



Sri Lanka Institute of Information Technology

**Hotel Reservation Platform from Restful Services
And
WSO2 Enterprise Integrator**

Group Members

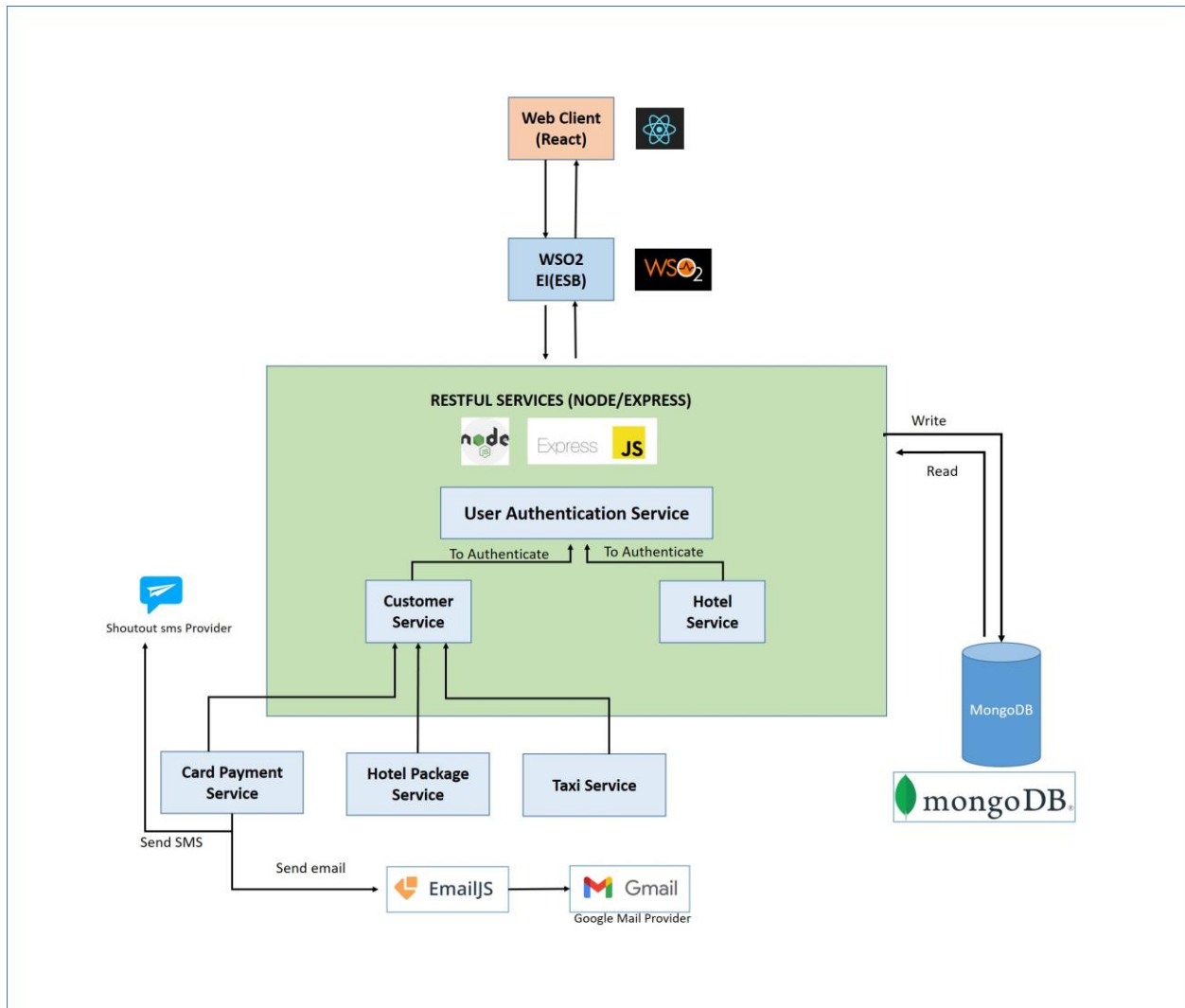
Student ID	Name with initials
IT20219420	Senadeera S.A.V.J
IT20218744	Wanigasinghe W.W.L.K.G
IT20236250	Perera G.U.L.K
IT20106874	Nethmini G.A.H.A

Contents

1. High Level Architectural Diagram of the System	5
2. Service Interfaces exposed by each service	6
2.1 User Authentication Service	6
2.2 Package Service	7
2.3 Taxi Service	8
2.4 Order Service	9
2.5 SMS Service	9
3. Workflows explained with code Snippets	10
3.1 Process of Registering a New User	10
3.2 Process of Log In	11
3.3 Process of Create Hotel Reservation	12
3.4 Process of Cancel Hotel Reservation	13
3.5 Process of Get All Hotel Reservation	14
3.6 Process of Retrieving a Ordered Hotel Reservation.....	15
3.7 Process of Creating New Hotel Package	16
3.8 Process of Get all Package	17
3.9 Process of Edit Hotel Package Details.....	18
3.10 Process of Retrieving a Hotel Package	19
3.11 Process of Deleting a Hotel Package.....	20
3.12 Process of Create Hotel Admin	21
3.13 Process of Removing User from a Hotel Admin	22
3.14 Process of Get all Users	23
3.15 Process of Blocking a User	24
3.16 Process of Book Taxi.....	25
3.17 Process of Get All Taxi Booking	26
3.18 Process of Retrieving a Taxi Reservation.....	27
3.19 Process of Cancelling Taxi Reservation.....	28
4. Screenshots of the User Interfaces	29
4.1 Register	29
4.2 Log In.....	29
4.3 All Hotel Packages	30
4.4 Reservation List	30

4.5 Booked Taxi	31
4.6 Users List.....	31
4.7 Package List	32
4.8 Add Package	32
4.9 Reservation List	33
4.10 Taxi Reservation List	33
4.11 Admin Menu.....	34
4.12 Cart.....	34
4.13 Payment.....	35
4.14 Hotel Admin Menu	36
4.15 Customer Menu.....	36
4.16 Hotel Package Filter	37
4.17 Reservation Confirmation SMS.....	37
4.18 Reservation Confirmation Email.....	38
4.19 WS02 Integration Studio	38
5. Appendix.....	39
5.1 Order Service	39
5.2 Package Service	45
5.3 User Authentication Service	50
5.4 Taxi Service	56
5.5 SMS Service	60
5.6 Frontend	64

1. High Level Architectural Diagram of the System

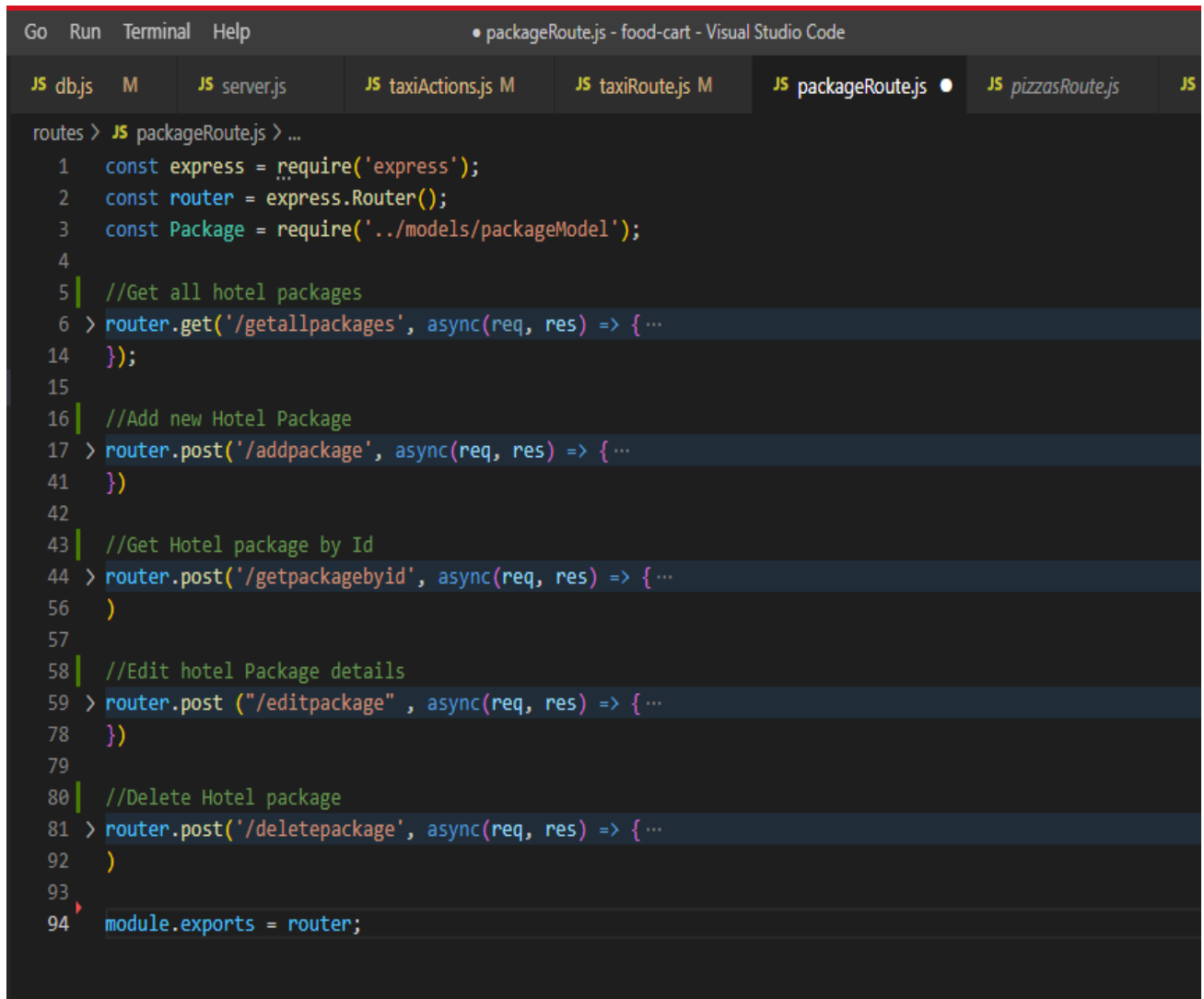


2. Service Interfaces exposed by each service

2.1 User Authentication Service

```
JS db.js M JS server.js JS taxiActions.js M JS taxiRoute.js M JS userRoute.js JS packageRoute.js JS ordersRoute.js
routes > JS userRoute.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const mongoose = require('mongoose');
4  const User = require('../models/userModel');
5
6  //Customer Registration
7  > router.post('/register', async (req, res) => { ...
22  })
23
24  //Customer Login
25  > router.post('/login', async (req, res) => { ...
56  })
57
58  //Get all user Details
59  > router.get('/getallusers', async(req, res) => { ...
67  });
68
69  //Delete or block user
70  > router.post('/deleteuser', async(req, res) => { ...
81  )
82
83  //Create hotel admin
84  > router.post('/makehoteladmin', async (req, res) => { ...
98  })
99
100 //Remove hotel admin
101 > router.post('/removehoteladmin', async (req, res) => { ...
115  })
116
117 module.exports = router;
118
```

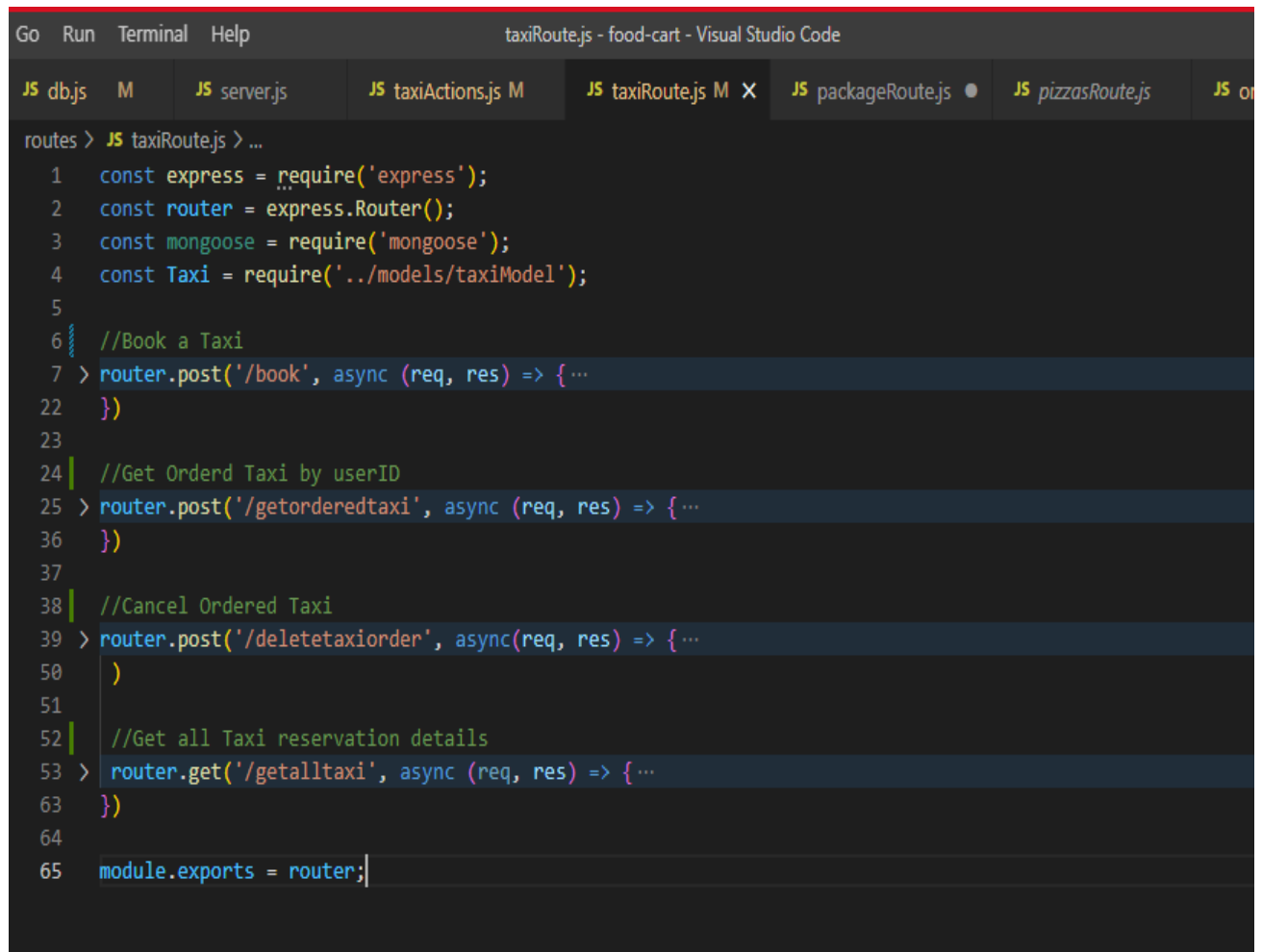
2.2 Package Service



The screenshot shows the Visual Studio Code editor interface. The top bar indicates the current file is 'packageRoute.js' in a workspace named 'food-cart'. The Explorer sidebar on the left shows a file structure with 'db.js', 'server.js', 'taxiActions.js', 'taxiRoute.js', 'packageRoute.js' (selected), and 'pizzasRoute.js'. The main editor area displays the content of 'packageRoute.js' with line numbers 1 through 94. The code defines an Express.js router for handling hotel packages, including endpoints for getting all packages, adding a new package, getting a package by ID, editing package details, and deleting a package. The router is exported as the module's exports.

```
1  const express = require('express');
2  const router = express.Router();
3  const Package = require('../models/packageModel');
4
5  //Get all hotel packages
6  > router.get('/getallpackages', async(req, res) => { ...
14  });
15
16  //Add new Hotel Package
17  > router.post('/addpackage', async(req, res) => { ...
41  });
42
43  //Get Hotel package by Id
44  > router.post('/getpackagebyid', async(req, res) => { ...
56  )
57
58  //Edit hotel Package details
59  > router.post ("/editpackage" , async(req, res) => { ...
78  })
79
80  //Delete Hotel package
81  > router.post('/deletepackage', async(req, res) => { ...
92  )
93
94  module.exports = router;
```

2.3 Taxi Service



```
Go Run Terminal Help taxiRoute.js - food-cart - Visual Studio Code
JS db.js M JS server.js JS taxiActions.js M JS taxiRoute.js M X JS packageRoute.js JS pizzasRoute.js JS or
routes > JS taxiRoute.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const mongoose = require('mongoose');
4  const Taxi = require('../models/taxiModel');
5
6  //Book a Taxi
7  > router.post('/book', async (req, res) => { ...
22  })
23
24  //Get Orderd Taxi by userID
25  > router.post('/getorderedtaxi', async (req, res) => { ...
36  })
37
38  //Cancel Ordered Taxi
39  > router.post('/deletetaxiorder', async(req, res) => { ...
50  })
51
52  //Get all Taxi reservation details
53  > router.get('/getalltaxi', async (req, res) => { ...
63  })
64
65  module.exports = router;
```


2.4 Order Service

```
Go Run Terminal Help • ordersRoute.js - food-cart - Visual Studio Code
JS db.js M JS server.js JS taxiActions.js M JS taxiRoute.js JS packageRoute.js JS ordersRoute.js M JS orderA

routes > JS ordersRoute.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const stripe = require('stripe')('sk_test_51Kry14IppFV46PpQ87ZKsImeawBxxAYfdwBG9xfSZWPKj8000wW6mSAjzDXUVjmnA6
4  const { v4: uuidv4 } = require('uuid');
5  const Order = require('../models/orderModel');
6
7  //Create Reservations
8  > router.post('/placeorder', async (req, res) => { ...
56  })
57
58  //Get User Created Reservations
59  > router.post('/getuserorders', async (req, res) => { ...
70  })
71
72  //Get all reservation
73  > router.get('/getallorders', async (req, res) => { ...
83  })
84
85  // Create active customer
86  > router.post('/checkedin', async (req, res) => { ...
100  })
101
102  //Get reservation by id
103  > router.post('/getorderbyid', async(req, res) => { ...
115  )
116
117  //Cancel reservation
118  > router.post('/deleteorder', async(req, res) => { ...
129  )
130
131  module.exports = router;
```

2.5 SMS Service

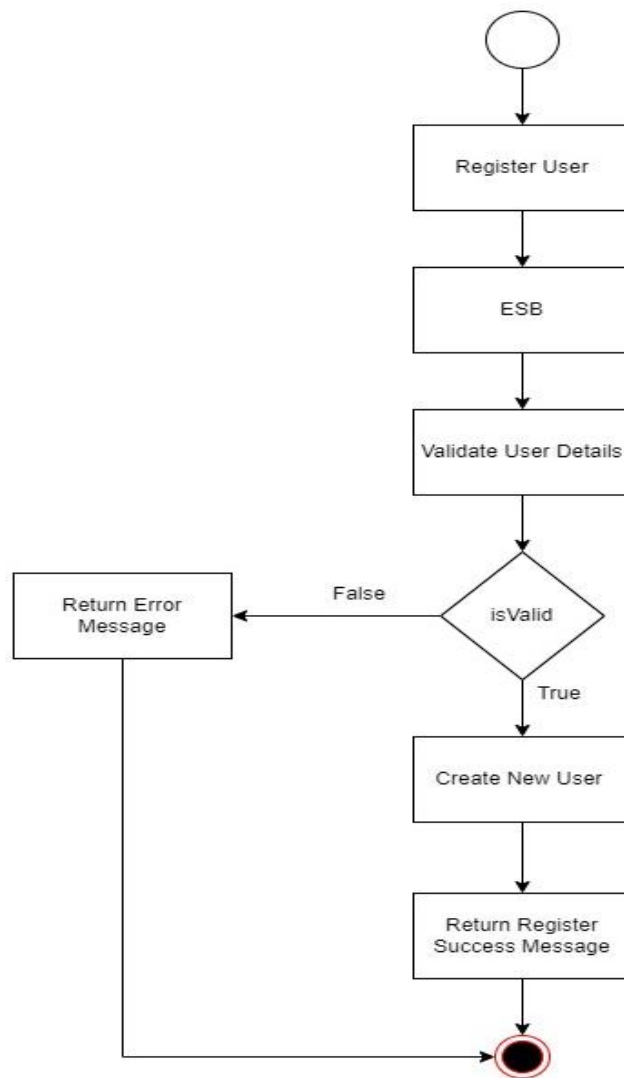
```
FilterPkg.js 2, M Checkout.js 8, M sms.js M Registerscreen.js 9+, M Loginscreen.js 9, M Navibar.js 2, M

backends > sms_service > routes > sms.js > ...
1  const express = require('express');
2  const router = express.Router();
3  var ShoutoutClient = require('shoutout-sdk');
4
5  //Reservation Confirmation SMS
6  > router.post('/send', async(req, res) => { ...
32  });
33
34  module.exports = router;
```

3. Workflows explained with code Snippets

3.1 Process of Registering a New User

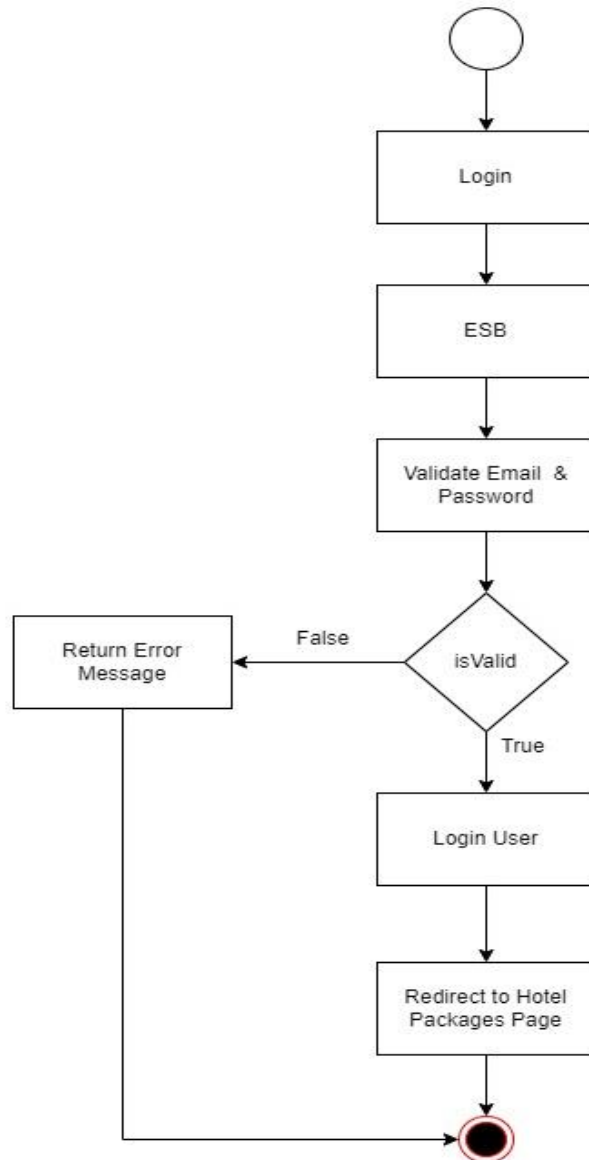
Workflow diagram



A POST request with the details (name, email, phone, password, confirm password) will be sent to the user authentication service through ESB (WSO2 EI). If the request is valid, a new user is created by adding new JSON document to the collection in mongo DB. The password is converted to a hash code for better privacy using bcrypt and stored. When querying, password is not passing unless specially asked for it. Once the registration is successful the user will be directed to the home page.

3.2 Process of Log In

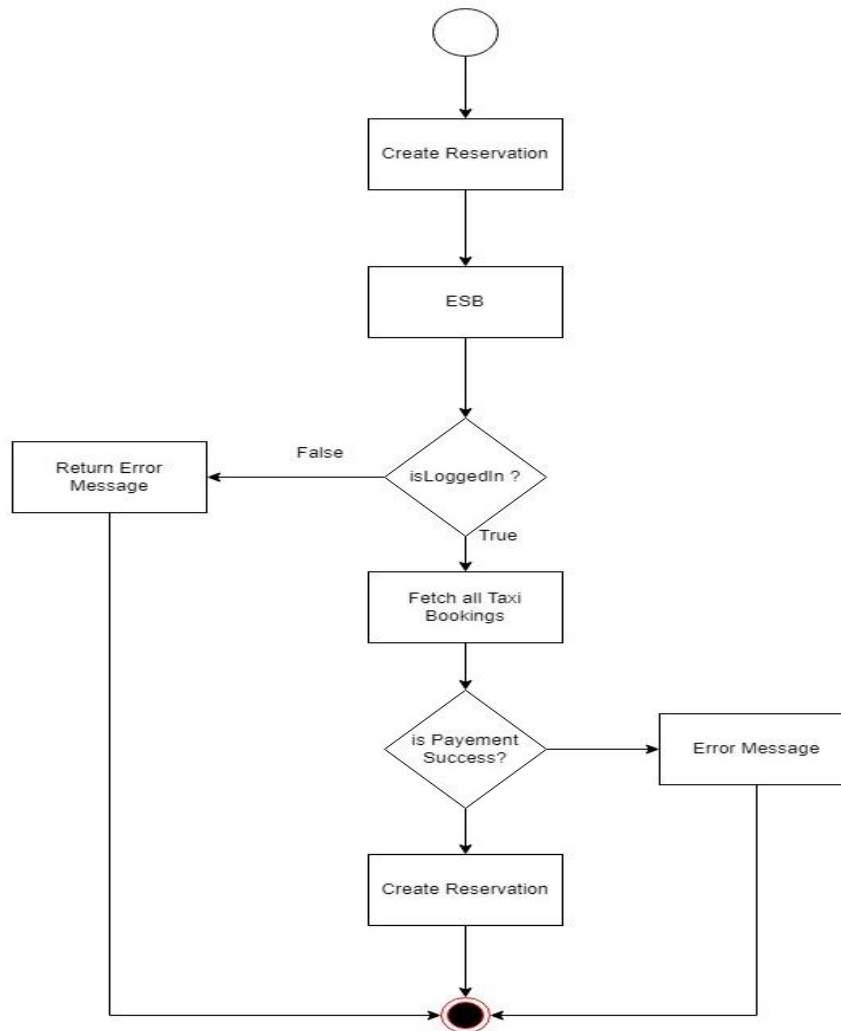
Workflow diagram



A POST request with email and password will be sent to the user authentication service through ESB (WSO2 EI). Once request is received the system check the availability of the user by searching for the email and match the passwords. If the passwords match, then the user is directed to the home page. Otherwise, an error message will be generated and if the user is not found , the user will be directed to the registration to create new user.

3.3 Process of Create Hotel Reservation

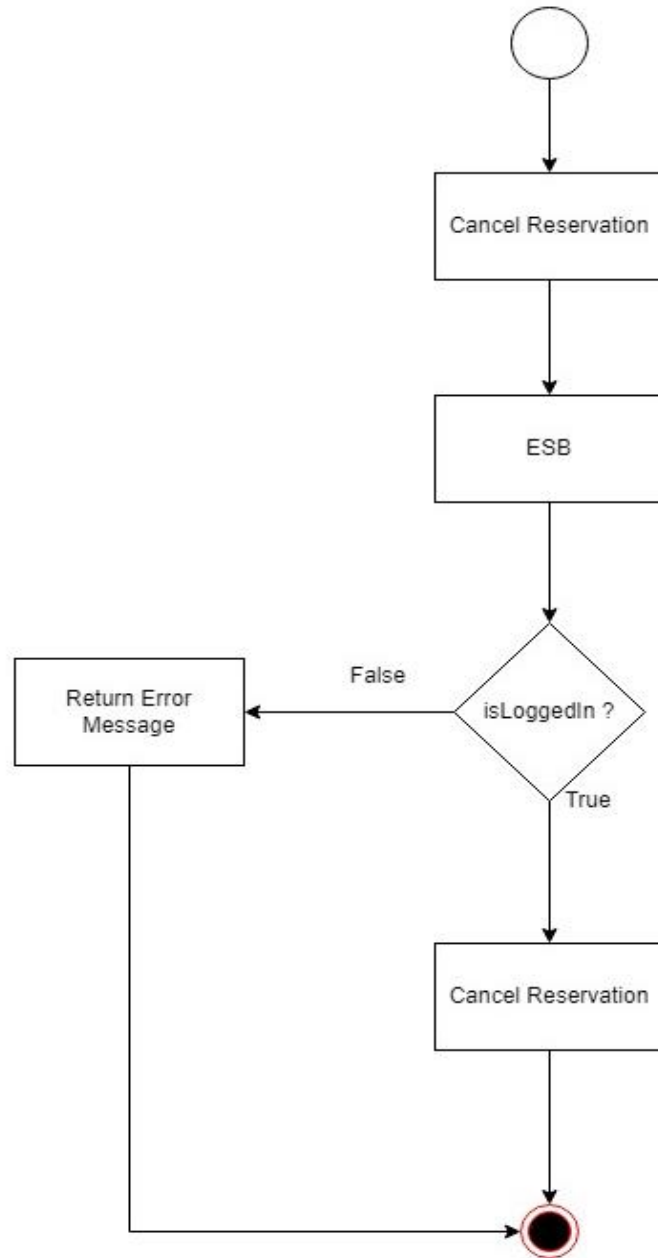
Workflow diagram



A POST request will be sent to the hotel package reservation service. To create a new order, user must be authenticated. Request body must have an array of objects where each object must have the package id and its reserved days. This function will iterate the array of objects and check if each object has a package id and then it checks if there is an existing package in the database with the package id in the object. Additionally, it allows user to select the number of days from their selection list as they offer it for limited number of days. If any of the validations fail, respective error messages will be sent back to the client. Else, an object with the validated package array and total payment value (package price * number of days selected) will be returned. A new order will be created by setting the user's id, package, and payment value fields. This order object is saved and if it fails to save an error message is sent back to the client. Else, the saved object will be returned back to the client which will be required to save the taxi details (if required) and payment.

3.4 Process of Cancel Hotel Reservation

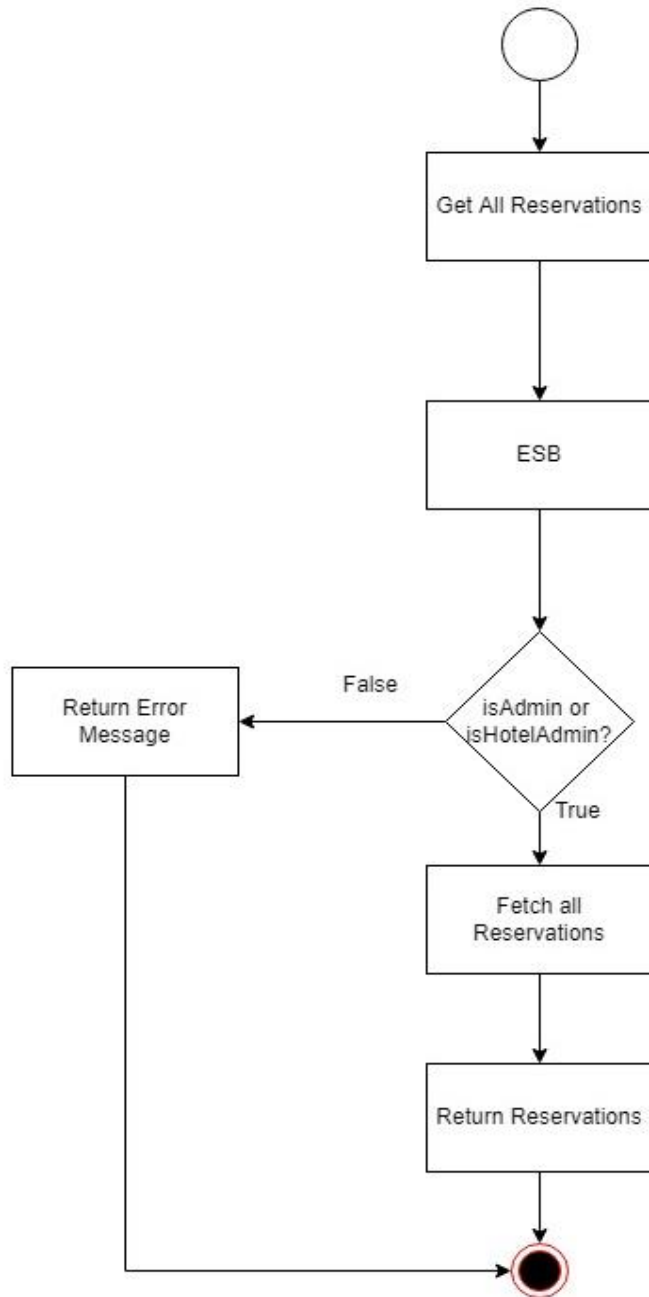
Workflow diagram



DELETE request will be sent with order id as a request parameter toward the hotel package reservation service through the ESB. An admin can cancel ordered packages. The ordered details will be deleted from database. If order was deleted, then send successfully message will be sent back. Else error response will be sent with 400 status code.

3.5 Process of Get All Hotel Reservation

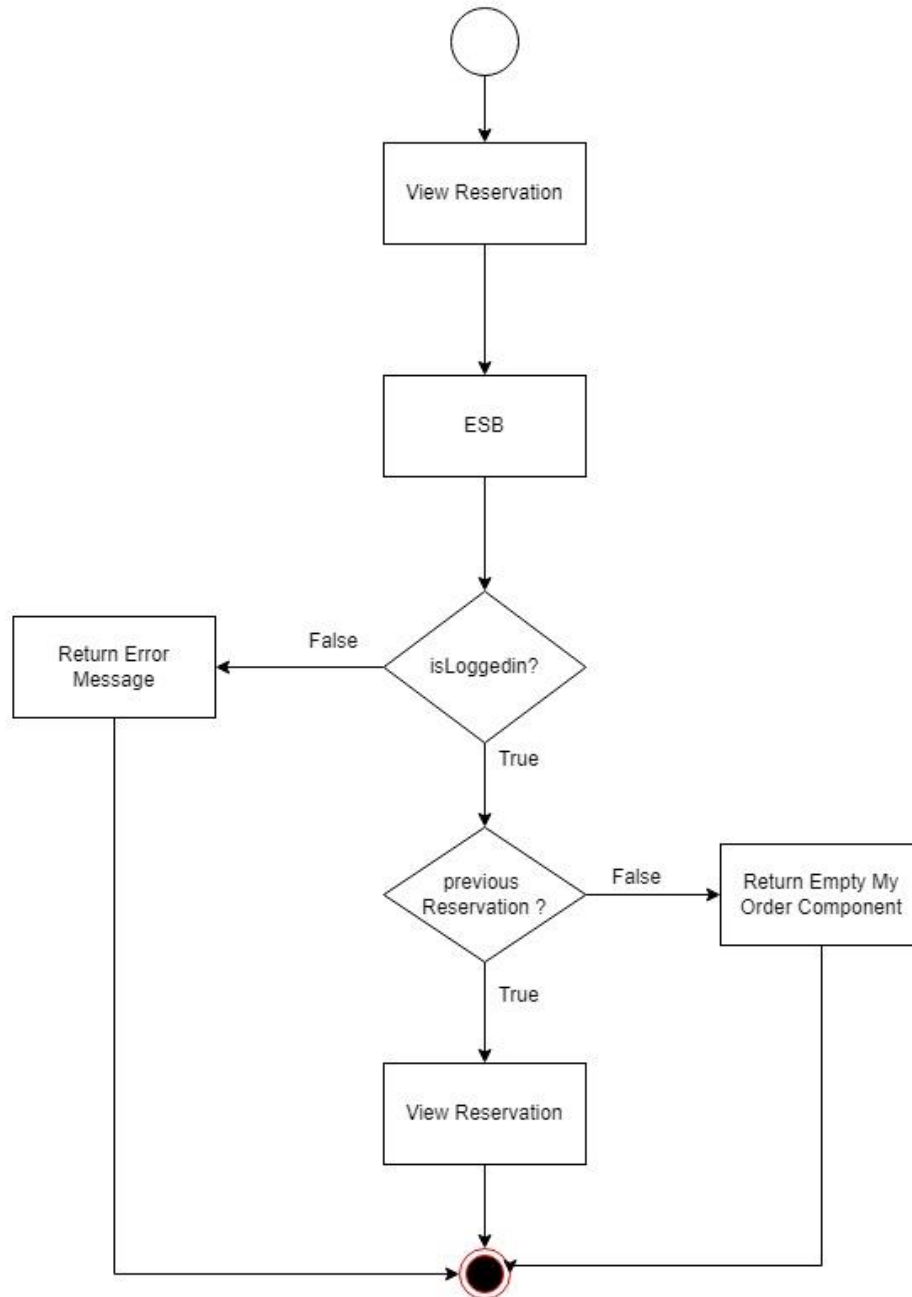
Workflow diagram



GET request will be sent to list reservations toward the hotel package reservation service through the ESB. An admin can list all the reservations done by the users. The reservations will be fetched from the order model. If the ordered packages are available, then those order details will be sent back. If there is error, it will return error message with 400 status code.

3.6 Process of Retrieving a Ordered Hotel Reservation

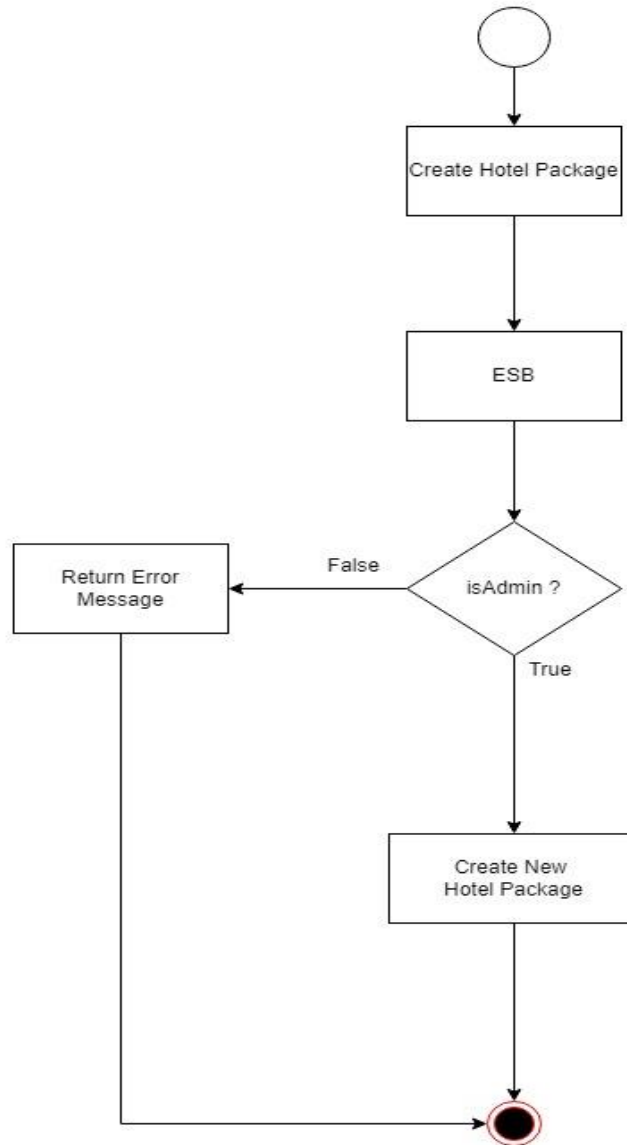
Workflow diagram



GET request will be sent to list reservations toward the hotel package reservation service through the ESB with the parameter of order id. Order details will be display according to the required order id. The order will be fetched from the order model. If the ordered package is available that package details will be sent back. If there is error, it will return error message with 400 status code.

3.7 Process of Creating New Hotel Package

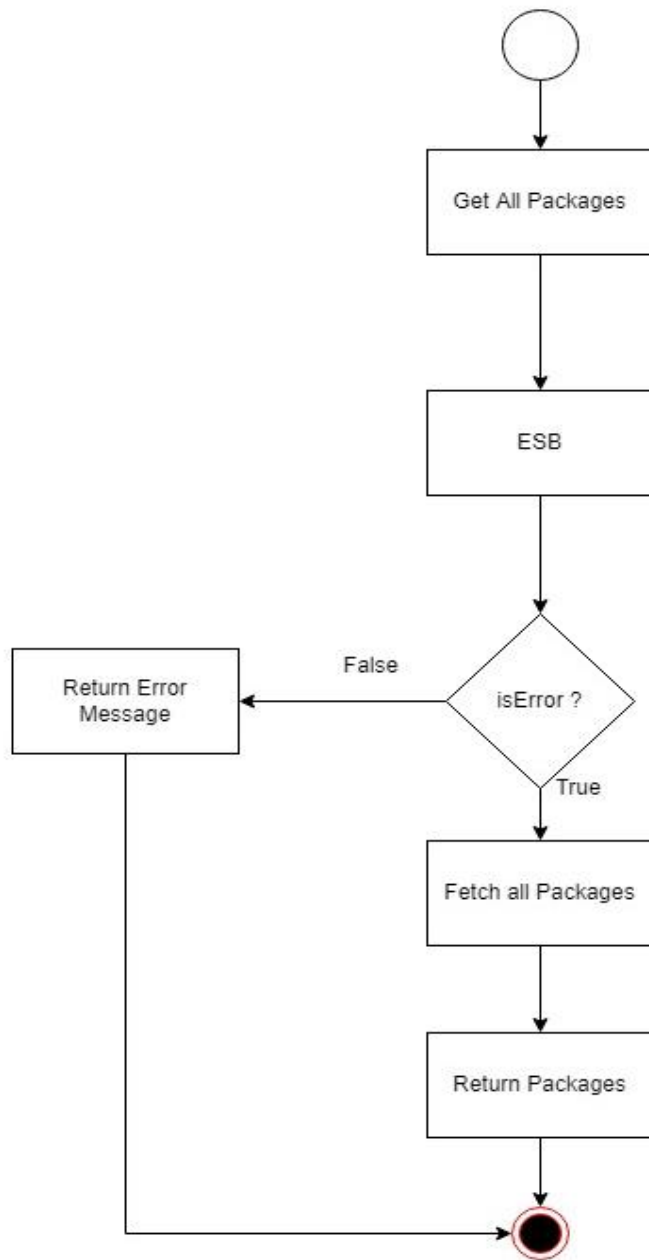
Workflow diagram



POST request will be sent to the hotel package service through the ESB. The request body must have name, small variant price, medium variant price, large variant price, image, description, and category. If there is a parameter with package id, then package will be fetched. If there is no package id, then new package will be initialized, and data will be appended to package model we created, and data will be saved. If there is package, then the fetched package will be updated with the requested data. If an exception occurred, then it will return an error response with 400 status code. If data is saved, then a response with successful message and will be send back with the package data.

3.8 Process of Get all Package

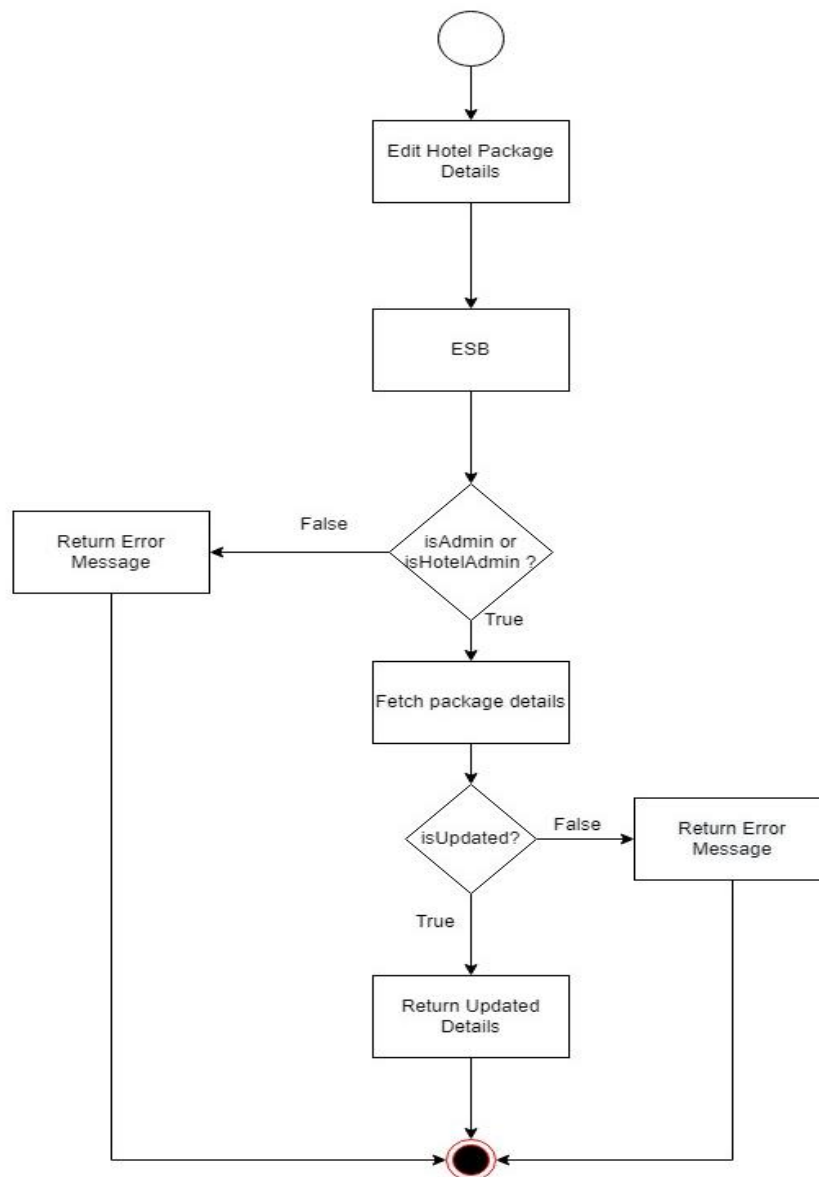
Workflow diagram



GET request will be sent to list packages toward the hotel package service through the ESB. An admin can list all the packages available. The packages will be fetched from the package model. If the package is available package details will be sent back. If there is error, it will return error message with 400 status code.

3.9 Process of Edit Hotel Package Details

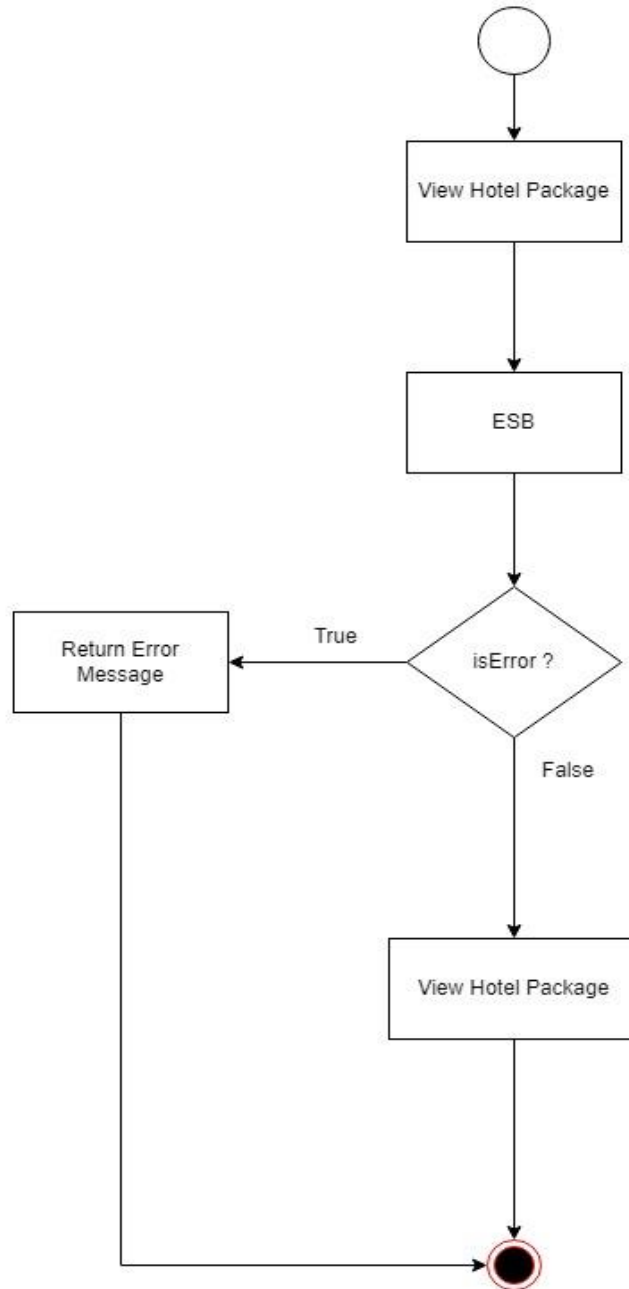
Workflow diagram



To update a package PUT request will be send to the hotel package service through the ESB. An admin can update the package details. The request must have a parameter packageid and then save package function will be invoke in order to update package. Creating the package and updating the package details will be done using the same function to facilitate reusability. Since the packageid is present in request parameter it will be used to fetch the package and request new body data will be appended to this and then updated or else error message will be sent back.

3.10 Process of Retrieving a Hotel Package

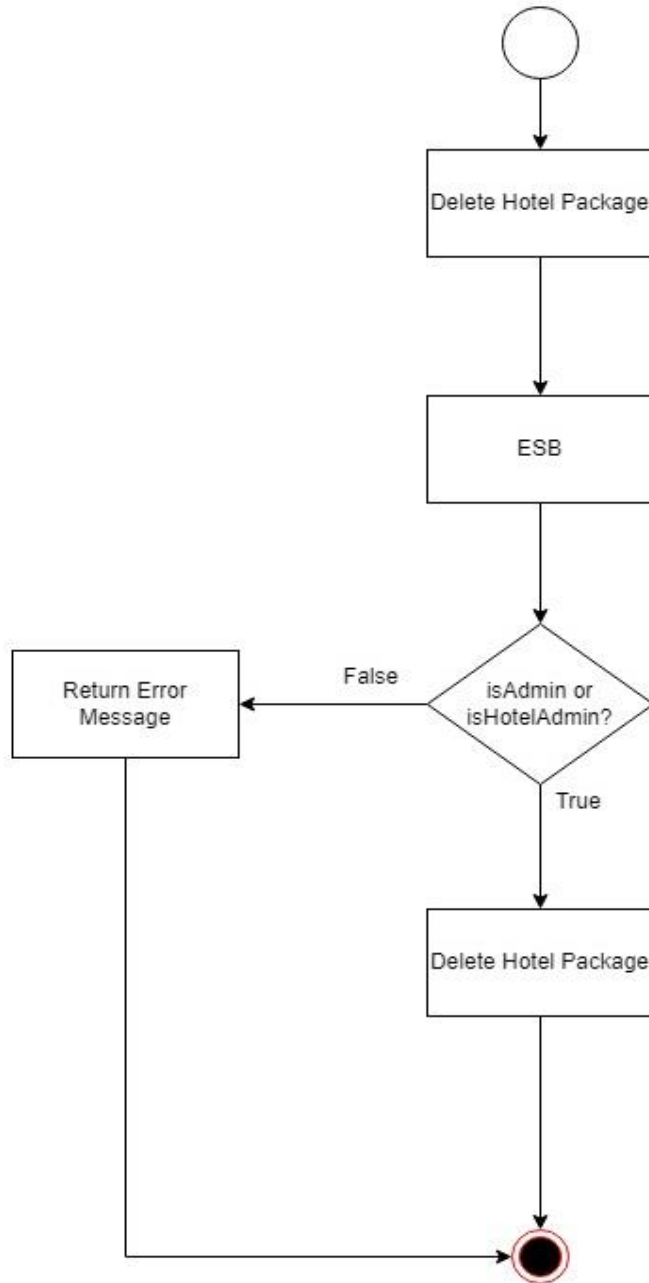
Workflow diagram



GET request will be sent to list packages toward the hotel package service through the ESB with the parameter of package id. Package will be display according to the required package id. The package will be fetched from the package model. If the package is available package details will be sent back. If there is error, it will return error message with 400 status code.

3.11 Process of Deleting a Hotel Package

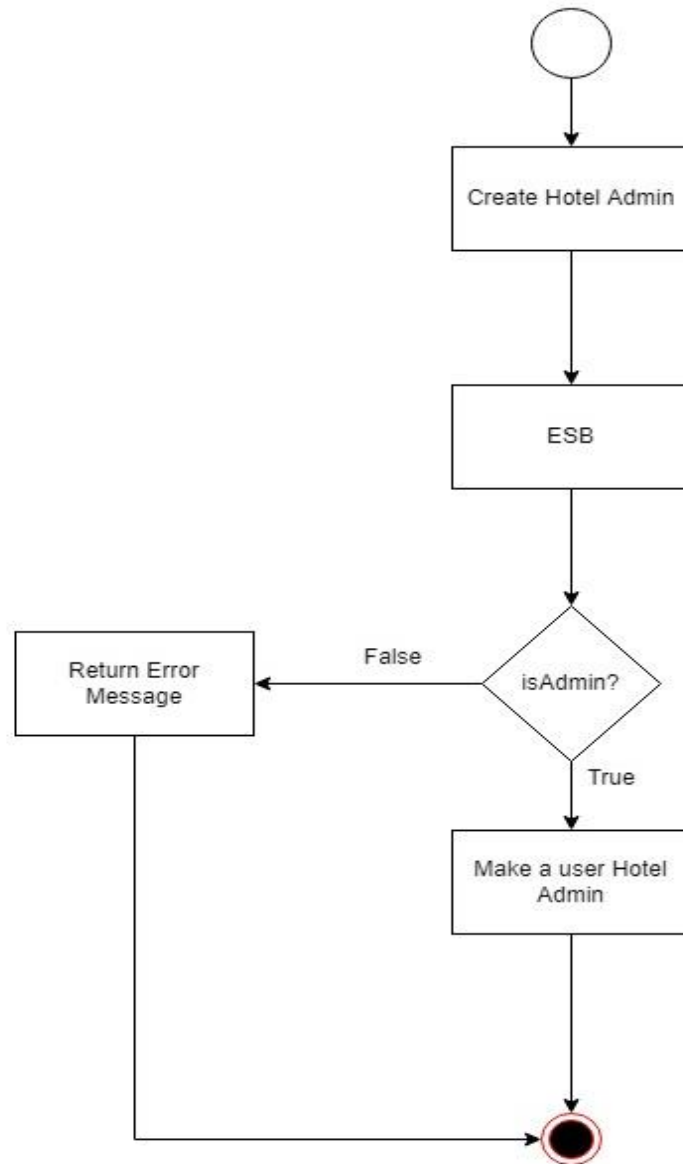
Workflow diagram



DELETE request will be sent with package id as a request parameter toward the hotel package service through the ESB. An admin can delete a package. The packages will be deleted from database. If package was deleted, then send successfully message will be sent back. Else error response will be sent with 400 status code.

3.12 Process of Create Hotel Admin

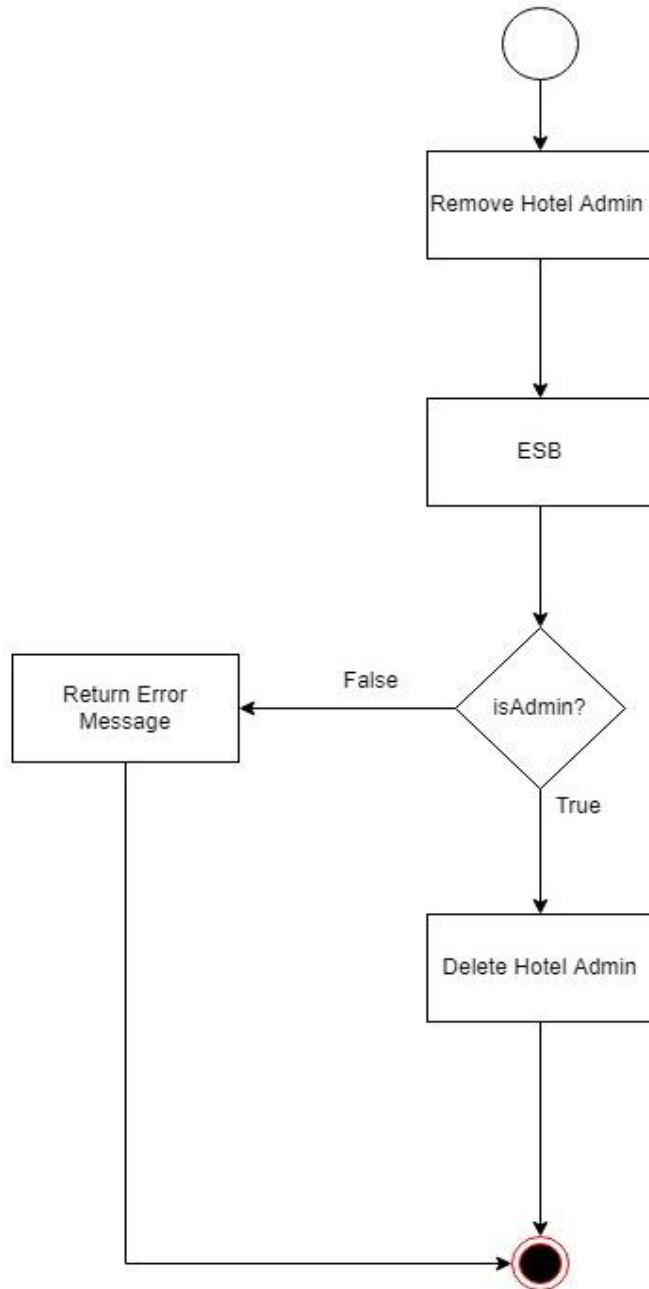
Workflow diagram



A POST request will be sent to the user authentication service through ESB (WSO2 EI). Once request is received the system give access to user to play role as a hotel admin also, then the user able to see the hotel admin dashboard and also able to do hotel admin functionalities. Otherwise, an error message will be generated and if unable to give the access.

3.13 Process of Removing User from a Hotel Admin

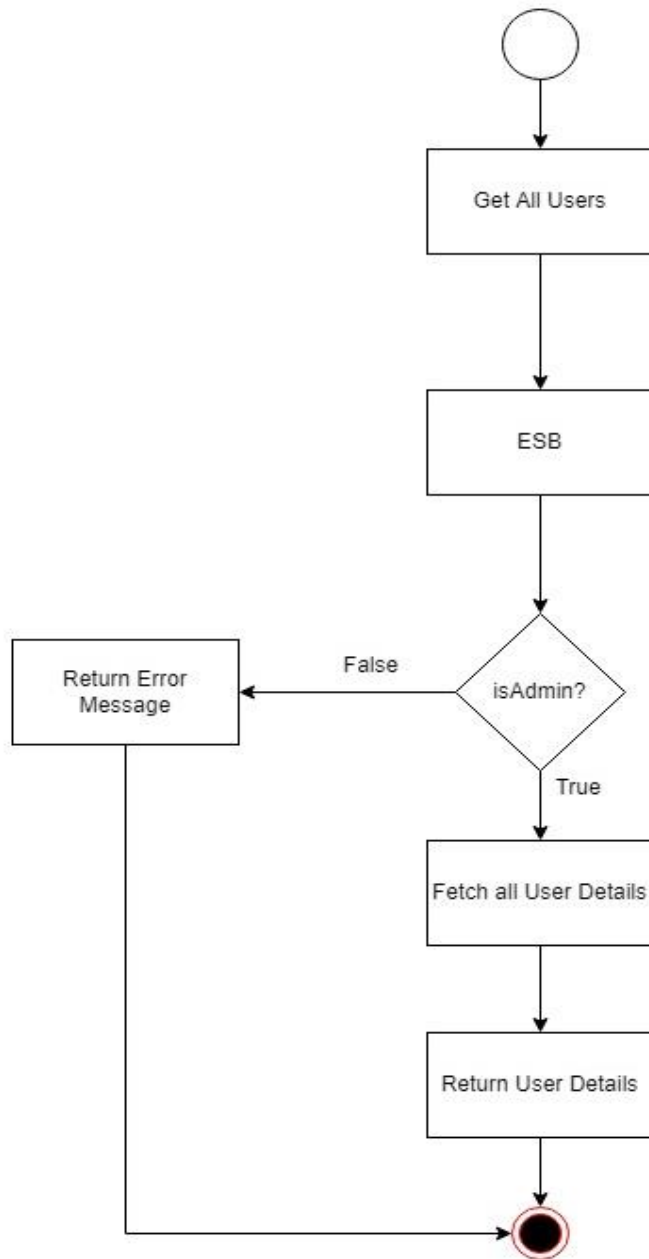
Workflow diagram



A POST request will be sent to the user authentication service through ESB (WSO2 EI). Once request is received the system deny access for user to play role as a hotel admin also, then the user won't be able to see the hotel admin dashboard and also unable to do hotel admin functionalities. If request doesn't send correctly, an error message will be generated.

3.14 Process of Get all Users

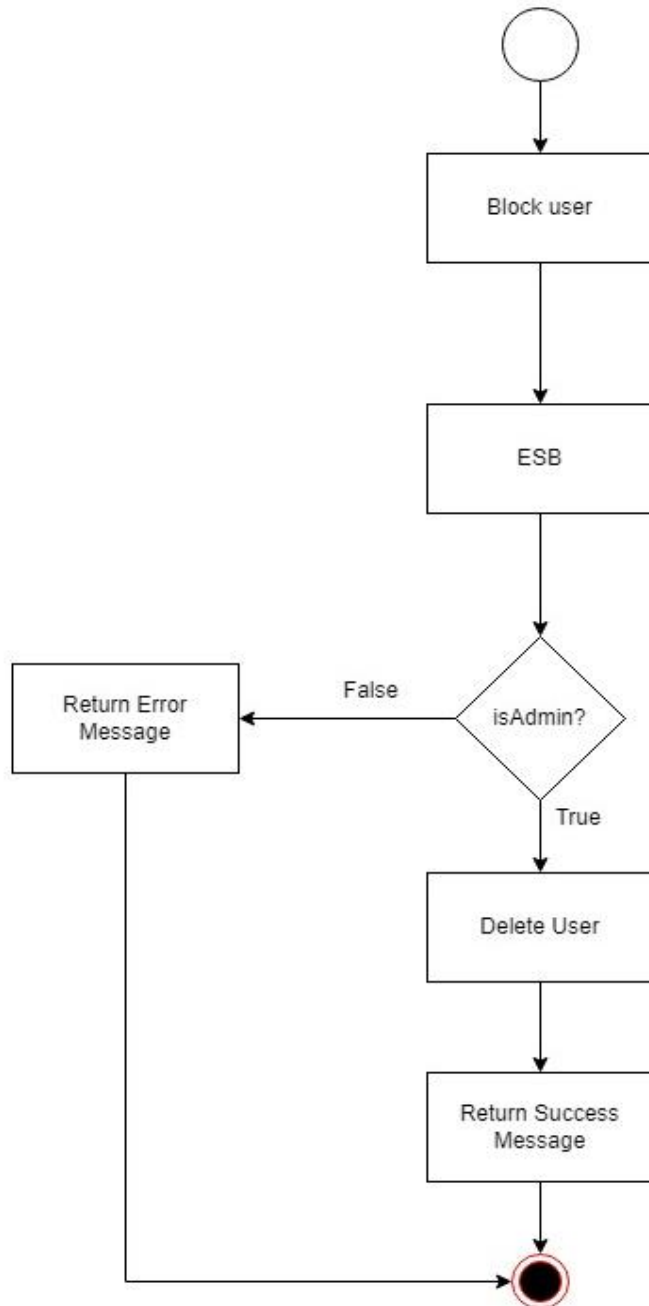
Workflow diagram



GET request will be sent to list users toward the user authentication service through the ESB. An admin can list all the users. The users will be fetched from the user model. If the users are available user details will be sent back. If there is error, it will return error message with 400 status code.

3.15 Process of Blocking a User

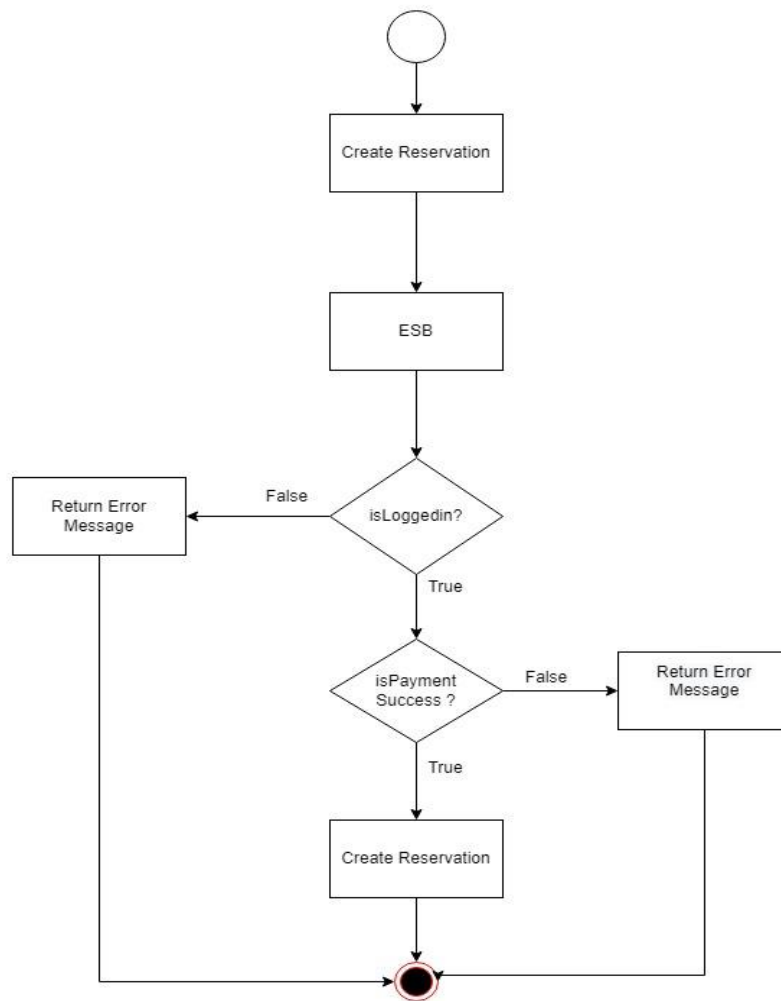
Workflow diagram



DELETE request will be sent with user id as a request parameter toward the user authentication service through the ESB. An admin can block a user. The user will be deleted from database. If user was blocked, then send successfully message will be sent back. Else error response will be sent with 400 status code.

3.16 Process of Book Taxi

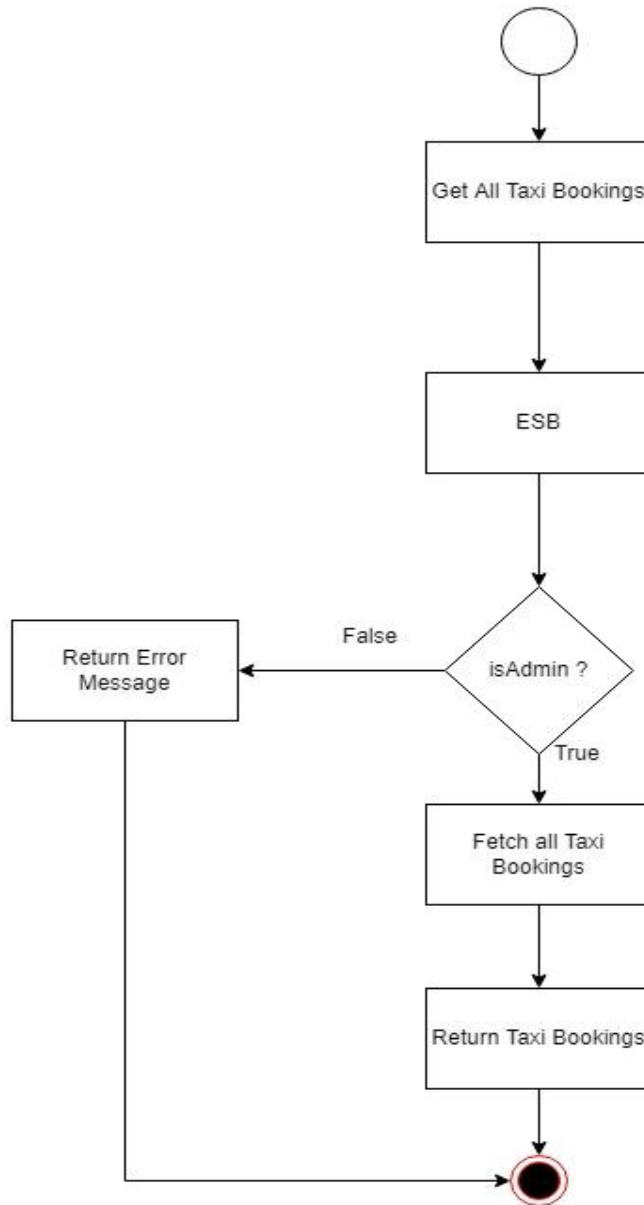
Workflow diagram



A POST request will be sent to the taxi reservation service. To create a book a taxi, user must be authenticated. Request body must have an array of objects where each object must have the package id, booking date and its reserved days. If any of the validations fail, respective error messages will be sent back to the client. A new taxi order will be created by setting the user's address and booking date fields. This order object is saved and if it fails to save an error message is sent back to the client. Else, the saved object will be returned back to the client which will be required to save the taxi details.

3.17 Process of Get All Taxi Booking

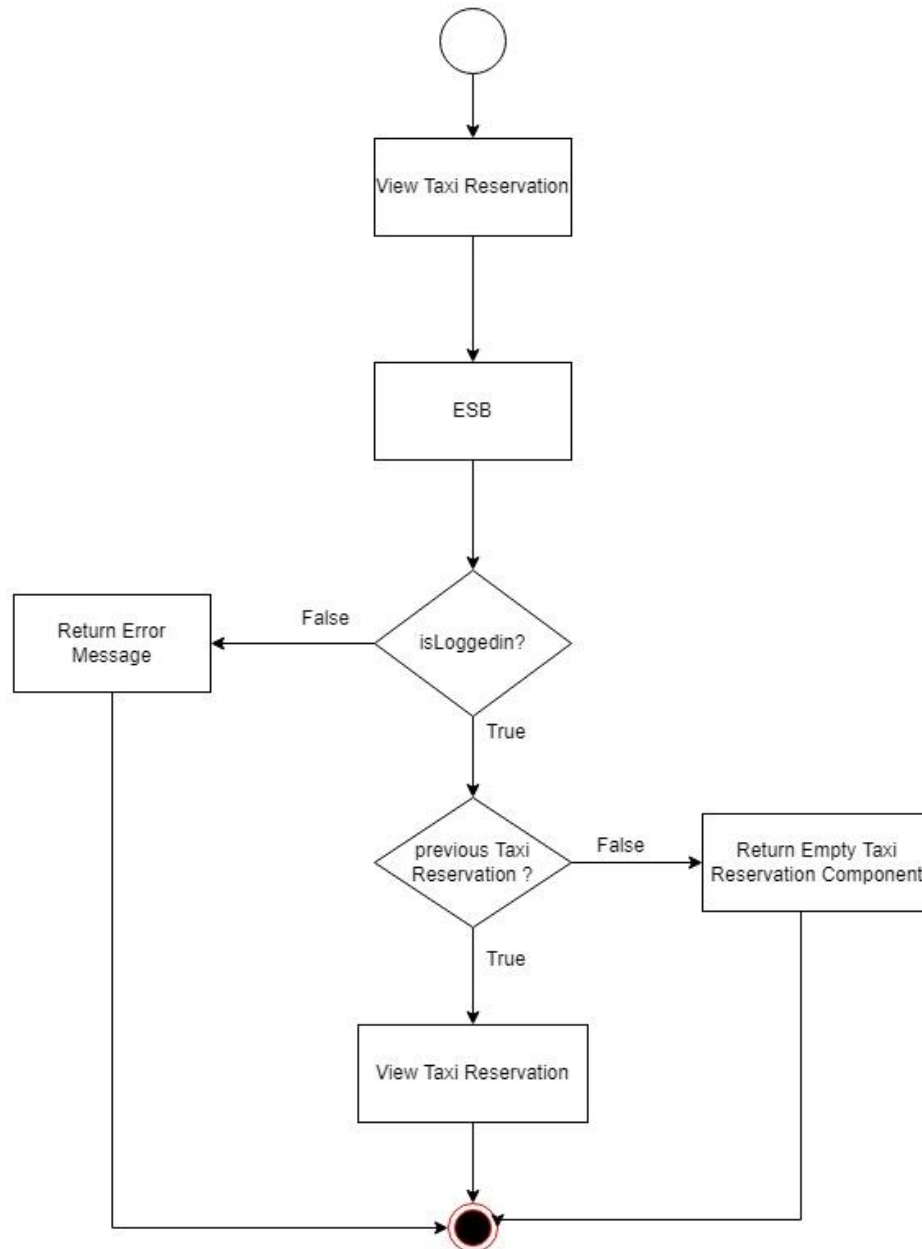
Workflow diagram



GET request will be sent to list taxi reservations toward the taxi reservation service through the ESB. An admin can list all the taxi reservations done by the users. The taxi reservations will be fetched from the order model. If the order is available taxi order details will be sent back. If there is error, it will return error message with 400 status code.

3.18 Process of Retrieving a Taxi Reservation

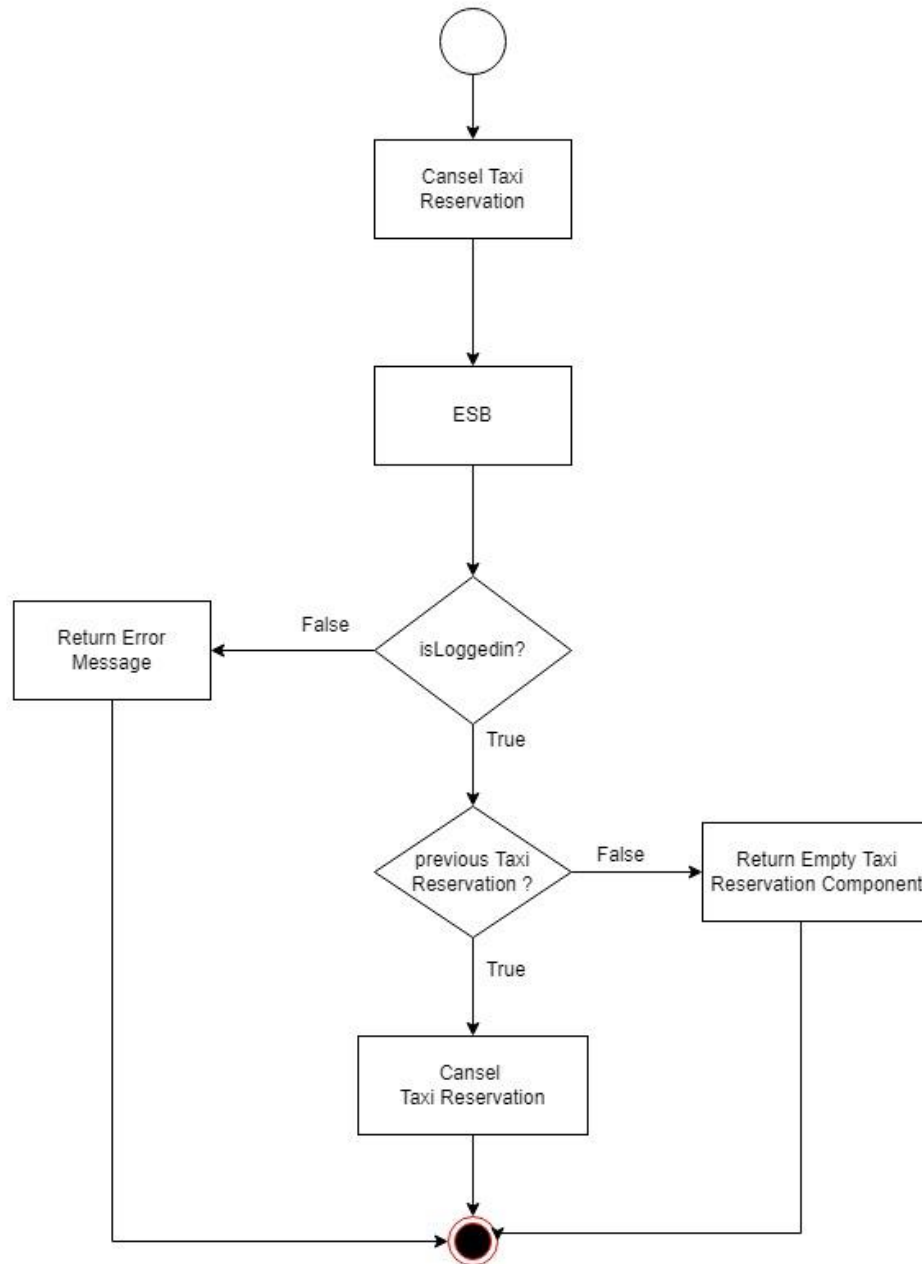
Workflow diagram



GET request will be sent to list taxi reservations toward the taxi reservation service through the ESB with the parameter of order id. Taxi reservation details will display according to the required order id. The order will be fetched from the order model. If the taxi is available taxi reservation details will be sent back. If there is error, it will return error message with 400 status code.

3.19 Process of Cancelling Taxi Reservation

Workflow diagram



DELETE request will be sent with order id as a request parameter toward the taxi reservation service through the ESB. An admin can cancel taxi reservation. The taxi reservation details will be deleted from database. If order was deleted, then send successfully message will be sent back. Else error response will be sent with 400 status code.

4. Screenshots of the User Interfaces

4.1 Register

The screenshot shows the 'Register' page of the Kingsbury Hotel Reservations website. The page has a dark header with the site name and navigation links. A central dark box contains the registration form with fields for Name, Email, Phone, Password, and Confirm Password, followed by a red 'REGISTER' button and a 'Click here to Login' link. The footer contains four columns: 'Visit Us' with a map, 'Address' with the hotel's location, 'Terms' with a list of terms, and 'Services' with a list of services.

Kingsbury Hotel Reservations [Login](#) [Add to Cart](#)

Register

[REGISTER](#)

[Click here to Login](#)

Visit Us
[View larger map](#)
The Kingsbury Hotel Parking Lot
The Kingsbury Colombo
Refined hotel with 4 restaurants & a spa

Address
26/D
Malabe
Kaduvela
Colombo

Terms
Pay visa
Copy Right
Must Agree
Must Read

Services
Hotel Packages
Taxi Service

4.2 Log In

The screenshot shows the 'Login' page of the Kingsbury Hotel Reservations website. The page has a dark header with the site name and navigation links. A central dark box contains the login form with fields for Email and Password, followed by a red 'LOGIN' button and a 'Click here to Register' link. The footer is identical to the Register page, containing 'Visit Us', 'Address', 'Terms', and 'Services' sections.

Kingsbury Hotel Reservations [Login](#) [Add to Cart](#)

Login

[LOGIN](#)

[Click here to Register](#)

Visit Us
[View larger map](#)
The Kingsbury Hotel Parking Lot
The Kingsbury Colombo
Refined hotel with 4 restaurants & a spa
Galadari Hotel
Upscale lodging with 3 restaurants

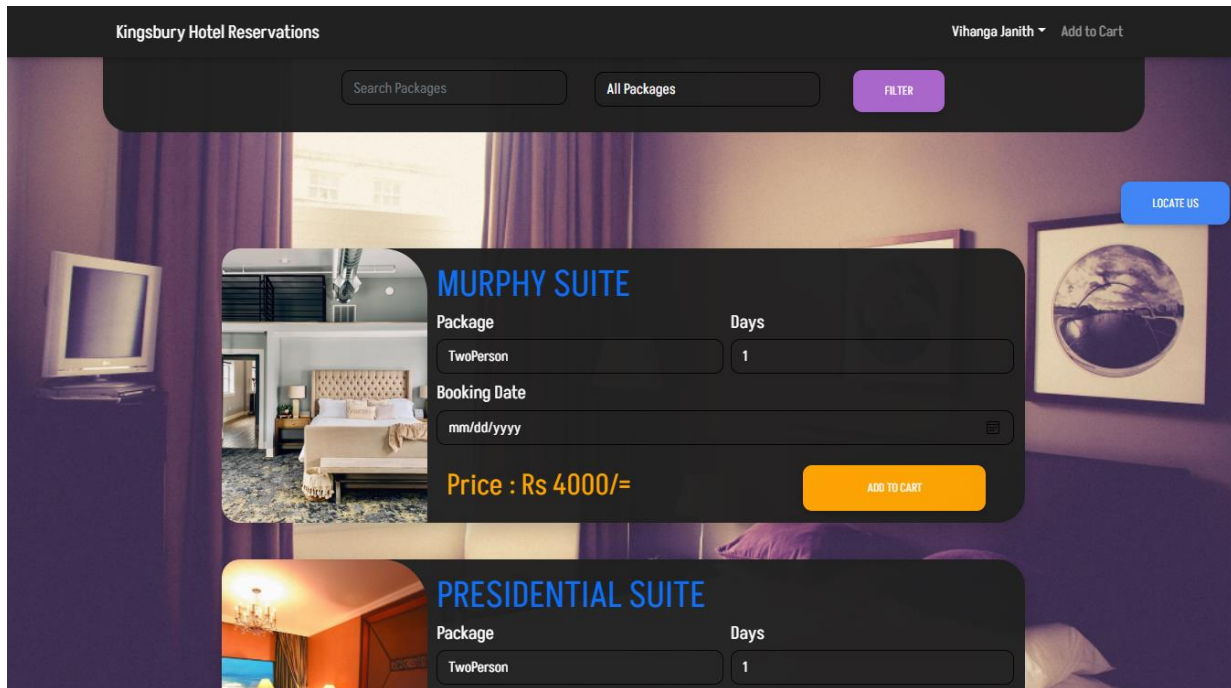
Address
26/D
Malabe
Kaduvela
Colombo

Terms
Pay visa
Copy Right
Must Agree
Must Read

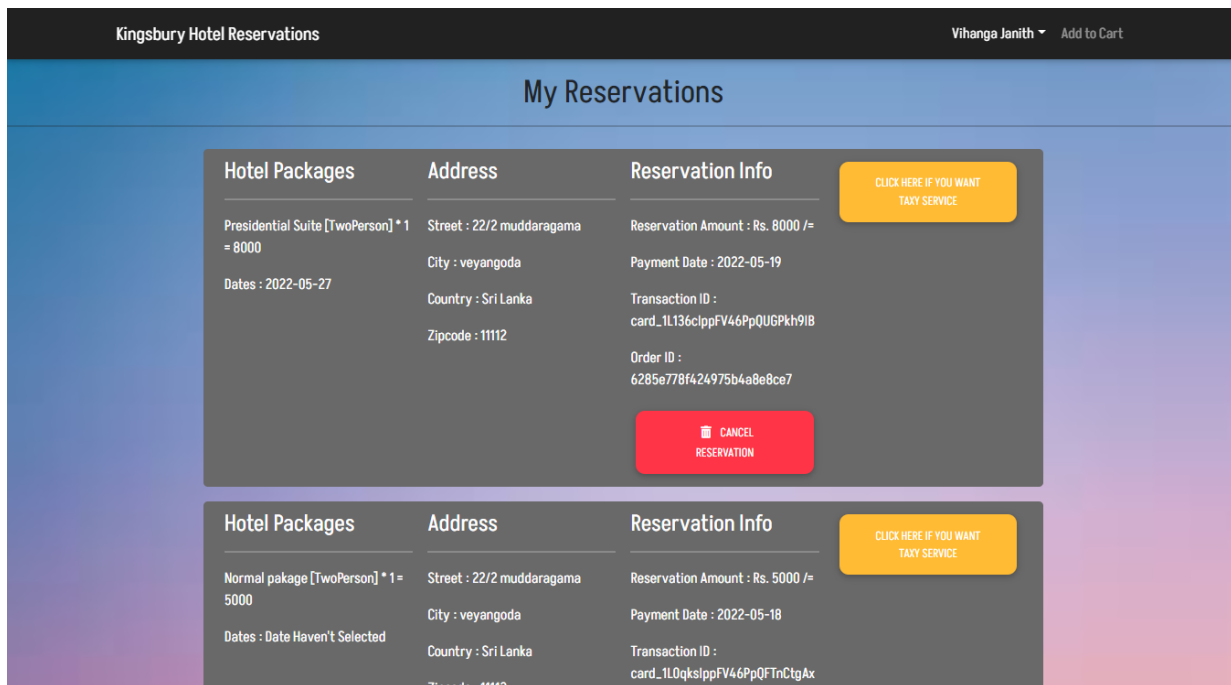
Services
Hotel Packages
Taxi Service

[f](#) [t](#) [i](#)

4.3 All Hotel Packages



4.4 Reservation List



4.5 Booked Taxi

Kingsbury Hotel Reservations Vihanga Janith ▾ [Add to Cart](#)

Booked Taxi Services

1) Booked Taxi Info

CANCEL TAXI RESERVATION

Address: 22/2 muddaragama, veyangoda, Sri Lanka

Booking date : 2022-05-26

Booking Created Date : 2022-05-08

2) Booked Taxi Info

CANCEL TAXI RESERVATION

Address: 22/2 muddaragama, veyangoda, Sri Lanka

Booking date : 2022-05-21

Booking Created Date : 2022-05-08

[Visit Us](#)
[Address](#)
[Terms](#)
[Services](#)

4.6 Users List













Kingsbury Hotel Reservations Vihanga Janith ▾ [Add to Cart](#)

[Admin Panel](#)
[Userslist](#)
[Packages List](#)
[Add New Package](#)
[Reservations List](#)
[Taxi List](#)

Users List

	Name	email	System Admins	Hotel Admins	All Users	User Actions	Create Hotel Admins
1)	Vihanga Janith (System Admin)	vihangajaniith99@gmail.com	System Admin	Hotel Admin	Staff	<div>BLOCK USER</div>	<div>REMOVE HOTEL ADMIN</div>
2)	Sandaru	sandarusenadeera2@gmail.com	Not Admin	Hotel Admin	Staff	<div>BLOCK USER</div>	<div>REMOVE HOTEL ADMIN</div>
3)	john	v4h4g4n@gmail.com	Not Admin	Not Hotel Admin	User	<div>BLOCK USER</div>	<div>MAKE HOTEL ADMIN</div>
4)	S.A.V.J.SENADEERA	it20219420@my.sliit.lk	Not Admin	Not Hotel Admin	User	<div>BLOCK USER</div>	<div>MAKE HOTEL ADMIN</div>
5)	hiks	kavindupro40@gmail.com	Not Admin	Not Hotel Admin	User	<div>BLOCK USER</div>	<div>MAKE HOTEL ADMIN</div>
6)	hiruni	hiruni@gmail.com	Not Admin	Hotel Admin	Staff	<div>BLOCK USER</div>	<div>REMOVE HOTEL ADMIN</div>

4.7 Package List

Kingsbury Hotel Reservations				Vihanga Janith ▾	Add to Cart
Admin Panel					
Userslist Packages List Add New Package Reservations List Taxi List					
Packages List					
	Name	Prices	Category	Actions	
1)	Murphy Suite	Two Person : 4000 Four Person : 7000 Family : 9000	normal	 EDIT	 DELETE
2)	Presidential Suite	Two Person : 8000 Four Person : 10000 Family : 13000	luxury	 EDIT	 DELETE
3)	Junior Suite	Two Person : 2000 Four Person : 5000 Family : 8000	normal	 EDIT	 DELETE
4)	Executive Suite	Two Person : 5000 Four Person : 8000 Family : 10000	luxury	 EDIT	 DELETE
5)	Connecting room	Two Person : 3000 Four Person : 6000 Family : 8000	normal	 EDIT	 DELETE
6)	Cabana	Two Person : 7000 Four Person : 10000 Family : 13000	luxury	 EDIT	 DELETE

4.8 Add Package

Kingsbury Hotel Reservations				Vihanga Janith ▾	Add to Cart
Admin Panel					
Userslist Packages List Add New Package Reservations List Taxi List					
Add Package					
<div> <div>Name</div> <div>Two Person Price</div> <div>Four Person Price</div> <div>Family Price</div> <div>Image</div> <div>Description</div> <div>Category</div> <div>ADD PACKAGE</div> </div>					
Visit Us					
<div> <div>View larger map</div> <div>The Kingsbury Hotel Parking Lot</div> </div>					
Address					
<div> <div>26/0</div> <div>Malabe</div> <div>Endunke</div> </div>					
Terms					
<div> <div>Pay visa</div> <div>Copy Right</div> <div>Mark Janith</div> </div>					
Services					
<div> <div>Hotel Packages</div> <div>Taxi Service</div> </div>					

4.9 Reservation List

Kingsbury Hotel Reservations							Vihanga Janith ▾	Add to Cart
Admin Panel								
Userslist Packages List Add New Package Reservations List Taxi List								
Reservations List								
Order ID	Email	User ID	Amount	Payed Date	Booked Date	Status	Actions	
6285e778f424975b4a8e8ce7	vihangajanith99@gmail.com	62643d41ce472cac74e68bf9	8000	2022-05-19	2022-05-27	CHECKED IN	DELETE RESERVATION	
6285701de3b2ea5de32ec966	kamindu@gmail.com	62842073a3ae42fa9ab8511a	6000	2022-05-18	2022-05-27	CHECKED IN	DELETE RESERVATION	
628529d8e1c3baa2a299c2cb	vihangajanith99@gmail.com	62643d41ce472cac74e68bf9	11000	2022-05-18	2022-05-26 2022-05-21	CHECKED IN	DELETE RESERVATION	
628528f39dff4d1868aa3491	vihangajanith99@gmail.com	62643d41ce472cac74e68bf9	20000	2022-05-18	2022-05-28	CHECKED IN	DELETE RESERVATION	
62852508fb602b153e45477	vihangajanith99@gmail.com	62643d41ce472cac74e68bf9	42000	2022-05-18	2022-05-28	CHECKED IN	DELETE RESERVATION	
62851a6eeafa0ac21e61b1e4	kamindu@gmail.com	62842073a3ae42fa9ab8511a	5000	2022-05-18	2022-05-20	CHECKED IN	DELETE RESERVATION	

4.10 Taxi Reservation List

Kingsbury Hotel Reservations							Vihanga Janith ▾	Add to Cart
Admin Panel								
Userslist Packages List Add New Package Reservations List Taxi List								
Taxi List								
Taxi Booking ID	Customer Name	Address	Booked Date	Phone	Reservation Date	Actions		
62851acbeafa0ac21e61b1ef	Kamindu Gayantha	ratnapura, ratnapura, United States	2022-05-20	715273881	2022-05-18	DELETE RESERVATION		
627771ef1ae8223e0b7d40c8	Vihanga Janith	22/2 muddaragama, veyangoda, Sri Lanka	2022-05-26	710115442	2022-05-08	DELETE RESERVATION		
627767f6dda9bb416f8dd479	Vihanga	22/2 muddaragama, veyangoda, Sri Lanka	2022-05-26	710115442	2022-05-08	DELETE RESERVATION		

Visit Us

Address

26/D
Malabe
Kaduvela
Colombo

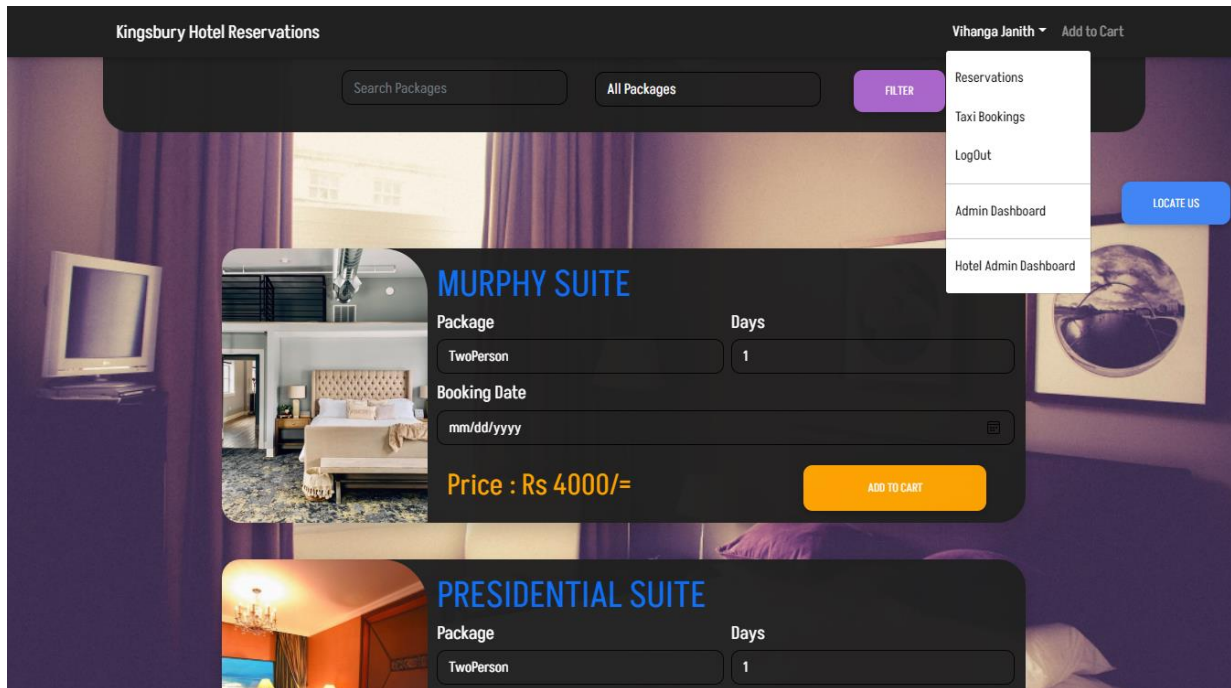
Terms

Pay visa
Copy Right
Must Agree
Must Read

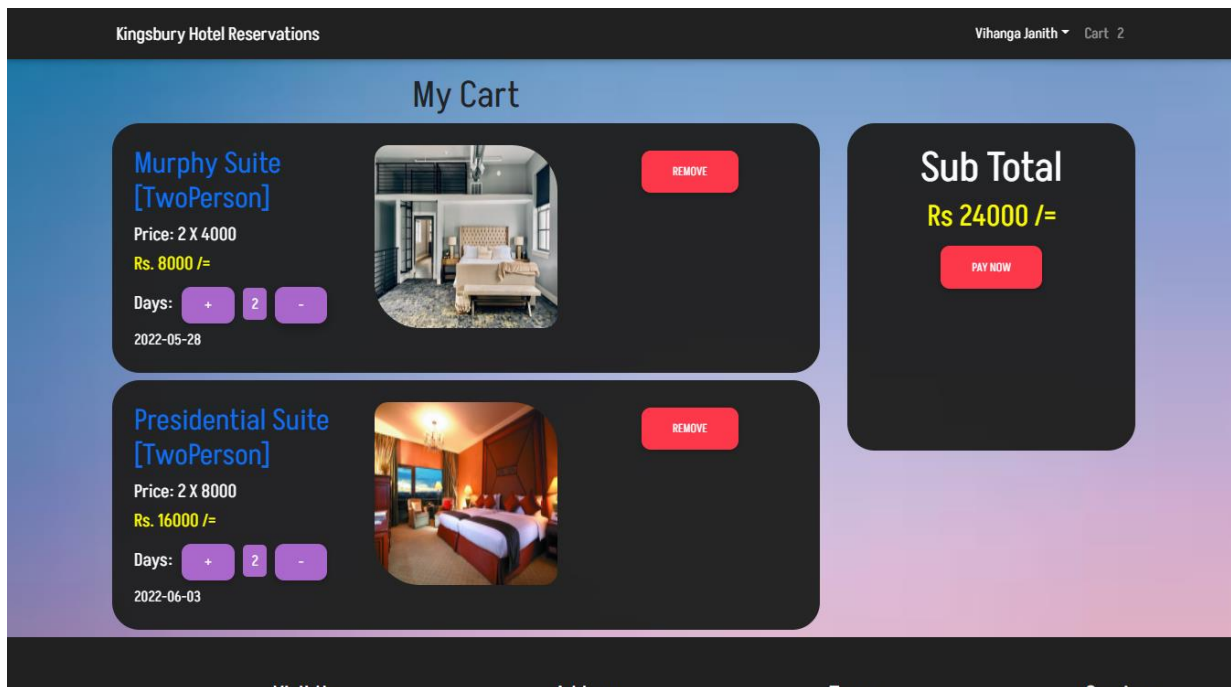
Services

Hotel Packages
Taxi Service

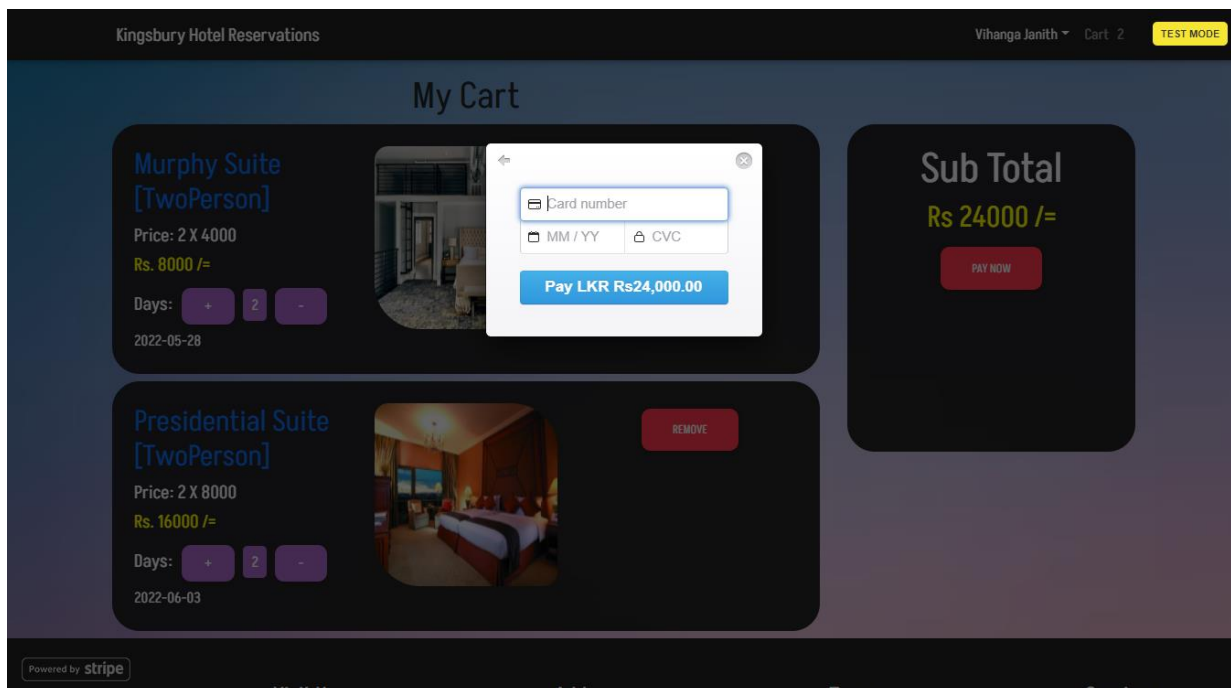
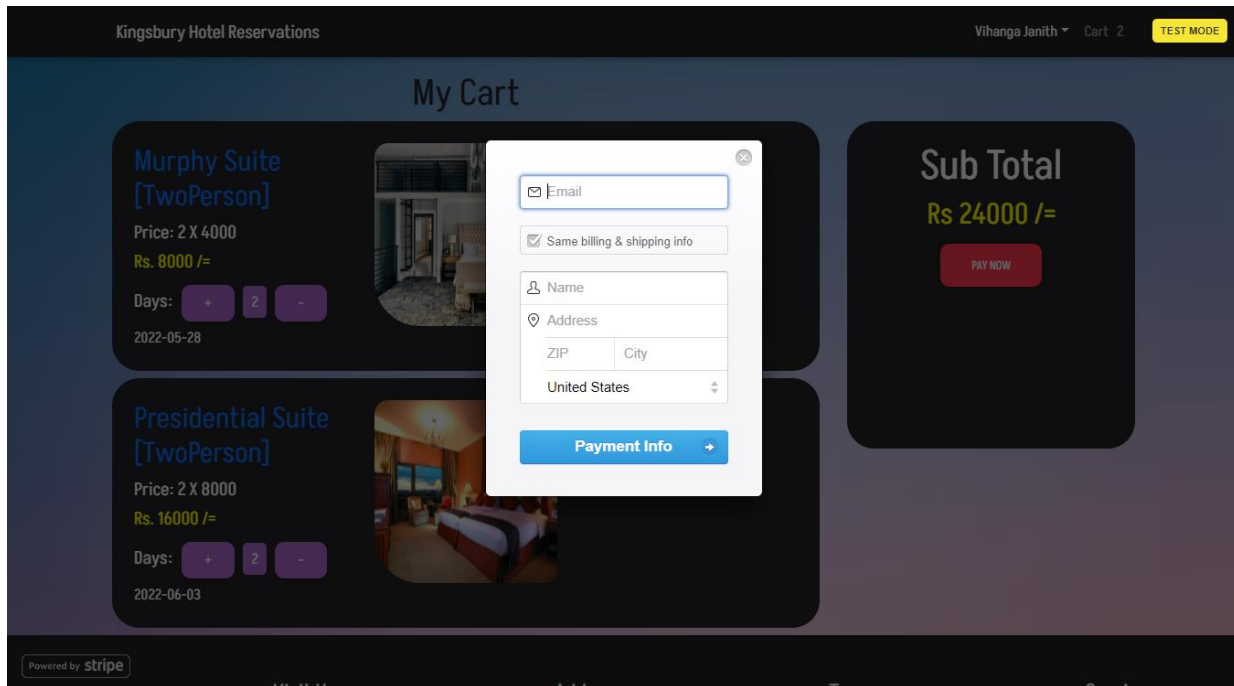
4.11 Admin Menu



4.12 Cart



4.13 Payment




4.14 Hotel Admin Menu

Kingsbury Hotel Reservations

Sandaru ▾ Add to Cart

Search Packages All Packages FILTER


Reservations
Taxi Bookings
LogOut
Hotel Admin Dashboard LOCATE US

 **MURPHY SUITE**

Package Days
TwoPerson 1

Booking Date
mm/dd/yyyy

Price : Rs 4000/= ADD TO CART

 **PRESIDENTIAL SUITE**

Package Days
TwoPerson 1


4.15 Customer Menu

Kingsbury Hotel Reservations

john ▾ Add to Cart

Search Packages All Packages FILTER


Reservations
Taxi Bookings
LogOut
LOCATE US

 **MURPHY SUITE**

Package Days
TwoPerson 1

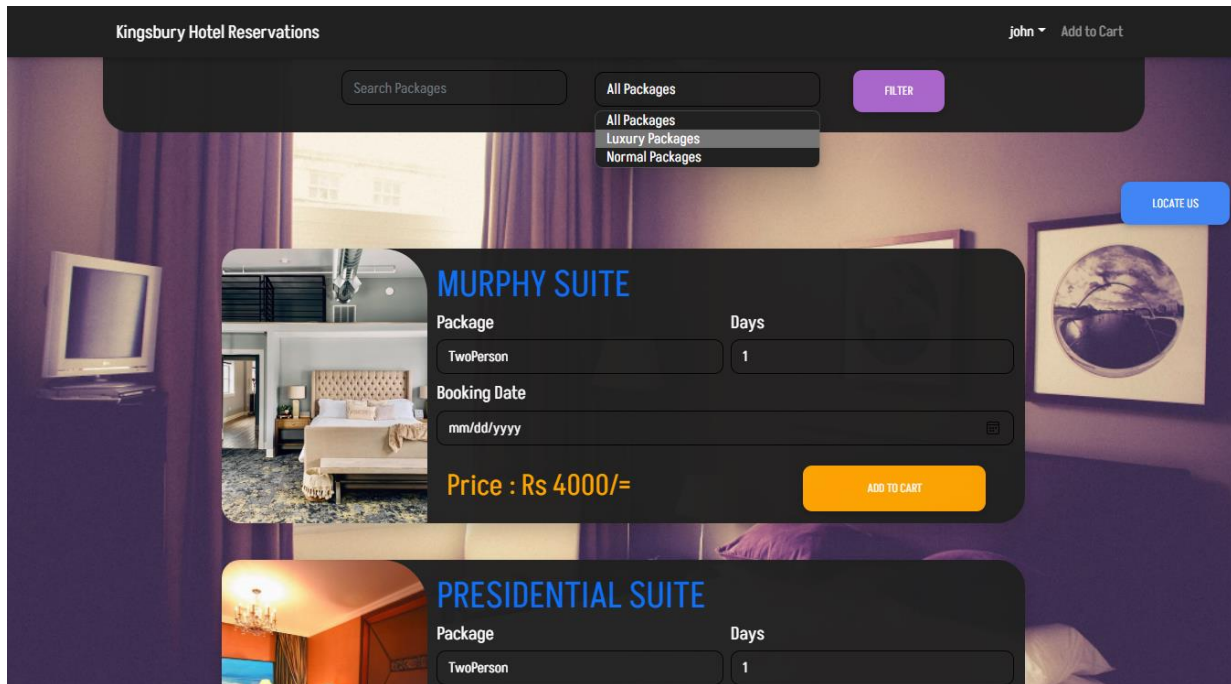
Booking Date
mm/dd/yyyy

Price : Rs 4000/= ADD TO CART

 **PRESIDENTIAL SUITE**

Package Days
TwoPerson 1

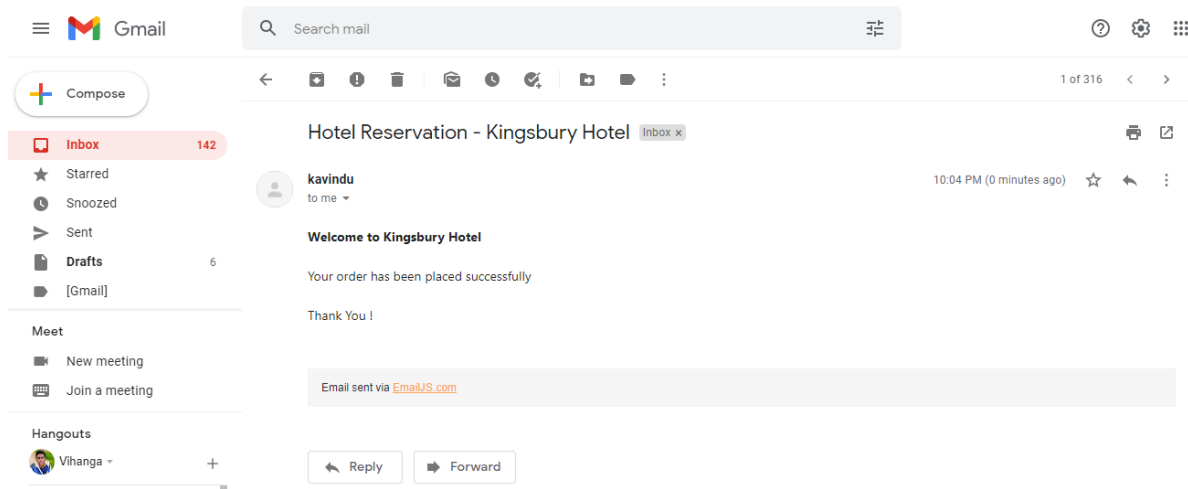
4.16 Hotel Package Filter



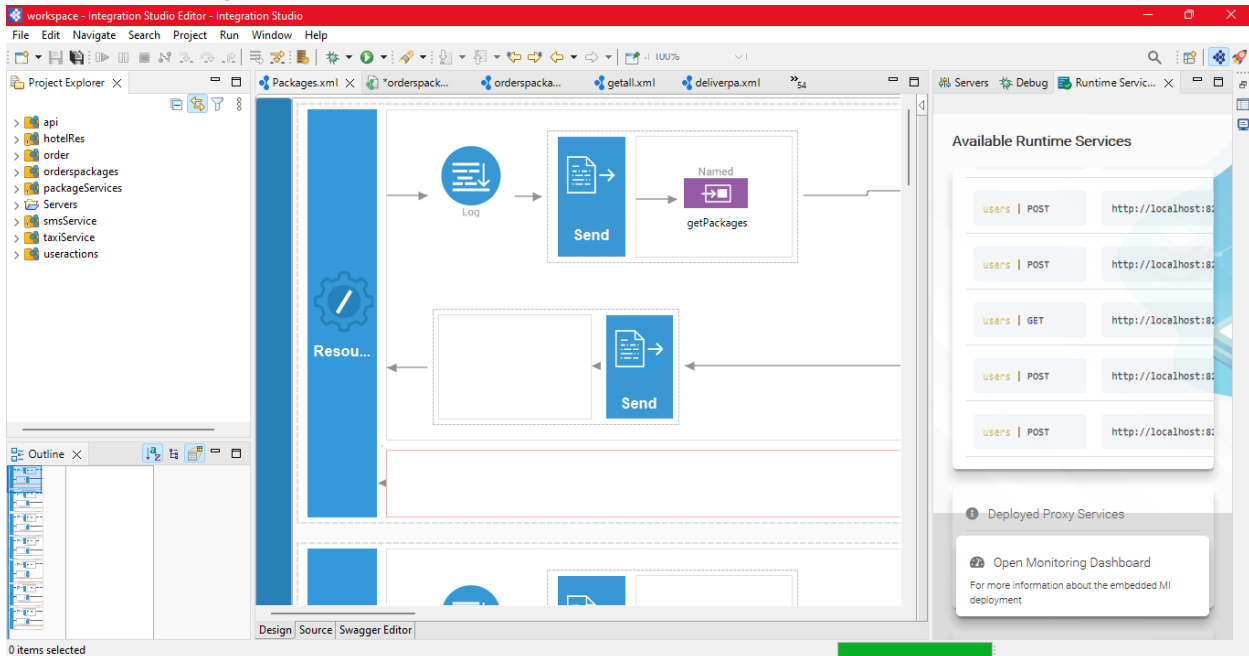
4.17 Reservation Confirmation SMS



4.18 Reservation Confirmation Email



4.19 WS2 Integration Studio



5. Appendix

5.1 Order Service

OrderModel.js

```
const mongoose = require('mongoose');

const orderSchema = mongoose.Schema({
  name: {
    type: String,
    require
  },
  email: {
    type: String,
    require
  },
  userid: {
    type: String,
    require
  },
  orderItems : [],

  shippingAddress: {
    type: Object
  },
  orderAmount: {
    type: Number,
    require
  },
  isDelivered: {
    type: Boolean,
    require,
    default: false
  },
  transactionId: {
    type: String,
    require
  },
},
```

```

{
  timestamps: true
})

module.exports = mongoose.model("orders",orderSchema)

```

ordersRoute.js

```

const express = require('express');
const router = express.Router();
const stripe =
require('stripe')('sk_test_51Kry14IppFV46PpQ87ZKsImeawBxxAYfdwBG9xfSZWPKj8000wW6m
SAjzDXUVjmnA6vlnoopGU0aSIUZVmV60mng00liG38ZCh')
const { v4: uuidv4 } = require('uuid');
const Order = require('../models/orderModel');

//Create Reservation
router.post('/placeorder', async (req, res) => {

  const {token,subtotal, currentUser, cartItems} = req.body

  try{
    const customer = await stripe.customers.create({
      email: token.email,
      source: token.id
    })
    const payment = await stripe.charges.create({
      amount: subtotal * 100,
      currency: 'lkr',
      customer: customer.id,
      receipt_email: token.email,
    },{
      idempotencyKey: uuidv4()
    } )

    if(payment){
      const newOrder = new Order({
        name: currentUser.name,
        email: currentUser.email,

```



```

        userid: currentUser._id,
        orderItems: cartItems,
        date: cartItems.date,
        orderAmount: subtotal,
        shippingAddress: {
            street: token.card.address_line1,
            city: token.card.address_city,
            country: token.card.address_country,
            pincode: token.card.address_zip
        },
        transactionId: payment.source.id,

    })
    newOrder.save()
    res.send("order placed successfully")
} else {
    res.send("payment failed")
}

}

catch(error){
    return res.status(400).json({message: error});
}

})

//Get User Created Reservations
router.post('/getuserorders', async (req, res) => {
    const {userid} = req.body

    try{
        const orders = await Order.find({userid : userid}).sort({_id : -1})
        res.send(orders)

    } catch(e){
        return res.status(400).json({message: e});
    }

})

//Get ALL Reservations
router.get('/getallorders', async (req, res) => {

    try{
        const orders = await Order.find({}).sort({_id : -1})

```

```

    res.send(orders)

  }catch(e){
    return res.status(400).json({message: e});
  }
})

//Create Active Customer
router.post('/deliverorder', async (req, res) => {
  const orderid = req.body.orderid

  try{
    const order = await Order.findOne({_id: orderid})
    order.isDelivered = true
    await order.save()
    res.send("order delivered successfully")

  }catch(e){
    return res.status(400).json({message: e});
  }
})

//Get Reservation By ID
router.post('/getorderbyid', async(req, res) => {
  const orderid = req.body.orderid
  try{
    const order = await Order.findOne({_id: orderid});
    res.send(order);

  }catch(e){

    return res.status(400).json({message: e});
  }
}

)

//Cancel Reservation
router.post('/deleteorder', async(req, res) => {
  const orderid = req.body.orderid
  try{
    await Order.findOneAndDelete({_id: orderid});
    res.send("booking deleted successfully");

  }catch(e){

```

```

        return res.status(400).json({message: e});
    }
}
)

module.exports = router;

```

db.js

```

const mongoose = require('mongoose');

var mongoURL = 'mongodb+srv://vjs9c:vjs9c@cluster0.8kweo.mongodb.net/mern-pizza'

mongoose.connect(mongoURL, { useNewUrlParser: true, useUnifiedTopology: true });

var db = mongoose.connection;

db.on('connected', () => {
    console.log('Connected to MongoDB');
})

db.on('error', () => {
    console.log('Error connecting to MongoDB: ');
})

module.exports = mongoose

```

Package.json

```
{
  "name": "travel",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "clouinary": "^1.29.1",
    "cors": "^2.8.5",
    "dotenv": "^16.0.0",
    "express": "^4.17.3",
    "mongoose": "^6.3.1",
    "multer": "^1.4.4",
    "nodemon": "^2.0.15",
    "router": "^1.3.6",
    "shoutout-sdk": "^3.0.4",
    "stripe": "^8.219.0",
    "uuid": "^8.3.2"
  }
}
```

Server.js

```
const express = require('express');

const db = require('./db');
const cors = require('cors')

const app = express();

app.use(cors())
app.use(express.json());

const ordersRoute = require('./routes/ordersRoute.js');
```

```

app.use('/api/orders/', ordersRoute);

app.get('/', (req, res) => {
  res.send('Server Working');
});

const port = process.env.PORT || 5001;

app.listen(port, () => console.log(`Order Service is started on port ${port}`));

```

5.2 Package Service

packageModel.js

```

const mongoose = require('mongoose');

const packageSchema = mongoose.Schema({
  name: {
    type: "string",
    require
  },
  variants: [],
  prices: [],
  category: {
    type: "string",
    require
  },
  image: {
    type: "string",
    require
  },
  description: {
    type: "string",
    require
  }
},
{
  timestamps: true
})

```

```
const packageModel = mongoose.model("HotelPackages",packageSchema)

module.exports = packageModel
```

PackageRoute.js

```
const express = require('express');
const router = express.Router();
const Package = require('../models/packageModel');

//Get ALL Hotel Packages
router.get('/getallpackages', async(req, res) => {
  try{
    const packages = await Package.find({});
    res.send(packages);
  }catch(e){
    return res.status(400).json({message: e});
  }
});

//Add New Hotel Package
router.post('/addpackage', async(req, res) => {
  const package = req.body.pack;

  try{

    const newpackage = new Package({
      name: package.name,
      image: package.image,
      variants: ['TwoPerson', 'FourPerson', 'Family'],
      description: package.description,
      category: package.category,
      prices: [package.prices]

    })

    await newpackage.save();
    res.send("Package added successfully");
  }catch(e){
```

```

        return res.status(400).json({message: e});
    }

    })

    //Get Hotel Package By ID
    router.post('/getpackagebyid', async(req, res) => {
        const packageid = req.body.packageid
        try{
            const package = await Package.findOne({_id: packageid});
            res.send(package);

        }catch(e){

            return res.status(400).json({message: e});
        }
    }

    )

    //Edit Hotel Package Details
    router.post ("/editpackage" , async(req, res) => {

        const editedpackage = req.body.editedpackage;
        try{
            const package = await Package.findOneAndUpdate({_id: editedpackage._id},
editedpackage);
            package.name = editedpackage.name;
            package.image = editedpackage.image;
            package.description = editedpackage.description;
            package.category = editedpackage.category;
            package.prices = [editedpackage.prices];

            await package.save();
            res.send("Package edited successfully");

        }catch(e){
            return res.status(400).json({message: e});
        }

    })

    //Delete Hotel Package
    router.post('/deletepackage', async(req, res) => {
        const packageid = req.body.packageid

```

```

    try{
      await Package.findOneAndDelete({_id: packageid});
      res.send("Package deleted successfully");

    }catch(e){

      return res.status(400).json({message: e});
    }
  }
)

module.exports = router;

```

db.js

```

const mongoose = require('mongoose');

var mongoURL = 'mongodb+srv://vjs9c:vjs9c@cluster0.8kweo.mongodb.net/mern-pizza'

mongoose.connect(mongoURL, { useNewUrlParser: true, useUnifiedTopology: true });

var db = mongoose.connection;

db.on('connected', () => {
  console.log('Connected to MongoDB');
})

db.on('error', () => {
  console.log('Error connecting to MongoDB: ');
})

module.exports = mongoose

```


Package.json

```
{
  "name": "travel",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "clouddinary": "^1.29.1",
    "cors": "^2.8.5",
    "dotenv": "^16.0.0",
    "express": "^4.17.3",
    "mongoose": "^6.3.1",
    "multer": "^1.4.4",
    "nodemon": "^2.0.15",
    "router": "^1.3.6",
    "shoutout-sdk": "^3.0.4",
    "stripe": "^8.219.0",
    "uuid": "^8.3.2"
  }
}
```

Server.js

```
const express = require('express');

const db = require('./db');
const cors = require('cors')

const app = express();

app.use(cors())
app.use(express.json());

const packageRoute = require('./routes/packageRoute.js');
```

```

app.use('/api/packages/', packageRoute);

app.get('/', (req, res) => {
  res.send('Server Working');
});

const port = process.env.PORT || 5000;

app.listen(port, () => console.log(`Package Server is started on port ${port}`));

```

5.3 User Authentication Service

userModel.js

```

const mongoose = require('mongoose');

const userSchema = mongoose.Schema({
  name: {
    type: "string",
    default: "User0",
  },
  email: {
    type: "string",
    require
  },
  password: {
    type: "string",
    require
  },
  isAdmin: {
    type: "boolean",
    require,
    default: false
  },
  isHotelAdmin: {
    type: "boolean",

```

```

        require,
        default: false
    },
    phone: {
        type: "Number",
        require,
    }
},
{
    timestamps: true
})

module.exports = mongoose.model("user",userSchema)

```

userRoute.js

```

const express = require('express');
const router = express.Router();
const mongoose = require('mongoose');
const User = require('../models/userModel');

//Customer Registration
router.post('/register', async (req, res) => {

    const { name, email, password , phone } = req.body;
    const newUser = new User({name, email, password , phone});

    try {
        newUser.save()
        res.send("user registered")
    } catch (e) {
        return res.status(400).json({message: e});
    }

})

//Customer Login
router.post('/login', async (req, res) => {

```

```

const { email, password } = req.body;

try{
  const user = await User.find({email, password})

  if(user.length > 0){

    const currentUser = {
      name: user[0].name,
      email: user[0].email,
      isAdmin: user[0].isAdmin,
      phone: user[0].phone,
      _id: user[0]._id,
      isHotelAdmin: user[0].isHotelAdmin,

    }
    res.send(currentUser)

  }
  else{
    return res.status(400).json({message: "user login failed"})
  }
}catch(e){
  return res.status(400).json({message: "something went wrong"});
}

}))

//Get ALL User Details
router.get('/getallusers', async(req, res) => {
  try{
    const users = await User.find({});
    res.send(users);
  }catch(e){
    return res.status(400).json({message: e});
  }

});

//Delete or Block User
router.post('/deleteuser', async(req, res) => {

```

```

    const userid = req.body.userid
    try{
      await User.findOneAndDelete({_id: userid});
      res.send("User deleted successfully");

    }catch(e){

      return res.status(400).json({message: e});
    }
  }
)
//Create Hotel Admin
router.post('/makehoteladmin', async (req, res) => {
  const userid = req.body.userid

  try{
    const hadmin = await User.findOne({_id: userid})
    hadmin.isHotelAdmin = true
    await hadmin.save()
    res.send("H admin successfully")

  }catch(e){
    return res.status(400).json({message: e});
  }

})
//Remove Hotel Admin
router.post('/removehoteladmin', async (req, res) => {
  const userid = req.body.userid

  try{
    const hadmin = await User.findOne({_id: userid})
    hadmin.isHotelAdmin = false
    await hadmin.save()
    res.send("H admin removed successfully")

  }catch(e){
    return res.status(400).json({message: e});
  }

})

module.exports = router;

```

db.js

```
const mongoose = require('mongoose');

var mongoURL = 'mongodb+srv://vjs9c:vjs9c@cluster0.8kweo.mongodb.net/mern-pizza'

mongoose.connect(mongoURL, { useNewUrlParser: true, useUnifiedTopology: true });


var db = mongoose.connection;

db.on('connected', () => {
  console.log('Connected to MongoDB');
})

db.on('error', () => {
  console.log('Error connecting to MongoDB: ');
})

module.exports = mongoose
```

package.json

```
{
  "name": "travel",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "clouinary": "^1.29.1",
    "cors": "^2.8.5",
    "dotenv": "^16.0.0",
    "express": "^4.17.3",
    "mongoose": "^6.3.1",
    "multer": "^1.4.4",
  }
}
```

```
"nodemon": "^2.0.15",  
"router": "^1.3.6",  
"shoutout-sdk": "^3.0.4",  
"stripe": "^8.219.0",  
"uuid": "^8.3.2"  
}  
}
```

Server.js

```
const express = require('express');  
  
const db = require('./db');  
const cors = require('cors')  
  
const app = express();  
  
app.use(cors())  
app.use(express.json());  
  
const userRoute = require('./routes/userRoute.js');  
  
app.use('/api/users/', userRoute);  
  
app.get('/', (req, res) => {  
  res.send('Server Working');  
});  
  
const port = process.env.PORT || 5003;  
  
app.listen(port, () => console.log(`User Authentication Service is started on  
port ${port}`));
```

5.4 Taxi Service

taxiModel.js

```
const mongoose = require('mongoose');

const taxiSchema = mongoose.Schema({
  id: {
    type: "string",
  },
  userid: {
    type: "string",
  },
  name: {
    type: "string",
    default: "User0",
  },
  email: {
    type: "string",
    require
  },
  address: {
    type: "string",
    require
  },
  date: {
    type: "string",
    require,
    default: "not booked"
  },
  phone: {
    type: "Number",
    require,
  }
},
{
  timestamps: true
})
```



```
module.exports = mongoose.model("taxiUsers",taxiSchema)
```

taxiRoute.js

```
const express = require('express');
const router = express.Router();
const mongoose = require('mongoose');
const Taxi = require('../models/taxiModel');

//Book Taxi
router.post('/book', async (req, res) => {

    const { id , userid ,name, email, address,date ,phone } = req.body;
    const bookTaxi = new Taxi({id , userid, name, email, address,date ,phone})

    try {
        bookTaxi.save()
        res.send("taxi booked")

    } catch (e) {
        return res.status(400).json({message: e});
    }

})

//Get Ordered Taxi By User ID
router.post('/getorderedtaxi', async (req, res) => {
    const {userid} = req.body

    try{
        const taxiorders = await Taxi.find({userid : userid}).sort({_id : -1})
        res.send(taxiorders)

    }catch(e){
        return res.status(400).json({message: e});
    }

})

//Cancel Ordered Taxi
router.post('/deletetaxiorder', async(req, res) => {
    const taxiid = req.body.taxiid
```

```

    try{
      await Taxi.findOneAndDelete({_id: taxiid});
      res.send("booking deleted successfully");

    }catch(e){

      return res.status(400).json({message: e});
    }
  }
)
//Get ALL Taxi Bookings
router.get('/getalltaxi', async (req, res) => {

  try{
    const taxi = await Taxi.find({}).sort({_id : -1})
    res.send(taxi)

  }catch(e){
    return res.status(400).json({message: e});
  }

})

module.exports = router;

```

db.js

```

const mongoose = require('mongoose');

var mongoURL = 'mongodb+srv://vjs9c:vjs9c@cluster0.8kweo.mongodb.net/mern-pizza'
mongoose.connect(mongoURL, { useNewUrlParser: true, useUnifiedTopology: true });

var db = mongoose.connection;

db.on('connected', () => {
  console.log('Connected to MongoDB');
})

db.on('error', () => {
  console.log('Error connecting to MongoDB: ');
})

```

```
module.exports = mongoose
```

package.json

```
{
  "name": "travel",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "cloundinary": "^1.29.1",
    "cors": "^2.8.5",
    "dotenv": "^16.0.0",
    "express": "^4.17.3",
    "mongoose": "^6.3.1",
    "multer": "^1.4.4",
    "nodemon": "^2.0.15",
    "router": "^1.3.6",
    "shoutout-sdk": "^3.0.4",
    "stripe": "^8.219.0",
    "uuid": "^8.3.2"
  }
}
```

Server.js

```
const express = require('express');

const Taxi = require('./models/taxiModel');

const db = require('./db');
const cors = require('cors')

const app = express();
```

```
app.use(cors())
app.use(express.json());

const taxiRoute = require('./routes/taxiRoute.js');

app.use('/api/taxi/', taxiRoute);

app.get('/', (req, res) => {
  res.send('Server Working');
});

const port = process.env.PORT || 5002;

app.listen(port, () => console.log(`Taxi Service is started on port ${port}`));
```

5.5 SMS Service

Sms.js

```
const express = require('express');
const router = express.Router();
var ShoutoutClient = require('shoutout-sdk');

//Reservation Confirmation SMS
router.post('/send', async(req, res) => {
var apiKey =
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOiIxZDA3MDgzMC1kNDEzLTExZWMTYjlkOS1lZjg3OTRmZDFkNTQiLCJzdWIiOiJTSE9VVE9VVf9BUeU1fVFNFUlIsIm1hdCI6MTY1MjU5MzkwOSwiZXhwIjoxOTY4MjEzMTA5LCJzY29wZXMiOnsiYWN0aXZpdGllcyI6WyJyZWFKIiwid3JpdGUxSwibWVzc2FnZSI6MTY3OTg1Iiwic29fdXNlcl9yb2x1IjoiaXNlciIsInNvX3Byb2ZpbGUiOiJhbGwiLCJzb191c2VyX25hbWUiOiJiLCJzb19hcGlrZXkiOiJub251In0.HDHTT87S3o55S7tI8G-1KpayqPpmqiftJYNTh9K1RC8';

var debug = true, verifySSL = false;

const { destinations } = req.body;
```

```

var client = new ShoutoutClient(apiKey, debug, verifySSL);

var message = {
  "content": {"sms":
    "Welcome to Covanro Hotel Reservations || Your Reservation Has Been Placed
    Successfully !",},
  "destinations": destinations,
  "source": "ShoutDEMO",
  "transports": ["SMS"]
};

client.sendMessage(message, (error, result) => {
  if (error) {
    console.error('Error sending message!',error);
  } else {
    console.log('Sending message successful!',result);
  }
});

});

module.exports = router;

```

db.js

```

const mongoose = require('mongoose');

var mongoURL = 'mongodb+srv://vjs9c:vjs9c@cluster0.8kweo.mongodb.net/mern-pizza'

mongoose.connect(mongoURL, { useNewUrlParser: true, useUnifiedTopology: true });

var db = mongoose.connection;

db.on('connected', () => {
  console.log('Connected to MongoDB');
})

db.on('error', () => {
  console.log('Error connecting to MongoDB: ');
})

```

```
module.exports = mongoose
```

package.json

```
{
  "name": "travel",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "cloudinary": "^1.29.1",
    "cors": "^2.8.5",
    "dotenv": "^16.0.0",
    "express": "^4.17.3",
    "mongoose": "^6.3.1",
    "multer": "^1.4.4",
    "nodemon": "^2.0.15",
    "router": "^1.3.6",
    "shoutout-sdk": "^3.0.4",
    "stripe": "^8.219.0",
    "uuid": "^8.3.2"
  }
}
```

Server.js

```
const express = require('express');

const db = require('./db');
const cors = require('cors')

const app = express();

app.use(cors())
app.use(express.json());

const sms = require('./routes/sms.js');

app.use('/api/sms/', sms);

app.get('/', (req, res) => {
  res.send('Server Working');
});

const port = process.env.PORT || 5004;

app.listen(port, () => console.log(`Sms Service started on port ${port}`));
```

5.6 Frontend

packageActions.js

```
import axios from "axios";

export const getAllPackages = () => async (dispatch) => {
  dispatch({ type: "GET_PACKAGES_REQUEST" });

  try {
    const response = await axios.get(
      // "http://localhost:5000/api/packages/getallpackages"
      "http://localhost:8290/packages/getpackages"
    );
    console.log(response);
    dispatch({ type: "GET_PACKAGES_SUCCESS", payload: response.data });
  } catch (error) {
    dispatch({ type: "GET_PACKAGES_FAILED", payload: error });
  }
};

export const filterPackages = (searchkey, category) => async (dispatch) => {
  var filteredPackages;
  dispatch({ type: "GET_PACKAGES_REQUEST" });

  try {
    const response = await axios.get(
      // "http://localhost:5000/api/packages/getallpackages"
      "http://localhost:8290/packages/getpackages"
    );
    filteredPackages = response.data.filter((packages) =>
      packages.name.toLowerCase().includes(searchkey)
    );

    if (category !== "all") {
      filteredPackages = response.data.filter(
        (packages) => packages.category.toLowerCase() === category
      );
    }

    dispatch({ type: "GET_PACKAGES_SUCCESS", payload: filteredPackages });
  } catch (error) {
    dispatch({ type: "GET_PACKAGES_FAILED", payload: error });
  }
};
```



```

export const addPackage = (pack) => async (dispatch) => {
  dispatch({ type: "ADD_PACKAGE_REQUEST" });

  try {
    const response = await axios.post(
      // "http://localhost:5000/api/packages/addpackage",
      "http://localhost:8290/packages/addpackage",
      { pack }
    );
    console.log(response);
    dispatch({ type: "ADD_PACKAGE_SUCCESS" });
    alert("Package added successfully!");
  } catch (error) {
    dispatch({ type: "ADD_PACKAGE_FAILED", payload: error });
  }
};

export const getPackageById = (packageid) => async (dispatch) => {
  dispatch({ type: "GET_PACKAGEBYID_REQUEST" });

  try {
    const response = await axios.post(
      // "http://localhost:5000/api/packages/getpackagebyid",
      "http://localhost:8290/packages/getpackagebyid",
      { packageid }
    );
    console.log(response);
    dispatch({ type: "GET_PACKAGEBYID_SUCCESS", payload: response.data });
  } catch (error) {
    dispatch({ type: "GET_PACKAGEBYID_FAILED", payload: error });
  }
};

export const editPackage = (editedpackage) => async (dispatch) => {
  dispatch({ type: "EDIT_PACKAGE_REQUEST" });

  try {
    const response = await axios.post(
      // "http://localhost:5000/api/packages/editpackage",

      "http://localhost:8290/packages/editbyid",
      { editedpackage }
    );
    console.log(response);
    dispatch({ type: "EDIT_PACKAGE_SUCCESS" });
  }
};

```

```

    window.location.href = "/admin/packageslist";
  } catch (error) {
    dispatch({ type: "EDIT_PACKAGE_FAILED", payload: error });
  }
};

export const deletePackage = (packageid) => async (dispatch) => {
  try {
    const response = await axios.post(
      // "http://localhost:5000/api/packages/delepackage",
      "http://localhost:8290/packages/delepackage",
      { packageid }
    );
    alert("Package Deleted Successfully");
    console.log(response);
    window.location.reload();
  } catch (error) {
    alert("Package Deletion Failed");
    console.log(error);
  }
};

```

cartActions.js

```

export const addToCart = (pack, quantity, varient, date) => (dispatch, getState) =>
{

  var cartItem = {
    name: pack.name,
    _id: pack._id,
    image: pack.image,
    varient: varient,
    quantity: Number(quantity),
    date: date,
    prices : pack.prices,
    price: pack.prices[0][varient] * quantity,

  }
}

```

```

    console.log(cartItem)

    if(cartItem.quantity >10) {
        alert("Sorry, You can only pay for 10 days at a time");
    }
    else{
        if(cartItem.quantity <1){
            dispatch({type: 'DELETE_FROM_CART', payload: pack})
        }

        else{
            dispatch({type: 'ADD_TO_CART', payload: cartItem})
        }
    }

    const cartItems = getState().cartReducer.cartItems;
    localStorage.setItem('cartItems', JSON.stringify(cartItems))

}

export const deleteFromCart = (pack) => (dispatch, getState) => {
    dispatch({type: 'DELETE_FROM_CART', payload: pack})

    const cartItems = getState().cartReducer.cartItems;
    localStorage.setItem('cartItems', JSON.stringify(cartItems))
}

```

orderAction.js

```

import axios from "axios";

export const placeOrder = (token, subtotal) => async (dispatch, getState) => {
    dispatch({ type: "PLACE_ORDER_REQUEST" });
    const currentUser = getState().loginUserReducer.currentUser;
    const cartItems = getState().cartReducer.cartItems;

    try {
        const response = await axios.post(
            "http://localhost:8290/apis/placeorder",
            { token, subtotal, currentUser, cartItems }
        );
    }
}

```

```

    dispatch({ type: "PLACE_ORDER_SUCCESS" });
    console.log(response);
    localStorage.removeItem('cartItems')

  } catch (error) {
    dispatch({ type: "PLACE_ORDER_FAILED" });
    console.log(error);
  }
};

export const getUserOrders = () => async (dispatch, getState) => {

  const currentUser = getState().loginUserReducer.currentUser;

  dispatch({ type: "GET_USER_ORDERS_REQUEST" });

  try {
    const response = await axios.post(
      "http://localhost:8290/apis/getuserorders",
      { userid: currentUser._id }
    );

    console.log(response);
    dispatch({ type: "GET_USER_ORDERS_SUCCESS", payload: response.data });
  } catch (error) {
    dispatch({ type: "GET_USER_ORDERS_FAILED", payload: error });
  }
};

export const getAllOrders = () => async (dispatch, getState) => {
  const currentUser = getState().loginUserReducer.currentUser;

  dispatch({ type: "GET_ALLORDERS_REQUEST" });

  try {

```

```

const response = await axios.get(
  "http://localhost:8290/apis/getallorder"
);

console.log(response);
dispatch({ type: "GET_ALLORDERS_SUCCESS", payload: response.data });
} catch (error) {
  dispatch({ type: "GET_ALLORDERS_FAILED", payload: error });
}
};

export const deliverOrder = (orderid) => async (dispatch) => {
  try {
    const response = await axios.post(
      "http://localhost:8290/apis/deliverorder",
      { orderid }
    );

    console.log(response);

    const orders = await axios.get(
      "http://localhost:8290/apis/getallorder"
    )

    dispatch({ type: "GET_ALLORDERS_SUCCESS", payload: orders.data });

  } catch (error) {

  }
};

export const getOrderById = (orderid) => async (dispatch) => {
  dispatch({ type: "GET_ORDERBYID_REQUEST" });

  try {
    const response = await axios.post(
      "http://localhost:8290/apis/getorderbyid",
      { orderid }
    );
    console.log(response);
    dispatch({ type: "GET_ORDERBYID_SUCCESS", payload: response.data });
  } catch (error) {
    dispatch({ type: "GET_ORDERBYID_FAILED", payload: error });
  }
}

```

```

};

export const deleteOrder = (orderid) => async (dispatch) => {
  try {
    if(window.confirm('Do you want to cancel your reservation ?')){
      const response = await axios.post(
        "http://localhost:8290/apis/deleteorder",
        { orderid }
      );
      alert("Reservation Cancelled Successfully");
      console.log(response);
      window.location.reload();
    }
  } catch (error) {
    alert("order Deletion Failed");
    console.log(error);
  }
};

```

taxiAction.js

```

import axios from 'axios';

export const bookTaxi = (taxi) => async dispatch => {

  dispatch({type: 'BOOK_TAXI_REQUEST'})

  try{

    const response = await axios.post(
      // 'http://localhost:5000/api/taxi/book',
      // 'http://localhost:5002/api/taxi/book',
      "http://localhost:8290/taxiroute/booktaxi",
      taxi)
    console.log(response)
    dispatch({type: 'BOOK_TAXI_SUCCESS'})
    alert('Taxi Booked Successfully')
    window.location.href = '/taxiord'
  }

```

```

    } catch (error) {
      dispatch({ type: 'BOOK_TAXI_FAILURE', payload: error })
    }
  }
}

export const getOrderedTaxi = () => async (dispatch, getState) => {

  const currentUser = getState().loginUserReducer.currentUser;

  dispatch({ type: "GET_TAXI_ORDERS_REQUEST" });

  try {
    const response = await axios.post(
      // "http://localhost:5000/api/taxi/getorderedtaxi",
      // "http://localhost:5002/api/taxi/getorderedtaxi",
      "http://localhost:8290/taxiroute/orderedtaxi",
      { userid: currentUser._id }
    );

    console.log(response);
    dispatch({ type: "GET_TAXI_ORDERS_SUCCESS", payload: response.data });
  } catch (error) {
    dispatch({ type: "GET_TAXI_ORDERS_FAILED", payload: error });
  }
};

export const deleteTaxiOrder = (taxiid) => async (dispatch) => {
  try {
    const response = await axios.post(
      // "http://localhost:5000/api/taxi/deletetaxiorder",
      // "http://localhost:5002/api/taxi/deletetaxiorder",
      "http://localhost:8290/taxiroute/taxidelete",
      { taxiid }
    );
    alert("Booking Deleted Successfully");
    console.log(response);
    window.location.reload();

  } catch (error) {
    alert("booking Deletion Failed");
    console.log(error);
  }
}

```

```

};

export const getAlltaxi = () => async (dispatch, getState) => {
  const currentUser = getState().loginUserReducer.currentUser;

  dispatch({ type: "GET_ALLTAXI_REQUEST" });

  try {
    const response = await axios.get(
      // "http://localhost:5000/api/taxi/getalltaxi"
      // "http://localhost:5002/api/taxi/getalltaxi"
      "http://localhost:8290/taxiroute/getall"
    );

    console.log(response);
    dispatch({ type: "GET_ALLTAXI_SUCCESS", payload: response.data });
  } catch (error) {
    dispatch({ type: "GET_ALLTAXI_FAILED", payload: error });
  }
};

```

userActions.js

```

import axios from 'axios';

export const registerUser = (user) => async dispatch => {

  dispatch({type: 'USER_REGISTER_REQUEST'})

  try{

    const response = await axios.post(
      // 'http://localhost:5000/api/users/register'
      // 'http://localhost:5003/api/users/register'
      'http://localhost:8290/users/userRegister'
    , user)
    console.log(response)
    dispatch({type: 'USER_REGISTER_SUCCESS'})
  }

```



```

    }catch(error){
      dispatch({type: 'USER_REGISTER_FAILURE', payload: error})
    }
  }
}

export const loginUser = (user) => async dispatch => {

  dispatch({type: 'USER_LOGIN_REQUEST'})

  try{
    const response = await axios.post(
      // 'http://localhost:5000/api/users/login'
      // 'http://localhost:5003/api/users/login'
      'http://localhost:8290/users/userLogin'
    , user)
    console.log(response)
    dispatch({type: 'USER_LOGIN_SUCCESS', payload: response.data})

    localStorage.setItem('currentUser', JSON.stringify(response.data))
    window.location.href = '/'
  }catch(error){
    dispatch({type: 'USER_LOGIN_FAILURE', payload: error})
  }
}

export const logoutUser = () =>dispatch => {
  localStorage.removeItem('currentUser')
  localStorage.removeItem('cartItems')

  window.location.href="/login"
}

export const getAllUsers = () => async (dispatch) => {
  dispatch({ type: "GET_USERS_REQUEST" });

  try {
    const response = await axios.get(
      // 'http://localhost:5000/api/users/getallusers'
      // 'http://localhost:5003/api/users/getallusers'
      'http://localhost:8290/users/getUsers'
    )
  }
}

```

```

    );
    console.log(response);
    dispatch({ type: "GET_USERS_SUCCESS", payload: response.data });
  } catch (error) {
    dispatch({ type: "GET_USERS_FAILED", payload: error });
  }
};

export const deleteUser = (userid) => async (dispatch) => {
  try {
    const response = await axios.post(
      // "http://localhost:5000/api/users/deleteuser",
      // "http://localhost:5003/api/users/deleteuser",
      "http://localhost:8290/users/deleteUser",

      { userid }
    );
    alert("User Deleted Successfully");
    console.log(response);
    window.location.reload();
  } catch (error) {
    alert("User Deletion Failed");
    console.log(error);
  }
};

export const makeHotelAdmin = (userid) => async (dispatch) => {
  try {
    const response = await axios.post(
      // "http://localhost:5000/api/users/makehoteladmin",
      // "http://localhost:5003/api/users/makehoteladmin",
      "http://localhost:8290/users/makeHotelAdmin",

      { userid }
    );

    console.log(response);
    alert("Hotel Admin ");

    const users = await axios.get(
      // "http://localhost:5000/api/users/getAllUsers"
      // "http://localhost:5003/api/users/getAllUsers"
      'http://localhost:8290/users/getUsers'
    );
  }
};

```

```

    )

    dispatch({ type: "GET_USERS_SUCCESS", payload: users.data });

  } catch (error) {

  }

};

export const removeHotelAdmin = (userid) => async (dispatch) => {
  try {
    const response = await axios.post(
      // "http://localhost:5000/api/users/removehoteladmin",
      // "http://localhost:5003/api/users/removehoteladmin",
      "http://localhost:8290/users/removeHotelAdmin",

      { userid }
    );

    console.log(response);
    alert("Hotel Admin removed ");

    const users = await axios.get(
      // "http://localhost:5000/api/users/getAllUsers"
      // "http://localhost:5003/api/users/getAllUsers"
      'http://localhost:8290/users/getUsers'
    )

    dispatch({ type: "GET_USERS_SUCCESS", payload: users.data });

  } catch (error) {

  }

};

```

Checkout.js

```
import React from 'react'
import StripeCheckout from 'react-stripe-checkout'
import { useDispatch, useSelector } from 'react-redux'
import { placeOrder } from '../actions/orderAction'
import Error from '../components/Error';
import Loading from '../components/Loading';
import Success from '../components/Success';
import emailjs from 'emailjs-com';
import axios from 'axios';
import { useEffect, useState } from 'react'

export default function Checkout({subtotal}) {

  const userState = useSelector(state => state.loginUserReducer);
  const {currentUser} = userState
  const orderstate = useSelector(state=>state.placeOrderReducer)
  const {loading,error,success} = orderstate
  const [destinations,setdestinations] = useState("");
  const [NewSms, setsms] = useState('')

  const dispatch = useDispatch()

  function tokenHandler(token) {
    console.log(token)
    dispatch(placeOrder(token,subtotal))
  }

  function sendSms (e){

    e.preventDefault()

    const destinations=["+94"+currentUser.phone];
    const NewSms ={

      destinations:destinations

    };
    console.log(NewSms);
    axios.post(
      // "http://localhost:5004/api/sms/send",
      "http://localhost:8290/sms/sendsms",
    )
  }
}
```

```

        NewSms).then(()=>{
            alert("success");
        }).catch((err)=>{
            alert(err);
        })

    }

    function emailsend (e){

        e.preventDefault();

        emailjs.sendForm('service_sy66xoc', 'template_fdhg1ym', e.target,
'user_iKzC1886DPbUEbUDz01bY')
            .then((result) => {

                console.log(result.text);
            }, (error) => {
                console.log(error.text);
            });
        e.target.reset();

    }

    if(success==true){

        document.getElementById('btn1').click();
        document.getElementById('btn2').click();

    }

    return (
        <div >
            {loading && (<Loading/>)}
        </div>
    )

```

```

{success && (<div class="alert alert-success alert-dismissible fade show"
role="alert">
<strong>Hotel Reservation Successfull</strong>

</div>)}

{error && (<div class="alert alert-danger alert-dismissible fade show"
role="alert">
<strong>Please Login To Place Orders</strong>

</div>)}}

    <StripeCheckout

        amount={subtotal * 100}
        shippingAddress
            token={tokenHandler}
            stripeKey='pk_test_51Kry14IppFV46PpQTbz9VzLa6keUgG1GtRyUYcQNrXjBBpzN33DaO
8mgGbJFXVmL3EmYb4chPIrcLFRFfNbvrNNO00PTpkrLAU'
            currency="LKR"
        >

            <button type='submit' className="btn btn-danger">Pay Now</button>

        </StripeCheckout>

        <div>
            <form onSubmit={emailsend}>
                <input name="email1" value={currentUser.email}
style={{visibility:"hidden"}}></input>
                <button id='btn1' type='submit' className="btn btn-danger"
style={{visibility:"hidden"}}>Email</button>
            </form>

            <form onSubmit={sendSms}>

                <button style={{visibility:"hidden"}} id='btn2' type='submit'
className="btn btn-danger">Sms</button>
            </form>
        </div>
    </div>

```

```
)  
}
```

FilterPkg.js

```
import React, { useState, useEffect } from "react";  
import { useDispatch, useSelector } from "react-redux";  
import { filterPackages } from '../actions/packageActions'  
  
export default function FilterPkg() {  
  const dispatch = useDispatch();  
  const [searchkey, setsearchkey] = useState("");  
  const [category, setcategory] = useState('all');  
  
  return (  
    <div className="sidebar mt-5 " >  
      <div className="container mt-5" >  
        <div className="row justify-content-center p-3 mb-5"  
" style={{backgroundColor: "#212121",  
color: "white" , borderRadius: "0px 0px 30px 30px", opacity: '.98' }} >  
  
          <div className="col-md-3" >  
            <input type="text" className="form-control" value={searchkey}  
onChange={(e) => {setsearchkey(e.target.value)}}  
placeholder="Search Packages" style={{ backgroundColor:  
"#212121", color: "white", borderRadius: "10px", border: "black solid 1px"}} />  
  
          </div>  
  
          <div className="col-md-3 mt-2">  
            <select className="form-control" value={category} onChange={(e)  
=> {setcategory(e.target.value)}}  
style={{ backgroundColor: "#212121", color: "white",  
borderRadius: "10px", border: "black solid 1px"}}>  
              <option value='all'>All Packages</option>  
              <option value='luxury'>Luxury Packages</option>  
              <option value='normal'>Normal Packages</option>  
            </select>  
  
          </div>  
  
          <div className="col-md-1">  
            <button
```

```

onClick={()=>dispatch(filterPackages(searchkey,category))} type="button"
class="btn btn-secondary btn-rounded">Filter</button>

    </div>

</div>

</div>
</div>
)
}

```

Footer.js

```

import React from "react";
import "../HeaderFooter.css";

function Footer() {
  return (
    <div className="mainfooter">
      <div className="ftr">
        <div className="row">
          <div className="col">
            <h4>Visit Us</h4>
            <iframe
src="https://www.google.com/maps/embed?pb=!1m14!1m8!1m3!1d1652.6692574132821!2d79.8419227322014!3d6.932750605071532!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x3ae259259a6157fb%3A0x8c0cebb288af4419!2sThe%20Kingsbury%20Colombo!5e0!3m2!1sen!2s!4v1652944507499!5m2!1sen!2s!k"></iframe>

            </div>
            { /* Column1 */}
            <div className="col">
              <h4>Address</h4>
              <h6 className="list-unstyled">
                <li>26/D</li>
                <li>Malabe</li>
                <li>Kaduwela</li>
                <li>Colombo</li>
              </h6>
            </div>
            { /* Column2 */}
            <div className="col">
              <h4>Terms</h4>

```



```

<div class="modal custom-fade" id="staticBackdrop" data-backdrop="static" data-
keyboard="false" tabindex="-1" aria-labelledby="staticBackdropLabel" aria-
hidden="true">
    <div class="modal-dialog modal-lg">
        <div class="modal-content">
            <div class="modal-body ftmodal-body ">
                <div class="text-right"> <i class="fa fa-close close ftclose"
data-dismiss="modal"></i> </div>
                <div class="row">
                    <div class="col-md-5">
                        <br/>
                        <div class="text-center mt-5">  </div>
                    </div>
                    <div class="col-md-6">
                        <div class="text-white mt-4"><span class="intro-1
ftintro-1">Only For Online Payment</span>
                            <div class="mt-2"> <span class="intro-2">An e-
commerce payment system facilitates the acceptance of electronic payment for
online transactions. Also known as a subcomponent of electronic data interchange,
e-commerce payment systems have become increasingly popular due to the widespread
use of the internet-based shopping and banking.</span> </div>
                            <div class="mt-4 mb-5"> <button class="btn ftbtn-
primary btn-primary" data-dismiss="modal"><i class="fa fa-close"></i>
Close</button> </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>

<div class="modal custom-fade" id="staticBackdrop2" data-backdrop="static" data-
keyboard="false" tabindex="-1" aria-labelledby="staticBackdropLabel" aria-
hidden="true">
    <div class="modal-dialog modal-lg">
        <div class="modal-content">
            <div class="modal-body ftmodal-body ">
                <div class="text-right"> <i class="fa fa-close close ftclose"
data-dismiss="modal"></i> </div>
            </div>
        </div>
    </div>

```

```

        <div class="row">
            <div class="col-md-5">
                <br/>
                <div class="text-center mt-5">  </div>
            </div>
            <div class="col-md-6">
                <div class="text-white mt-4"> <span class="intro-1
ftintro-1">Do not Copy Right</span>
                    <div class="mt-2"> <span class="intro-2">Our Dream
Travel Company website do not copy. Website copywriting is the process of writing
digital content for landing pages, product pages, blog posts, and everything in
between. Compelling copy can keep your website visitors engaged and lead them to
take actions that are both important to you and meaningful to them.</span> </div>
                    <div class="mt-4 mb-5"> <button class="btn ftbtn-
primary btn-primary" data-dismiss="modal"><i class="fa fa-close"></i>
Close</button> </div>
                </div>
            </div>
        </div>
    </div>
</div>

<div class="modal custom-fade" id="staticBackdrop3" data-backdrop="static" data-
keyboard="false" tabindex="-1" aria-labelledby="staticBackdropLabel" aria-
hidden="true">
    <div class="modal-dialog modal-lg">
        <div class="modal-content">
            <div class="modal-body ftmodal-body ">
                <div class="text-right"> <i class="fa fa-close close ftclose"
data-dismiss="modal"></i> </div>
                <div class="row">
                    <div class="col-md-5">
                        <br/>
                        <div class="text-center mt-4">  </div>
                    </div>
                    <div class="col-md-6">
                        <div class="text-white mt-4"> <span class="intro-1
ftintro-1">You have to Agree with Condition</span>

```

```

        <div class="mt-2"> <span class="intro-2">In this
case, agreed on its own means "It is accepted". agree is the verb form. If you
use a verb on its own, it is an imperative: telling somebody to do something. So,
if you simply say agree, you are telling the other person to agree with
you.</span> </div>

        <div class="mt-4 mb-5"> <button class="btn ftbtn-
primary btn-primary" data-dismiss="modal"><i class="fa fa-close"></i>
Close</button> </div>

    </div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>

<div class="modal custom-fade" id="staticBackdrop4" data-backdrop="static" data-
keyboard="false" tabindex="-1" aria-labelledby="staticBackdropLabel" aria-
hidden="true">
    <div class="modal-dialog modal-lg">
        <div class="modal-content">
            <div class="modal-body ftmodal-body ">
                <div class="text-right"> <i class="fa fa-close close ftclose"
data-dismiss="modal"></i> </div>
                <div class="row">
                    <div class="col-md-5">
                        <br/>
                        <div class="text-center mt-5">  </div>
                    </div>
                    <div class="col-md-6">
                        <div class="text-white mt-4"> <span class="intro-1
ftintro-1">Must Read Each and Every Fields</span>
                            <div class="mt-2"> <span class="intro-2">Meaning of
must-read in English. something that many people want to read or that a
particular group of people should read: His quick wit and dazzling writing style
have made his column a must-read. The book is a must-read for anyone interested
in Cold War diplomacy.</span> </div>
                            <div class="mt-4 mb-5"> <button class="btn ftbtn-
primary btn-primary" data-dismiss="modal"><i class="fa fa-close"></i>
Close</button> </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>

```

```

        </div>
      </div>
    </div>
  </div>
</div>

    </div>
  </div>
</div>
);
}

export default Footer;

```

Loading.js

```

import React from "react";

export default function Loading() {
  return (
    <div>
      <div classname="spinner-grow" role="status" style={{height:"80px",
width:"80px"}}>

        </div>
        <div class="spinner-border" role="status">

</div>
    </div>
  );
}

```

Navibar.js

```
import React from "react";

import {Navbar, Nav, NavDropdown, Container} from 'react-bootstrap';
import {useSelector,useDispatch} from 'react-redux';
import {logoutUser} from '../actions/userActions'
import Checkout from './Checkout'

export default function Navibar() {

  const cartstate = useSelector(state => state.cartReducer);
  const userState = useSelector(state => state.loginUserReducer);
  const {currentUser} = userState
  const dispatch = useDispatch();

  return (
    <div>
      <Navbar bg="dark" variant="dark" expand="lg" fixed={'top'}>
        <Container>

          <Navbar.Brand href="/homepkg" classname="shadow p-3 mb-5 bg-white rounded">Kingsbury Hotel Reservations</Navbar.Brand>

          <Navbar.Toggle aria-controls="basic-navbar-nav" />
          <Navbar.Collapse className="justify-content-end">
            <Nav className="justify-content-end" >

              {currentUser ? ( <NavDropdown title={currentUser.name} id="collasible-nav-dropdown">
                <NavDropdown.Item href="/ord">Reservations</NavDropdown.Item>
                <NavDropdown.Item href="/taxiord">Taxi Bookings</NavDropdown.Item>
                <NavDropdown.Item href="" onClick={() =>{dispatch(logoutUser())}}>Log Out</NavDropdown.Item>
                {currentUser.isAdmin ? ( <NavDropdown.Divider />):""}
                {currentUser.isAdmin ? ( <NavDropdown.Item href="/admin/orderslist">Admin Dashboard</NavDropdown.Item>):""}
                {currentUser.isHotelAdmin ? ( <NavDropdown.Divider />):""}
              ) : null}

            </Nav>
          </Navbar.Collapse>
        </Navbar>
      </div>
    )
  }
```

```

        {currentUser.isHotelAdmin ? ( <NavDropdown.Item
href="/hoteladmin/orderslist"> Hotel Admin Dashboard</NavDropdown.Item>):""}
        </NavDropdown>

) :
( <Nav.Link href="/login">Login</Nav.Link>)}

        {cartstate.cartItems.length === 0 ? <Nav.Link>Add to Cart</Nav.Link> :
(<Nav.Link href="/cart">Cart &nbsp;   {cartstate.cartItems.length}</Nav.Link>)}

        </Nav>
        </Navbar.Collapse>
        </Container>

</Navbar>

        </div>
    );
}

```

Packages.js

```

import React, { useState } from "react";
import { Modal } from "react-bootstrap";
import { useDispatch, useSelector } from "react-redux";
import { addToCart } from "../actions/cartActions";

export default function Packages({ pack }) {
    const [quantity, setquantity] = useState(1);
    const [varient, setvarient] = useState("TwoPerson");
    const [date, setdate] = useState("Date Haven't Selected");

    const [show, setShow] = useState(false);

    const handleClose = () => setShow(false);
    const handleShow = () => setShow(true);

```

```

const dispatch = useDispatch();
function addtocart() {
  {
    dispatch(addToCart(pack, quantity, varient, date));
  }
}

const disablePastDate = () => {
  const today = new Date();
  const dd = String(today.getDate()).padStart(2, "0");
  const mm = String(today.getMonth() + 1).padStart(2, "0");
  const yyyy = today.getFullYear();

  return yyyy + "-" + mm + "-" + dd;
};

return (
  <div className="sss">

    <div
      class="shadow mb-3"
      style={{backgroundColor:"#212121",
        color:"white" , borderRadius:"30px",opacity: '.98' }}>

      <div class="row " >
        <div class="col-sm-3 text-left">
          <div onClick={handleShow}>
            <img
              src={pack.image}
              classname="img-fluid"
              alt={pack.name}
              style={{ height: "300px", width: "225px", borderRadius: "20px 50px 0px 20px", objectFit: "cover", objectPosition: "50%"}}
            />

          </div>
        </div>
      </div>
    </div>
  </div>

```



```

<div class="col-sm-9">

  <div className="flex-container ">

    <div className="w-100 m-1 ml-2 mt-2 text-left ">
      <h9 style={{color: '#0D6EF7' , fontSize:"40px"
}}>{pack.name.toUpperCase()}</h9>
      <h1 style={{marginBottom: '7px'}}>Package</h1>
      <select
        className="form-control"
        value={variant}
        onChange={(e) => {
          setvariant(e.target.value);
        }}
        style={{ backgroundColor: "#212121", color: "white", borderRadius:
"10px" , border: "white solid 1px !important"}}
      >
        {pack.varients.map((variant) => {
          return <option value={variant}> {variant}</option>;
        })}
      </select>
    </div>

    <div className="w-100 m-1 pr-2 mt-2 text-left"> <h9 style={{visibility:
'hidden' , fontSize:"40px"}} >{pack.name}</h9>
    <h1 style={{marginBottom: '7px'}}>Days</h1>
    <select
      className="form-control"
      value={quantity}
      onChange={(e) => {
        setquantity(e.target.value);
      }}
      style={{ backgroundColor: "#212121", color: "white", borderRadius:
"10px", border: "black solid 1px"}}
    >
      {[...Array(10).keys()].map((x, i) => {
        return <option value={i + 1}> {i + 1}</option>;
      })}
    </select>
  </div>
</div>

```

```

<div className="w-2 m-1 ml-2 pr-2 text-left">
  <h1> Booking Date</h1>
  <input
    type="date"
    required
    className="form-control"
    name="departureDate"
    placeholder="YY/MM/DD"
    min={disablePastDate()}
    minDate={new Date()}
    value={date}
    onChange={(e) => {
      setdate(e.target.value);
    }}
    style={{ backgroundColor: "#212121", color: "white", borderRadius:
"20px", border: "black solid 1px"}}
  />
</div>

<div className="flex-container text-left mt-3">
  <div className="m-1 w-100 col-12 col-sm-6 col-md-7 text-left">
    <h2 className="mt-1" style={{color:"orange"}}>
      Price : Rs {pack.prices[0][varient] * quantity}/=
    </h2>
  </div>

  <div className=" w-100 col-6 col-md-5">
    <button className="btn" onClick={addtocart}
      style={{backgroundColor: "orange", height: "50px" ,width: "200px"}}>

      Add to cart
    </button>
  </div>

</div>

```

```

</div>

</div>
<Modal show={show} onHide={handleClose} >
  <Modal.Header style={{backgroundColor:"#212121",
    color:"white" , opacity: '.98' }} closeButton>
    <Modal.Title style={{color: '#0D6EF7' , fontSize:"40px"
}}>{pack.name.toUpperCase()}</Modal.Title>
  </Modal.Header>

  <Modal.Body style={{backgroundColor:"#212121",
    color:"white" , opacity: '.98' }}>
    <div className="">
      <img
        src={pack.image}
        className="img-fluid"
        alt={pack.name}
        style={{ height: "380px", width: "450px" , borderRadius: "20px 50px
0px 20px", objectFit: "cover" }}
      />

    </div>
    <p style={{fontSize:"20px" }}>{pack.description}</p>
  </Modal.Body>

  <Modal.Footer style={{backgroundColor:"#212121",
    color:"white" , opacity: '.98' }}>

    <button className="btn btn-danger" onClick={handleClose}>
      Close
    </button>
  </Modal.Footer>
</Modal>
</div>
</div>
);
}

```

cartReducer.js

```
export const cartReducer = (state={cartItems:[]}, action) => {

  switch(action.type){
    case 'ADD_TO_CART':
      const alreadyExists = state.cartItems.find(item => item._id ===
action.payload._id)
      if(alreadyExists){
        return{
          ...state,
          cartItems: state.cartItems.map(item=> item._id ===
action.payload._id ? action.payload : item)
        }
      }
      else{

        return {
          ...state,
          cartItems: [...state.cartItems, action.payload]
        }
      }
    case 'DELETE_FROM_CART': return {
      ...state,
      cartItems: state.cartItems.filter(item => item._id !== action.payload._id)
    }
    default: return state
  }
}
```

OrderReducer.js

```
export const placeOrderReducer = (state={}, action) => {

  switch(action.type){
    case 'PLACE_ORDER_REQUEST': return {
      loading: true
    }
  }
}
```

```

    }
    case 'PLACE_ORDER_SUCCESS': return {
      loading: false,
      success: true
    }
    case 'PLACE_ORDER_FAILED': return {
      loading: false,
      error: action.payload
    }
    default: return state
  }
}

export const getUserOrdersReducer = (state = {orders : []}, action) => {

  switch (action.type) {
    case 'GET_USER_ORDERS_REQUEST': return {
      loadings: true,
      ...state
    }
    case 'GET_USER_ORDERS_SUCCESS': return {
      loadings: false,
      orders: action.payload
    }
    case 'GET_USER_ORDERS_FAILED':return {
      error: action.payload,
      loadings: false
    }
    default: return state;
  }
}

export const getAllOrdersReducer = (state = {orders : []}, action) => {

  switch (action.type) {
    case 'GET_ALLORDERS_REQUEST': return {
      loadings: true,
      ...state
    }
    case 'GET_ALLORDERS_SUCCESS': return {
      loadings: false,

```

```

        orders: action.payload
      }
      case 'GET_ALLORDERS_FAILED':return {
        error: action.payload,
        loadings: false
      }
      default: return state;
    }
  }
}

export const getOrderByIdReducer = (state = {}, action) => {
  switch (action.type) {
    case 'GET_ORDERBYID_REQUEST': return {
      loadings: true,
      ...state
    }
    case 'GET_ORDERBYID_SUCCESS': return {
      loadings: false,
      order: action.payload
    }
    case 'GET_ORDERBYID_FAILED':return {
      error: action.payload,
      loadings: false
    }
    default: return state;
  }
}

```

packageReducer.js

```

export const getAllPackagesReducer = (state = {packages : []}, action) => {
  switch (action.type) {
    case 'GET_PACKAGES_REQUEST': return {
      loading: true,
      ...state
    }
    case 'GET_PACKAGES_SUCCESS': return {
      loading: false,
      packages: action.payload
    }
    case 'GET_PACKAGES_FAILED':return {

```

```

        error: action.payload,
        loadings: false
    }
    default: return state;
}
}

export const addPackageReducer = (state = {}, action) => {

    switch (action.type) {
        case 'ADD_PACKAGE_REQUEST': return {
            loadings: true,
            ...state
        }
        case 'ADD_PACKAGE_SUCCESS': return {
            loadings: false,
            success: true
        }
        case 'ADD_PACKAGE_FAILED':return {

            error: action.payload,
            loadings: false
        }
        default: return state;
    }
}

export const getPackageByIdReducer = (state = {}, action) => {

    switch (action.type) {
        case 'GET_PACKAGEBYID_REQUEST': return {
            loadings: true,
            ...state
        }
        case 'GET_PACKAGEBYID_SUCCESS': return {
            loadings: false,
            packag: action.payload
        }
        case 'GET_PACKAGEBYID_FAILED':return {

            error: action.payload,
            loadings: false
        }
    }
}

```

```

    }
    default: return state;
  }
}

export const editPackageReducer = (state = {}, action) => {

  switch (action.type) {
    case 'EDIT_PACKAGE_REQUEST': return {
      editloadings: true,
      ...state
    }
    case 'EDIT_PACKAGE_SUCCESS': return {
      editloadings: false,
      editsuccess: true
    }
    case 'EDIT_PACKAGE_FAILED':return {

      editerror: action.payload,
      editloadings: false
    }
    default: return state;
  }
}

```

taxiReducer.js

```

export const bookTaxiReducer = (state = {}, action) => {

  switch (action.type) {
    case 'BOOK_TAXI_REQUEST': return {
      loadings: true,
      ...state
    }
    case 'BOOK_TAXI_SUCCESS': return {
      loadings: false,
      success: true
    }
    case 'BOOK_TAXI_FAILED':return {

      error: action.payload,
      loadings: false
    }
  }
}

```



```

        default: return state;
    }
}

export const getOrderedTaxireducer = (state = {taxis : []}, action) => {

    switch (action.type) {
        case 'GET_TAXI_ORDERS_REQUEST': return {
            loadings: true,
            ...state
        }
        case 'GET_TAXI_ORDERS_SUCCESS': return {
            loadings: false,
            taxis: action.payload
        }
        case 'GET_TAXI_ORDERS_FAILED':return {

            error: action.payload,
            loadings: false
        }
        default: return state;
    }
}

export const getAllTaxiReducer = (state = {taxis : []}, action) => {

    switch (action.type) {
        case 'GET_ALLTAXI_REQUEST': return {
            loading: true,
            ...state
        }
        case 'GET_ALLTAXI_SUCCESS': return {
            loadings: false,
            taxis: action.payload
        }
        case 'GET_ALLTAXI_FAILED':return {

            error: action.payload,
            loading: false
        }
        default: return state;
    }
}

```

userReducer.js

```
export const registerUserReducer = (state={}, action) => {

  switch(action.type){
    case 'USER_REGISTER_REQUEST': return {
      loading: true
    }
    case 'USER_REGISTER_SUCCESS': return {
      loading: false,
      success: true
    }
    case 'USER_REGISTER_FAILED': return {
      loading: false,
      error: action.payload
    }
    default: return state
  }
}

export const loginUserReducer = (state={}, action) => {

  switch(action.type)
  {

    case 'USER_LOGIN_REQUEST': return {
      loading: true
    }
    case 'USER_LOGIN_SUCCESS': return {
      loading: false,
      success: true,
      currentUser: action.payload
    }
    case 'USER_LOGIN_FAILED': return {
      loading: false,
      error: action.payload
    }

    default: return state
  }
}

export const getAllUsersReducer = (state = {users : []}, action) => {
```

```

    switch (action.type) {
      case 'GET_USERS_REQUEST': return {
        loading: true,
        ...state
      }
      case 'GET_USERS_SUCCESS': return {
        loading: false,
        users: action.payload
      }
      case 'GET_USERS_FAILED': return {
        error: action.payload,
        loading: false
      }
      default: return state;
    }
  }
}

```

AddPackage.js

```

import React, {useState, useEffect} from 'react'
import {useDispatch, useSelector} from 'react-redux'
import {addPackage} from '../actions/packageActions'
import Loading from '../components/Loading'
import Error from '../components/Error'
import Success from '../components/Success'
import Swal from 'sweetalert2'

export default function Addpackage() {
  const [name, setname] = useState('')
  const [smallprice, setsmallprice] = useState('')
  const [mediumprice, setmediumprice] = useState('')
  const [largeprice, setlargeprice] = useState('')
  const [image, setimage] = useState('')
  const [description, setdescription] = useState('')
  const [category, setcategory] = useState('')

  const dispatch = useDispatch()
  const addpackagestate = useSelector(state => state.addPackageReducer)
  const {success, error, loading} = addpackagestate

  function formHandler(e){

```

```

e.preventDefault()

const pack = {
  name,
  image,
  description,
  category,
  prices: {
    TwoPerson: smallprice,
    FourPerson: mediumprice,
    Family: largeprice
  }
}
console.log(pack)

dispatch(addPackage(pack));
}

return (
  <div >

    <div className='justify-content-center' style={{textAlign: 'center',
backgroundColor: '#FFF'}}>
      <div >
        <h2>Add Package</h2>
        {loading && <Loading/>}

{success && (<div class="alert alert-success alert-dismissible fade show"
role="alert">
<strong>Hotel Package Added Success</strong>

</div>))}

      <div >
        <form onSubmit={formHandler}
          className='col-md-7 p-3 m-3 text-left' style={{display: 'inline-
block'}}>
          <input id="form12" className="form-control" type="text" value={name}
placeholder="Name"
            onChange={(e) =>{setname(e.target.value)}} />

          <input type="text" className="form-control mt-2 " value={smallprice}
placeholder="Two Person Price"

```

```

        onChange={(e) =>{setsmallprice(e.target.value)} }/>

<input type="text"  className="form-control mt-2" value={mediumprice}
placeholder="Four Person Price"
        onChange={(e) =>{setmediumprice(e.target.value)} }/>

<input type="text"  className="form-control mt-2" value={largeprice}
placeholder="Family Price"
        onChange={(e) =>{setlargeprice(e.target.value)} }/>

<input type="text"  className="form-control mt-2" value={image}
placeholder="Image"
        onChange={(e) =>{setimage(e.target.value)} }/>

<input type="text"  className="form-control mt-2" value={description}
placeholder="Description"
        onChange={(e) =>{setdescription(e.target.value)} }/>

<input type="text"  className="form-control mt-2" value={category}
placeholder="Category"
        onChange={(e) =>{setcategory(e.target.value)} }/>

    <br/>

<button type="submit" className="btn btn-success" >
    Add package
</button>

</form>

</div>
</div>
</div>

</div>
)
}

```

Adminscreen.js

```
import React, { useEffect } from 'react'
import {useSelector,useDispatch} from 'react-redux';
import { Link, Route , Switch } from 'react-router-dom';

import Orderslist from './Orderslist';
import Userslist from './Userslist';

export default function Adminscreen() {

  const userState = useSelector(state => state.loginUserReducer);
  const {currentUser} = userState
  const dispatch = useDispatch();

  useEffect(() => {
    if(!currentUser.isAdmin){
      window.location.href = '/'
    }
  })

  return (
    <div className="mt-5">

      <nav className="navbar navbar-expand-lg navbar navbar-dark bg-success "
style={{backgroundColor: "black"}}>
        <a className="navbar-brand " href="/admin/userslist">Admin Panel</a>

        <button className="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-
Label="Toggle navigation">
          <span className="navbar-toggler-icon"></span>
        </button>
        <div className="collapse navbar-collapse justify-content-center text-left"
id="navbarNav">
          <ul className="navbar-nav">
            <li className="nav-item mr-2">
              <Link className="nav-link" to="/admin/userslist">Userslist </Link>
              { /* <a className="nav-link" href="/admin/userslist">Users List <span
className="sr-only">(current)</span></a> */}
            </li>
            <li className="nav-item mr-2 ">
              <Link className="nav-link " to="/admin/packageslist">Packages List</Link>
```

```

    </li>
    <li className="nav-item mr-2">
      <Link className="nav-link" to="/admin/addpackage">Add New Package</Link>
    </li>
    <li className="nav-item mr-2">
      <Link className="nav-link " to="/admin/orderslist">Reservations
List</Link>
    </li>

    <li className="nav-item mr-2">
      <Link className="nav-link " to="/admin/taxilist">Taxi List</Link>
    </li>
  </ul>

</div>
</nav>

{ /* <Switch>
  { /* <Route path="/admin" component={Pizzaslist} />

    <Route path="/admin/userslist" component={Userslist} />
    <Route path="/admin/pizzaslist" component={Pizzaslist} />
    <Route path="/admin/addpizza" component={Addpizza} />
    <Route path="/admin/orderslist" component={Orderslist} />
    <Route path="/admin/editpizza/:pizzaid" component={Editpizza} />

    <Route path="/admin/edpizza/:pizzaid" component={Edpizza} />

    <Route path="/admin/edit/:pizzaid" component={Edit} />
  </Switch> */}
</div>
)
}

```

Cartscreen.js

```
import React from 'react'
import {useSelector,useDispatch} from 'react-redux'
import { addToCart } from '../actions/cartActions';
import {deleteFromCart} from '../actions/cartActions';
import Checkout from '../components/Checkout'

export default function Cartscreen() {

  const cartstate = useSelector(state => state.cartReducer);
  const cartItems = cartstate.cartItems;

  var subtotal = cartItems.reduce((total,item)=>total+item.price,0);
  const dispatch = useDispatch();

  return (
    <div className="row justify-content-center" style={{marginTop:"70px"}}>
      <div className="col-md-7">
        <h2 style={{fontSize:'40px'}}>My Cart</h2>
        {cartItems.map(item => {

          return <div className="flex-container mb-2
" style={{backgroundColor:"#212121",
          color:"white" , borderRadius:"30px",opacity: '.98' , padding:
'20px'}}>

            <div className="text-left m-1 w-100">
              <h2 style={{color: '#0D6EF7' }}>{item.name}
[ {item.varient} ]</h2>
              <h1>Price: {item.quantity} X {item.prices[0][item.varient]}
</h1>

              <h1 style={{color:"#FFFF00"}}> Rs. {item.price} /=</h1>
              <h1 style={{display: 'inline'}}>Days: </h1>
              <button class="btn btn-secondary btn-sm"
onClick={()=>{dispatch(addToCart(item,item.quantity+1,item.va
rient,item.date))}}>
              <strong style={{fontSize: "16px"}}>+</strong>
              </button>
              &nbsp;
            </div>
          )}
```



```

        <b style={{fontSize: "16px" , backgroundColor:"#AB69CC"
,padding: "9px",borderRadius:"5px"}}>{item.quantity}</b>
        &nbsp;

        <button class="btn btn-secondary btn-sm"
        onClick={()=>{dispatch(addToCart(item,item.quantity-
1,item.varient,item.date))}}>
        <strong style={{fontSize: "16px"}}>-</strong>
        </button>
        <br/>
        <b>{item.date}</b>

        /* <i className="fa fa-minus" aria-hidden="true"
onClick={()=>{dispatch(addToCart(item,item.quantity-1,item.varient))}}></i> */

    </div>

    <div className="m-1 w-100">

<img src={item.image} className="" style={{height:"200px", width:"200px"
,borderRadius:"15px 50px"}}/>

    </div>

    <div className="m-1 w-100">
        <button type="button" class="btn btn-danger btn-rounded"

        onClick={()=>{dispatch(deleteFromCart(item))}}> Remove
        </button>

    </div>
</div>
    ))}

</div>

<div className="col-md-3 mt-5" >

    <div className="mt-2" style={{backgroundColor:"#212121",

```

```

        color:"white" , borderRadius:"30px",opacity: '.98' , padding:
'20px'}}}>
        <h2 style={{fontSize:'45px'}}>Sub Total</h2>
        <h2 style={{color:"#FFFF00"}}> Rs {subtotal} /=</h2>
        <Checkout subtotal={subtotal}/>
      </div>
    </div>
  </div>
)
}

```

EditPackage.js

```

import React, { useEffect, useState } from "react";
import { useDispatch, useSelector } from "react-redux";
import { getPackageById , editPackage} from "../actions/packageActions";

import Loading from "../components/Loading";
import Error from "../components/Error";
import Success from "../components/Success";

export default function Editpizza({ match }) {
  const [name, setname] = useState("");
  const [smallprice, setsmallprice] = useState("");
  const [mediumprice, setmediumprice] = useState("");
  const [largeprice, setlargeprice] = useState("");
  const [image, setimage] = useState("");
  const [description, setdescription] = useState("");
  const [category, setcategory] = useState("");

  const getpackagebyidstate = useSelector((state) =>
state.getPackageByIdReducer);
  const { packag, error, loading } = getpackagebyidstate;

  const editpackagestate = useSelector((state) => state.editPackageReducer);
  const { editsuccess, editerror, editloading } = editpackagestate;

  const dispatch = useDispatch();

  useEffect(() => {

```

```

    if(packag){

        if(packag._id==match.params.packageid){

            setname(packag.name);
            setsmallprice(packag.prices[0]["TwoPerson"])
            setmediumprice(packag.prices[0]["FourPerson"])
            setlargeprice(packag.prices[0]["Family"])
            setimage(packag.image);
            setdescription(packag.description);
            setcategory(packag.category);
        }
        else{
            dispatch(getPackageById(match.params.packageid));
        }
    }
    else{
        dispatch(getPackageById(match.params.packageid));
    }

}, [packag , dispatch]);

function formHandler(e) {
    e.preventDefault();

    const editedpackage = {
        _id: match.params.packageid,
        name,
        image,
        description,
        category,
        prices: {
            TwoPerson: smallprice,
            FourPerson: mediumprice,
            Family: largeprice,
        },
    };

    dispatch(editPackage(editedpackage))
}
return (
    <div>

```

```

    <div className="justify-content-center" style={{ textAlign: "center" }}>
      <div>
        <hr/>
        <h2>Edit Packages</h2>
        {editsuccess && (<div class="alert alert-success alert-dismissible fade
show" role="alert">
<strong>Hotel Package Updated Success</strong>

</div>))}

      <div>
        <form
          onSubmit={formHandler}
          className='col-md-7 p-3 m-3 text-left'
          style={{ display: "inline-block" }}
        >

          <input
            className="form-control"
            type="text"
            value={name}
            placeholder="Name"
            onChange={(e) => {
              setname(e.target.value);
            }}
          />

          <input
            type="text"
            className="form-control"
            value={smallprice}
            placeholder="Small Varient Price"
            onChange={(e) => {
              setsmallprice(e.target.value);
            }}
          />

          <input
            type="text"
            className="form-control"
            value={mediumprice}
            placeholder="Medium Varient Price"
            onChange={(e) => {
              setmediumprice(e.target.value);
            }}
          />

```

```

/>

<input
  type="text"
  className="form-control"
  value={largeprice}
  placeholder="Large Varient Price"
  onChange={(e) => {
    setlargeprice(e.target.value);
  }}
/>

<input
  type="text"
  className="form-control"
  value={image}
  placeholder="Image"
  onChange={(e) => {
    setimage(e.target.value);
  }}
/>

<input
  type="text"
  className="form-control"
  value={description}
  placeholder="Description"
  onChange={(e) => {
    setdescription(e.target.value);
  }}
/>

<input
  type="text"
  className="form-control"
  value={category}
  placeholder="Category"
  onChange={(e) => {
    setcategory(e.target.value);
  }}
/>

<br />

<button type="submit" className="btn btn-primary">

```

```

        Edit Package
      </button>
    </form>
  </div>
</div>
</div>
</div>
</div>
);
}

```

HomeScreenPkg.js

```

import React, { useEffect, useState, useRef } from 'react'
import { useDispatch, useSelector } from 'react-redux'

import Packages from '../components/Packages'
import { getAllPackages } from '../actions/packageActions'
import Loading from './Loading'
import Error from '../components/Error'
import FilterPkg from '../components/FilterPkg'

function HomeScreenPkg() {

  const dispatch = useDispatch()

  const packagestate = useSelector(state => state.getAllPackagesReducer)
  const { packages, error, loading } = packagestate

  useEffect(() => {
    dispatch(getAllPackages())
  }, [])

  const divRef = useRef();

  return (
    <div
      style={{

```

```

        backgroundImage: "url(" +
"https://res.cloudinary.com/vihanga/image/upload/v1651916442/ds/markus-spiske-
g5ZIXjzRGds-unsplash_shvxqj.jpg" + ")",
        backgroundColor: 'center',
        backgroundSize: 'cover',
        backgroundRepeat: 'no-repeat',
        backgroundAttachment: 'fixed',
    }}
    >

    <FilterPkg/>
    <button style={{marginLeft:"90%"}} className="btn btn-primary"
        onClick={() => {
            divRef.current.scrollToView({ behavior: "smooth" });
        }}
    > Locate Us

    </button>
    <div className="wrapper" >

        {loading && (<h1>Loading...</h1>)}

        <div>
        <div className="row justify-content-center">
            {loading ? (<Loading/>) : error ? (<Error error="something went wrong"/>) :
(
                packages.map(pack => {

                    return <div className="col-md-8 p-1"key={pack._id} >
                        <div className="m-3" ref={divRef}>
                            <Packages pack={pack}/>
                        </div>
                    </div >

                })
            )
        }
    </div>

```

```

    })

</div >

    </div>
    </div >

    </div >
  )
}

export default HomeScreenPkg

```

HotelAdminScreen.js

```

import React, { useEffect } from 'react'
import {useSelector,useDispatch} from 'react-redux';
import { Link, Route , Switch } from 'react-router-dom';

import Orderslist from './Orderslist';
import Userslist from './Userslist';

import Packageslist from './Packageslist';
import Addpackage from './Addpackage';
import Editpackage from './Editpackage';

export default function HotelAdminscreen() {

  const userState = useSelector(state => state.loginUserReducer);
  const {currentUser} = userState
  const dispatch = useDispatch();

  useEffect(() => {
    if(!currentUser.isHotelAdmin){
      window.location.href = '/'
    }
  })

```



```

    })

    return (
      <div className="mt-5">

        <nav className="navbar navbar-expand-lg navbar navbar-dark bg-success "
style={{backgroundColor: "black"}}>
          <a className="navbar-brand " href="/orderslist">Admin Panel</a>

          <button className="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-
Label="Toggle navigation">
            <span className="navbar-toggler-icon"></span>
          </button>
          <div className="collapse navbar-collapse justify-content-center text-left"
id="navbarNav">
            <ul className="navbar-nav">

              <li className="nav-item mr-2 ">
                <Link className="nav-link " to="/hoteladmin/packageslist">Packages
List</Link>
              </li>
              <li className="nav-item mr-2">
                <Link className="nav-link" to="/hoteladmin/addpackage">Add New
Package</Link>
              </li>
              <li className="nav-item mr-2">
                <Link className="nav-link " to="/hoteladmin/orderslist">Reservations
List</Link>
              </li>
            </ul>

          </div>
        </nav>

        <Switch>
          { /* <Route path="/admin" component={Pizzaslist} /> */ }
          <Route path="/hoteladmin/userslist" component={Userslist} />

          <Route path="/hoteladmin/orderslist" component={Orderslist} />

          <Route path="/hoteladmin/addpackage" component={Addpackage} />
          <Route path="/hoteladmin/packageslist" component={Packageslist} />
        </Switch>
      </div>
    )
  )
}

```

```

        <Route path="/hoteladmin/editpackage/:packageid" component={Editpackage}
/>

    </Switch>
  </div>
)
}

```

Loginscreen.js

```

import React, { useState, useEffect } from "react";
import { useDispatch, useSelector } from "react-redux";
import { loginUser } from '../actions/userActions'
import Error from "../Error";
import Loading from "../Loading";
import TextField from "@material-ui/core/TextField";

export default function Loginscreen() {

  const [email, setemail] = useState("");
  const [password, setpassword] = useState("");
  const loginstate = useSelector(state=>state.loginUserReducer)
  const {error,loading} = loginstate

  const dispatch = useDispatch();

  useEffect(() => {

    if(localStorage.getItem("currentUser"))
    {
      window.location.href = "/homepkg"
    }

  })

  const validateEmail = (email) => {
    return email.match(
      /^((([^<>()[]\.\,;\s@\"']+)(\.[^<>()[]\.\,;\s@\"']+)*)|(\".+\"))@((\[[0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\])|((a-zA-Z-0-9)+\.)+[a-zA-Z]{2,}))$/
    );
  };

```

```

};

function login(){

  if(email==""){
    alert("Please enter Your Email")
  }

  else{
    if(!validateEmail(email)){
      alert("Please Enter a valid Email")
    }

    else
    if(password==""){
      alert("Please enter Your Password")
    }

    else{

      const user = {email,password }
      console.log(user)
      dispatch(loginUser(user))

    }
  }

}

return (
  <div className="p-2" style={{marginTop: '5%'}}>
    <div className="row justify-content-center">
      <div className="col-md-5 mt-5 text-left p-3 mb-5" style={{backgroundColor:"#212121",
        color:"white" , borderRadius:"30px",opacity: '.98' , padding:
'20px'}}>
        <h2 className="text-center m-2 " style={{ fontSize: "35px" }}>
          Login
        </h2>

        {error &&( <Error/>)}
      </div>
    </div>
  </div>
)

```

```

<div >

  <input type="text"
    className="form-control"
    placeholder="Email"
    required
    value={email}
    onChange={(e)=>{setemail(e.target.value)}}
    style={{ backgroundColor: "#212121", color: "white",
borderRadius: "10px" , border: "white solid 1px !important"}}
  />

  <input
    type="text"
    className="form-control"
    placeholder="Password"
    required
    value={password}
    onChange={(e)=>{setpassword(e.target.value)}}
    style={{ backgroundColor: "#212121", color: "white",
borderRadius: "10px" , border: "white solid 1px !important"}}
  />

  <button type="submit" className="btn btn-danger mt-2 mb-2 "
    onClick={login} style={{marginLeft:"-1px"}}>
    Login
  </button>
  <br/>
  <a href="/register"
    style={{color:"white", textDecoration: "none"}} className="mt-2">
    Klik here to Register</a>
  </div>
</div>
</div>
</div>
)
}

```

OrderedTaxi.js

```
import React, { useEffect,useState } from 'react'
import { useDispatch, useSelector } from 'react-redux'

import {getOrderedTaxi, deleteTaxiOrder} from '../actions/taxiActions'
import Loading from '../components/Loading'
import Error from '../components/Error'

export default function OrderedTaxi() {

    const dispatch = useDispatch()

    const orderedtaxistate = useSelector(state=>state.getOrderedTaxireducer)
    const {taxis,error,loading} = orderedtaxistate

    useEffect(() => {
        dispatch(getOrderedTaxi())
    },[])
    return (
        <div className="mt-5 container text-left">

            <div    className="text-left w-100 m-1 p-1 mt-5 ">

                <h2 className="mt-3" style={{fontSize: "25px" }}>Booked Taxi
Services</h2>

                <hr/>

            </div>

            {taxis && taxis.map((taxi,index)=>{
return <div className="col-md-8 m-2" style={{backgroundColor:"dimgray",
color:"white" , borderRadius:"15px"}}>
                <div className="flex-container">
```

```

        <div className="row text-left w-100 m-1 p-4 ">

        <div className="col-8">

        <h2 style={{fontSize: "25px" }}> {index+1}) Booked Taxi
Info</h2> <hr/>
        <p>Address: {taxi.address}</p>
        <p>Booking date : {taxi.date}</p>
        <p> Booking Created Date :
{taxi.createdAt.substring(0,10)}</p>
        </div>

        <div className="col">
        <button className="btn btn-danger"
onClick={()=>{dispatch(deleteTaxiOrder(taxi._id))}} >
        <i className="fa fa-trash" style={{ color: "white" }}></i>
        &nbsp; Cancel Taxi Reservation
        </button>

        </div>

        </div>

    </div>
</div>

    )
}

```

Orders.js

```

import React, { useEffect,useState } from 'react'
import { useDispatch, useSelector } from 'react-redux'

import {getUserOrders, deleteOrder} from '../actions/orderAction'
import Loading from '../components/Loading'
import Error from '../components/Error'

```

```
import { Link } from "react-router-dom";

export default function Orders() {

  const dispatch = useDispatch()

  const orderstate = useSelector(state=>state.getUserOrdersReducer)
  const {orders,error,loading} = orderstate

  useEffect(() => {
    dispatch(getUserOrders())

  },[])
  return (

    <div className="wrapper" >
      <div style={{marginTop:"70px"}}>
        <h2 style={{fontSize:"35px"}}>My Reservations</h2>
        <hr/>

<div className="row justify-content-center">
  {loading && (<Loading/>)}
  {error && (<Error error="something went wrong"/>)}
  {orders && orders.map(order=>{
    return <div className="col-md-8 m-2 rounded-lg"
style={{backgroundColor:"dimgray",
color:"white"}}>
      <div className="flex-container">
        <div className="text-left w-100 m-1 p-1" >
          <h2 style={{fontSize: "25px" }}>Hotel Packages</h2>
          <hr/>
          {order.orderItems.map(item=>{
            return <div >
<p>{item.name} [{item.varient}] * {item.quantity} = {item.price}</p>

            <p>Dates : {item.date}</p>
            </div>
          })}
        </div>

      </div>

    </div>
  )}
```

```

<div className="text-left w-100 m-1 p-1 ">

  <h2 style={{fontSize: "25px" }}>Address</h2>
  <hr/>
  <p>Street : {order.shippingAddress.street}</p>
  <p>City : {order.shippingAddress.city}</p>
  <p>Country : {order.shippingAddress.country}</p>
  <p>Zipcode : {order.shippingAddress.pincod}</p>
</div>

<div className="text-left w-100 m-1 p-1 ">

  <h2 style={{fontSize: "25px" }}>Reservation Info</h2> <hr/>
  <p>Reservation Amount : Rs. {order.orderAmount} /=</p>
  <p>Payment Date : {order.createdAt.substring(0,10)}</p>
  <p>Transaction ID : {order.transactionId}</p>
  <p>Order ID : {order._id}</p>

  <button className="btn btn-danger"
    onClick={()=>{dispatch(deleteOrder(order._id))}} >
    <i className="fa fa-trash" style={{ color: "white" }}></i>
    &nbsp; Cancel Reservation
  </button>

</div>

<div className="text-left w-100 m-1 p-1 ">

  <Link to={` /taxy/${order._id}`} className>
    <button className="btn btn-warning">
      Click here if you want Taxy Service
    </button>
  </Link>
</div>

</div>
</div>

  )})
</div>

```



```

    </div>
  </div>
)
}

```

Orderslist.js

```

import React, { useEffect, useState } from "react";
import { useDispatch, useSelector } from "react-redux";
import Loading from "../components/Loading";
import Error from "../components/Error";

import { getAllOrders ,deliverOrder, deleteOrder} from "../actions/orderAction";

export default function Orderslist() {
  const dispatch = useDispatch();
  const getordersstate = useSelector((state) => state.getAllOrdersReducer);
  const { orders, error, loading } = getordersstate;

  useEffect(() => {
    dispatch(getAllOrders());
  }, []);
  return (
    <div className="container-fluid" style={{backgroundColor: '#FFFF'}}>
      <h2>Reservations List</h2>
      {loading && <Loading />}
      {error && <Error error="something went wrong" />}
      <table className="table table-striped">
        <thead className="thead table-dark">
          <tr>
            <th>Order ID</th>
            <th>Email</th>
            <th>User ID</th>
            <th>Amount</th>
            <th>Payed Date</th>
            <th>Booked Date</th>
            <th>Status</th>
            <th>Actions</th>
          </tr>
        </thead>

        <tbody>
          {orders &&

```

```

orders.map((order) => {
  return (
    <tr>
      <td>{order._id}</td>
      <td>{order.email}</td>
      <td>{order.userid}</td>
      <td>{order.orderAmount}</td>
      <td>{order.createdAt.substring(0, 10)}</td>
      <td>
        {order.orderItems.map(item=>{
          return <div >
            {item.date}
          </div>
        })}
      </td>

      <td>
        {order.isDelivered ? (
          <button className="btn btn-warning"

            > Active Now </button>)
          : (
            <button className="btn btn-success"
              onClick={() => {dispatch(deliverOrder(order._id))}}
            > Checked In </button>
          )}
      </td>
      <td>
        <button className="btn btn-danger"
          onClick={() => {dispatch(deleteOrder(order._id))}} >
          <i className="fa fa-trash" style={{ color: "white" }}></i>
          &nbsp; Delete Reservation
        </button>
      </td>
    </tr>
  );
})}
</tbody>
</table>
</div>
);
}

```

Packageslist.js

```
import React, { useEffect, useState } from "react";
import { useDispatch, useSelector } from "react-redux";
import Loading from "../components/Loading";
import Error from "../components/Error";
import { getAllPackages, deletePackage } from "../actions/packageActions";

import { Link } from "react-router-dom";

export default function Packageslist() {
  const dispatch = useDispatch();

  const packagestate = useSelector((state) => state.getAllPackagesReducer);
  const { packages, error, loading } = packagestate;

  useEffect(() => {
    dispatch(getAllPackages());
  }, []);

  return (
    <div className="col-md-12" style={{backgroundColor: '#FFFF'}}>
      <h2>Packages List</h2>
      {error && <Error error="something went wrong" />}
      {loading && <Loading />}

      <table className="table table-striped">
        <thead className="thead table-dark">
          <tr>
            <th></th>
            <th style={{textAlign: "center"}}>Name</th>
            <th>Prices</th>
            <th>Category</th>
            <th>Actions</th>
          </tr>
        </thead>
        <tbody>
          {packages &&
            packages.map((pack, index) => {
              return (
                <tr>
                  <td>{index+1}</td>
                  <td style={{textAlign: "center"}}>{pack.name}</td>
                  <td>
                    Two Person : {pack.prices[0]["TwoPerson"]} <br />

```

```

        Four Person : {pack.prices[0]["FourPerson"]} <br />
        Family : {pack.prices[0]["Family"]}
    </td>

    <td>{pack.category}</td>
    <td>
        <Link to={`/admin/editpackage/${pack._id}`} className>
            <button className="btn btn-warning">
                <i className="fa fa-edit"></i>
                &nbsp;   Edit
            </button>
        </Link>

        &nbsp;   &nbsp;  
        <button className="btn btn-danger"
            onClick={()=>{dispatch(deletePackage(pack._id))}} >
            <i className="fa fa-trash" style={{ color: "white" }}></i>
            Delete
        </button>
    </td>
</tr>
);
    })}
</tbody>
</table>
</div>
);
}

```

Registerscreen.js

```

import React, { useState, useEffect } from "react";
import { useDispatch, useSelector } from "react-redux";
import { registerUser } from '../actions/userActions'
import Error from "../components/Error";
import Loading from "../components/Loading";
import Success from "../components/Success";

export default function Registerscreen() {
    const [name, setname] = useState("");
    const [email, setemail] = useState("");
    const [password, setpassword] = useState("");
    const [cpassword, setcpassword] = useState("");
    const [phone, setphone] = useState("");

```

```

const registerstate = useSelector(state=>state.registerUserReducer)
const {error,loading,success} = registerstate

const dispatch = useDispatch();

const validateEmail = (email) => {
  return email.match(
    /^((([^<>()[]\.\,;\s@"']+(\.([^<>()[]\.\,;\s@""]+)*)|("\.+\\"))@((\[[0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,})))$/
  );
};

function register(){

  if(!validateEmail(email)){
    alert("Your Email is not valid");
  }
  else{
    if(phone==""){
      alert("Please enter phone number")
    }
    else
    if(name==""){
      alert("Please enter your name")
    }

    else
    if(password.length<6){
      alert("Password must be atleast 6 characters")
    }

    else
    if(password!=cpassword){
      alert("password and confirm password do not match")
    }
  }
}

```



```

    <strong> Email already use</strong>

</div>) : ''}

    <div >
      <input
        type="text"
        className="form-control"
        placeholder="Name"
        required
        value={name}
        onChange={(e)=>{setname(e.target.value)}}
        style={{ backgroundColor: "#212121", color: "white",
borderRadius: "10px" , border: "white solid 1px !important"}}
      />
      <input type="text"
        className="form-control"
        placeholder="Email"
        required
        value={email}
        onChange={(e)=>{setemail(e.target.value)}}
        style={{ backgroundColor: "#212121", color: "white",
borderRadius: "10px" , border: "white solid 1px !important"}}
      />
      <input
        type="text"
        className="form-control"
        placeholder="Phone"
        required
        value={phone}
        onChange={(e)=>{setphone(e.target.value)}}
        style={{ backgroundColor: "#212121", color: "white",
borderRadius: "10px" , border: "white solid 1px !important"}}
      />
      <input
        type="text"
        className="form-control"
        placeholder="Password"
        required
        value={password}
        onChange={(e)=>{setpassword(e.target.value)}}

```

```

        style={{ backgroundColor: "#212121", color: "white",
borderRadius: "10px" , border: "white solid 1px !important"}}

        />
        <input
            type="text"
            className="form-control"
            placeholder="Confirm Password"
            required
            value={cpassword}
            onChange={(e)=>{setcpassword(e.target.value)}}
            style={{ backgroundColor: "#212121", color: "white",
borderRadius: "10px" , border: "white solid 1px !important"}}
        />

        <button type="submit" className="btn btn-danger mt-2 mb-2"
            onClick={register}>
            Register
        </button>
        <br/>
        <a href="/login" style={{color:"white", textDecoration: "none"}}
className="mt-2 ml-2"> Klik here to Login</a>
    </div>
</div>
</div>
</div>
);
}

```

Taxilist.js

```

import React, { useEffect, useState } from "react";
import { useDispatch, useSelector } from "react-redux";
import Loading from "../components/Loading";
import Error from "../components/Error";

import { getAllOrders ,deliverOrder, deleteOrder} from "../actions/orderAction";

import {getAlltaxi, deleteTaxiOrder} from '../actions/taxiActions'

export default function Taxilist() {
    const dispatch = useDispatch();
    const gettaxistate = useSelector((state) => state.getAllTaxiReducer);
    const { taxis, error, loading } = gettaxistate;

```



```

useEffect(() => {
  dispatch(getAlltaxi());
}, []);

return (

  <div className="container-fluid" style={{backgroundColor: '#FFFF'}}>
    <h2>Taxi List</h2>

    {error && <Error error="something went wrong" />}
    <table className="table table-striped">
      <thead className="thead table-dark">
        <tr>
          <th>Taxi Booking ID</th>
          <th>Customer Name</th>
          <th>Address</th>

          <th>Booked Date</th>
          <th>Phone</th>
          <th>Reservation Date</th>
          <th>Actions</th>
        </tr>
      </thead>

      <tbody>
        {taxis &&
          taxis.map((taxi) => {
            return (
              <tr>
                <td>{taxi._id}</td>
                <td>{taxi.name}</td>
                <td>{taxi.address}</td>
                <td>{taxi.date}</td>
                <td>{taxi.phone}</td>

                <td>{taxi.createdAt.substring(0, 10)}</td>
              </tr>
            )
          })
        }
      </tbody>
    </table>
  </div>
)

```

```

        <td>
        <button className="btn btn-danger"
            onClick={()=>{dispatch(deleteTaxiOrder(taxi._id))}} >
            <i className="fa fa-trash" style={{ color: "white" }}></i>
            &nbsp; Delete Reservation
        </button>
        </td>
    </tr>
    </tbody>
</table>
</div>
);
}

```

Taxy.js

```

import React,{ useEffect, useState} from 'react'
import {useSelector,useDispatch} from 'react-redux';
import {getOrderByid} from '../actions/orderAction';
import {getUserOrders} from '../actions/orderAction'
import {bookTaxi} from '../actions/taxiActions';
import Loading from '../components/Loading';
import Error from '../components/Error';
import {Link} from 'react-router-dom';
import SimpleImageSlider from "react-simple-image-slider";

export default function Taxy({match}) {

    const [name, setname] = useState("");
    const [address, setaddress] = useState("");
    const [email, setemail] = useState("");
    const [date, setdate] = useState("");

    const userState = useSelector(state => state.loginUserReducer);
    const {currentUser} = userState

    const getorderbyidstate = useSelector((state) => state.getOrderByidReducer);
    const { order, success, loading } = getorderbyidstate;

    const dispatch = useDispatch();

```

```

useEffect(() => {

  if(order){

    if(order._id==match.params.orderid){

      setname(order.name);
      setaddress(order.shippingAddress.street + ", " +
order.shippingAddress.city + ", " + order.shippingAddress.country);
      setemail(order.email)

      setdate(order.orderItems[order.orderItems.Length-1].date)

    }
    else{
      dispatch(getOrderById(match.params.orderid));
    }
  }
  else{
    dispatch(getOrderById(match.params.orderid));
  }

}, [order , dispatch]);

function formHandler(e){
  e.preventDefault()

  const taxi ={
    id: match.params.orderid,
    userid: currentUser._id,
    name: order.name,
    address,
    email: order.email,
    date,
    phone: currentUser.phone,
  }
  console.log(taxi)

  dispatch(bookTaxi(taxi));

```

```

    }

    const images = [
      {url:
'https://res.cloudinary.com/vihanga/image/upload/v1651903372/ds/781967_zdilas.jpg
'},
      {url:
'https://res.cloudinary.com/vihanga/image/upload/v1651903199/ds/781965_cn1wlh.jpg
'},
      { url:
'https://res.cloudinary.com/vihanga/image/upload/v1651903373/ds/781969_tnv9wh.jpg
'},
      { url:
'https://res.cloudinary.com/vihanga/image/upload/v1651903373/ds/781970_aak0jg.jpg
'},
      { url:
'https://res.cloudinary.com/vihanga/image/upload/v1651903373/ds/781971_qxqfqr.jpg
'},
    ];

    return (
      <div>

        <div className="mt-5 text-center" style={{textAlign: 'center'}}>

          <nav class="navbar sticky-top navbar-collapse justify-content-center text-
left navbar-light bg-light mt-5 p-3 ">
            <a class="navbar-brand" href="" style={{marginLeft: '90px',}}>

              
              <strong>Taxi Service</strong>
            </a>
          </nav>

```

```

<div className="container">

  <div className="w3-container">

    <SimpleImageSlider style={{}}
      width={1125}
      height={504}
      images={images}
      showBullets={true}
      showNavs={true}
    />

  </div>

</div>

</div>

  <div >

    {success && (<h1>Deliver success</h1>)}

  </div>

  <div class="form-outline ml-5 p-3">

    <div class="form-group ml-5 p-3" style={{backgroundColor:"#212121",
      color:"white" , borderRadius:"30px",opacity: '.98', width:"600px" }}>
      <form onSubmit={formHandler} className="container text-left"
style={{width:'400px'}} >
        <div class="w3-container">
          <h2>Book Taxi </h2>
        </div>

        <label style={{marginBottom: '-10px'}}>Address</label>

```

```

<input
    id="form12" class="form-control"
    type="text"
    value={address}
    placeholder="address"
    onChange={(e) => {
        setaddress(e.target.value);
    }}
/>

<label className="mt-2 " style={{marginBottom: '-10px'}}>Date</label>
<input
    className="form-control"
    type="text"
    value={date}
    placeholder="address"
    onChange={(e) => {
        setdate(e.target.value);
    }}
/>
<button type="submit" className="btn btn-warning" style={{marginTop: '20px'}} >
    Book Taxi
</button>

    </form>
  </div>
</div>
</div>
)
}

```

Userslist.js

```
import React, { useEffect,useState } from 'react'
import { useDispatch, useSelector } from 'react-redux'
import { getAllUsers, deleteUser, makeHotelAdmin, removeHotelAdmin } from
'../actions/userActions';

export default function Userslist() {

  const dispatch = useDispatch();
  const getUsersstate = useSelector((state) => state.getAllUsersReducer);
  const { users, error, loading } = getUsersstate;
  const userState = useSelector(state => state.loginUserReducer);
  const {currentUser} = userState

  useEffect(() => {
    dispatch(getAllUsers());
  }, []);
  return (
    <div className="container-fluid" style={{backgroundColor: '#FFFF'}}>
      <h2>Users List</h2>
      <table className="table table-striped">
        <thead className="thead table-dark">
          <tr>
            <th></th>
            <th style={{ textAlign: "center" }}>Name</th>
            <th>email</th>
            <th>System Admins</th>
            <th>Hotel Admins</th>
            {currentUser.isAdmin ? (
              <th>All Users</th>
            ) : (<th></th>)}
            {currentUser.isAdmin ? (
              <th>User Actions</th>
            ) : (<th></th>)}

            {currentUser.isAdmin ? (
              <th>Create Hotel Admins</th>
            ) : (<th></th>)}

          </tr>
```

```

</thead>
<tbody>
  {users &&
    users.map((user,index) => {
      return (
        <tr>
          <td>{index+1}</td>
          <td style={{ textAlign: "center" }}>{user.name}
{user.isAdmin===true?
      ( <h4 style={{ color:"green"}}> (System Admin)</h4> ):( "")

        </td>
        <td>{user.email}
        </td>
        <td>

          { /* {user.isAdmin ? "Yes" : "No"} */}
          {user.isAdmin===true?
            ( <h4 style={{ color:"green"}}> System Admin</h4> ):( "Not
Admin")

          }

        </td>
        <td>
{user.isHotelAdmin===true?
      ( <h4 style={{ color:"green"}}>Hotel Admin</h4> ):( "Not Hotel
Admin")

    }
    </td>

    <td>
{user.isHotelAdmin===true?
      ( "Staff" ):( "User" )
    }
    </td>

    <td>
{currentUser.isAdmin? (

      <button type="button" className="btn btn-danger"
onClick={()=>{dispatch(deleteUser(user._id))}} >
      <i className="fa fa-trash" style={{ color: "white" }}></i>
      Block User

```



```

        </button>

       ):(null)}}
    </td>

    <td>
        {currentUser.isAdmin? (

        user.isHotelAdmin ? (
            <button className="btn btn-secondary"
                onClick={() => {dispatch(removeHotelAdmin(user._id))}}
            > Remove Hotel Admin </button>
        ) : (
            <button className="btn btn-success"

            onClick={() => {dispatch(makeHotelAdmin(user._id))}}
            > Make Hotel Admin </button>
        )

       ):(null)}}

    </td>

</tr>
);
}}
</tbody>
</table>
</div>
)
}

```

App.js

```
import logo from "../logo.svg";
import "../App.css";
import { BrowserRouter, Route, Link, Switch, Redirect } from "react-router-dom";
import bootstrap from "../node_modules/bootstrap/dist/css/bootstrap.min.css";
import Navibar from "../components/Navibar";
import Cartscreen from "../screens/Cartscreen";
import Registerscreen from "../screens/Registerscreen";
import Loginscreen from "../screens/Loginscreen";
import Orders from "../screens/Orders";
import Adminscreen from "../screens/Adminscreen";
import Taxy from "../screens/Taxy";
import OrderedTaxi from "../screens/OrderedTaxi";
import HotelAdminScreen from "../screens/HotelAdminScreen";
import Orderslist from "../screens/Orderslist";
import Userslist from "../screens/Userslist";
import Taxilist from "../screens/Taxilist";
import Addpackage from "../screens/Addpackage";
import HomeScreenPkg from "../screens/HomeScreenPkg";
import Packageslist from "../screens/Packageslist";
import Editpackage from "../screens/Editpackage";
import Footer from "../components/Footer";

function App() {
  return (
    <div className="App">
      <Navibar />
      <BrowserRouter>
        <Route exact path="/">
          <Redirect to="/homepkg" />
        </Route>
        { /*
        <Route path="/" exact component={Homescreen}/> */}
        <Route path="/cart" exact component={Cartscreen} />
        <Route path="/register" exact component={Registerscreen} />
        <Route path="/login" exact component={Loginscreen} />
        <Route path="/ord" exact component={Orders} />
        <Route path="/admin" component={Adminscreen} />
        { /* <Route path="/taxy/:orderid?/:orderamount?" exact
component={Taxy}/> */}
        <Route path="/taxy/:orderid" exact component={Taxy} />
        <Route path="/taxiord" exact component={OrderedTaxi} />
        <Route path="/hoteladmin" component={HotelAdminScreen} />
        <Route path="/admin/userslist" component={Userslist} />
```

```

        <Route path="/admin/orderslist" component={Orderslist} />
        <Route path="/admin/taxilist" component={Taxilist} />
        <Route path="/admin/addpackage" component={Addpackage} />
        <Route path="/homepkg" component={HomeScreenPkg} />
        <Route path="/admin/packageslist" component={Packageslist} />
        <Route path="/admin/editpackage/:packageid" component={Editpackage} />
    </BrowserRouter>
    <Footer />
  </div>
);
}

export default App;

```

store.js

```

import {combineReducers} from 'redux';

import {createStore, applyMiddleware} from 'redux'

import thunk from 'redux-thunk'

import {composeWithDevTools} from 'redux-devtools-extension'

import { getAllPackagesReducer , addPackageReducer, getPackageByIdReducer,
editPackageReducer} from './reducers/packageReducer';

import { cartReducer } from './reducers/cartReducer';

import { registerUserReducer, loginUserReducer, getAllUsersReducer } from
'./reducers/userReducer';

import { placeOrderReducer , getUserOrdersReducer ,getAllOrdersReducer,
getOrderByIdReducer} from './reducers/orderReducer';

import { bookTaxiReducer , getOrderedTaxireducer , getAllTaxiReducer} from
'./reducers/taxiReducer';

const finalReducer = combineReducers({

```

```

    cartReducer: cartReducer,
    registerUserReducer: registerUserReducer,
    loginUserReducer: loginUserReducer,
    placeOrderReducer: placeOrderReducer,
    getUserOrdersReducer: getUserOrdersReducer,
    getAllOrdersReducer: getAllOrdersReducer,
    getAllUsersReducer: getAllUsersReducer,
    getOrderByIdReducer: getOrderByIdReducer,
    bookTaxiReducer: bookTaxiReducer,
    getOrderedTaxiReducer: getOrderedTaxiReducer,
    getAllPackagesReducer: getAllPackagesReducer,
    addPackageReducer: addPackageReducer,
    getPackageByIdReducer: getPackageByIdReducer,
    editPackageReducer: editPackageReducer,

    getAllTaxiReducer: getAllTaxiReducer

  })

const cartItems =
localStorage.getItem('cartItems') ? JSON.parse(localStorage.getItem('cartItems')) : [
]
const currentUser =
localStorage.getItem('currentUser') ? JSON.parse(localStorage.getItem('currentUser')) : null

const initialState = {
  cartReducer: {
    cartItems: cartItems
  },
  loginUserReducer: {
    currentUser: currentUser
  }
}

const composeEnhancers = composeWithDevTools({})

const store = createStore(finalReducer, initialState,
composeEnhancers(applyMiddleware(thunk)))

export default store;

```

The End!