Sri Lanka Institute of Information Technology

Online Tourism Application

Category: Tourism

Project Report

Application Frameworks Project 2020
Year 3 Semester 1
Final Exam

Submitted by:

IT18002034	D.S.R.C.V Perera

Question 01 – Introduction

Tripper, the introducing online tourism application facilitates its users to view and apply for the travel packages, which are added by the admins. A travel package consists of its own name, destination country, days of visit, and the cost for the travel package. Besides the users can register to the system using the sign-up form and a login user is implemented to access the system. In addition, a login for admins is also implemented to add new travel packages, and to manipulate their information, and by other hand to view the user information who are registered to the system.

The project is implemented and designed using MERN (mongoDB + Express + ReactJS + Node) full stack principles. And all type of CRUD functions are performed in this application.

Question 02 – Main functionalities

1.Add packages by admin

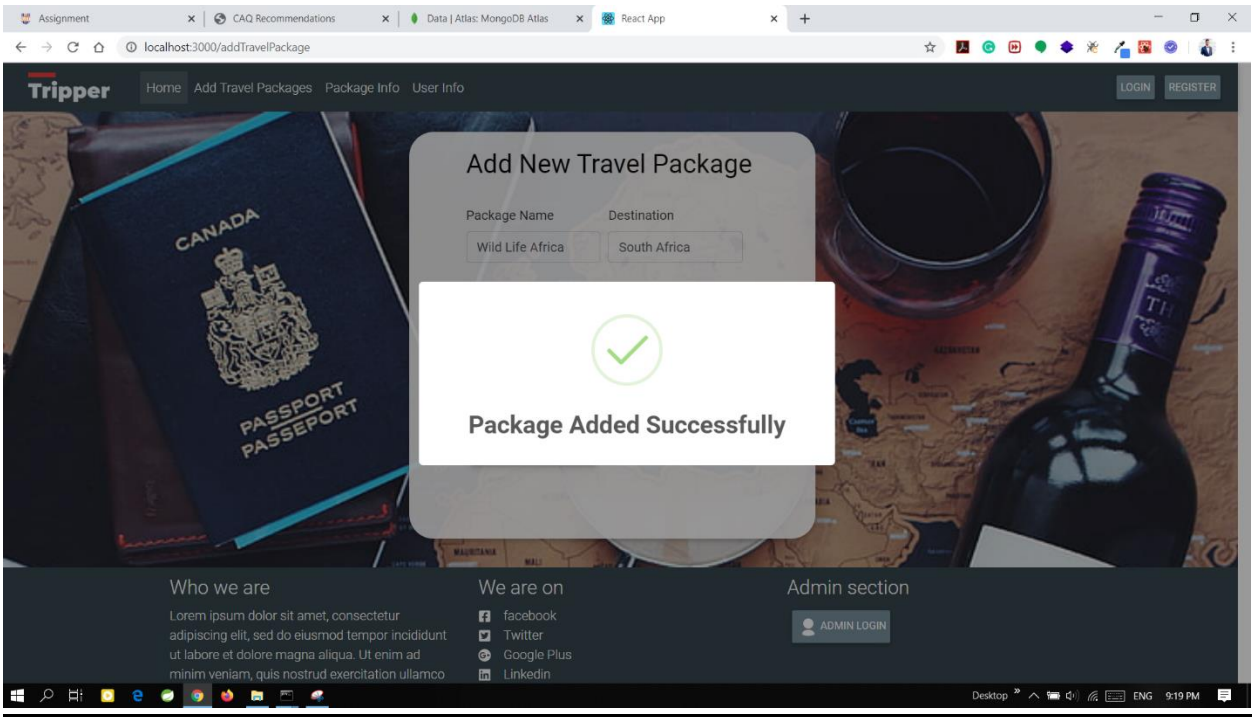

Figure 1.0

The system only allows the admins to add new travel packages. The Admins can add new packages by entering the name, destination, days, and cost of the packages. If the package is added to the system, it will display the pop-up message on the screen as shown in figure 1.0.

Backend

Model – package.js

```
var mongoose = require('mongoose');
const schema = mongoose.Schema;

const packageSchema = new schema({
    packageName: {
        type: String,
        required: true
    },
    destination: {
        type: String,
        required: true
```

```
    },
    days: {
        type: Number,
        required: true
    },
    cost: {
        type: Number,
        required: true
    },


})
module.exports = Package = mongoose.model('packages', packageSchema)
```

Routes – AdminRoutes.js

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

/*
    method : POST
    description : create new travelpackage
    params : Body {
        package name,destination,noofdays,cost
    }
*/
router.post('/createPackage', (req, res) => {

    const pack = new Package({
        packageName: req.body.packageName,
        destination: req.body.destination,
        days: req.body.days,
        cost: req.body.cost

    })
    pack.save().then(pkg => {
        console.log("package Added")
        try {
            res.status(200).send({
                message: 'package created successfully !',
                data: pkg,
                messageCode: "1000"
            })
```

```
        } catch (err) {
            res.status(502).send({
                message: 'OOPS ! server error',
                error: err
            })
        }

    })
})
})
module.exports = router;
```

Frontend

AddTravelPackages.js

```
import React, { Component } from 'react'
import axios from 'axios';
import Swal from 'sweetalert2'

class AddTravelPackage extends Component {
    constructor(props) {
        super(props);
        this.state = {
            packageName: '',
            destination: '',
            days: '',
            cost: '',


            packRes: '',

        };
    }
    updatepkName = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            packageName: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.packageName);
```

```
        });

        console.log("New state DIRECTLY after setState:", this.state.packageName)
;


    }

    updateDestination = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            destination: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.destination);
        });

        console.log("New state DIRECTLY after setState:", this.state.destination)
;
        console.log("New state DIRECTLY after setState:", this.state.destination)
;
    }
    updateDuration = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            days: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.days);
        });

        console.log("New state DIRECTLY after setState:", this.state.days);
        console.log("New state DIRECTLY after setState:", this.state.days);
    }
    updateCost = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            cost: parseInt(event.target.value)
        }, () => {
```

```
            console.log("New state in ASYNC callback:", this.state.cost);
        });

        console.log("New state DIRECTLY after setState:", this.state.cost);
        console.log("New state DIRECTLY after setState:", this.state.cost);
    }


    addNewPackage = (event) => {
        if ((this.state.packageName === "") || (this.state.destination === "") ||
 (this.state.days === "") || (this.state.cost === "")) {
            Swal.fire({
                position: 'middle',
                icon: 'error',
                title: 'Oops...',
                text: 'Check your inputs again!',
                timer: 1500
            })

        }

        event.preventDefault();
        // alert("successfully added")
        let pkBody = JSON.stringify(
            {
                "packageName": this.state.packageName,
                "destination": this.state.destination,
                "days": this.state.days,
                "cost": this.state.cost,

            }
        );
        axios({
            headers: {
                //'Content-Type ': 'application/x-www-form-
urlencoded;charset=UTF-8',
                // 'Content-Type': 'application/json, text/plain, */*',
                'Content-Type': 'application/json;charset=UTF-8',
            },

            method: 'POST',
            url: 'http://localhost:3800/api/admins/createPackage',
            data: pkBody,

        })
```

```
            .then(response => {
                console.log("Arrived to send request")
                this.setState({
                    packRes: response.data
                })

            })
            .then(() => {
                Swal.fire({
                    position: 'middle',
                    icon: 'success',
                    title: 'Package Added Successfully',
                    showConfirmButton: false,
                    timer: 3500
                })
            })

            .catch((console.log("ISSUES !")))

    }


    render() {
        return (
            <div style={{ backgroundImage: 'url("https://c4.wallpaperflare.com/wa
llpaper/714/452/950/passport-map-wine-of-greece-magnifier-wallpaper-
preview.jpg")', backgroundSize: "cover", position: "relative", height: "560px" }}
>
                <div className="card" style={{ opacity: 0.9, borderRadius: 30, po
sition: 'absolute', marginTop: 25, height: 500, width: 500, justifyContent: 'cent
er', marginLeft: 500 }} >
                    <div className="card-body">
                        <form style={{ marginLeft: 50 }}>
                            <h2 style={{ color: 'black' }}>Add New Travel Package
</h2>
                            <br />
                            <div className="form-row">
                                <div className="col-md-5 mb-4">
                                    <label htmlFor="validationDefault01">Package
Name</label>
                                    <input type="text" className="form-
control" id="validationDefault01" placeholder="package name" required
                                        onChange={this.updatepkName} />
                                </div>
                                <div className="col-md-5 mb-3">
```

```
                                    <label htmlFor="validationDefault02">Destinat
ion</label>
                                    <input type="text" className="form-
control" id="validationDefault02" placeholder="destination" required
                                        onChange={this.updateDestination} />
                                </div>
                            </div>
                            <div className="form-row">
                                <div className="col-md-10 mb-3">
                                    <label htmlFor="inputEmail4">Duration</label>
                                    <input type="email" className="form-
control" id="inputEmail4" placeholder="no of days"
                                        onChange={this.updateDuration} />
                                </div>
                            </div>
                            <div className="form-row">
                                <div className="col-md-10 mb-3">
                                    <label htmlFor="validationDefault04">Cost</la
bel>
                                    <input type="text" className="form-
control" id="validationDefault04" placeholder="Cost" required
                                        onChange={this.updateCost} />
                                </div>

                            </div>

                            <button onClick={this.addNewPackage} className="btn b
tn-primary" type="submit">Add Package</button>
                        </form>
                    </div>
                </div>
            </div>
        )
    }
}
export default AddTravelPackage;
```

2. Search packages by package name

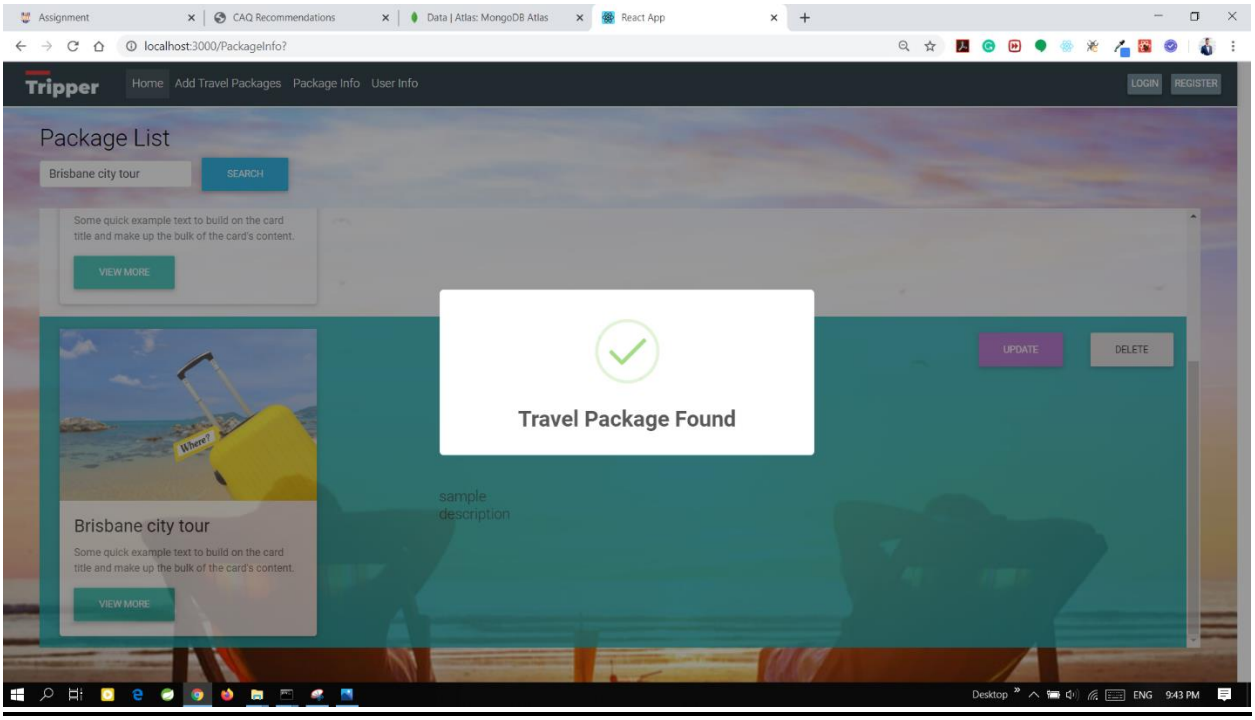

Figure 2.0

An admin can search the travel packages by entering the name of the package as the search key and then the particular the result will be highlighted in the list as shown in figure 2.0. If the program can identify the matched package name, it will show the pop-up message on the screen.

Backend

Routes – AdminRoutes.js

```
/*
method : GET
    description : search package by packagenName
*/
router.get('/getbyPackageName/:Name', (req, res) => {
    Package.findOne({
        packageName: req.params.Name
    }).then(pkg => {


        try {
            res.status(200).send({
                message: 'Employee retrived successfully ok !',
                data: pkg
```

```
            })

        } catch (err) {
            res.status(502).send({
                message: 'OOPS ! server error',
                error: err
            })
        }

    })
})
```

Frontend

PackageInfo.js

```
import React, { Component } from 'react'
import axios from 'axios';
import { MdAccountCircle } from "react-icons/md";
import Modal from 'react-modal'
import Swal from 'sweetalert2'
import { wait } from '@testing-library/react';
import { MDBBtn, MDBCard, MDBCardBody, MDBCardImage, MDBCardTitle, MDBCardText, M
DBCol } from 'mdbreact';
import { MdLocationOn } from "react-icons/md";
import { MdDateRange } from "react-icons/md";
import { AiFillDollarCircle } from "react-icons/ai";
import { GiCancel } from "react-icons/gi";



function sleep(milliseconds) {
    const date = Date.now();
    let currentDate = null;
    do {
        currentDate = Date.now();
    } while (currentDate - date < milliseconds);
}
class PackageInfo extends Component {
    constructor(props) {
        super(props);
        this.state = {
            packDetails: [],

            searchKey: "",
```

```
            searchResult: "",


            currentPkgName: "",

            showModal: false,

            packageName: '',
            destination: '',
            days: '',
            cost: '',


            pkgRes: '',

            upPackageName: '',
            upDestination: '',
            upDays: '',
            upCost: '',

            currentPkgID: '',
            refresh: false



        };
    }
    componentDidMount() {
        this.loadData();
    }


    loadData = () => {


        let baseURL = "http://localhost:3800/api/admins/getAllPackages";
        axios({
            headers: {
                'Content-Type': 'application/json;charset=UTF-8',
            },


            method: 'GET',
            url: baseURL,
```

```
        })
            .then(response => {
                this.setState({
                    packDetails: response.data
                })
                console.log("success")
                console.log(this.state.packDetails)
                var len = this.state.packDetails.length;

                console.log(len + "this is the lenth")

            })
        //setting the auto refresh time as every 40 secs
        //.then(setInterval(this.loadData, 40000))


    }


    deletePackage = (_id) => {
        console.log("arrived to delete")
        console.log(_id + " this is id")

        let baseURL = "http://localhost:3800/api/admins/deletePackage/" + _id;
        console.log("2nd ok")

        axios({


            method: 'DELETE',
            url: baseURL,


        })

            .then(response => {
                console.log(response.status)
                if (response.status === 200) {
                    Swal.fire({
                        position: 'center',
                        icon: 'success',
                        title: 'Package deleted successfully',
                        showConfirmButton: false,
                        timer: 3000
                    })
```

```
            }



        }).then(this.doFrefresh)

    //    window.location.reload(false)

    // .then(setTimeout(() => { console.log("World!"); }, 5000))
    //  .then(window.location.reload(true), 4000)
    //setting the auto refresh time as every 40 secs
    //.then(setInterval(this.loadData, 40000))


}
doFrefresh() {
    sleep(2000);
    window.location.reload(false)
}
captureInput = event => {
    event.preventDefault()
    console.log(event.target.value)
    var val = event.target.value
    console.log(val + "this is the value retrived")
    this.setState({
        searchKey: event.target.value
    }, () => {
        console.log("New state in ASYNC callback:", this.state.searchKey);
    });

    console.log("New state DIRECTLY after setState:", this.state.searchKey);
    console.log("New state DIRECTLY after setState:", this.state.searchKey);
}

searchPackage = (event) => {


    event.preventDefault();
    axios({
        headers: {
            'Content-Type': 'application/json;charset=UTF-8',
        },
```

```
            method: 'GET',
            url: 'http://localhost:3800/api/admins/getbyPackageName/' + this.stat
e.searchKey,
        })
            .then(response => {
                console.log("Arrived to send request")
                if (response.status === 200) {

                    this.setState({
                        searchResult: response.data
                    })
                    if (this.state.searchResult.data === null) {
                        Swal.fire({
                            position: 'middle',
                            icon: 'error',
                            title: 'Oops...',
                            text: 'No results found!',
                            timer: 1500
                        })
                    }
                    else {
                        Swal.fire({
                            position: 'middle',
                            icon: 'success',
                            title: 'Travel Package Found',
                            showConfirmButton: false,
                            timer: 1500
                        })


                        console.log(this.state.searchResult.data.packageName)
                        this.setState({
                            currentPkgName: this.state.searchResult.data.packageN
ame
                        })
                        console.log("this is package name  =Found=" + this.state.
currentPkgName)
                    }

                }

            })
            .catch((console.log("ISSUES !")))
```

```
    }
    updatePackageName = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            packageName: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.packageName);
        });

        console.log("New state DIRECTLY after setState:", this.state.packageName)
;


    }

    updateDestination = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            destination: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.destination);
        });

        console.log("New state DIRECTLY after setState:", this.state.destination)
;
        console.log("New state DIRECTLY after setState:", this.state.destination)
;
    }
    updatedays = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            days: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.days);
        });

        console.log("New state DIRECTLY after setState:", this.state.days);
```

```
        console.log("New state DIRECTLY after setState:", this.state.days);
    }
    updatecost = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            cost: parseInt(event.target.value)
        }, () => {
            console.log("New state in ASYNC callback:", this.state.cost);
        });

        console.log("New state DIRECTLY after setState:", this.state.cost);
        console.log("New state DIRECTLY after setState:", this.state.cost);
    }

    go = (packageName, destination, days, cost) => {
        this.setState({
            upPackageName: packageName,
            upDestination: destination,
            upDays: days,
            upCost: cost,

        })
        console.log(this.state.upPackageName + " this is the name to show first")

    }

    updatePakagedeatils = (_id) => {

        //event.preventDefault();
        // alert("successfully added")
        if (this.state.packageName === "" ||
            this.state.destination === "" ||
            this.state.days === "" ||
            this.state.cost === "") {
            Swal.fire({
                position: 'middle',
                icon: 'error',
                title: 'Oops...',
                text: 'Check your Inputs!',
                timer: 1500
            })
        }
```

```
        else {
            let pkgbody = JSON.stringify(
                {
                    "packageName": this.state.packageName,
                    "destination": this.state.destination,
                    "days": this.state.days,
                    "cost": this.state.cost

                }
            );
            axios({
                headers: {
                    'Content-Type': 'application/json;charset=UTF-8',
                },

                method: 'PUT',
                url: 'http://localhost:3800/api/admins/updatePackage/' + _id,
                data: pkgbody,

            })
                .then(response => {
                    console.log("Arrived to send request")
                    this.setState({
                        pkgRes: response.data
                    })
                })
                .then(() => {
                    Swal.fire({
                        position: 'middle',
                        icon: 'success',
                        title: 'Package Updated Successfully',
                        showConfirmButton: false,
                        timer: 3500
                    })
                    this.setState({
                        showModal: false
                    })
                })

                .catch((console.log("ISSUES !")))


        }
```

```
    }


    render() {

        return (
            <div style={{ backgroundImage: 'url("https://i.inews.co.uk/content/uploads/2020/05/holiday.jpg")', backgroundSize: "cover", position: "relative", height: "800px" }}>
                <div style={{ marginLeft: 50, marginRight: 50, marginBottom: 200, }}>
                    <br />
                    <h2 style={{ color: "black", justifyContent: 'center', alignItems: 'center' }}>Package List </h2>

                    <form className="form-inline my-2 my-lg-0">
                        <input className="form-control mr-sm-2" type="search" placeholder="Package Name" aria-label="Search" onChange={this.captureInput} required />
                        <button className="btn btn-info my-2 my-sm-0" type="submit" onClick={this.searchPackage}>Search</button>
                    </form>
                    <br></br>

                    <table className="table table-light" style={{ height: 600, width: "100%", overflow: "auto", display: "block", opacity: 0.8 }}>


                        <thead>

                        </thead>
                        <tbody style={{ opacity: 1 }}>
                            {this.state.packDetails.map(pkg =>

                                <tr style={{ backgroundColor: this.state.currentPkgName === pkg.packageName ? '#22aeb5' : '' }} >

                                    {/* MODA for update package------------------------------------------------------------------------------------------------------------ */}
                                    <Modal isOpen={this.state.showModal}
                                        style={{
                                            overlay: {
                                                position: 'fixed',
```

```
                                        top: 0,
                                        left: 0,
                                        right: 0,
                                        bottom: 0,
                                        backgroundColor: 'rgba(255, 255,
255, 0.2)',


                                    },
                                    content: {
                                        position: 'absolute',
                                        top: '80px',
                                        left: '540px',
                                        right: '540px',
                                        bottom: '100px',
                                        border: '4px solid #ccc',
                                        background: '#fff',
                                        overflow: 'auto',
                                        // WebkitOverflowScrolling: 'touc
h',

                                        borderRadius: '30px',
                                        outline: '5px',
                                        padding: '20px'
                                    }
                                }}
                            >
                                <GiCancel style={{ marginLeft: 520 }} siz
e={30} onClick={() => this.setState({ showModal: false })} />
                                <form style={{ marginBottom: 100, marginL
eft: 100 }}>

                                    <br />
                                    <br />
                                    <h2 style={{ color: 'black' }}>Update
 Package Details</h2>

                                    <br />
                                    <div className="form-row">
                                        <div className="col-md-4 mb-3">
                                            <label htmlFor="validationDef
ault01">Package Name</label>

                                            <input type="text" className=
"form-
control" id="validationDefault01" placeholder={this.state.upPackageName} required
                                                onChange={this.updatePack
ageName} />

                                        </div>
```

```
                                        <div className="col-md-4 mb-3">
                                            <label htmlFor="validationDef
ault02">Destination</label>
                                            <input type="text" className=
"form-
control" id="validationDefault02" placeholder={this.state.upDestination} required
                                                onChange={this.updateDest
ination} />
                                        </div>
                                    </div>
                                    {/* <div className="form-row">
                                        <div className="form-group col-
md-8">
                                            <label htmlFor="validationDef
ault04">Days</label>
                                            <input type="email" classNam
e="form-control" id="validationDefault01" placeholder={this.state.upDays}
                                                onChange={this.updatedays
} />
                                        </div>
                                    </div> */}
                                    <div className="form-row">
                                        <div className="form-group col-
md-8">
                                            <label htmlFor="validationDef
ault02">Days</label>
                                            <input type="text" className=
"form-control" id="validationDefault02" placeholder={this.state.upDays} required
                                                onChange={this.updatedays
} />
                                        </div>
                                    </div>


                                    <div className="form-row">
                                        <div className="form-group col-
md-8">
                                            <label htmlFor="validationDef
ault04">Cost</label>
                                            <input type="text" className=
"form-control" id="validationDefault04" placeholder={this.state.upCost}
                                                onChange={this.updatecost
} />
                                        </div>
```

```
                                        </div>
                                        <div className="form-row">
                                            <div className="col-md-4 mb-3">
                                                <button onClick={() => this.u
pdatePakagedeatils(this.state.currentPkgID)} className="btn btn-
primary" type="submit" >Update</button>
                                            </div>
                                            <div className="col-md-4 mb-3">
                                                <button onClick={() => this.s
etState({ showModal: false })} className="btn btn-primary" >Close</button>
                                            </div>
                                        </div>

                                    </form>
                                </Modal>
                                {/* MODAL-------------------------------------
-------------------------------------------------------------------------
---------- */}
                                <td style={{}}><MDBCol>
                                    <MDBCard style={{ width: "22rem" }}>
                                        <MDBCardImage className="img-
fluid" src="https://marketingland.com/wp-content/ml-loads/2016/05/travel-search-
suitcasebeach-ss-1920.png" waves />
                                        <MDBCardBody>
                                            <MDBCardTitle>{pkg.packageName}</
MDBCardTitle>

                                            <MDBCardText>
                                                Some quick example text to bu
ild on the card title and make

                                                up the bulk of the card&amp;apos;
s content.
          </MDBCardText>

                                            <MDBBtn href="https://www.google.
com/">View more</MDBBtn>
                                        </MDBCardBody>
                                    </MDBCard>
                                </MDBCol>
                                </td>
                                <td style={{ paddingLeft: 140, marginTop: 10
}}> <MdLocationOn size={20} /> {pkg.destination} <br /><br /><MdDateRange size={2
0} /> {pkg.days} Days <br /><br /><AiFillDollarCircle size={20} />  {pkg.cost}
                                    <br /><br /> <h2>Description</h2> <br /><
br /> <h5>sample description</h5> </td>
                                <td style={{ paddingLeft: 140 }}></td>
                                <td style={{ paddingLeft: 140 }}></td>
```

```
                                    <td style={{ paddingLeft: 140 }}></td>
                                    <td style={{ paddingLeft: 100 }}> <button typ
e="button" className="btn btn-secondary" onClick={() => {
                                        this.go(pkg.packageName, pkg.destination,
 pkg.days, pkg.cost)
                                        this.setState({
                                            currentPkgID: pkg._id
                                        })

                                        this.setState({ showModal: true });
                                    }}>Update</button></td>

                                    <td><button type="button" onClick={() => this
.deletePackage(pkg._id)} className="btn btn-light">Delete</button></td>
                                    <hr />

                                </tr>

                            )}


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

export default PackageInfo;
```

3. Update and delete travel packages by admin

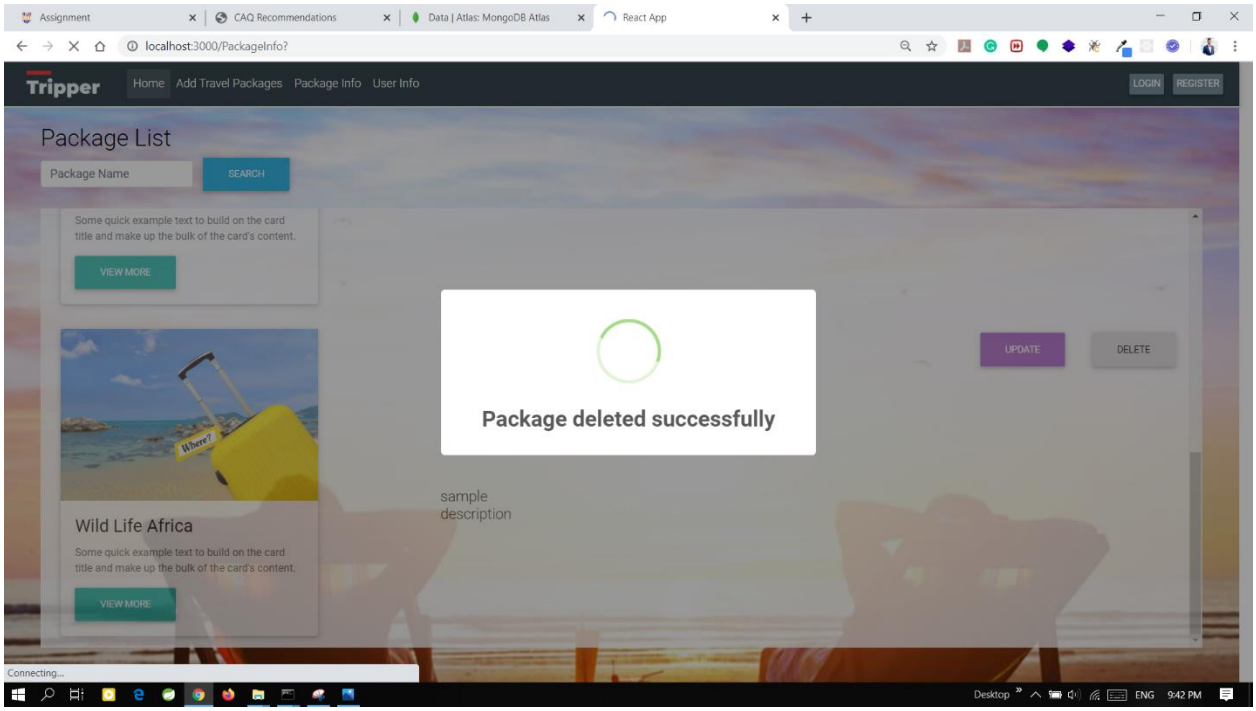

Figure 2.1

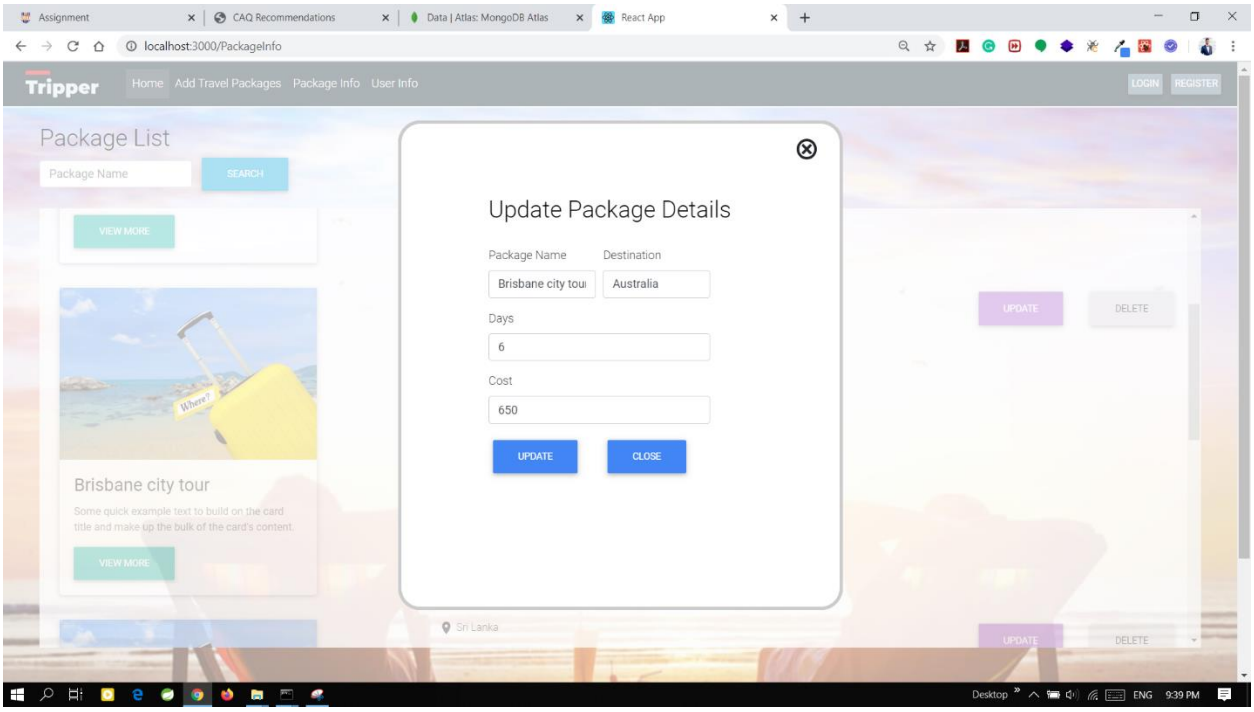

Figure 2.2

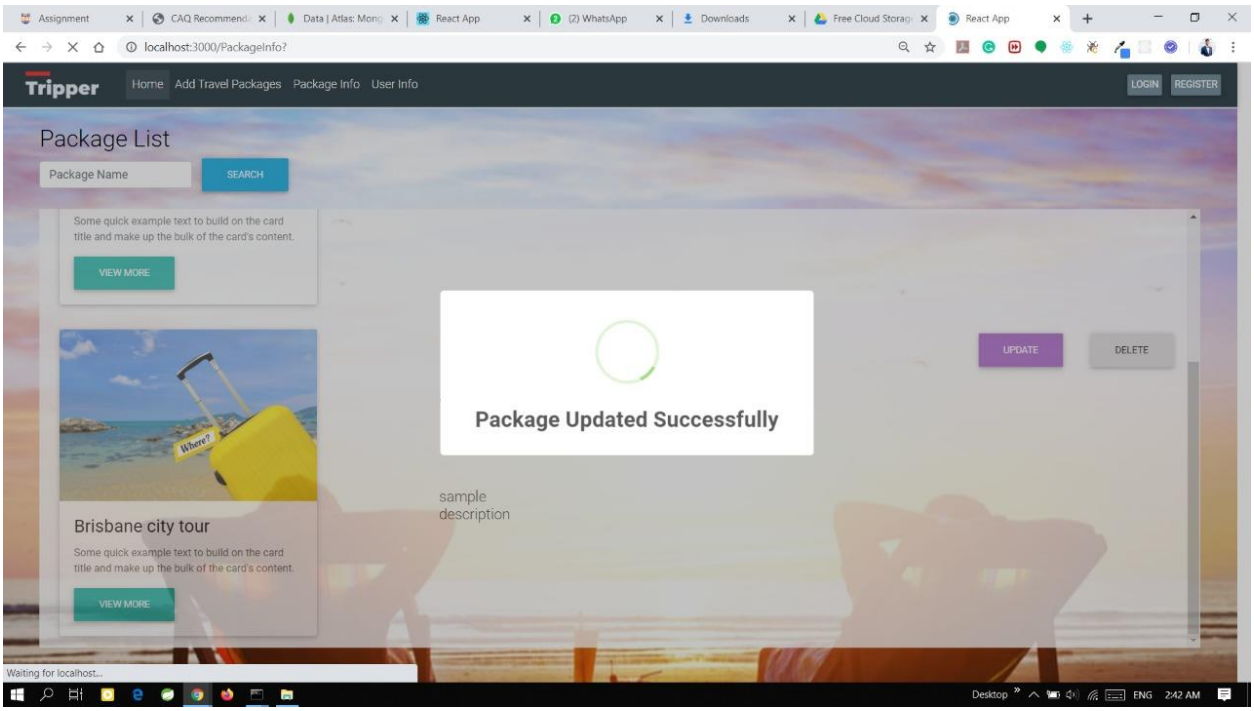

Figure 2.3

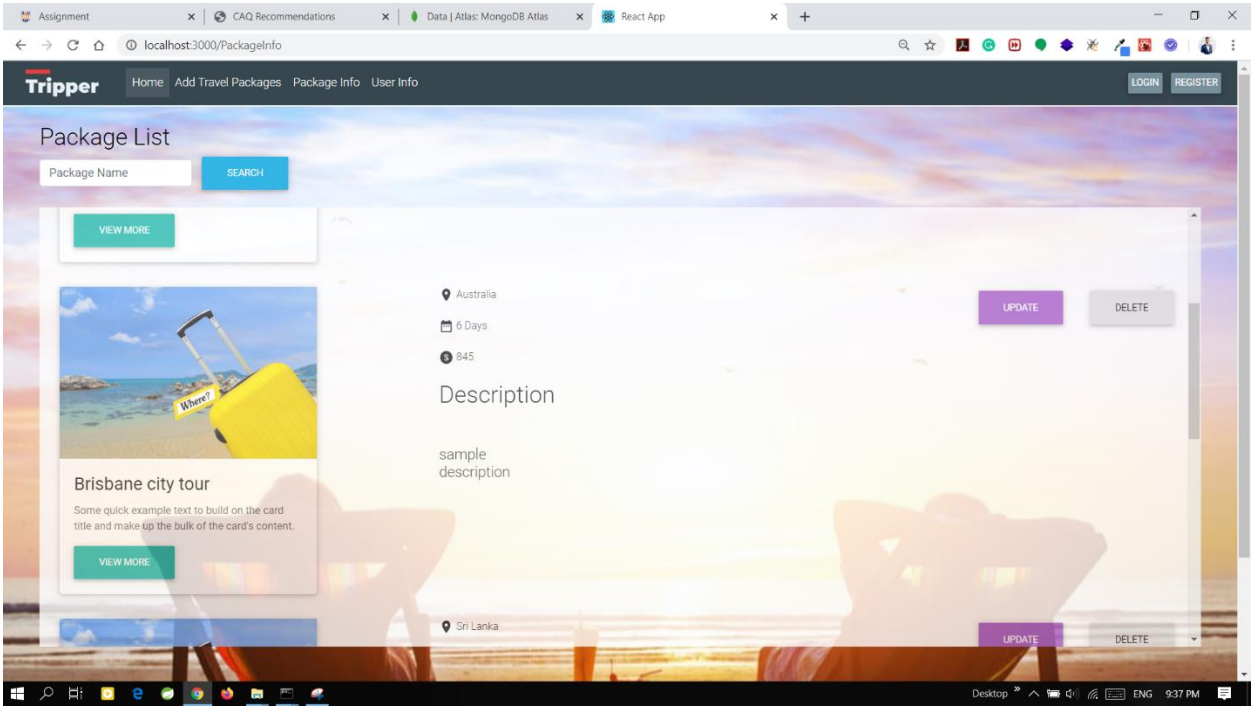

Figure 2.4

Admin can update and delete the previously added package details by clicking on the update and delete buttons as shown in figure 2.4. Figure 2.2 illustrates the update interface for the travel

packages. If the update operation happened successfully the successful message will be popped up on the screen as shown in figure 2.3. It will proceed in the same manner for the delete method. (figure 2.1)

Backend

Routes – AdminRoutes.js

```
/*
    method : PUT
    description : UPDATE travelpackage
    params : Body {
        package name,destination,noofdays,cost
    }
*/

router.put('/updatePackage/:id', (req, res) => {
    var id = req.params.id;
    Package.findOne({ _id: id }).then(pkg => {
        if (Package) {
            console.log("found package")

            pkg.packageName = req.body.packageName,
                pkg.destination = req.body.destination,
                pkg.days = req.body.days,
                pkg.cost = req.body.cost

            pkg.save()
            res.status(200).send({
                message: 'Package updated successfully !',
                messageCode: '1000',
                data: pkg
            })


        }
    })

})

/*
method : DELETE
    description : delete package by id
*/
router.delete('/deletePackage/:id', (req, res) => {
```

```
    Package.findById(req.params.id).then(pkg => {
        if (pkg) {
            pkg.remove();
            res.send(
                {
                    message: pkg.packageName + ' package was deleted successfully
',
                    data: pkg
                }
            )
        }
        else {
            res.send({
                message: "No such package"
            })
        }

    }).catch(err => {
        res.send(err)
    })
})
```

Frontend

PackageInfo.js

```
import React, { Component } from 'react'
import axios from 'axios';
import { MdAccountCircle } from "react-icons/md";
import Modal from 'react-modal'
import Swal from 'sweetalert2'
import { wait } from '@testing-library/react';
import { MDBBtn, MDBCard, MDBCardBody, MDBCardImage, MDBCardTitle, MDBCardText, M
DBCol } from 'mdbreact';
import { MdLocationOn } from "react-icons/md";
import { MdDateRange } from "react-icons/md";
import { AiFillDollarCircle } from "react-icons/ai";
import { GiCancel } from "react-icons/gi";
```

```
function sleep(milliseconds) {
    const date = Date.now();
    let currentDate = null;
    do {
        currentDate = Date.now();
    } while (currentDate - date < milliseconds);
}
class PackageInfo extends Component {
    constructor(props) {
        super(props);
        this.state = {
            packDetails: [],

            searchKey: "",
            searchResult: "",


            currentPkgName: "",

            showModal: false,

            packageName: '',
            destination: '',
            days: '',
            cost: '',


            pkgRes: '',

            upPackageName: '',
            upDestination: '',
            upDays: '',
            upCost: '',

            currentPkgID: '',
            refresh: false



        };
    }
    componentDidMount() {
        this.loadData();
    }
```

```
loadData = () => {


    let baseURL = "http://localhost:3800/api/admins/getAllPackages";
    axios({
        headers: {
            'Content-Type': 'application/json;charset=UTF-8',
        },


        method: 'GET',
        url: baseURL,


    })
        .then(response => {
            this.setState({
                packDetails: response.data
            })
            console.log("success")
            console.log(this.state.packDetails)
            var len = this.state.packDetails.length;

            console.log(len + "this is the lenth")

        })
    //setting the auto refresh time as every 40 secs
    //.then(setInterval(this.loadData, 40000))


}


deletePackage = (_id) => {
    console.log("arrived to delete")
    console.log(_id + " this is id")

    let baseURL = "http://localhost:3800/api/admins/deletePackage/" + _id;
    console.log("2nd ok")

    axios({


        method: 'DELETE',
        url: baseURL,
```

```
        })

            .then(response => {
                console.log(response.status)
                if (response.status === 200) {
                    Swal.fire({
                        position: 'center',
                        icon: 'success',
                        title: 'Package deleted successfully',
                        showConfirmButton: false,
                        timer: 3000
                    })



                }



            }).then(this.doFrefresh)

        //    window.location.reload(false)

        // .then(setTimeout(() => { console.log("World!"); }, 5000))
        //  .then(window.location.reload(true), 4000)
        //setting the auto refresh time as every 40 secs
        //.then(setInterval(this.loadData, 40000))


    }
    doFrefresh() {
        sleep(2000);
        window.location.reload(false)
    }
    captureInput = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            searchKey: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.searchKey);
        });
```

```
        console.log("New state DIRECTLY after setState:", this.state.searchKey);
        console.log("New state DIRECTLY after setState:", this.state.searchKey);
    }

    searchPackage = (event) => {


        event.preventDefault();
        axios({
            headers: {
                'Content-Type': 'application/json;charset=UTF-8',
            },

            method: 'GET',
            url: 'http://localhost:3800/api/admins/getbyPackageName/' + this.stat
e.searchKey,
        })
            .then(response => {
                console.log("Arrived to send request")
                if (response.status === 200) {

                    this.setState({
                        searchResult: response.data
                    })
                    if (this.state.searchResult.data === null) {
                        Swal.fire({
                            position: 'middle',
                            icon: 'error',
                            title: 'Oops...',
                            text: 'No results found!',
                            timer: 1500
                        })
                    }
                    else {
                        Swal.fire({
                            position: 'middle',
                            icon: 'success',
                            title: 'Travel Package Found',
                            showConfirmButton: false,
                            timer: 1500
                        })


                        console.log(this.state.searchResult.data.packageName)
```

```
                        this.setState({
                            currentPkgName: this.state.searchResult.data.packageN
ame
                        })
                        console.log("this is package name  =Found=" + this.state.
currentPkgName)
                    }

                }

            })
            .catch((console.log("ISSUES !")))

    }
    updatePackageName = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            packageName: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.packageName);
        });

        console.log("New state DIRECTLY after setState:", this.state.packageName)
;

    }

    updateDestination = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            destination: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.destination);
        });

        console.log("New state DIRECTLY after setState:", this.state.destination)
;
        console.log("New state DIRECTLY after setState:", this.state.destination)
;
```

```
    }
    updatedays = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            days: event.target.value
        }, () => {
            console.log("New state in ASYNC callback:", this.state.days);
        });

        console.log("New state DIRECTLY after setState:", this.state.days);
        console.log("New state DIRECTLY after setState:", this.state.days);
    }
    updatecost = event => {
        event.preventDefault()
        console.log(event.target.value)
        var val = event.target.value
        console.log(val + "this is the value retrived")
        this.setState({
            cost: parseInt(event.target.value)
        }, () => {
            console.log("New state in ASYNC callback:", this.state.cost);
        });

        console.log("New state DIRECTLY after setState:", this.state.cost);
        console.log("New state DIRECTLY after setState:", this.state.cost);
    }

    go = (packageName, destination, days, cost) => {
        this.setState({
            upPackageName: packageName,
            upDestination: destination,
            upDays: days,
            upCost: cost,

        })
        console.log(this.state.upPackageName + " this is the name to show first")

    }

    updatePakagedeatils = (_id) => {

        //event.preventDefault();
```

```
        // alert("successfully added")
        if (this.state.packageName === "" ||
            this.state.destination === "" ||
            this.state.days === "" ||
            this.state.cost === "") {
            Swal.fire({
                position: 'middle',
                icon: 'error',
                title: 'Oops...',
                text: 'Check your Inputs!',
                timer: 1500
            })
        }
        else {
            let pkgbody = JSON.stringify(
                {
                    "packageName": this.state.packageName,
                    "destination": this.state.destination,
                    "days": this.state.days,
                    "cost": this.state.cost

                }
            );
            axios({
                headers: {
                    'Content-Type': 'application/json;charset=UTF-8',
                },

                method: 'PUT',
                url: 'http://localhost:3800/api/admins/updatePackage/' + _id,
                data: pkgbody,

            })
                .then(response => {
                    console.log("Arrived to send request")
                    this.setState({
                        pkgRes: response.data
                    })
                })
                .then(() => {
                    Swal.fire({
                        position: 'middle',
                        icon: 'success',
                        title: 'Package Updated Successfully',
                        showConfirmButton: false,
```

```
                        timer: 3500
                    })
                    this.setState({
                        showModal: false
                    })
                })

                .catch((console.log("ISSUES !")))


        }


    }


    render() {

        return (
            <div style={{ backgroundImage: 'url("https://i.inews.co.uk/content/uploads/2020/05/holiday.jpg")', backgroundSize: "cover", position: "relative", height: "800px" }}>
                <div style={{ marginLeft: 50, marginRight: 50, marginBottom: 200, }}>
                    <br />
                    <h2 style={{ color: "black", justifyContent: 'center', alignItems: 'center' }}>Package List </h2>

                    <form className="form-inline my-2 my-lg-0">
                        <input className="form-control mr-sm-2" type="search" placeholder="Package Name" aria-label="Search" onChange={this.captureInput} required />
                        <button className="btn btn-info my-2 my-sm-0" type="submit" onClick={this.searchPackage}>Search</button>
                    </form>
                    <br></br>

                    <table className="table table-light" style={{ height: 600, width: "100%", overflow: "auto", display: "block", opacity: 0.8 }}>


                        <thead>

                        </thead>
                        <tbody style={{ opacity: 1 }}>
```

```
                            {this.state.packDetails.map(pkg =>

                                <tr style={{ backgroundColor: this.state.currentP
kgName === pkg.packageName ? '#22aeb5' : '' }} >

                                    {/* MODA for update package------------------
------------------------------------------------------------------------------
---------------------------- */}
                                    <Modal isOpen={this.state.showModal}
                                        style={{
                                            overlay: {
                                                position: 'fixed',
                                                top: 0,
                                                left: 0,
                                                right: 0,
                                                bottom: 0,
                                                backgroundColor: 'rgba(255, 255,
255, 0.2)',


                                            },
                                            content: {
                                                position: 'absolute',
                                                top: '80px',
                                                left: '540px',
                                                right: '540px',
                                                bottom: '100px',
                                                border: '4px solid #ccc',
                                                background: '#fff',
                                                overflow: 'auto',
                                                // WebkitOverflowScrolling: 'touc
h',

                                                borderRadius: '30px',
                                                outline: '5px',
                                                padding: '20px'
                                            }
                                        }}
                                    >
                                        <GiCancel style={{ marginLeft: 520 }} siz
e={30} onClick={() => this.setState({ showModal: false })} />
                                        <form style={{ marginBottom: 100, marginL
eft: 100 }}>

                                            <br />
                                            <br />
```

```
                                        <h2 style={{ color: 'black' }}>Update
 Package Details</h2>
                                        <br />
                                        <div className="form-row">
                                            <div className="col-md-4 mb-3">
                                                <label htmlFor="validationDef
ault01">Package Name</label>
                                                <input type="text" className=
"form-
control" id="validationDefault01" placeholder={this.state.upPackageName} required
                                                    onChange={this.updatePack
ageName} />
                                            </div>
                                            <div className="col-md-4 mb-3">
                                                <label htmlFor="validationDef
ault02">Destination</label>
                                                <input type="text" className=
"form-
control" id="validationDefault02" placeholder={this.state.upDestination} required
                                                    onChange={this.updateDest
ination} />
                                            </div>
                                        </div>
                                        {/* <div className="form-row">
                                            <div className="form-group col-
md-8">
                                                <label htmlFor="validationDef
ault04">Days</label>
                                                <input type="email" className
="form-control" id="validationDefault01" placeholder={this.state.upDays}
                                                    onChange={this.updatedays
} />
                                            </div>
                                        </div> */}
                                        <div className="form-row">
                                            <div className="form-group col-
md-8">
                                                <label htmlFor="validationDef
ault02">Days</label>
                                                <input type="text" className=
"form-control" id="validationDefault02" placeholder={this.state.upDays} required
                                                    onChange={this.updatedays
} />
                                            </div>
                                        </div>
```

```
                                        <div className="form-row">
                                            <div className="form-group col-
md-8">
                                                <label htmlFor="validationDef
ault04">Cost</label>
                                                <input type="text" className=
"form-control" id="validationDefault04" placeholder={this.state.upCost}
                                                    onChange={this.updatecost
} />
                                            </div>

                                        </div>
                                        <div className="form-row">
                                            <div className="col-md-4 mb-3">
                                                <button onClick={() => this.u
pdatePakagedeatils(this.state.currentPkgID)} className="btn btn-
primary" type="submit" >Update</button>
                                            </div>
                                            <div className="col-md-4 mb-3">
                                                <button onClick={() => this.s
etState({ showModal: false })} className="btn btn-primary" >Close</button>
                                            </div>
                                        </div>

                                    </form>
                                </Modal>
                                {/* MODAL-------------------------------------
-------------------------------------------------------------------------
---------- */}
                                <td style={{}}><MDBCol>
                                    <MDBCard style={{ width: "22rem" }}>
                                        <MDBCardImage className="img-
fluid" src="https://marketingland.com/wp-content/ml-loads/2016/05/travel-search-
suitcasebeach-ss-1920.png" waves />
                                        <MDBCardBody>
                                            <MDBCardTitle>{pkg.packageName}</
MDBCardTitle>
                                            <MDBCardText>
                                                Some quick example text to bu
ild on the card title and make
                                                up the bulk of the card&amp;apos;
s content.
                    </MDBCardText>
```

```
                                                <MDBBtn href="https://www.google.
com/">View more</MDBBtn>
                                            </MDBCardBody>
                                        </MDBCard>
                                    </MDBCol>
                                    </td>
                                    <td style={{ paddingLeft: 140, marginTop: 10
}}> <MdLocationOn size={20} /> {pkg.destination} <br /><br /><MdDateRange size={2
0} /> {pkg.days} Days <br /><br /><AiFillDollarCircle size={20} />  {pkg.cost}
                                        <br /><br /> <h2>Description</h2> <br /><
br /> <h5>sample description</h5> </td>
                                    <td style={{ paddingLeft: 140 }}></td>
                                    <td style={{ paddingLeft: 140 }}></td>
                                    <td style={{ paddingLeft: 140 }}></td>
                                    <td style={{ paddingLeft: 100 }}> <button typ
e="button" className="btn btn-secondary" onClick={() => {
                                        this.go(pkg.packageName, pkg.destination,
 pkg.days, pkg.cost)
                                        this.setState({
                                            currentPkgID: pkg._id
                                        })

                                        this.setState({ showModal: true });
                                    }}>Update</button></td>

                                    <td><button type="button" onClick={() => this
.deletePackage(pkg._id)} className="btn btn-light">Delete</button></td>
                                    <hr />

                                </tr>

                            )}


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

export default PackageInfo;
```

Other functionalities

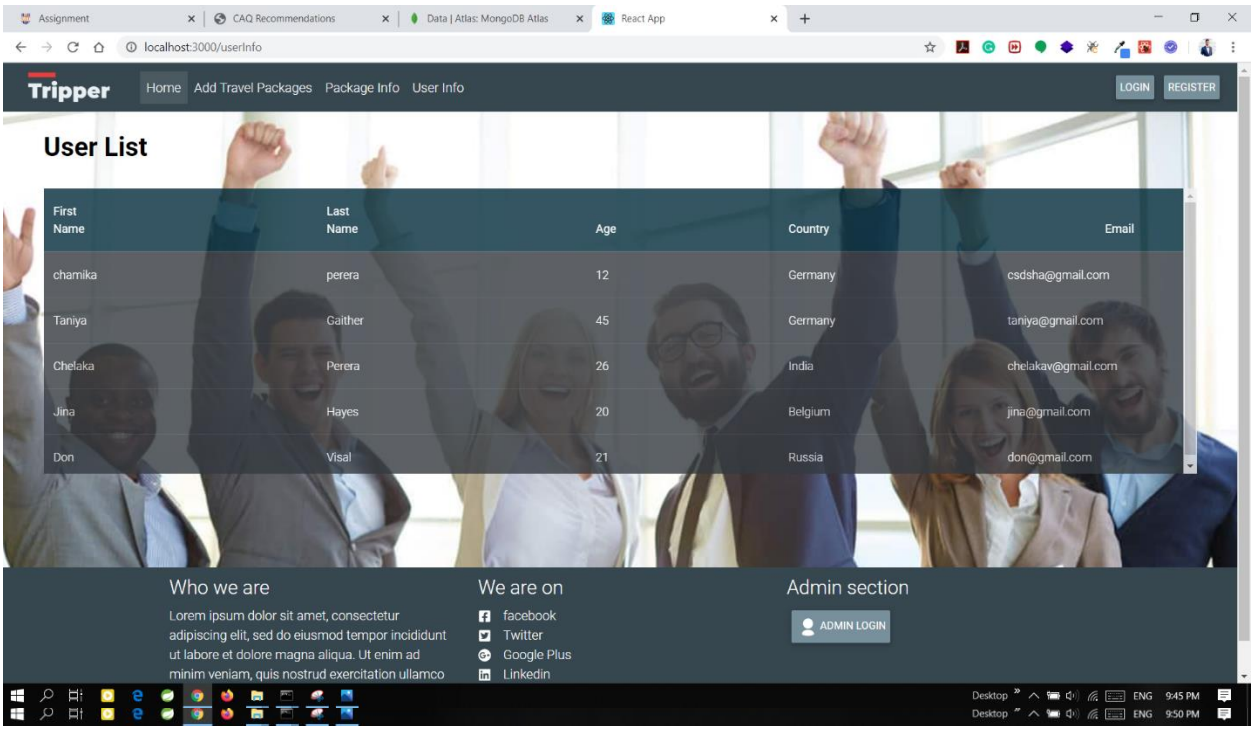

Figure 2.5

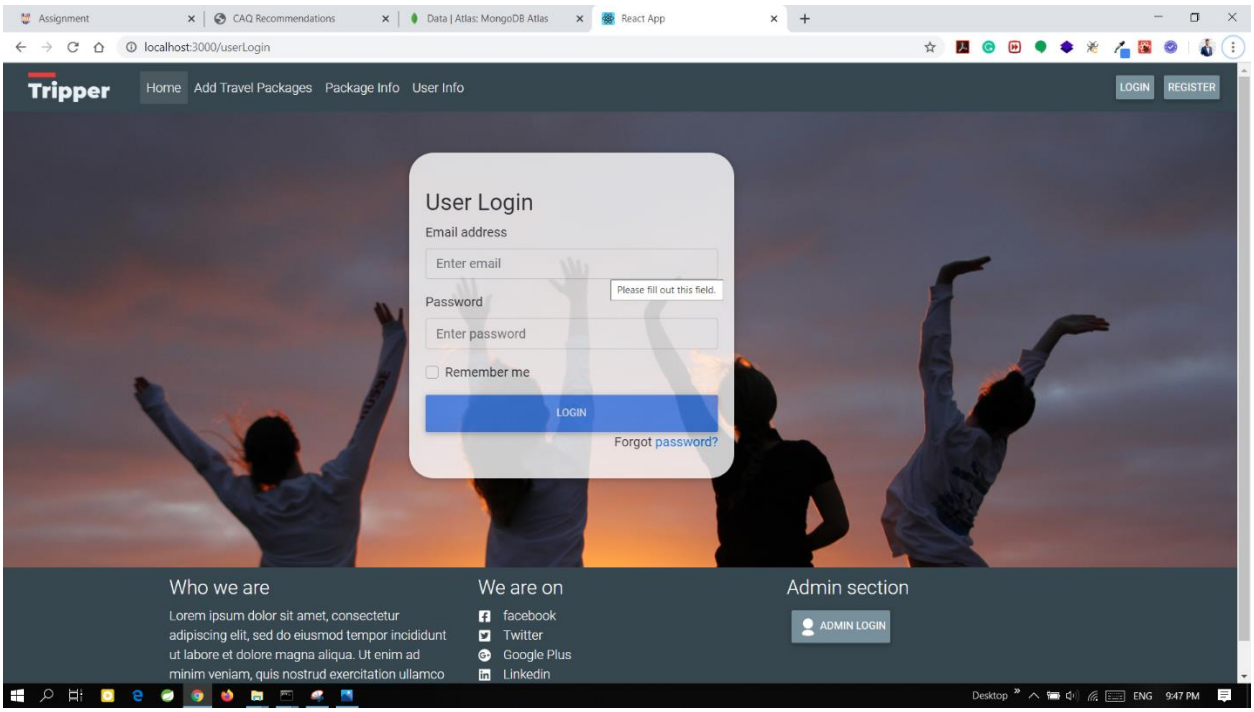

Figure 2.6

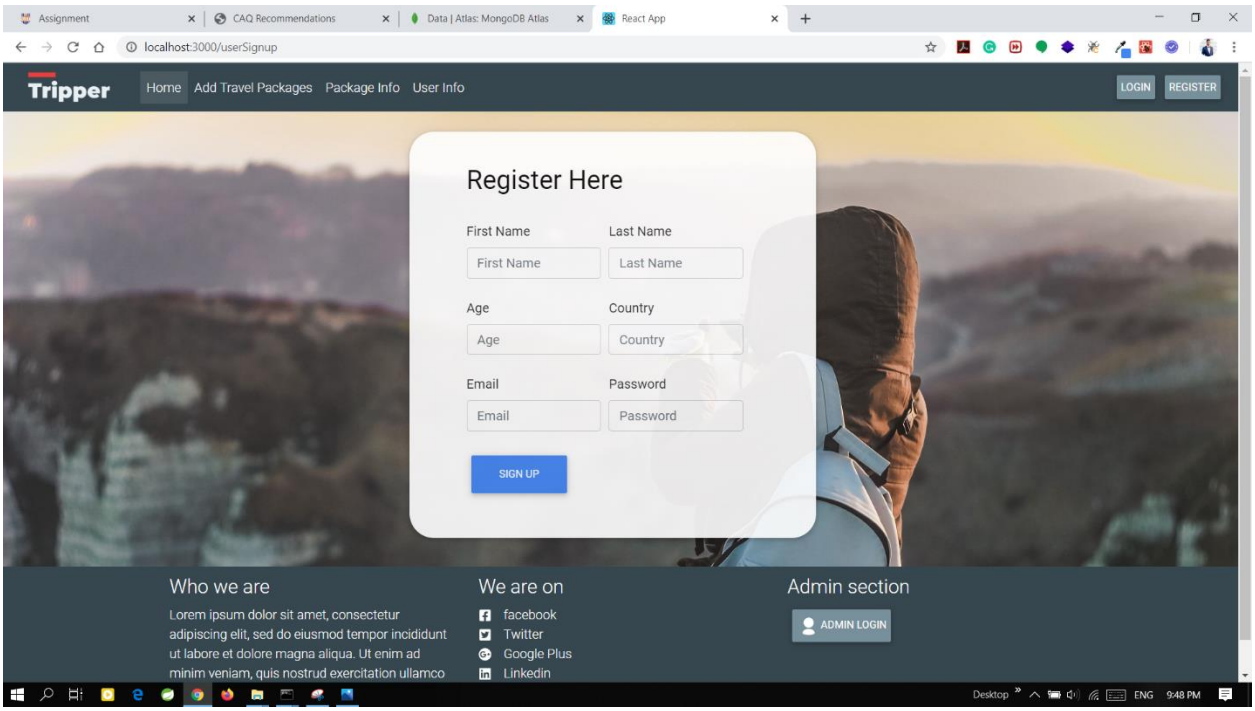

Figure 2.7

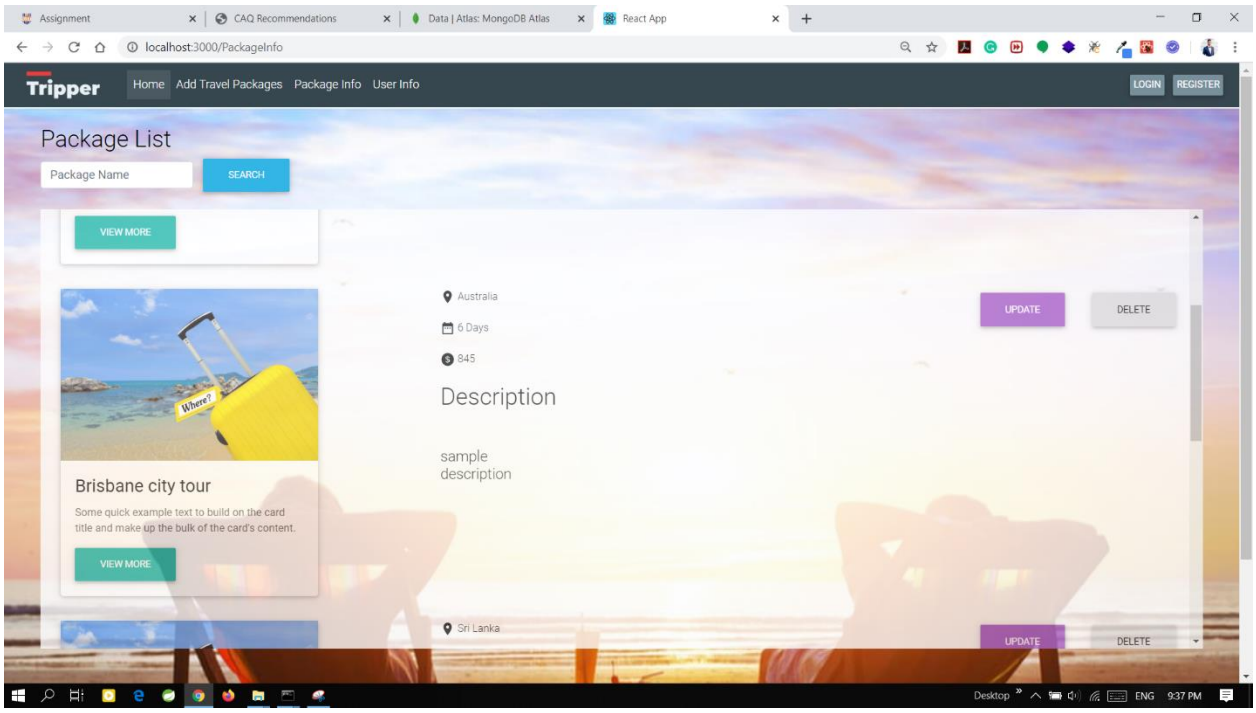

Figure 2.8

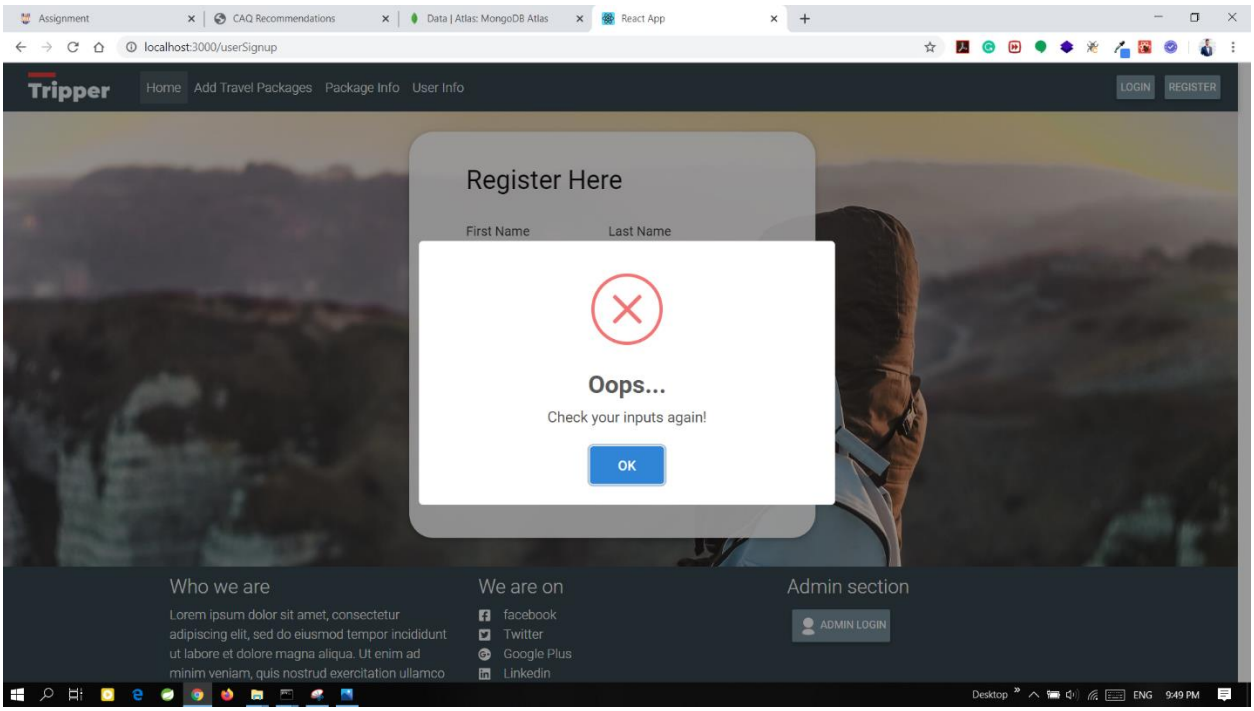

Figure 2.9

The user can view package details which have added by the admin (figure 1.0) and search travel packages by entering the package name as a search key in the search bar. All functionalities can be done by only the registered users. Therefore, the user should have to register to the system first (figure 2.7), then the user can log in to the system (figure 2.8) and do the functionalities as mentioned above.

The admin can directly log in to the system using his login credentials and can add travel packages to the system, view travel package details, update and delete package details and view the registered users' details (figure 2.5).

Except for figure 2.9, all the screenshots that have attached in this report are for the successful scenarios. If some worst scenario is triggered, the error pop-up message will be displayed on the screen as shown in figure 2.9.

Question 03 – RESTful web services

Routes – AdminRoutes.js

```
/*
    method : POST
    description : create new admin
    params : Body {
        username,password,
    }
*/
router.post('/create', (req, res) => {

    const admin = new Admins({
        username: req.body.username,
        password: req.body.password,

    })
    admin.save().then(adm => {
        console.log("admin Added")
        try {
            res.status(200).send({
                message: 'admin created successfully !',
                data: adm
            })

        } catch (err) {
            res.status(502).send({
                message: 'OOPS ! server error',
                error: err
            })
        }

    })
})

/*
    method : POST
    description : admin login
    params : Body {
        email,password
    }
*/
```

```
router.post('/adminLogin', (req, res) => {
    Admins.findOne({
        username: req.body.username,
        password: req.body.password
    }).then(adm => {
        if (adm) {
            console.log("found")
            res.send({
                message: 'successfully logged in ',
                data: adm,
                messageCode: "1000"
            })
        }
        else {
            console.log(" not found")
            res.send({
                message: 'Invalid admin credentials',
                data: adm,
                messageCode: "1001"
            })
        }

    })
})

/*
    method : POST
    description : create new travelpackage
    params : Body {
        package name,destination,noofdays,cost
    }
*/
router.post('/createPackage', (req, res) => {

    const pack = new Package({
        packageName: req.body.packageName,
        destination: req.body.destination,
        days: req.body.days,
        cost: req.body.cost

    })
    pack.save().then(pkg => {
        console.log("package Added")
        try {
            res.status(200).send({
```

```
                message: 'package created successfully !',
                data: pkg,
                messageCode: "1000"
            })

        } catch (err) {
            res.status(502).send({
                message: 'OOPS ! server error',
                error: err
            })
        }

    })
})

/*
    method : GET
    description : get all package details

*/
router.get('/getAllPackages', (req, res) => {
    const pkg = Package.find().then(pkgs => {
        res.send(pkgs);
    })
    console.log(pkg)
})



/*
    method : PUT
    description : UPDATE travelpackage
    params : Body {
        package name,destination,noofdays,cost
    }
*/

router.put('/updatePackage/:id', (req, res) => {
    var id = req.params.id;
    Package.findOne({ _id: id }).then(pkg => {
        if (Package) {
            console.log("found package")

            pkg.packageName = req.body.packageName,
                pkg.destination = req.body.destination,
```

```
                pkg.days = req.body.days,
                pkg.cost = req.body.cost

            pkg.save()
            res.status(200).send({
                message: 'Package updated successfully !',
                messageCode: '1000',
                data: pkg
            })


        }
    })

})

/*
method : DELETE
    description : delete package by id
*/
router.delete('/deletePackage/:id', (req, res) => {

    Package.findById(req.params.id).then(pkg => {
        if (pkg) {
            pkg.remove();
            res.send(
                {
                    message: pkg.packageName + ' package was deleted successfully
',
                    data: pkg
                }
            )
        }
        else {
            res.send({
                message: "No such package"
            })
        }

    }).catch(err => {
        res.send(err)
    })
})
```

```
/*
method : GET
    description : search package by packagenName
*/
router.get('/getbyPackageName/:Name', (req, res) => {
    Package.findOne({
        packageName: req.params.Name
    }).then(pkg => {


        try {
            res.status(200).send({
                message: 'Employee retrived successfully ok !',
                data: pkg
            })

        } catch (err) {
            res.status(502).send({
                message: 'OOPS ! server error',
                error: err
            })
        }

    })
})
```

Users.js

```
/*
    method : POST
    description : create new user
    params : Body {
        firstName,lastName,age,country,email,password
    }
*/
router.post('/create', (req, res) => {

    const user = new Users({
        firstName: req.body.firstName,
        lastName: req.body.lastName,
        age: req.body.age,
        country: req.body.country,
        email: req.body.email,
        password: req.body.password
```

```
    })
    user.save().then(userr => {
        console.log("user Added")
        try {
            res.status(200).send({
                message: 'user created successfully !',
                data: userr
            })

        } catch (err) {
            res.status(502).send({
                message: 'OOPS ! server error',
                error: err
            })
        }

    })
})

/*
    method : POST
    description : user login
    params : Body {
        email,password
    }
*/
router.post('/userLogin', (req, res) => {
    Users.findOne({
        email: req.body.email,
        password: req.body.password
    }).then(user => {
        if (user) {
            console.log("found")
            res.send({
                message: 'successfully logged in ',
                data: user,
                messageCode: "1000"
            })
        }
        else {
            console.log(" not found")
            res.send({
                message: 'Invalid user credentials',
                data: user,
                messageCode: "1001"
```

```
            })
        }

    })
})

/*
    method : GET
    description : get all user details

*/
router.get('/getAllUserInfo', (req, res) => {
    const user = Users.find().then(user => {
        res.send(user);
    })
    console.log(user)
})
```

Question 04 – Mongo query

```
/*
    method : POST
    description : user login
    params : Body {
        email,password
    }
*/
router.post('/userLogin', (req, res) => {
    Users.findOne({
        email: req.body.email,
        password: req.body.password

    }).then(user => {
        if (user) {
            console.log("found")
            res.send({
                message: 'successfully logged in ',
                data: user,
                messageCode: "1000"
            })
        }
        else {
            console.log(" not found")
            res.send({
                message: 'Invalid user credentials',
                data: user,
                messageCode: "1001"
            })
        }

    })
})
```

findOne method is one of the methods to query in MongoDB

Question 06 – Home Page