();
 20    }, [])
 21
 22
 23    return (
 24        <div style={{marginLeft:"400px",fontSize:"20px"}}>
 25
 26
 27            <div>
 28                <div>
 29                    <div>
 30                        <h1>Insurance Types
 31                        <a href="/adminInsurance/adminAddInsuranceType"><button className="btn btn-primary" type="submit" style={{float:'right'}}>Add New Type</button></a></h1>
 32                    </div>
 33                </div>
 34            </div>
 35
 36            <div class="container">
 37
 38                <div class="row col-md-6 col-md-offset-2 custyle tabeldiv">
 39
 40                    <table class="table table-striped custab column">
 41                        <thead>
 42                            <tr>
 43                                <th>Insurance ID</th>
 44                                <th>Insurance Type</th>
 45                                <th>Company Name</th>
 46                                <th>Price</th>
 47                                <th>Description</th>
 48                                <th class="text-center">Action</th>
 49                            </tr>
 50                        </thead>
 51                        <tbody>
 52                            {
 53                                insurancetype.map(post => [
 54                                    <tr>
 55                                        <td>{post.insuranceID}</td>
 56                                        <td>{post.insuranceType}</td>
 57                                        <td>{post.companyName}</td>
 58                                        <td>{post.price}</td>
 59                                        <td>{post.description}</td>
 60                                        <td class="text-center"><a class='btn btn-info btn-xs' href={`/adminInsurance/editType/${post._id}`}>
 61                                            <span class="glyphicon glyphicon-edit"></span> Edit</a>
 62                                            <a href={`/adminInsurance/deleteType/${post._id}`} class="btn btn-danger btn-xs"><span class="glyphicon glyphicon-trash"></span> Delete</a>
 63                                        </td>
 64                                    ])}
 65                                </tbody>
 66                            </table>
 67                        </div>
 68
 69                    </div>
 70
 71                </div>
 72            </div>
 73        </div>
 74    )
 75
 76    export default AdminViewInsuranceType
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
```

File Edit Selection View Go Run Terminal Help

adminAddInsuranceType.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

```
frontend > src > components > adminAddInsuranceType.js U x
```

```
Signinjs Profilejs ManageUserjs adminAddInsuranceType.js U x
```

```
1 import React, { useState } from 'react';
2 import './style/adminTour.css';
3 import axios from 'axios';
4
5 import './style/admin.css';
6 import ImgLogo from '../images/logo.png';
7 import HouseIcon from '@material-ui/icons/House';
8 import ConfirmationNumberIcon from '@material-ui/icons/ConfirmationNumber';
9 import LocalTaxiIcon from '@material-ui/icons/localTaxi';
10 import ExploreIcon from '@material-ui/icons/Explore';
11 import DescriptionIcon from '@material-ui/icons/Description';
12 import SecurityIcon from '@material-ui/icons/Security';
13 import EventIcon from '@material-ui/icons/Event';
14 import GroupIcon from '@material-ui/icons/Group';
15
16
17 export default function AdminAddInsuranceType() {
18
19     const [insuranceID, setId] = useState("");
20     const [insuranceType, setType] = useState("");
21     const [companyName, setCompany] = useState("");
22     const [price, setPrice] = useState("");
23     const [description, setDescription] = useState("");
24     const [ImageLink, setImageLink] = useState("");
25
26     function sendData(e) {
27         e.preventDefault();
28
29         const newtype = {
30             insuranceID,
31             insuranceType,
32             companyName,
33             price,
34             description,
35             ImageLink
36         }
37
38     }
39 }
```

File Edit Selection View Go Run Terminal Help

adminAddInsuranceType.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

```
frontend > src > components > adminAddInsuranceType.js U x
```

```
Signinjs Profilejs ManageUserjs adminAddInsuranceType.js U x
```

```
<div>
    <div>
        <div>
            <div>
                <div>
                    <div>
                        <div>
                            <div>
                                <div>
                                    <div>
                                        <div>
                                            <div>
                                                <div>
                                                    <div>
                                                        <div>
                                                            <div>
                                                                <div>
                                                                    <div>
                                                                        <div>
                                                                            <div>
                                                                                <div>
                                                                                    <div>
                                                                                        <div>
                                                                                            <div>
                                                                                                <div>
                                                                                                    <div>
                                                                                                        <div>
                                                                                                            <div>
                                                                                                                <div>
                                                                                                                    <div>
                                                                                                                        <div>
                                                                                                                            <div>
                                                                                                                                <div>
                                                                                                                                    <div>
                                                                ................................................................
```

File Edit Selection View Go Run Terminal Help

adminAddInsuranceType.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

```
 125      </div>
 126      <hr />
 127    </div>
 128  </div>
 129</div>
 130</div>
 131</div>
 132</div>
 133</div>
 134</div>
 135</div>
 136</div>
 137</div>
 138</div>
 139</div>
 140</div>
 141</div>
 142</div>
 143</div>
 144</div>
 145</div>
 146</div>
 147</div>
 148</div>
 149</div>
 150</div>
 151</div>
 152</div>
 153</div>
 154</div>
 155</div>
 156</div>
 157</div>
 158</div>
```

File Edit Selection View Go Run Terminal Help

adminAddInsuranceType.js - Travel-Agency-Tour-Planning-Management-System - Visual Studio Code

```
 197      </div>
 198      </div>
 199      </div>
 200</div>
 201      </div>
 202    </div>
 203  </div>
 204</div>
 205</div>
 206</div>
 207</div>
 208</div>
 209</div>
 210</div>
 211</div>
 212</div>
 213</div>
 214</div>
 215</div>
 216</div>
 217</div>
 218</div>
 219</div>
 220</div>
 221</div>
 222</div>
 223</div>
 224</div>
 225</div>
 226</div>
 227</div>
 228</div>
 229</div>
 230</div>
 231</div>
 232</div>
 233</div>
```



```

frontend > src > components > InsuranceFeed.js ...
1  import React, { Component } from "react";
2  //import React, { useState, useEffect } from 'react'
3  import "../style/packagefeed.css";
4  import axios from "axios";
5
6
7
8  export default class InsuranceFeed extends Component {
9
10    constructor(props) {
11      super(props);
12
13      this.state = {
14        posts: []
15      };
16    }
17
18    componentDidMount() {
19      this.retrieveInsurance();
20    }
21
22    retrieveInsurance() {
23      axios.get("http://localhost:8000/InsuranceType/viewAllInsuranceType").then(res => {
24        if (res.data.success) {
25          this.setState({
26            posts: res.data.existingPosts
27          });
28        }
29      });
30      console.log(this.state.posts);
31    }
32
33    filterData(posts, searchKey) {
34      const result = posts.filter((post) =>
35        post.insuranceType.toLowerCase().includes(searchKey) || post.insuranceType.includes(searchKey) ||
36        post.companyName.toLowerCase().includes(searchKey) || post.companyName.includes(searchKey)
37      );
38
39      this.setState({ posts: result })
40    }
41
42    handleSearchArea = (e) => {
43      const searchkey = e.currentTarget.value;
44      console.log(e.currentTarget.value);
45      axios.get("http://localhost:8000/InsuranceType/viewAllInsuranceType").then(res => {
46        if (res.data.success) {
47          this.filterData(res.data.existingPosts, searchKey)
48        }
49      });
50    }
51
52    render() {
53      return (
54        <div style={{fontSize:"20px"}}>
55          <div className="feed">
56            <div className="container">
57
58              <h3 style={{fontSize:"20px",textAlign:"center",marginTop:20}}>
59                Insurance
60              </h3>
61              <div class="searchBar">
62                <div class="searchBar">
63                  <div id="row3 searchBar">
64                    <div id="custom-search-input">
65                      <div class="input-group col-md-12 search">
66                        <input
67                          type="text"
68                          class=" form-control"
69                          placeholder="Search"
70                          name="searchQueue"
71                          onChange={this.handleChangeSearchArea}
72                        />
73                        <span class="input-group-btn">
74                          <button class="btn btn-danger" type="button" style={{marginLeft:5,marginTop:10}}>
75                            <span class="glyphicon glyphicon-search"></span>
76                          </button>
77                        </span>
78                      </div>
79                    </div>
80                  </div>
81                </div>
82              </div>
83
84              <hr />
85
86              <div className="row">
87                {this.state.posts.map(post => (
88                  <div className="col-md-4">
89                    <div className="card mb-4 text-white bg-dark">
90                      <img className="card-img-top" src={post.ImageLink} alt="Card image cap" />
91                      <div className="card-body">
92                        <h5 style={{fontSize:"20px"}} className="card-title">{post.insuranceType}</h5>
93                        <p style={{fontSize:"20px",color: "#C8C0E" }} className="card-text">Company : {post.companyName} </p>
94                        <a href={"/Insurance/" + post._id} style={{fontSize:"20px", backgroundColor: "#000000", color: "black" }}>View</a>
95                      </div>
96                    </div>
97                  ))
98                )
99              </div>
100
101              <hr />
102            </div>
103          </div>
104
105          <hr />
106        </div>
107      </div>
108    )
109  }
110
111}
112

```

Figure 110

2.3.3.8 Event Management

Routes

```

const router = require("express").Router();
const json = require("body-parser");
let Event = require("../models/events");

router.route("/add").post((req,res)=> {
    const eventType = req.body.eventType;
    const location = req.body.location;
    const date = req.body.date;
    const price = Number(req.body.price);

    const newEvent = new Event({
        eventType,
        location,
        date,
        price
    });

    newEvent.save().then(()=>{
        res.json("event added")
    }).catch((err)=>{
        console.log(err);
    })
})

```

```

router.route("/").get((req,res)=>{
    Event.find().then((events)=>{
        res.json(events)
    }).catch((err)=>{
        console.log(err)
    })
});

router.route("/update/:id").put(async(req,res)=>{
    let userId = req.params.id;
    const {eventType,location,date,price} = req.body;

    const updateEvent = {
        eventType,
        location,
        date,
        price
    }

    const update = await Event.findByIdAndUpdate(userId,updateEvent).then(()=>{
        res.status(200).send({status: "event updated"})
    }).catch((err)=>{
        console.log(err);
        res.status(500).send({status: "error updating data",error: err.message});
    })
})

```

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER**: Shows the project structure under "OPEN EDITORS". The "backend" folder contains "routes" which includes "events.js". Other files like "reportGenerator2.js", "addevent.js", and "addevent.js" are also listed.
- EDITOR**: The "events.js" file is open in the editor. The code handles routes for events, including deletion and retrieval. It uses async/await for database operations and handles errors.
- STATUS BAR**: Shows the file path "backend / routes > events.js > put() callback", line 48, column 50, and other settings like "Spaces: 4", "UTF-8", "CRLF", and "JavaScript".

```
66     })
67   }
68 }
69
70 router.route("/delete:id").delete(async (req,res)=>{
71   let userId = req.params.id;
72
73   await Event.findByIdAndDelete(userId).exec(deletenewEvent).then(() => {
74     res.status(200).send({status: "Event deleted"});
75   }).catch((err)=>{
76     console.log(err.message);
77     res.status(500).send({status: "error delete data",error: err.message});
78   })
79 }
80
81 router.route("/get:id").get(async (req,res)=>{
82   let userId = req.params.id;
83   const user = await Event.findById(userId)
84   .then((Event)=>{
85     res.status(200).send({status: "event fetched",Event})
86   }).catch((err)=>{
87     console.log(err.message);
88     res.status(500).send({status: "Error with get event"});
89   })
90 }
91
92 module.exports = router;
```

Figure 111

Frontend

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER**: Shows the project structure under "OPEN EDITORS". The "frontend" folder contains "src" which includes "components" and "admin". Inside "components", there is an "addevent.js" file.
- EDITOR**: The "addevent.js" file is open in the editor. It defines a component "Addevent" that handles state for event type, location, date, and price. It has a "sendData" function that sends a POST request to "http://localhost:8000/add" using axios.
- STATUS BAR**: Shows the file path "frontend / src / components > addevent.js > Addevent > sendData", line 48, column 9, and other settings like "Spaces: 4", "UTF-8", "CRLF", and "JavaScript".

```
27
28
29 function Addevent() {
30
31   const [eventType, setEventType] = useState("");
32   const [location, setLocation] = useState("");
33   const [date, setDate] = useState("");
34   const [price, setPrice] = useState("");
35
36
37   function sendData(e){
38
39     e.preventDefault();
40
41     const newEvent ={
42       eventType,
43       location,
44       date,
45       price
46
47     }
48
49
50     axios.post("http://localhost:8000/add",newEvent).then(()=>{
51       alert("event added")
52
53     }).catch((err)=>{
54       alert(err)
55     })
56
57   }
58
59 }
```

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER**: Shows the project structure under "TRAVEL-AGENCY-TOUR-PLANNING-...". The "components" folder is expanded, showing files like AccommodationBookPage.js, Admin_Airline_page.js, and addevent.js.
- OPEN EDITORS**: Three editors are open:
 - reportGenerator2.js (frontend/src/components)
 - events.js (frontend/src/components)
 - addevent.js (frontend/src/components)
- CODE**: The "addevent.js" editor is active, displaying the following code:

```
frontend > src > components > addevent.js > Addevent
62     const errors = setErrors(eventType,location,date,price) ->
63         this.setState({errors:errors});
64     return Object.values
65 }; /*

66
67
68
69 /* function Alldata(){ */
70
71 const [Events , setEvents] = useState([]);
72
73 useEffect(() => {
74     //function getEvents(){
75     axios.get("http://localhost:8000/")
76         .then((res)=>{
77             setEvents(res.data);
78
79         }).catch((err)=>{
80             alert(err.message);
81         })
82
83     },
84     //        getEvents();
85
86     //    },
87     [1]
88
89
90
91
92
93 // }
```

The code uses React hooks like useState and useEffect to manage state and fetch data from a local API endpoint.

The screenshot shows a code editor interface with the following details:

- Explorer View:** Shows a tree structure of files and folders. The 'components' folder under 'src/admin' contains several files, including 'AccommodationBookPage.js', 'AccommodationMainPage.js', 'Add_New_Car_And_Airport_Taxis.js', 'Add_New_Guide_Bodyjs', 'AddAccommodation.js', 'AddDocument.js', and 'addevent.js'. The 'addevent.js' file is currently selected.
- Editor View:** The main area displays the content of the 'addevent.js' file. The code is written in JavaScript and includes logic for deleting events via an axios delete request to 'http://localhost:8000/delete/\${id}'.
- Status Bar:** At the bottom, it shows 'master' as the current branch, and the status bar indicates 'Ln 48, Col 9' with 'Spaces: 4' and 'UTF-8' encoding.

```
// axios.delete(`post/delete/${id}`).then(() => {
// axios.delete(`http://localhost:8000/delete/${id}`).then(() => {
function DeleteEvent(id) {
  axios.delete(`http://localhost:8000/delete/${id}`).then(() => {
    alert("Successfully deleted");
  }).catch((err) => {
    alert(err);
  });
}
/*function onDelete(id){
onDelete = (id)=>{
  alert(id);
  axios.delete(`post/delete/${id}`).then((res)=>{
    alert("Deleted sucessfully");
  });
};
*/
return(

```

Figure 112

2.4 Testing

2.4.1 User Management

Group ID: ITP2021_S2_MTR_G05									
Project title: Travel Agency and Tour Planning									
Testing Function: User Registration and Sign-in into the system.									
Test case ID: TC001		Test designed by: Reg. No- IT20142964 Name- Dulakshi Hansini R.M							
Test priority (High/Medium/Low):		High							
Test Description: User must create a user account before sign-in to the system. For that user must fill an application in registration page. Then user can log into the system with given email and password for the user registration.									
Preconditions (if there are any): User must create a valid user account before sign-in.									
Test steps:									
<p>User must visit the web site. Click the button Sign-up in the header. Fill User registration form. Click REGISTER button after filling the form. Click Sign-in button. Enter Email and password given in the registration. Click Sign-out button to sign-out from account.</p>									
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments				
TC001	Check user enter an invalid password or email for the valid user account login.	System sends an error message as “Email and password don’t match”	Error message was showed as Email and password don’t match	Pass	Error message was shown and the test case worked as described.				

Testing Function: Admin can delete and view user profiles

Test case ID: TC002	Test designed by: Reg. No- IT20142964 Name- Dulakshi Hansini R.M									
Test priority (High/Medium/Low):	Medium									
Test Description: Admin can log in to the system by giving correct email address and password. After login to the system admin can view all user profile details in manage user page by clicking user management button which is in the side bar.										
Preconditions (if there are any): Admin must sign into the system by giving correct email and password and move to admin panel										
Test steps:										
Step1: Admin must login to the website. Step2: Admin should Click user management button which is in the admin panel sidebar. Step3: It navigate to Admin dashboard and click view user profiles button Step4: Admin can see all the users that registered into the system. Step5: Admin can click delete button, if admin wants to delete any user profile in the system Step6: It will display the confirmation message that confirm the deletion Step7: Admin wants to confirm it Step8: Therefore, the record will be deleted and updated the user profiles details										
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments					
TC002	Click on delete button and check whether the user profile deletion is successful	Admin can see “Successfully deleted” message	Admin can see “Successfully deleted” message	Pass	Admin can delete user profile is successfully completed					

2.4.2 Accommodation reservation Management

Project ID:									
Project Name: Travel Agency Tour Planning Management System									
Testing Function: Book an accommodation									
Test case ID:TC003		Test case Designed By: ID: IT20204648 Name: Nethu Nipuna M.							
Test Priority (High/Medium/Low)		High							
Test Description: registered customer who visit the website can reservation accommodations on their own. Then total cost can be calculated. You can easily book the accommodation through the website.									
Test steps: Step1: Register to the web site Step2: log in to the web site Step3: go to accommodation page Step4: customer can search accommodations what he or she want Step5: click book now Step6: fill the given form and click book now button									
Test ID	Test Inputs	Expected output	Actual output	Result (pass/fail)	Comments				
TC003	Name, email, booking date, due date, No of accommodations, No of adult, No of child	Show total price and when press book now button system shows success message.	Success message displayed	pass	Booking function Successfully completed				

Testing Function: insert an accommodation									
Test case ID:TC004		Test case Designed By: ID: IT20204648 Name: Nethu Nipuna M.							
Test Priority (High/Medium/Low)		High							
Test Description: registered admin can log in to admin panel. Admin can view all accommodations and add new one, edit or delete accommodations from the database									
Test steps: Step1: log in to the admin side Step2: go to accommodation admin page Step3: go to add new accommodation page Step4: fill the form Step5: click submit button Step6: newly added accommodation must display in all accommodation page									
Test ID	Test Inputs	Expected output	Actual output	Result (pass/fail)	Comments				
TC004	Accommodation name, location, phone number, owner, price, type, image link	Display success message and show in all accommodation page	Success message displayed. Newly added item show in the all accommodation page	pass	Add new accommodation function successfully completed				

2.4.3 Reservation of Ticket Management

Group ID: ITP2021_S2_MTR_G05										
Project title: Travel Agency and Tour Planning										
Testing Function: Airline Tickets Booking										
Test case ID: TC005	Test designed by: Reg. No- IT20046552 Name- Abewardhanage S. R.D									
Test priority (High/Medium/Low):	High									
Test Description: Registered Customer who visit the website they can easily Find Airline Reservation page and book the Airline Ticket Successfully. Customer Should pay their payment using correct payment details through using website.										
Preconditions (if there are any): User must sign up or sign in and move to Ticket Booking home page.										
Test steps:										
<p>Step1: User must visit the web site.</p> <p>Step2: User must Register to the website</p> <p>Step3: User must login to the website</p> <p>Step4: User should Click Tickets button in navigation bar.</p> <p>Step5: User can see Four types of ticket in Ticket Home page and can Click Airline Ticket Book Button.</p> <p>Step6: User can see Timetable details and can search using starting places, destination places and flight numbers.</p> <p>Step7: After getting Idea, User should Fill airline booking form correctly and click book now button.</p> <p>Step8: User must fill the payment form.</p> <p>Step9: User must click the pay now button to confirm the Ticket Booking.</p>										
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments					
TC006	Check User Fill the Airline booking Form Details and payment details with Valid Data.	User can See “Thank you for your Booking. Welcome” message.	User can See “Thank you for your Booking. Welcome” message	Pass	Booking Function Successfully Completed.					

Testing Function: Insert Bus Details									
Test case ID: TC007		Test designed by: Reg. No- IT20046552 Name- Abewardhanage S. R.D							
Test priority (High/Medium/Low):		High							
Test Description: Registered admin can log in to admin panel using correct email address and password. Admin can add new bus details to a Bus Timetable and admin can view, update, and delete bus details from database.									
Preconditions (if there are any): Admin must sign up or sign in and move to admin panel.									
Test steps:									
<p>Step1: Admin must Register to the website.</p> <p>Step2: Admin must login to the website.</p> <p>Step3: Admin should Click Tickets button in Admin Panel.</p> <p>Step4: Admin can see Insert Bus Page Button. Click the button and can go to an insert Bus page.</p> <p>Step5: Admin should Fill Insert Bus Form correctly and click Add now button.</p> <p>Step6: Admin can see new added Bus Details in the Table and can search using starting places, destination places. As well as those details updated in Bus Timetable.</p>									
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments				
TC006	Check Admin Fill the Insert bus Form Details with Valid Data.	Admin can See “Bus added Successfully” message.	Admin can See “Bus added Successfully” message	Pass	Add new Bus Function Successfully Completed.				

2.4.4 Reservation of Ground Resource Management

Group ID: ITP2021_S2_MTR_G05										
Project title: Travel Agency and Tour Planning										
Testing Function: Add New Travel Guide										
Test case ID: TC007	Test designed by: Reg. No- IT20186906 Name- Tharuka Gayashan F.									
Test priority (High/Medium/Low):	Medium									
Test Description: Registered admin can log in to admin panel using correct email address and password. The administrator can add a new travel guide to the system. Go to the guide management page and the guide can be added to the system by entering valid information.										
Preconditions (if there are any): Must be an admin and must be Login to the System.										
Test steps:										
Step1: Admin must visit to the web application. Step2: Admin must Register to the website Step3: Admin must login to the website Step4: Admin should click Guide Management button in admin panel. Step5: Admin must enter the correct information in the form. Step6: Admin should click the insert button. Step7: Admin can see the “Guide added successfully” message.										
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments					
TC007	Inputs that belong to validations must be provided the admin	Admin can see “Guide added successfully” message.	Admin can see “Guide added successfully” message.	Pass	Ground Resource function Successfully Completed.					

Testing Function: Add New Travel Guide					
Test case ID: TC008	Test designed by: Reg. No- IT20186906 Name- Tharuka Gayashan F.				
Test priority (High/Medium/Low):	High				
Test Description:	Registered customer can log in to system using correct email address and password. The customer can make a new car reservation. For that customer has to go to the car profile and give the correct details. Then click the book now button.				
Preconditions (if there are any):	User must sign up or sign in to the system.				
Test steps:	Step1: Customer must Register to the website. Step2: Customer must login to the website. Step3: Customer should Click Ground reservation button in navigation bar. Step4: The customer is given 3 options. And the customer must click the car button. Step5: Customer must enter the correct information in the form. Step6: User should click the Book Now button. Step7: Customer can see the "Car reservation successfully" message.				
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments
TC008	Inputs that belong to validations must be provided the customer	Customer can see "Car reservation successfully" message.	Customer can see "Car reservation successfully" message.	Pass	Car Reservation Function Successfully Completed.

2.4.5 Travel Document Management

Group ID: ITP2021_S2_MTR_G05										
Project title: Travel Agency and Tour Planning										
Testing Function: Add Travel Document										
Test case ID: TC009	Test designed by: Reg. No- IT20172350 Name- M.P.O.M Gomes.									
Test priority (High/Medium/Low):	Medium									
Test Description: Registered customer can log in to system using correct email address and password. The Customer can add travel Document to the system from Travel document function. Go to the Travel Document Add page and the Travel Document can be added to the system by entering valid information.										
Preconditions (if there are any): Must be a Registered user and must be Login to the System.										
Test steps:										
Step1: Customer must visit to the web application. Step2: Customer must Register to the website Step3: Customer must login to the website Step4: Customer should click Document button in home navigation bar. Step5: Customer must enter the correct information in the form in Travel Document Add page. Step6: Customer should click the Submit button.										
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments					
TC009	Inputs valid details into Travel document Add form.	Customer can see “Travel Document added successfully” message.	Customer can see “Travel Document added successfully” message.	Pass	Travel Document Add function Successfully Completed.					

Testing Function: Update User's Travel Document									
Test case ID: TC010		Test designed by: Reg. No- IT20172350 Name- M.P.O.M. Gomes							
Test priority (High/Medium/Low):		Medium							
<p>Test Description: Registered admin can log in to admin panel using correct email address and password. The administrator can view customers added travel document details and update them in the system. Go to the User's Travel Document page and the selected travel Document can be updated in the system by entering valid information and click submit button at last.</p>									
<p>Preconditions (if there are any): Must be an admin and must be Login to the System.</p>									
<p>Test steps:</p> <p>Step1: Admin must visit to the web application. Step2: Admin must Register to the website Step3: Admin must login to the website Step4: Admin should click User's Travel Document button in admin panel. Step5: Admin must click updated button from selected travel document. Step6: Admin must enter the updated information in the form. Step7: Admin should click the submit button.</p>									
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments				
TC010	Inputs that updated details into the form	Admin can see "Travel Document update successfully" message.	Admin can see "Travel Document update successfully" message.	Pass	Travel document update Function Successfully Completed.				

2.4.6 Tour Package Management

Group ID: ITP2021_S2_MTR_G05										
Project title: Travel Agency and Tour Planning										
Testing Function: User input No of adults and child calculate the price for the tour package.										
Test case ID: TC011	Test designed by: Reg. No- IT20255510 Name- Y. N Jayasekara									
Test priority (High/Medium/Low):	Medium									
Test Description: User first select Tour package what he/she wants. Then user must fill the reservation details with No of Adults and child. Then user can see the price for the No of adults and child separately and full amount for the tour package. Then user must confirm the tour reservation by fill payment details and proceed it.										
Preconditions (if there are any): User must select the travel package for what he/she wants.										
Test steps:										
User mut go to the web site. User must select the tour in navigation bar. User must select tour package among Tour packages. User must select a date for the tour package. User input No of adults and children User clicks Procced button User must fill the payment form User must click the submit button to confirm the reservation.										
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments					
TC011	User input No of child and adults as 1 and 2.	System will calculate price for the child and adults separately and display the total amount of the child and adults' price as a tour package amount (Total:250000/=)	Total:250000/=	Pass	Output showed as expected.					

Testing Function: Admin can view, add, update, delete tour packages

Test case ID: TC012		Test designed by: Reg. No- IT20255510 Name- Y.N Jayasekara							
Test priority (High/Medium/Low):		Medium							
Test Description: Tour Package Manager/Admin can log in to admin dashboard. Admin can view all tour packages and add new one, edit or delete tour packages from the database.									
Preconditions (if there are any): Must be an admin and must be Login to the System.									
Test steps: Admin must go to the admin panel. Admin must select the tour package section in dashboard.									
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments				
TC012	packageID: TP001 packageName: Universal Studios Singapore PackageType: Family & Holiday NoOfDays: 5 MealPlan: Breakfast and Dinner Transportation: Airport Transfers Activities: Ride the roller coasters Location List: 8 Sentosa Gateway, Sentosa Island, Singapore 098269, Singapore PriceForAdult: 100000 PriceForChild: 50000 (Admin must insert new package data to form)	Success message display redirect to current page.	Success message display and redirect to current page.	Pass	Output showed as expected.				

2.4.7 Travel Insurance Management

Group ID: ITP2021_S2_MTR_G05									
Project title: Travel Agency and Tour Planning									
Testing Function: Add new insurance type									
Test case ID: TC013		Test case Designed by: Reg. No- IT20163204 Name- Manimendra N. H.							
Test priority (High/Medium/Low):		Medium							
Test Description: Registered admin can log in to admin panel. The admin can add a new insurance types to the system.									
Preconditions (if there are any): Login account must be an admin account.									
Test steps:									
Step1: Admin login to the admin panel. Step2: Admin go to insurance page by clicking insurance button in admin panel. Step3: Admin go to add new insurance page by clicking add button. Step4: Admin must fill the form. Step5: Admin should click the add button. Step6: Admin can see the successful message. Step7: Admin can see newly added insurance type display in all insurance types page.									
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments				
TC013	Admin input insurance ID, insurance type, price, description and image link	Admin can see success message.	Success message displayed.	Pass	Add new insurance type function successfully completed.				

Testing Function: Insurance Reservation									
Test case ID: TC014		Test designed by: Reg. No - IT20163204 Name - Manimendra N. H.							
Test priority (High/Medium/Low):		High							
Test Description: Registered customer can log in to system. The customer can reserve the insurance easily.									
Preconditions (if there are any): User must sign up or sign in to the system.									
Test steps:									
Step1: Customer login to the system. Step2: Customer go to insurance page by clicking insurance button in navigation bar. Step3: Customer can search insurance what he/she want Step4: Customer can select the insurance what he/she want Step5: Customer fill the form. Step6: Customer click the apply button for reserve the insurance. Step7: Customer can see the successful message.									
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments				
TC014	User input due date and reservation date	Admin can see success message.	Success message displayed.	Pass	Insurance reservation function successfully completed.				

2.4.8 Event Management

Group ID: ITP2021_S2_MTR_G05									
Project title: Travel Agency and Tour Planning									
Testing Function: Add a new event									
Test case ID: TC015		Test case Designed by: ID: IT20142728 Name: Ranawaka. T.D							
Test priority (High/Medium/Low):		Medium							
Test Description: The event manager of the company can login to the system with admin credentials and add new events to the database.									
Preconditions (if there are any): Must be a Registered user and must be Login to the System.									
Test steps:									
Step 1:login to the website as the event manager Step 2:go to add new events page Step 3: add a new event to database									
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments				
TC015	Event type-entertainment Location - Matara Date- 2/09/2021 Price-1200rs	Show newly added events to the event manager as a table	Error message show “cannot submit with empty fields”	Pass	Every field is required in this form				

Testing Function: Delete an added event								
Test case ID: TC016			Test case Designed by: ID: IT20142728 Name: Ranawaka. T.D					
Test priority (High/Medium/Low):			Medium					
Test Description: The event manager of the company can login to the system with admin credentials and delete the events from database which are not available.								
Preconditions (if there are any): Must be an admin and must be Login to the System.								
Test steps:								
Step 1: login to the website as the event manager Step 2: go to add new events page Step 3: click delete button from each record Step 4: delete event from database								
Test ID	Test Inputs	Expected Output	Actual Output	Result (Pass/Fail)	Comments			
TC016	Event ID (auto generated by database)	Events which are select to delete will deleted from database using event ID	Show message "Event deleted"	Pass	Event manager can delete any event that he wants to delete			

3. Conclusion

It's been almost close to 6 months since our project was started. Finally, we come up with the system that all client requirements are fulfilled. We are really delighted as a group to have finished this assignment successfully. As second year second semester undergraduate students, the best reward we can earn is being able to apply our whole knowledge and talents to impress our client. Having to work on such a high-level project in our second year of academic life is a key element in generating future great professionals. Let's hope we took huge advantage of this great chance. This project was not as simple as we had expected. It was somewhat hard because that the project will be held out during the Covid 19 pandemic situation. We aimed to achieve two main goals through this project. One is to get professional experience and develop our knowledge. The second goal is to provide excellent service to our clients and establish our reputation in IT field.

In this project, the project's main goal was to cover the Travel Agency System with all functions according to client expectation. In this system, we've gone through eight different functions for each eight members. The functions are namely, User management system, Tickets reservation management system, Tour package management system, Travel documentation management system, Ground reservation management system, Accommodation management system, Travel insurance management system, and an Event management system.

Basically, in the User management system contains the details of all customers of the company and also it handles the user admin authentication. Tickets reservation management system handles all the tickets reservations and its payments such as train, bus, air tickets. In Ground reservation management system contains all the ground reservations that customer expected. As same as Accommodation management system can reserve any accommodations such as hotel, motel, etc. In Tour package management system managing all the tour packages of the system with combination of purchasing system. Travel documentation management system managing all the client reports and documentation which are essential for the customer's travel purpose. Travel insurance management system is responsible for the security and safe of all clients. Moreover, in the Event management system customized and organize all the events and functions according to the customer willing. Besides of this every single subsystem implemented with retrieval, insert, update, and delete operations along with a report generation function and search function.

Based on this experience, we also hope to launch more successful projects in the future. Not only that but also, we got the chance to recognize our capabilities and weaknesses throughout this period when working with as a group.

4. References

- [1] *Lecture 5 - Final Report.*
- [2] *IEEE Citation Guidelines*
- [3] <https://ng.ant.design/docs/introduce/en>