Sri Lanka Institute of Information Technology

Rest API – Version 03

Project Report

Distributed System Module

Assignment 02

Project ID: **2022S1_REG_08**

Submitted by:

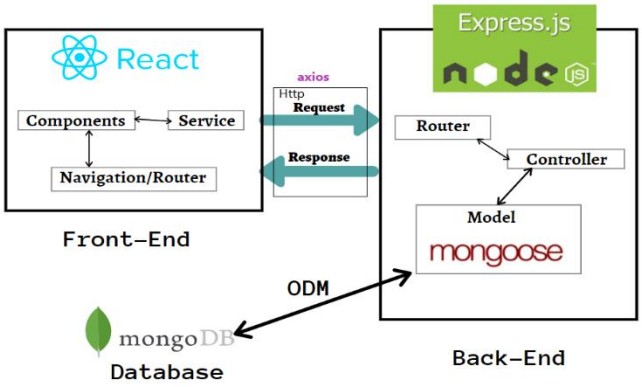
1. **IT20065294 - Prasadi W.V.C**
2. **IT20095062 - Gamage S. R. J**
3. **IT20075408 - Nandasiri K.G.N.D**
4. **IT20013950 - Lakshani N.V.M**

Contents

01. INTRODUTION

This is an Agri product online purchasing platform which is available for both buying and selling. We have mainly used Mern stack technology to develop this platform.

- Frontend - React
- Backend – Node.js, MongoDB
- Rest APIs – Exprss.js (To develop the backend, frontend and the database)



Mainly there are two types of actors as seller and buyer. And there are 3 services.

1. Seller service
2. Buyer service
3. Payment service (Dummy service)

1. Seller Service

First a seller needs to register to the website under the type of seller. After successfully register or login to a system the seller redirect to seller home page and it displays all the item that he/she previously added to the system. Under that page if seller wants to add new item to the site, then he/she should click the create product button and it displays create product form. After seller can add new product to the system by providing required details and that item details saved in the database. If seller is not provided the required details then the system will display error message. And also, by providing required details, the seller can update and delete product details that he/she previously added. And the seller can create, edit and delete product categories as well. Buyers can filter the products according to that categories.

2. Buyer Service

When buyer is logged into the system, that person will not be able to do any modification to the product details. Add Products, create categories button in header will not be visible for them. They can only go to the shop, cart, payment process and order history. All the products in database will be retrieved to the shop with its own details. They can search products by categories, newest items, oldest items, best sales, low price, high price. Then if they select product, it will be going to the cart. In the cart if buyer wants to remove product, he can do it there. And also, they can increase or decrease the product quantity as their wish. And under the cart buyer can see related the products also. Buyer can add any number of products to the cart which is displayed right below with a total bill for the selected items. Then they can select the payment Method. There will be two options for that

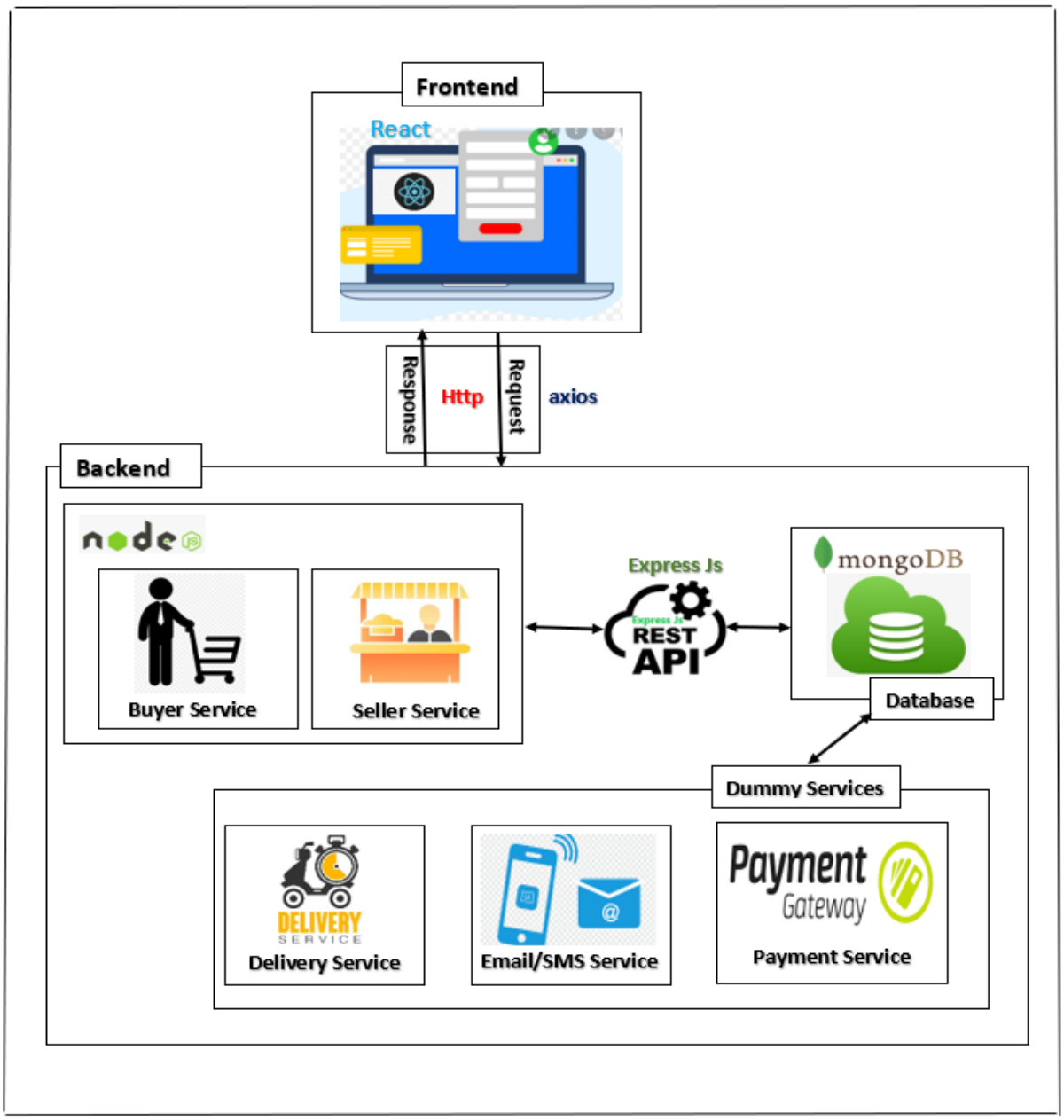
- Credit Card Payment

  When customer select credit card payment as the payment, customer should enter the card number, CVC code, the card holder's name, and the email address. If the details are valid system allow to make the payment. And email will be sent to the given email address. If the credentials not valid system not allow to make the payment.

- Mobile Payment

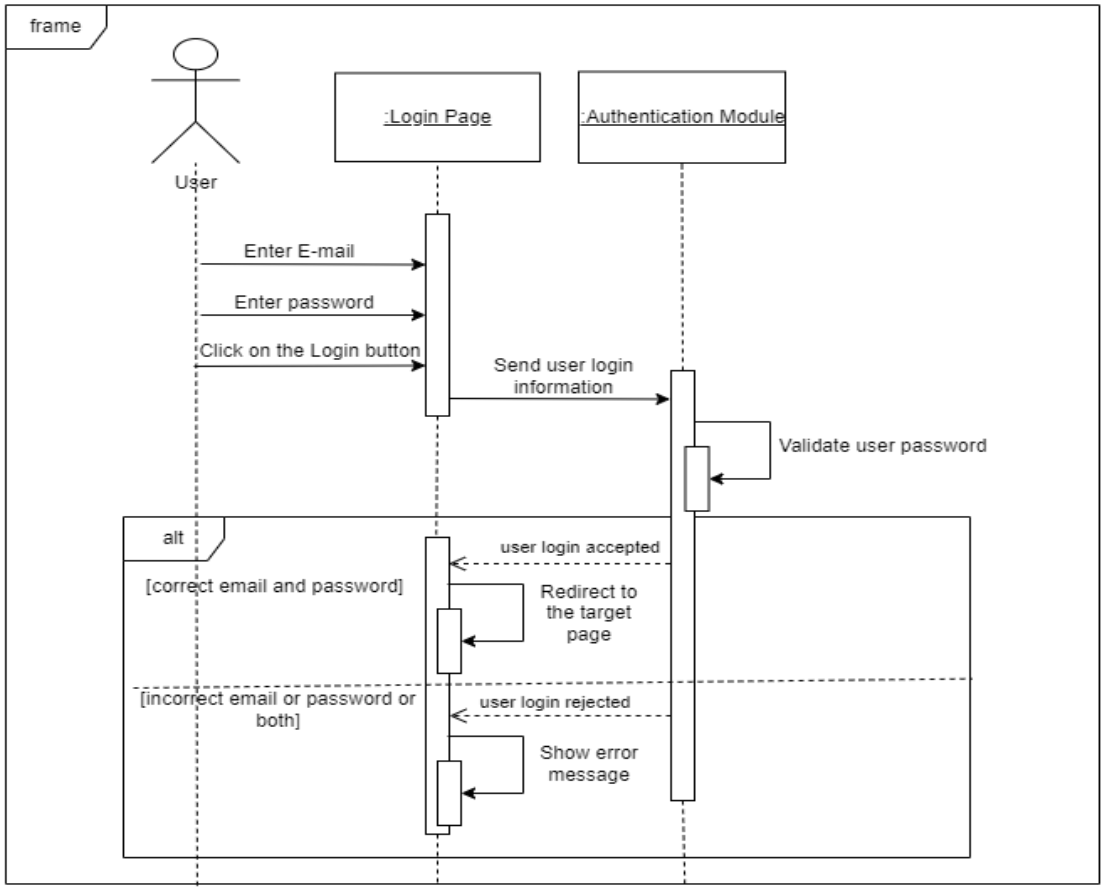
  When customer select Mobile payment as the payment, user needs to enter his phone number to request an OTP number with 6 digits. After requesting, the application will send a message to the provided mobile number with a 6-digit OTP (pin). After receiving the OTP number user can insert OTP to validate it. After validating the OTP, amount, and billing mobile number must provide to the "Payment" form. After that, a message will be sent to the given mobile number as payment successfully.

02. High-level Architectural Diagram



03. Workflow of the System

1. Login Service




```
// Password Encryption
            const passwordHash = await bcrypt.hash(password, 10)
            const newUser = new Users({
                name, email, password: passwordHash
            })
```

After the user register with his password. It is going to the database as encrypted password. For that we use a special node module called 'bcrypt' to make it more secure than other data. Password is shows as below after the encrypted.

[{_id:6282998ea901a70e24a4e57f}, {role:0}, {cart:Array}, {name:"Chethana"} {email:c@gmail.com}, {password:"$2b$10$pTsrO..VBbN3y5VWaW/pqOxjHjZrzCv4xC0s1aDqsSaInf9hd8aMK"}]

When the user is trying to logging to the system, the REST API compares the entered password with the data stored in the database. Then entered password compared with encrypted password.

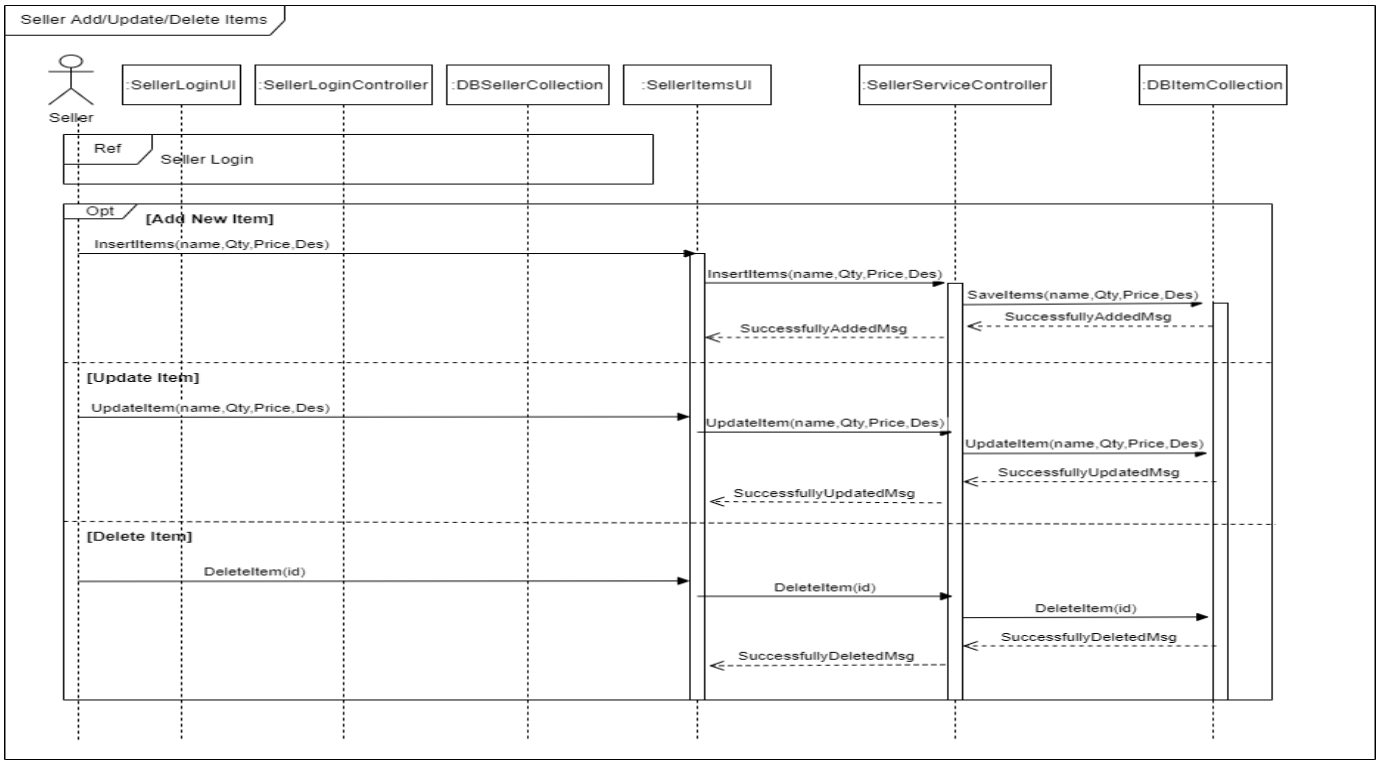
```
//Create jsonwebtoken to Authentication
            const accesstoken = createAccessToken({id: newUser._id})
            const refreshtoken = createRefreshToken({id: newUser._id})
```

In simple words authentication is the process of verifying that who the user is. And after the authentication there is a process of verifying user for what they have access to in our application. That's called Authorization.

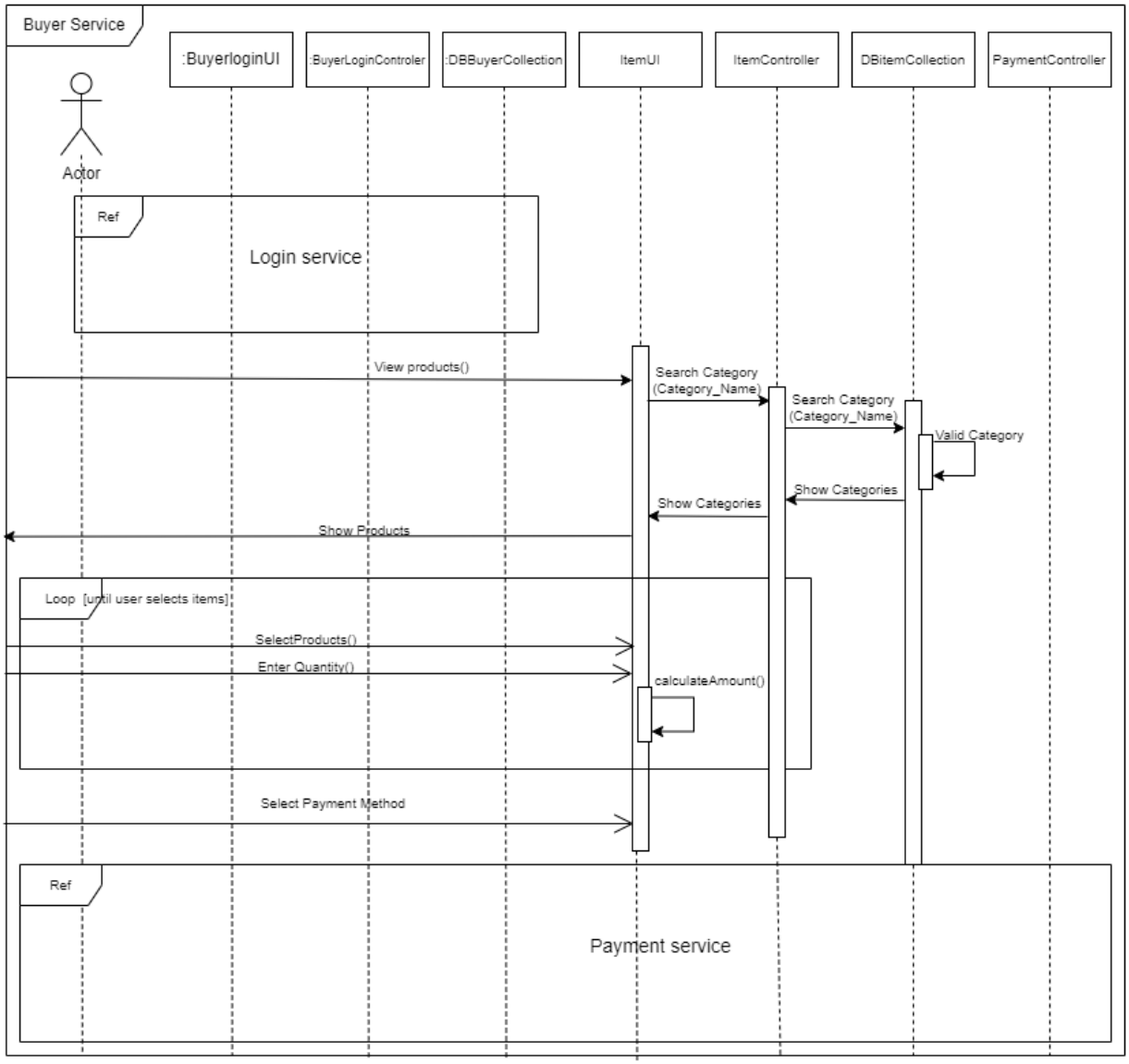
```
const authAdmin = async (req, res, next) => {
    try {
        // Get user information by id
        const user = await Users.findOne ({
            _id: req.user.id
        })

        if(user.role === 0 || user.role === 2)
            return res.status(400).json({msg: "Admin Resources Access Denied"})

        next();

    } catch (err) {
        return res.status(500).json({msg: err.message})
    }
}
```

In here user can't access the admin part. We assume user's role is 0 and Admin's user role to 1. Only if user role = 1, user can access the admin part.

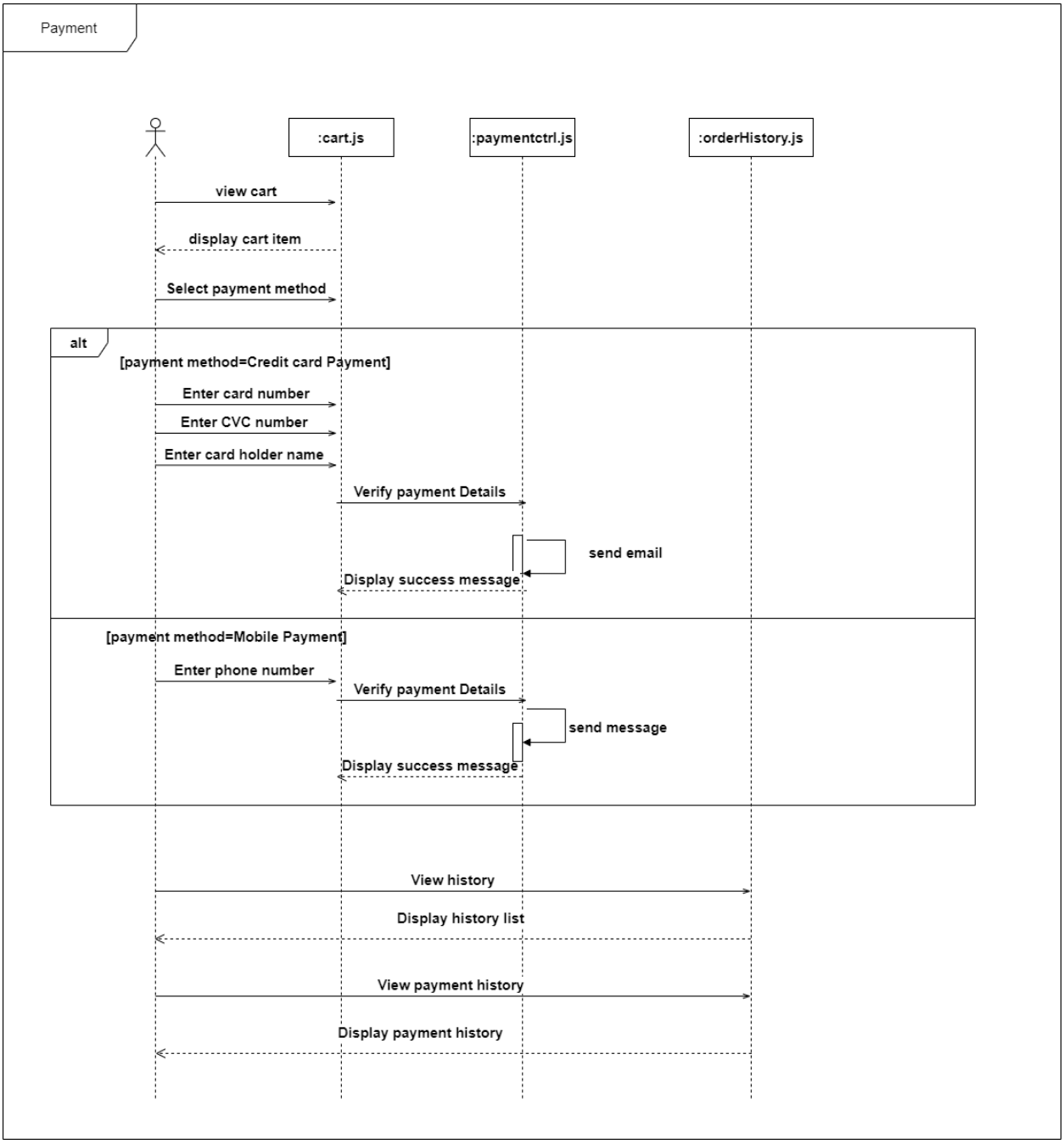
2. Seller service



3. Buyer Service



4. Payment Service



04. Appendix

01. Backend

1. server.js

```
//Import All Dependencies
require('dotenv').config()
const express = require('express')
const mongoose = require('mongoose')
const cors = require('cors')
const fileUpload = require('express-fileupload')
const cookieParser = require('cookie-parser')

const app = express()
app.use(express.json())
app.use(cookieParser())
app.use(cors())
app.use(fileUpload({
    useTempFiles: true
}))

//Routes
app.use('/user', require('./routes/userRoute'));
app.use('/api', require('./routes/categoryRoute'));
app.use('/api', require('./routes/upload'));
app.use('/api', require('./routes/productRoutes'));
app.use('/api', require('./routes/paymentRoute'));
app.use('/api', require('./routes/MsgRoute'));
app.use('/api', require('./routes/mobilepay'));
```

```
//Connect to MongoDB
const URI = process.env.MONGO_URI
mongoose.connect(URI, {
    useCreateIndex: true,
    useFindAndModify: false,
    useNewUrlParser: true,
    useUnifiedTopology: true
}, err =>{
    if(err) throw err;
    console.log('Connected to MongoDB')
});

//Connect to the Server
const PORT = process.env.PORT || 5100
app.listen(PORT, () =>{
    console.log('Server is running on port', PORT)
});
```

2. Models
    I. userModel.js

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

//Create Userschema
const userSchema = new mongoose.Schema({
    name: {
        type: String,
        required: true,
        trim: true
    },

    email: {
        type: String,
```

```
        required: true,
        unique: true
    },

    password: {
        type: String,
        required: true
    },

    role: {
        type: Number,
        default: 0
    },

    cart: {
        type: Array,
        default: []
    }

}, {
    timestamps: true
})

module.exports = mongoose.model('Users', userSchema)
```

II. productModel.js

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

//create productSchema
const productSchema = new mongoose.Schema ({
    product_id:{
        type: String,
```

```
        unique: true,
        trim: true,
        required: true
    },

    title:{
        type: String,
        trim: true,
        required: true
    },

    price:{
        type: Number,
        trim: true,
        required: true
    },

    description:{
        type: String,
        required: true
    },

    content:{
        type: String,
        required: true
    },

    images:{
        type: Object,
        required: true
    },
```

```
    category:{
        type: String,
        required: true
    },

    checked:{
        type: Boolean,
        default: false
    },

    sold:{
        type: Number,
        default: 0
    }
}, {
    timestamps: true //important
})


module.exports = mongoose.model("Products", productSchema)
```

III. categoryModel.js

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

//create categorySchema
const categorySchema = new mongoose.Schema ({
    name: {
        type: String,
        required: true,
        trim: true,
        unique: true
    }
```

```
}, {
    timestamps: true
})

module.exports = mongoose.model("Category", categorySchema)
```

IV. MobilePay.js

```
const mongoose = require("mongoose")

//create MobilePaySchema
var MPay = new mongoose.Schema(
    {
    phone: {type: String, required:true, default: 'default No', trim: true},
    amount: {type: String, required:true, default:'default 1200', trim: true},
    pin: {type: String, required:true, default:'default 123456', trim: true},
    }
)

module.exports = mongoose.model('mpays', MPay);
```

V. paymentModel.js

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

//create paymentSchema
const paymentSchema = new mongoose.Schema({
    user_id: {
        type: String,
        required: true
    },
    name:{
        type: String,
        required: true
```

```
    },
    email:{
        type: String,
        required: true
    },
    paymentID:{
        type: String,
        required: true
    },
    address:{
        type: Object,
        required: true
    },
    cart:{
        type: Array,
        default: []
    },
    status:{
        type: Boolean,
        default: false
    }
}, {
    timestamps: true
})



module.exports = mongoose.model("Payments", paymentSchema)
```

3. Controllers

I. userCtrl.js

```
const Users = require('../models/userModel');
```

```
const Payments = require('../models/paymentModel');
const bcrypt = require('bcrypt');
const jwt = require('jsonwebtoken');

const userCtrl = {

    //register the user
    register: async (req, res) => {
        try {
            const {name, email, password} = req.body;

            const user = await Users.findOne({email})
            if(user) return res.status(400).json({msg: "This E-mail is already
Used"})

            if(password.length < 6)
                return res.status(400).json({msg: "Password is at least 6 characters
long."})

            // Password Encryption
            const passwordHash = await bcrypt.hash(password, 10)
            const newUser = new Users({
                name, email, password: passwordHash
            })


            // Save mongodb
            await newUser.save();

            //Create jsonwebtoken to Authentication
            const accesstoken = createAccessToken({id: newUser._id})
            const refreshtoken = createRefreshToken({id: newUser._id})
```

```
            res.cookie('refreshtoken', refreshtoken, {
                httpOnly: true,
                path: '/user/refresh_token',
                maxAge: 7*24*60*60*1000 //7d
            })

            res.json({accesstoken})

            //res.json({msg: "Sucessfully Registered"})

        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    //login user
    login: async (req, res) => {
        try {
            const {email, password} = req.body;

            const user = await Users.findOne({email})
            if(!user) return res.status(400).json({msg: "User does not exist."})

            const isMatch = await bcrypt.compare(password, user.password)
            if(!isMatch) return res.status(400).json({msg: "Incorrect password."})

            // If login success , create access token and refresh token
            const accesstoken = createAccessToken({id: user._id})
            const refreshtoken = createRefreshToken({id: user._id})

            res.cookie('refreshtoken', refreshtoken, {
```

```
                httpOnly: true,
                path: '/user/refresh_token',
                maxAge: 7*24*60*60*1000 //7d
            })

            res.json({accesstoken})

        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    //logout user
    logout: async (req, res) => {
        try {
            res.clearCookie('refreshtoken', {path: '/user/refresh_token'})
            return res.json({msg: "Logged out"})

        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    refreshToken: (req, res) => {
        try {
            const rf_token = req.cookies.refreshtoken;
            if(!rf_token) return res.status(400).json({msg: "Please Login or
Register"})

            jwt.verify(rf_token, process.env.REFRESH_TOKEN_SECRET, (err, user) => {
                if(err) return res.status(400).json({msg: "Please Login or
Register"})
```

```
            const accesstoken = createAccessToken({id: user.id})

            res.json({accesstoken})
        })

    } catch (err) {
        return res.status(500).json({msg: err.message})
    }
},

//get user by ID
getUser: async (req, res) => {
    try {
        const user = await Users.findById(req.user.id).select('-password')
        if(!user) return res.status(400).json({msg: "User does not exist."})

        res.json(user);

    } catch (err) {
        return res.status(500).json({msg: err.message})
    }
},

//add items to the user's relavant cart
addCart: async (req, res) => {
    try {
        const user = await Users.findById(req.user.id)
        if(!user) return res.status(400).json({msg: "User does not exist."})

        await Users.findOneAndUpdate({_id: req.user.id}, {
            cart: req.body.cart
```

```
            })

            return res.json({msg: "Added to cart"})

        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    //payment history
    history: async(req, res) => {
        try {
            const history = await Payments.find({user_id: req.user.id});

            res.json(history);

        } catch (err) {
            return res.status(500).json({msg: err.message});
        }
    }
}

const createAccessToken = (user) => {
    return jwt.sign(user, process.env.ACCESS_TOKEN_SECRET, {expiresIn: '12m'})
}

const createRefreshToken = (user) =>{
    return jwt.sign(user, process.env.REFRESH_TOKEN_SECRET, {expiresIn: '7d'})
}

module.exports = userCtrl;
```

II. productCtrl.js

```
const Products = require('../models/productModel');


// Filter, sorting and paginating
class APIfeatures {
    constructor(query, queryString) {
        this.query = query;
        this.queryString = queryString;
    }

    filtering() {
       const queryObj = {...this.queryString} //queryString = req.query

       const excludedFields = ['page', 'sort', 'limit']
       excludedFields.forEach(el => delete(queryObj[el]))

       let queryStr = JSON.stringify(queryObj)
       queryStr = queryStr.replace(/\b(gte|gt|lt|lte|regex)\b/g, match => '$' +
match)

    //    gte = greater than or equal
    //    lte = lesser than or equal
    //    lt = lesser than
    //    gt = greater than

       this.query.find(JSON.parse(queryStr))

       return this;
    }

    sorting() {
        if(this.queryString.sort) {
```

```
            const sortBy = this.queryString.sort.split(',').join(' ')
            this.query = this.query.sort(sortBy)
        } else {
            this.query = this.query.sort('-createdAt')
        }

        return this;
    }

    paginating() {
        const page = this.queryString.page * 1 || 1
        const limit = this.queryString.limit * 1 || 9
        const skip = (page - 1) * limit;
        this.query = this.query.skip(skip).limit(limit)
        return this;
    }
}

const productCtrl = {

    //get products
    getProducts: async(req, res) =>{
        try {
            const features = new APIfeatures(Products.find(), req.query)
            .filtering().sorting().paginating()

            const products = await features.query

            res.json({
                status: 'success',
                result: products.length,
                products: products
```

```
            })

        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    //add product
    createProduct: async (req, res) => {
        try {
            const {product_id, title, price, description, content, images, category}
= req.body;
            if(!images) return res.status(400).json({msg: "No image upload"})

            const product = await Products.findOne({product_id})
            if(product)
                return res.status(400).json({msg: "This product already exists."})

            const newProduct = new Products ({
                product_id, title: title.toLowerCase(), price, description, content,
images, category
            });

            await newProduct.save()
            res.json({msg: "Created a product"})

        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    //update product
```

```
    updateProduct: async (req, res) => {
        try {
            const {title, price, description, content, images, category} = req.body;
            if(!images) return res.status(400).json({msg: "No image upload"})

            await Products.findOneAndUpdate({_id: req.params.id}, {
                title: title.toLowerCase(), price, description, content, images,
category
            })

            res.json({msg: "Updated a Product"})
        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    //delete product
    deleteProduct: async(req, res) =>{
        try {
            await Products.findByIdAndDelete(req.params.id)
            res.json({msg: "Deleted a Product"})
        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },
}

module.exports = productCtrl;
```

III. categoryCtrl.js

```
const Category = require('../models/categoryModel');
```

```
const categoryCtrl = {

    //get categories
    getCategories: async(req, res) =>{
        try {
            const categories = await Category.find()
            res.json(categories)
        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    //add categories
    createCategory: async (req, res) => {
        try {
            // if user have role = 1 ---> farmer
            // only farmer can create , delete and update category
            const {name} = req.body;
            const category = await Category.findOne({name})
            if(category) return res.status(400).json({msg: "This category already
exists."})

            const newCategory = new Category({name})

            await newCategory.save()
            res.json({msg: "Created a category"})

        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },
```

```
    //update categories
    updateCategory: async (req, res) => {
        try {
            const {name} = req.body;
            await Category.findOneAndUpdate({_id: req.params.id}, {name})

            res.json({msg: "Updated Category"})

        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },

    //delete category
    deleteCategory: async (req, res) => {
        try {
            await Category.findByIdAndDelete(req.params.id)
            res.json({msg: "Deleted a Category"})
        } catch (err) {
            return res.status(500).json({msg: err.message})
        }
    },
}

module.exports = categoryCtrl;
```

IV. paymentCtrl.js

```
const Payments = require('../models/paymentModel');
const Users = require('../models/userModel');
const Products = require('../models/productModel')
```

```
const paymentCtrl = {

    //get payments info
    getPayments: async (req, res) => {
        try {
            const payments = await Payments.find()
            res.json(payments)
        } catch (err) {
            return res.status(500).json({ msg: err.message })
        }
    },

    //add payment info
    createPayment: async (req, res) => {
        try {
            const user = await Users.findById(req.user.id).select('name email')
            if (!user) return res.status(400).json({ msg: "User does not exist." })

            const { cart, paymentID, address } = req.body;

            const { _id, name, email } = user;

            const newPayment = new Payments({
                user_id: _id, name, email, cart, paymentID, address
            })

            cart.filter(item => {
                return sold(item._id, item.quantity, item.sold)
            })

            await newPayment.save()
```

```
            res.json({ msg: "Payment Succesfull" })


        } catch (err) {
            return res.status(500).json({ msg: err.message })
        }
    }
}

//show sold items quantity
const sold = async (id, quantity, oldSold) => {
    await Products.findOneAndUpdate({ _id: id }, {
        sold: quantity + oldSold
    })
}

module.exports = paymentCtrl;
```

V. Mobilepay.js

```
const express = require("express");
const MobilePay = require("../models/MobilePay");
const Nexmo = require('nexmo')
const crypto = require ("crypto");

//init nexmo
const nexmo = new Nexmo({
    apiKey: API_KEY,
    apiSecret: API_SECERET
}, {debug: true});

const router = express.Router();

//add mobile payment details to the database=======================================
```

```
router.post('/', async (req, res) => {
    const {phone, amount, pin} = req.body;
    let pay = {
        "phone" : phone,
        "amount" : amount,
        "pin" : pin,
    };
    try{
        let newpayModel = new MobilePay(pay);
        await newpayModel.save();
        res.json({Message: 'Value inserted', Result: newpayModel});
    }catch (error) {
        res.status(404).send("Mobile payment not added");
    }
})

//Send OTP ======================================
router.post('/sendOTP', (req, res) => {
    const number = req.body.phone;
    const random = Math.floor(Math.random() * (9999 - 0001) + 1);
    nexmo.message.sendSms(
        'Vonage APIs', number, "OTP : "+ random, {type: 'unicode'},
        (err, responseData) => {
            if(err){
                console.log(err);
            }else{
                console.dir(responseData);
            }
        }
    );
})
```

```
//Send Details ======================================
router.post('/sendData', (req, res) => {
    const number = req.body.phone;
    const amount = req.body.amount;
    nexmo.message.sendSms(
        'Vonage APIs', number, "You have paid RS."+ amount +" Successfully.", {type:
'unicode'},
        (err, responseData) => {
            if(err){
                console.log(err);
            }else{
                console.dir(responseData);
            }
        }
    );
})


//Get all mobile payments============================================================
router.get('/getAll', async (req,res) => {
    try{
        const data = await MobilePay.find();
        res.json({Message : "All results fetched", Result: data})
    } catch (errror) {
        res.status(500).send("Cannot fetch all data");
    }
})

//Get a payment ====================================================================
router.get('/get', async (req,res) => {
    const{id} = req.body;
    try{
```

```
        const pay = await MobilePay.findById(id);
        res.json({Message: "Payment recieved", Result: pay});
    } catch (error) {
        res.status(500).send("Cannot get the Payment");
    }
})

//Delete payment ==================================================================
router.delete('/deleteMpay', async(req,res) => {
    const{id} = req.body;
    try{
        const pay = await MobilePay.findById(id);
        await pay.remove();
        res.json({Message: "Payment deleted", Result: pay});
    } catch (error) {
        res.status(500).send("Cannot delete Payment Details");
    }
})

//Update payment Details ==========================================================
router.put('/update/:id', async (req, res) => {
    const{id} = req.params;
    const {phone, amount, pin} = req.body;
    let data = {
        "phone" : phone,
        "amount" : amount,
        "pin" : pin,
    };
    try{
        const pay = await MobilePay.findByIdAndUpdate(id, data);
        res.json({Message: "Payment Updated Successfully..."});
    } catch (error) {
```

```
        res.status(500).send("Payment Not Updated")
    }
})



//=================================================================================
  module.exports = router;
//=================================================================================
```

4. Routes
   I. userRoute.js

```
const router = require('express').Router();
const userCtrl = require('../controllers/userCtrl');
const auth = require ('../middlewares/auth');

router.post('/register', userCtrl.register);
router.post('/login', userCtrl.login);
router.get('/logout', userCtrl.logout);
router.get('/refresh_token', userCtrl.refreshToken);
router.get('/infor', auth,  userCtrl.getUser);
router.patch('/addcart', auth, userCtrl.addCart);
router.get('/history', auth, userCtrl.history);

module.exports = router;
```

   II. productRoutes.js

```
const router = require('express').Router();
const productCtrl = require('../controllers/productCtrl');
const auth = require ('../middlewares/auth');
const authAdmin = require('../middlewares/authAdmin');

router.route('/products')
    .get(productCtrl.getProducts)
```

```
    .post( productCtrl.createProduct);

router.route('/products/:id')
    .put( productCtrl.updateProduct)
    .delete( productCtrl.deleteProduct);

module.exports = router;
```

III. upload.js

```
const router = require('express').Router();
const cloudinary = require('cloudinary');
const auth = require('../middlewares/auth');
const authAdmin = require('../middlewares/authAdmin');
const fs = require('fs')

//upload image on cloudinary
cloudinary.config({
    cloud_name: process.env.CLOUD_NAME,
    api_key: process.env.CLOUD_API_KEY,
    api_secret: process.env.CLOUD_API_SECRET
});

//Upload image only admin can use
router.post('/upload', auth, authAdmin, (req, res) => {
    try {
        console.log(req.files);
        if(!req.files || Object.keys(req.files).length === 0)
        return res.status(400).send({msg: 'No files were uploaded.'})

        const file = req.files.file;
        if(file.size > 1024*1024) {
```

```
            removeTmp(file.tempFilePath)
            return res.status(400).json({msg: "Size too large"})
        }

        if(file.mimetype !== 'image/jpeg' && file.mimetype !== 'image/png'){
            removeTmp(file.tempFilePath)
            return res.status(400).json({msg: "File format is incorrect."})
        }

        cloudinary.v2.uploader.upload(file.tempFilePath, {folder:
"Online_Agri_Product"}, async(err, result) => {
            if(err) throw err;
            removeTmp(file.tempFilePath);
            res.json({result});
        });


    } catch (err) {
        return res.status(500).json({msg: err.message})
    }
});

// Delete image only admin can use
router.post('/destroy', auth, authAdmin, (req, res) => {
    try {
        const {public_id} = req.body;
        if(!public_id) return res.status(400).json({msg: 'No images Selected'})

        cloudinary.v2.uploader.destroy(public_id, async(err, result) =>{
            if(err) throw err;

            res.json({msg: "Deleted Image"})
```

```
        })

    } catch (err) {
        return res.status(500).json({msg: err.message})
    }

});

const removeTmp = (path) => {
    fs.unlink(path, err => {
        if(err) throw err;
    })
}

module.exports = router;
```

IV. categoryRoute.js

```
const router = require('express').Router();
const categoryCtrl = require('../controllers/categoryCtrl');
const auth = require('../middlewares/auth');
const authAdmin = require('../middlewares/authAdmin');

router.route('/category')
    .get(categoryCtrl.getCategories)
    .post(auth, authAdmin, categoryCtrl.createCategory);

router.route('/category/:id')
    .put(auth, authAdmin, categoryCtrl.updateCategory)
    .delete(auth, authAdmin, categoryCtrl.deleteCategory);

module.exports = router;
```

V. mobilepay.js

```
const express = require("express");
const MobilePay = require("../models/MobilePay");

const router = express.Router();

//add mobile payment details to the database=======================================
router.post('/', async (req, res) => {
    const {phone, amount, pin} = req.body;
    let pay = {
        "phone" : phone,
        "amount" : amount,
        "pin" : pin,
    };
    try{
        let newpayModel = new MobilePay(pay);
        await newpayModel.save();
        res.json({Message: 'Value inserted', Result: newpayModel});
    }catch (error) {
        res.status(404).send("Mobile payment not added");
    }
})


//Get all mobile payments=========================================================
router.get('/getAll', async (req,res) => {
    try{
        const data = await MobilePay.find();
        res.json({Message : "All results fetched", Result: data})
    } catch (errror) {
```

```
        res.status(500).send("Cannot fetch all data");
    }
})

//Get a payment ==========================================================================
router.get('/get', async (req,res) => {
    const{id} = req.body;
    try{
        const pay = await MobilePay.findById(id);
        res.json({Message: "Payment recieved", Result: pay});
    } catch (error) {
        res.status(500).send("Cannot get the Payment");
    }
})

//Delete payment ==========================================================================
router.delete('/deleteMpay', async(req,res) => {
    const{id} = req.body;
    try{
        const pay = await MobilePay.findById(id);
        await pay.remove();
        res.json({Message: "Payment deleted", Result: pay});
    } catch (error) {
        res.status(500).send("Cannot delete Payment Details");
    }
})

//Update payment Details ==========================================================
router.put('/update/:id', async (req, res) => {
    const{id} = req.params;
    const {phone, amount, pin} = req.body;
    let data = {
```

```
        "phone" : phone,
        "amount" : amount,
        "pin" : pin,
    };
    try{
        const pay = await MobilePay.findByIdAndUpdate(id, data);
        res.json({Message: "Payment Updated Successfully..."});
    } catch (error) {
        res.status(500).send("Payment Not Updated")
    }
})

module.exports = router;
```

VI. paymentRoute.js

```
const router = require('express').Router()
const paymentCtrl = require('../controllers/paymentCtrl')
const auth = require('../middlewares/auth')
const authAdmin = require('../middlewares/authAdmin')


router.route('/payment')
    .get(auth, authAdmin, paymentCtrl.getPayments)
    .post(auth, paymentCtrl.createPayment)

module.exports = router;
```

5. Middlewares

I. auth.js

```
const jwt = require('jsonwebtoken');
```

```
const auth = (req, res, next) => {
    try {
        const token = req.header("Authorization")
        if(!token) return res.status(400).json({msg: "Invalid Authentication"})

        jwt.verify(token, process.env.ACCESS_TOKEN_SECRET, (err, user) => {
            if(err) return res.status(400).json({msg: "Invalid Authentication"})

            req.user = user;
            next();
        })

    } catch (err) {
        return res.status(500).json({msg: err.message})
    }
}

module.exports = auth;
```

II. authAdmin.js

```
const Users = require('../models/userModel');

const authAdmin = async (req, res, next) => {
    try {
        // Get user information by id
        const user = await Users.findOne ({
            _id: req.user.id
        })

        if(user.role === 0 || user.role === 2)
            return res.status(400).json({msg: "Admin Resources Access Denied"})
```

```
        next();

    } catch (err) {
        return res.status(500).json({msg: err.message})
    }
}

module.exports = authAdmin;
```

III. authDriver.js

```
const Users = require('../models/userModel');

const authDriver = async (req, res, next) => {
    try {
        // Get user information by id
        const user = await Users.findOne ({
            _id: req.user.id
        })

        if(user.role === 0 || user.role === 1)
            return res.status(400).json({msg: "Driver resources access denied"})

        next();

    } catch (err) {
        return res.status(500).json({msg: err.message})
    }
}

module.exports = authDriver;
```

01.Frontend

1. App.js

```
import React from 'react';
import {BrowserRouter as Router} from 'react-router-dom'
import {DataProvider} from './GlobalState'
import "../node_modules/bootstrap/dist/css/bootstrap.min.css";
import "../node_modules/bootstrap/dist/js/bootstrap.min.js";
import '../node_modules/font-awesome/css/font-awesome.min.css';
import './styles/Home.css'
import './styles/Products.css'
import './styles/PhoneBill.css'
import './styles/Utils.css'
import './styles/Login.css'
import './styles/Cart.css'
import './styles/History.css'
import './styles/Categories.css'
import Header from './components/Home/Header'
import MainPages from './components/MainPages/Pages'


function App() {
  return (
    <DataProvider>
      <Router>
        <div>
          <Header />
          <MainPages />
        </div>
      </Router>
```

```
    </DataProvider>
  );
}

export default App;
```

2. GlobalState.js

```
import React, {createContext, useState, useEffect} from 'react'
import ProductsAPI from './components/Api/ProductsApi';
import UserAPI from './components/Api/UserAPI';
import CategoriesAPI from './components/Api/CategoriesAPI';
import axios from 'axios'

export const GlobalState = createContext()

export const DataProvider =({children}) => {

    const [token, setToken] = useState(false)

    const refreshToken = async () => {
        const token = await axios.get('/user/refresh_token');
        console.log(token);
    }

    //get users
    useEffect(() =>{
        const firstLogin = localStorage.getItem('firstLogin')
        if(firstLogin){
```

```
            const refreshToken = async () =>{
                const res = await axios.get('/user/refresh_token')

                setToken(res.data.accesstoken)

                setTimeout(() => {
                    refreshToken()
                }, 10 * 60 * 1000)
            }
            refreshToken()
        }
    },[])

    const state = {
        token: [token, setToken],
        productsAPI: ProductsAPI(),
        userAPI: UserAPI(token),
        categoriesAPI: CategoriesAPI()
    }

    return (
        <GlobalState.Provider value={state}>
            {children}
        </GlobalState.Provider>
    )
}
```

3. Pages.js

```jsx
import React, {useContext} from 'react'
import {Switch, Route} from 'react-router-dom'
import Home from '../Home/Home';
import Login from '../Buyer/Login';
import Products from '../Farmer/Products';
import DetailProduct from '../Buyer/DetailProduct'
import Register from '../Buyer/Register'
import Cart from '../Buyer/Cart';
import Error from '../Home/Error';
import MobilePayMain from '../MobilePay/MobilePayMain';
import OrderHistory from '../Payment/OrderHistory';
import OrderDetails from '../Payment/OrderDetails';
import Categories from '../Farmer/Categories';
import CreateProduct from '../Farmer/CreateProduct';
import About from '../Home/About';
import Contact from '../Home/Contact'


import {GlobalState} from '../../GlobalState'


function Pages() {

    const state = useContext(GlobalState)
    const [isLogged] = state.userAPI.isLogged
    const [isAdmin] = state.userAPI.isAdmin

    return (
            <Switch>
                <Route path="/" exact component={Home} />
                <Route path="/product" exact component={Products} />
                <Route path="/detail/:id" exact component={DetailProduct} />
```

```
                <Route path="/login" exact component={isLogged ? Error : Login} />
                <Route path="/history" exact component={isLogged ? OrderHistory :
Error } />
                <Route path="/history/:id" exact component={isLogged ? OrderDetails :
Error } />
                <Route path="/register" exact component={Register} />
                <Route path="/about" exact component={About} />
                <Route path="/contact" exact component={Contact} />
                <Route path="/bill" exact component={MobilePayMain} />
                <Route path="/cart" exact component={Cart} />
                <Route path="/category" exact component={isAdmin ? Categories :
Error} />
                <Route path="/create_product" exact component={isAdmin ?
CreateProduct : Error} />
                <Route path="/edit_product/:id" exact component={isAdmin ?
CreateProduct : Error} />
                <Route path="*" exact component={Error} />
            </Switch>
        )
}

export default Pages;
```

4. Home

I. Header.js

```
import React, { useState, useContext } from 'react';
import { GlobalState } from '../../GlobalState';
import { NavLink, Link } from 'react-router-dom';
import axios from 'axios'
```

```
function Header() {

  const state = useContext(GlobalState);
  const [isLogged, setIsLogged] = state.userAPI.isLogged
  const [isAdmin, setIsAdmin] = state.userAPI.isAdmin
  const [cart] = state.userAPI.cart

  const logoutUser = async () => {
    await axios.get('/user/logout')

    localStorage.removeItem('firstLogin')

    window.location.href = "/";
  }

  const adminRouter = () => {
    return (
      <>
        <li><Link class="btn btn-white" to="/create_product">Create
Product</Link></li>
        <li><Link class="btn btn-white" to="/category">Categories</Link></li>
      </>
    )
  }

  const loggedRouter = () => {
    return (
      <>
        <li><Link className="btn btn-white ms-2 px-4" to="/history">History
</Link></li>
        <li><Link className="btn btn-outline-success ms-2 px-4 rounded-pill" to="/"
onClick={logoutUser}>Logout <i className="fa fa-sign-out me-2" /></Link></li>
```

```
      </>
    )
  }

  return (
    <div>
      <nav class="navbar navbar-expand-lg navbar-light shadow">
        <div class="container">
          <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul class="navbar-nav me-auto mb-2 mb-lg-0">
              <li class="nav-item">
                <a class="nav-link active" aria-current="page" href="/">Home</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="/about">About</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="/service">Services</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="/contact">Contact</a>
              </li>
              <li class="nav-item">
                <Link class="nav-link active " to="/product">
                  {isAdmin ? 'Products' : 'Shop'}
                </Link>
              </li>
```

```
            </ul>
            <Link class="nav-link active navbar-brand mx-auto fw-bolder fs-2" to="/">
              {isAdmin ? 'ADMIN' : 'Agri Product'}
            </Link>
            {isAdmin && adminRouter()}

            {
              isLogged ? loggedRouter() : <li><Link className ="btn btn-outline-
success ms-2 px-4 rounded-pill" to="/login">Login <i className="fa fa-sign-in me-2"/>
</Link></li>
            }

            <NavLink to="/register"> <button className="btn btn-outline-success ms-2
px-4 rounded-pill">
              Register <i className="fa fa-user-plus me-2" />
            </button> </NavLink>

            {
              isAdmin ? ''
                : <div className='cart-icon'>
                  <span>{cart.length}</span>
                  <NavLink to="/cart"> <button className="btn btn ms-2 px-4 rounded-
pill">
                    <i className="fa fa-2x fa-cart-arrow-down me-2" />
                  </button> </NavLink>
                </div>

            }

          </div>
        </div>
      </nav>
```

```
    </div >

  )
}

export default Header;
```

II. Home.js

```
import React from 'react'
import About from '../Home/About';
import Contact from './Contact';


const Home = () => {
  return (
    <div>
      <section id="home">
        <div className='container'>
          <div className='row justify-content-center'>
            <div className='col-md-8 mt-5'>
              <h1 className='display-4 fw-bolder mb-4 text-center'>Feels the Smart
Business</h1>
              <p className='lead text-center fs-4 mb-5'>
                Agriculture is the art and science of cultivating the soil,
                growing crops and raising livestock. It includes the preparation
                of plant and animal products for people to use and their
                distribution to markets. Agriculture provides most of the
```

```
                world's food and fabrics. people to use and their
                distribution to markets. Agriculture provides most of the
                world's food and fabrics.</p>



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

      </section>
      <About/>
      <Contact/>

    </div>
  )
}

export default Home;
```

III. LoadMore.js

```
import React, {useContext} from 'react'
import {GlobalState} from '../../GlobalState';

function LoadMore() {
    const state = useContext(GlobalState)
    const [page, setPage] = state.productsAPI.page
    const [result] = state.productsAPI.result

    return (
        <div className="load_more">
            {
```

```
                result < page * 9 ? ""
                : <button onClick={() => setPage(page+1)}>Load more</button>
            }
        </div>
    )
}

export default LoadMore
```

IV. Error.js

```
import React from 'react'
import img1 from '../../img/Error.png'

const Error = () => {
  return (
    <div>
      <div className='container my-5 py-5'>
        <div className='row'>
          <div className='col-md-6 mx-auto'>
            <h1><center>Error Not Found</center></h1>
            <img src={img1}
              alt="Error"
              className='img1' />
          </div>
        </div>
      </div>


    </div>
  )
```

```
}


export default Error
```

V. Filter.js

```
import React, { useContext } from 'react';
import { GlobalState } from '../../GlobalState';

function Filters() {
    const state = useContext(GlobalState)
    const [categories] = state.categoriesAPI.categories

    const [category, setCategory] = state.productsAPI.category
    const [sort, setSort] = state.productsAPI.sort
    const [search, setSearch] = state.productsAPI.search

    const handleCategory = e => {
        setCategory(e.target.value)
        setSearch('')
    }

    return (
        <div className='container filter py-5'>
            <div className="row_cat">
                <span>Filters: </span>
                <select name="category" className='form-control' value={category}
onChange={handleCategory} >
                    <option value=''>All Products</option>
                    {
                        categories.map(category => (
```

```
                            <option value={"category=" + category._id}
key={category._id}>
                                {category.name}
                            </option>
                        ))
                    }
                </select>
            </div>

            <input type="text" className='search form-control' value={search}
placeholder="Enter your search!"
                onChange={e => setSearch(e.target.value.toLowerCase())} />

            <div className="row_sorts">
                <span>Sort By: </span>
                <select className='form-control select_filter'  value={sort}
onChange={e => setSort(e.target.value)} >
                    <option value=''>Newest</option>
                    <option value='sort=oldest'>Oldest</option>
                    <option value='sort=-sold'>Best sales</option>
                    <option value='sort=-price'>Price: Hight-Low</option>
                    <option value='sort=price'>Price: Low-Hight</option>
                </select>
            </div>
        </div>

    )
}

export default Filters
```

5. Buyer

I. Cart.js

```
import React, { useContext, useState, useEffect } from 'react';
import { GlobalState } from '../../GlobalState';
import axios from 'axios';
import { Link } from 'react-router-dom';
import PayPalButton from '../Payment/PayPalButton';

function Cart() {
  const state = useContext(GlobalState)
  const [cart, setCart] = state.userAPI.cart
  const [token] = state.token
  //const [callback, setCallback] = state.userAPI.callback
  const [total, setTotal] = useState(0)

  useEffect(() => {
    const getTotal = () => {
      const total = cart.reduce((prev, item) => {
        return prev + (item.price * item.quantity)
      }, 0)

      setTotal(total)
    }

    getTotal()

  }, [cart])

  const addToCart = async (cart) =>{
    await axios.patch('/user/addcart', {cart}, {
        headers: {Authorization: token}
```

```
    })
}

  const increment = (id) => {
    cart.forEach(item => {
      if (item._id === id) {
        item.quantity += 1
      }
    })

    setCart([...cart])
    addToCart(cart)
  }

  const decrement = (id) => {
    cart.forEach(item => {
      if (item._id === id) {
        item.quantity === 1 ? item.quantity = 1 : item.quantity -= 1
      }
    })

    setCart([...cart])
    addToCart(cart)
  }

  const removeProduct = id =>{
    if(window.confirm("Do you want to delete this product?")){
        cart.forEach((item, index) => {
            if(item._id === id){
                cart.splice(index, 1)
            }
        })
```

```
        setCart([...cart])
        addToCart(cart)
    }
}

  const tranSuccess = async (payment) => {
    const { paymentID, address } = payment;

    await axios.post('/api/payment', { cart, paymentID, address }, {
      headers: { Authorization: token }
    })

    setCart([])
    addToCart([])
    alert("You have successfully Placed an Order.");

  }

  if (cart.length === 0)
    return <h2 style={{ textAlign: "center", fontSize: "5rem" }}>Your Cart is
Empty</h2>

  return (
    <div>
      {
        cart.map(product => (

          <div className="detail cart" key={product._id}>
            <img className='cart_imgs' src={product.images.url} alt="" />

            <div className="box-detail">
```

```
            <h2>{product.title}</h2>

            <h3>$ {product.price * product.quantity}</h3>
            <p className='cart_con'>{product.content}</p>
            <p className='cart_desc'>{product.description}</p>




            <div className="amount">
              <button onClick={() => decrement(product._id)}> - </button>
              <span>{product.quantity}</span>
              <button onClick={() => increment(product._id)}> + </button>
            </div>

            <Link to="/cart" className='btn btn-success btn-lg'>Buy Now</Link>

            <div className="delete"
              onClick={() => removeProduct(product._id)}>
              X
            </div>
          </div>
        </div>
      ))
    } <br/> <br/> <br/>
    <div className="total">
      <h1>Total Price: $ {total}</h1>
      <PayPalButton className="paypal"
      total = {total}
      tranSuccess={tranSuccess}/>
    </div> <br/> <br/> <br/>
```

```
    </div>
  )
}

export default Cart
```

II. DetailProduct.js

```
import React, { useState, useContext, useEffect } from 'react';
import { useParams, Link } from 'react-router-dom';
import { GlobalState } from '../../GlobalState';
import ProductItem from '../Buyer/ProductItem';

function DetailProduct() {
    const params = useParams()
    const state = useContext(GlobalState);
    const [products] = state.productsAPI.products
    const addCart = state.userAPI.addCart
    const [detailProduct, setDetailProduct] = useState([])

    useEffect(() => {
        if (params.id) {
            products.forEach(product => {
                if (product._id === params.id) setDetailProduct(product)
            })
        }
    }, [params.id, products]);
    if (detailProduct.length === 0) return null;

    return (
        <>
            <div className="detail">
```

```
                <img src={detailProduct.images.url} alt="" />
                <div className="box-detail">
                    <div className="row">
                        <h2> {detailProduct.title} </h2>
                        <h6> #id:{detailProduct.product_id} </h6>
                    </div>
                    <span>$ {detailProduct.price} </span>
                    <p> {detailProduct.description} </p>
                    <p> {detailProduct.content} </p>
                    <p>Sold: {detailProduct.sold}</p>

                    <Link className="btn btn-success btn-lg" to="/cart"
                    onClick={() => addCart(detailProduct)}>
                        Buy Now
                    </Link>
                </div>
            </div>

            <div>
                <h2>Related products</h2>
                <div className="products">
                    {
                        products.map(product => {
                            return product.category === detailProduct.category
                                ? <ProductItem key={product._id} product={product} />
: null
                        })
                    }
                </div>
            </div>
        </>
    )
```

```
}


export default DetailProduct
```

III. Login.js

```
import React, { useState } from 'react'
import { Link, NavLink } from 'react-router-dom'
import axios from 'axios'

function Login() {
  const [user, setUser] = useState({
    email: '',
    password: ''
  });

  const onChangeInput = e => {
    const { name, value } = e.target;
    setUser({ ...user, [name]: value })
  }

  const loginSubmit = async e => {
    e.preventDefault()
    try {
      await axios.post('/user/login', { ...user })

      localStorage.setItem('firstLogin', true)

      window.location.href = "/product";

    } catch (err) {
        alert(err.response.data.msg)
```

```
    }
  }

  return (
    <div>
      <div className="container shadow my-5">
        <div className="row">
          <div className="col-md-5 d-flex flex-column align-items-center text-dark
justify-content-center form">
            <h1 className="display-4 fw-bolder"> Welcome Back</h1>
            <p className="lead text-center">Enter Your Credentials to Login</p>
            <h5 className="mb-4">OR</h5>
            <NavLink to="/register" className="btn btn-outline-light rounded-pill pb-
2 w-50">Register</NavLink>
          </div>
          <div className="col-md-6 p-5">
            <h1 className="display-6 fw-bolder mb-5">LOGIN</h1>
            <form onSubmit={loginSubmit}>
              <div class="mb-3">
                <label for="exampleInputEmail1" class="form-label">Email
address</label>
                <input name="email" value={user.email} onChange={onChangeInput}
type="email" class="form-control" id="exampleInputEmail1" aria-
describedby="emailHelp" required />
              </div>
              <div class="mb-3">
                <label for="exampleInputPassword1" class="form-
label">Password</label>
                <input name="password" value={user.password} onChange={onChangeInput}
autoComplete="on" type="password" class="form-control" id="exampleInputPassword1"
required />
              </div>
```

```
              <div class="mb-3 form-check">
                <input type="checkbox" class="form-check-input" id="exampleCheck1" />
                <label class="form-check-label" for="exampleCheck1">Remember
Me</label>
              </div>
              <button type="submit" class="btn btn-success w-100 rounded-
pill">Login</button>
            </form>
          </div>
        </div>
      </div>
    </div>
  )
}

export default Login
```

IV. Register.js

```
import React, { useState } from 'react'
import { Link, NavLink } from 'react-router-dom'
import axios from 'axios'

function Register() {
  const [user, setUser] = useState({
    name: '',
    email: '',
    password: ''
  })

  const onChangeInput = e => {
    const { name, value } = e.target;
```

```
    setUser({ ...user, [name]: value })
  }

  const registerSubmit = async e => {
    e.preventDefault()
    try {
      await axios.post('/user/register', { ...user })

      localStorage.setItem('firstLogin', true)

      window.location.href = "/product";

    } catch (err) {
      alert(err.response.data.msg)
    }
  }
  return (
    <div>
      <div className="container shadow my-5">
        <div className="row justify-content-end">
          <div className="col-md-5 d-flex flex-column align-items-center text-dark
justify-content-center form order-2">
            <h1 className="display-4 fw-bolder text-center"> Welcome to the Store
</h1>
            <p className="lead text-center"> Enter Your Details to Register </p>
            <h5 className="mb-4">OR</h5>
            <NavLink to="/login" className="btn btn-outline-light rounded-pill pb-2
w-50">Login</NavLink>
          </div>
          <div className="col-md-6 p-5">
            <h1 className="display-6 fw-bolder mb-5">REGISTER</h1>
            <form onSubmit={registerSubmit}>
```

```
            <div class="mb-3">
              <label for="username" class="form-label">Your Name</label>
              <input type="text" name="name" class="form-control" required
                placeholder="Name" value={user.name} onChange={onChangeInput} />
            </div>

            <div class="mb-3">
              <label for="username" class="form-label">Your E-mail Address</label>
              <input type="email" name="email" class="form-control" required
                placeholder="Email" value={user.email} onChange={onChangeInput} />
            </div>

            <div class="mb-3">
              <label for="username" class="form-label">Password</label>
              <input type="password" name="password" class="form-control" required
autoComplete="on"
                placeholder="Password" value={user.password}
onChange={onChangeInput} />
            </div>

            <div class="mb-3 form-check">
              <input type="checkbox" class="form-check-input" id="exampleCheck1" />
              <label class="form-check-label" for="exampleCheck1">I Agree Terms and
Conditions</label>
            </div>
            <button type="submit" class="btn btn-success w-100 mt-4 rounded-
pill">Register</button>
          </form>
        </div>
      </div>
    </div>
```

```
    </div>
  )
}

export default Register;
```

V. ProductItem.js

```
import React from 'react';
import { Link } from 'react-router-dom'
import BtnRender from '../Farmer/BtnRender';
import axios from 'axios'

function ProductItem({ product, isAdmin, deleteProduct, handleCheck }) {

  return (
    <section className='py-4 container'>
      <div className='row justidy-content-center'>
        <div className='col-12'>

          <div className='product_card'>
            {
              isAdmin && <input type="checkbox" checked={product.checked}
                onChange={() => handleCheck(product._id)} />
            }

            <img src={product.images.url} alt="z" />

            <div className='product_box'>
              <h2 title={product.title}>{product.title}</h2>
              <span>${product.price}</span>
              <p>{product.description}</p>
```

```
            </div>

            <BtnRender product={product} deleteProduct={deleteProduct} />

          </div>
        </div>
      </div>
    </section>
  )
}

export default ProductItem;
```

6. Farmer

I. BtnRender.js

```
import React, { useContext } from 'react';
import { Link } from 'react-router-dom';
import { GlobalState } from '../../GlobalState'

function BtnRender({ product , deleteProduct}) {
    const state = useContext(GlobalState)
    const [isAdmin] = state.userAPI.isAdmin
    const addCart = state.userAPI.addCart

    return (
        <div className='row_btn'>
            {
                isAdmin ?
                    <>
                        <Link id="btn_buy" to="#!"
```

```
                    onClick={() =>deleteProduct(product._id,
product.images.public_id)}>
                        Delete
                    </Link>
                        <Link id="btn_view" to={`/edit_product/${product._id}`}>
                            Edit
                        </Link>
                    </>
                    : <>
                        <Link id="btn_buy" to="#!" onClick={() => addCart(product)}>
                            Buy
                        </Link>
                        <Link id="btn_view" to={`/detail/${product._id}`}>
                            View
                        </Link>
                    </>
            }
        </div>
    )
}

export default BtnRender;
```

II. Categories.js

```
import React, { useState, useContext } from 'react'
import { GlobalState } from '../../GlobalState'
import axios from 'axios'

function Categories() {
    const state = useContext(GlobalState)
    const [categories] = state.categoriesAPI.categories
```

```
    const [category, setCategory] = useState('')
    const [token] = state.token
    const [callback, setCallback] = state.categoriesAPI.callback
    const [onEdit, setOnEdit] = useState(false)
    const [id, setID] = useState('')

    const createCategory = async e => {
        e.preventDefault()
        try {
            if(onEdit) {
                const res = await axios.put(`/api/category/${id}`, {name: category},
{
                    headers: {Authorization: token}
                })
                alert(res.data.msg)
            } else {
                const res = await axios.post('/api/category', {name: category}, {
                    headers: {Authorization: token}
                })
                alert(res.data.msg)
            }
            setOnEdit(false)
            setCategory('')
            setCallback(!callback)

        } catch (err) {
            alert(err.response.data.msg)
        }
    }

    const editCategory = async (id, name) =>{
        setID(id)
```

```
        setCategory(name)
        setOnEdit(true)
    }

    const deleteCategory = async id =>{
        try {
            const res = await axios.delete(`/api/category/${id}`, {
                headers: {Authorization: token}
            })
            alert(res.data.msg)
            setCallback(!callback)
        } catch (err) {
            alert(err.response.data.msg)
        }
    }

    return (
        <div className='container'> <br/>
            <div className="categories_edit form-control">
            <form onSubmit={createCategory}>
                <label className="lbl_cat" htmlFor="Category" >Category</label>
                <input className='form-control crt_cat' type="text"
placeholder='Enter Category Name' name="category" value={category} required
                onChange={e => setCategory(e.target.value)} />

                <button className="btn btn-success cat_btn" type="submit">{onEdit?
"Update" : "Create"}</button>
            </form> <br/>

            <div className="">
                {
                    categories.map(category => (
```

```
                        <div className="cat_name" key={category._id}>
                            <p>{category.name}</p>
                            <div>
                                <button className='btn btn-warning edit_btn'
onClick={() => editCategory(category._id, category.name)}>Edit</button>
                                <button className='btn btn-danger dlt_btn'
onClick={() => deleteCategory(category._id)}>Delete</button>
                            </div>
                        </div>
                    ))
                }
            </div>
        </div>
        </div>
    )
}

export default Categories;
```

III. CreateProduct.js

```
import React, { useState, useContext, useEffect } from 'react'
import { GlobalState } from '../../GlobalState'
import Loading from '../utils/Loading';
import { useHistory, useParams } from 'react-router-dom'
import axios from 'axios'

const initialState = {
  product_id: '',
  title: '',
  price: 0,
```

```
  description: 'Soil fertility with more nutrients. Immunity of plant & leaves.
Nutrients uptake. White root developments. Protection pest attacks & deceases.
Chlorophyll content and energy. Vegetation growth.Fruiting and more yield. Chemical
toxicity in soil. pH of water and soil.',
  content: 'Protection',
  category: '',
  _id: ''
}

function CreateProduct() {
  const state = useContext(GlobalState)
  const [product, setProduct] = useState(initialState)
  const [categories] = state.categoriesAPI.categories
  const [images, setImages] = useState(false)
  const [loading, setLoading] = useState(false)


  const [isAdmin] = state.userAPI.isAdmin
  const [token] = state.token

  const history = useHistory()
  const param = useParams()

  const [products] = state.productsAPI.products
  const [onEdit, setOnEdit] = useState(false)
  //const [callback, setCallback] = state.productsAPI.callback

  useEffect(() => {
    if (param.id) {
      setOnEdit(true)
      products.forEach(product => {
        if (product._id === param.id) {
```

```
        setProduct(product)
        setImages(product.images)
      }
    })
  } else {
    setOnEdit(false)
    setProduct(initialState)
    setImages(false)
  }
}, [param.id, products])

const handleUpload = async e => {
  e.preventDefault()
  try {
    if (!isAdmin) return alert("You're not an admin")
    const file = e.target.files[0]

    if (!file) return alert("File not exist.")

    if (file.size > 1024 * 1024) // 1mb
      return alert("Size too large!")

    if (file.type !== 'image/jpeg' && file.type !== 'image/png') // 1mb
      return alert("File format is incorrect.")

    let formData = new FormData()
    formData.append('file', file)

    setLoading(true)
    const res = await axios.post('/api/upload', formData, {
      headers: { 'content-type': 'multipart/form-data', Authorization: token }
    })
```

```
      setLoading(false)
      setImages(res.data)

    } catch (err) {
      alert(err.response.data.msg)
    }
}

const handleDestroy = async () => {
    try {
      if (!isAdmin) return alert("You're not an admin")
      setLoading(true)
      await axios.post('/api/destroy', { public_id: images.public_id }, {
        headers: { Authorization: token }
      });

      setLoading(false)
      setImages(false)

    } catch (err) {
      alert(err.response.data.msg)
    }
}

const handleChangeInput = e => {
    const { name, value } = e.target
    setProduct({ ...product, [name]: value })
}

const handleSubmit = async e => {
    e.preventDefault()
    try {
```

```
    if (!isAdmin) return alert("You're not an admin")
    if (!images) return alert("No Image Upload")

    if (onEdit) {
      await axios.put(`/api/products/${product._id}`, { ...product, images }, {
        headers: { Authorization: token }
      })
    } else {
      await axios.post('/api/products', { ...product, images }, {
        headers: { Authorization: token }
      })

      setImages(false)
      setProduct(initialState)
      //setCallback(!callback)
      history.push('/product')
    }

  } catch (err) {
    alert(err.response.data.msg)
  }
}

const styleUpload = {
  display: images ? "block" : "none"
}

return (
  <div className="create_product">
    <div className="upload">
      <input type="file" name="file" id="file_up" onChange={handleUpload} />
      {
```

```
          loading ? <div id="file_img" style={styleUpload}> <Loading /></div>

            : <div id="file_img" >
              <img src={images ? images.url : ''} alt="" />
              <span onClick={handleDestroy}>X</span>
            </div>
        }

      </div>

      <form onSubmit={handleSubmit}>
        <div className="row">
          <label htmlFor="product_id">Product ID</label>
          <input type="text" name="product_id" id="product_id" required
            value={product.product_id} onChange={handleChangeInput}
disabled={product._id} />
        </div>

        <div className="row">
          <label htmlFor="title">Title</label>
          <input type="text" name="title" id="title" required
            value={product.title} onChange={handleChangeInput} />
        </div>

        <div className="row">
          <label htmlFor="price">Price</label>
          <input type="number" name="price" id="price" required
            value={product.price} onChange={handleChangeInput} />
        </div>

        <div className="row">
          <label htmlFor="description">Description</label>
```

```
          <textarea type="text" name="description" id="description" required
            value={product.description} rows="3" onChange={handleChangeInput} />
        </div>

        <div className="row">
          <label htmlFor="content">Content</label>
          <textarea type="text" name="content" id="content" required
            value={product.content} rows="4" onChange={handleChangeInput} />
        </div>

        <div className="row">
          <label htmlFor="categories">Categories: </label>
          <select name="category" value={product.category}
onChange={handleChangeInput} >
            <option value="">Please select a category</option>
            {
              categories.map(category => (
                <option value={category._id} key={category._id}>
                  {category.name}
                </option>
              ))
            }
          </select>
        </div>


        <button class="btn btn-success w-100 rounded-pill" type="submit">
          {onEdit ? "Update" : "Create"}
        </button>
      </form>
    </div>
  )
```

```
}


export default CreateProduct;
```

IV. Products.js

```
import React, { useState, useContext, useEffect } from 'react';
import { GlobalState } from '../../GlobalState';
import ProductItem from '../Buyer/ProductItem';
import Loading from '../utils/Loading';
import axios from 'axios'
import Filters from '../Home/Filters';
import LoadMore from '../Home/LoadMore';

function Prodcuts() {

  const state = useContext(GlobalState)
    const [products, setProducts] = state.productsAPI.products
    const [isAdmin] = state.userAPI.isAdmin
    const [token] = state.token
    //const [callback, setCallback] = state.productsAPI.callback
    const [loading, setLoading] = useState(false)
    const [isCheck, setIsCheck] = useState(false)

    const handleCheck = (id) =>{
        products.forEach(product => {
            if(product._id === id) product.checked = !product.checked
        })
        setProducts([...products])
    }

    const deleteProduct = async(id, public_id) => {
```

```
        try {
            setLoading(true)
            const destroyImg = axios.post('/api/destroy', {public_id},{
                headers: {Authorization: token}
            })
            const deleteProduct = axios.delete(`/api/products/${id}`, {
                headers: {Authorization: token}
            })

            await destroyImg
            await deleteProduct
            //setCallback(!callback)
            setLoading(false)
        } catch (err) {
            alert(err.response.data.msg)
        }
    }

    const checkAll = () =>{
        products.forEach(product => {
            product.checked = !isCheck
        })
        setProducts([...products])
        setIsCheck(!isCheck)
    }

    const deleteAll = () =>{
        products.forEach(product => {
            if(product.checked) deleteProduct(product._id, product.images.public_id)
        })
    }
```

```
    if(loading) return <div><Loading /></div>
    return (
        <>

        <Filters/>

        {
            isAdmin &&
            <div className="delete-all">
                <span>Select all</span>
                <input type="checkbox" checked={isCheck} onChange={checkAll} />
                <button onClick={deleteAll}>Delete ALL</button>
            </div>
        }

        <div className="products">
            {
                products.map(product => {
                    return <ProductItem key={product._id} product={product}
                    isAdmin={isAdmin} deleteProduct={deleteProduct}
handleCheck={handleCheck} />
                })
            }
        </div>

        <LoadMore/>
        {products.length === 0 && <Loading />}
        </>
    )
}
```

```
export default Prodcuts
```

7. Payment

I. OrderDetails.js

```
import React, {useState , useEffect , useContext} from 'react';
import {useParams} from 'react-router-dom';
import { GlobalState } from '../../GlobalState';

const OrderDetails = () => {
    const state = useContext(GlobalState)
    const [history] = state.userAPI.history
    const [orderDetails, setOrderDetails] = useState([])

    const params = useParams();

    useEffect(() => {
        if(params.id){
            history.forEach(item =>{
                if(item._id === params.id) setOrderDetails(item)
            })
        }
    },[params.id, history]);
    console.log(orderDetails);

    if(orderDetails.length === 0) return null;

  return (
    <div className="history-page">
            <table>
                <thead>
                    <tr>
```

```
                        <th>Name</th>
                        <th>Address</th>
                        <th>Postal Code</th>
                        <th>Country Code</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>{orderDetails.address.recipient_name}</td>
                        <td>{orderDetails.address.line1 + " - " +
orderDetails.address.city}</td>
                        <td>{orderDetails.address.postal_code}</td>
                        <td>{orderDetails.address.country_code}</td>
                    </tr>
                </tbody>
            </table>

            <table style={{margin: "30px 0px"}}>
                <thead>
                    <tr>
                        <th></th>
                        <th>Products</th>
                        <th>Quantity</th>
                        <th>Price</th>
                    </tr>
                </thead>
                <tbody>
                    {
                        orderDetails.cart.map(item =>(
                        <tr key={item._id}>
                            <td><img src={item.images.url} alt="" /></td>
                            <td>{item.title}</td>
```

```
                            <td>{item.quantity}</td>
                            <td>$ {item.price * item.quantity}</td>
                        </tr>
                        ))
                    }

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

export default OrderDetails
```

II. OrderHistory.js

```
import React, { useEffect , useContext } from 'react'
import { GlobalState } from '../../GlobalState'
import {Link} from 'react-router-dom'
import axios from 'axios'

function OrderHistory() {
    const state = useContext(GlobalState)
    const [history, setHistory] = state.userAPI.history
    const [isAdmin] = state.userAPI.isAdmin
    const [token] = state.token

    useEffect(() => {
        if(token){
            const getHistory = async() =>{
                if(isAdmin){
                    const res = await axios.get('/api/payment', {
```

```
                    headers: {Authorization: token}
                })
                setHistory(res.data)
            }else{
                const res = await axios.get('/user/history', {
                    headers: {Authorization: token}
                })
                setHistory(res.data)
            }
        }
        getHistory()
    }
  },[token, isAdmin, setHistory])

  return (
  <div className='container'> <br/> <br/>
      <h2><center>History</center></h2> <br/>
      <h4>You have {history.length} Order </h4> <br/><br/>

      <table class="table">
<thead>
  <tr>
    <th scope="col">#</th>
    <th scope="col">Payment ID</th>
    <th scope="col">Date of Purchased</th>
  </tr>
</thead>
<tbody>
{
                    history.map(items => (
                        <tr key={items._id}>
                            <th scope="row">1</th>
```

```
                                <td>{items.paymentID}</td>
                                <td>{new
Date(items.createdAt).toLocaleDateString()}</td>
                                <td><Link
to={`/history/${items._id}`}>View</Link></td>
                            </tr>
                        ))
                    }



  </tbody>
</table>

    </div>
  )
}

export default OrderHistory
```

III. PayPalButton.js

```
import React from 'react';
import PaypalExpressBtn from 'react-paypal-express-checkout';

export default class PayPalButton extends React.Component {
    render() {
        const onSuccess = (payment) => {
            // Congratulation, it came here means everything's fine!
            console.log("The payment was succeeded!", payment);
            // You can bind the "payment" object's value to your state or props or
whatever here, please see below for sample returned data
            this.props.tranSuccess(payment)
```

```
        }

        const onCancel = (data) => {
            // User pressed "cancel" or close Paypal's popup!
            console.log('The payment was cancelled!', data);
            // You can bind the "data" object's value to your state or props or
whatever here, please see below for sample returned data
        }

        const onError = (err) => {
            // The main Paypal's script cannot be loaded or somethings block the
loading of that script!
            console.log("Error!", err);
            // Because the Paypal's main script is loaded asynchronously from
"https://www.paypalobjects.com/api/checkout.js"
            // => sometimes it may take about 0.5 second for everything to get set,
or for the button to appear
        }

        let env = 'sandbox'; // you can set here to 'production' for production
        let currency = 'USD'; // or you can set this value from your props or state
        let total = this.props.total; // same as above, this is the total amount
(based on currency) to be paid by using Paypal express checkout
        // Document on Paypal's currency code:
https://developer.paypal.com/docs/classic/api/currency_codes/

        const client = {
            sandbox:
'ARWjg7sk5sujWnGSsKKLgFkHaadfGqkNU_FrL26vIqYUk7JHrVeBtZlRID0REwNDBKcwucwImJzd2cw2',
            production: 'YOUR-PRODUCTION-APP-ID',
        }
```

```
        // In order to get production's app-ID, you will have to send your app to
Paypal for approval first
        // For sandbox app-ID (after logging into your developer account, please
locate the "REST API apps" section, click "Create App"):
        //   => https://developer.paypal.com/docs/classic/lifecycle/sb_credentials/
        // For production app-ID:
        //   => https://developer.paypal.com/docs/classic/lifecycle/goingLive/

        // NB. You can also have many Paypal express checkout buttons on page, just
pass in the correct amount and they will work!
        let style = {
            size: 'large',
            color: 'blue',
            shape: 'rect',
            label: 'checkout',
            className: 'paypal',
            tagline: false
        }

        return (
            <PaypalExpressBtn
                env={env}
                client={client}
                currency={currency}
                total={total}
                onError={onError}
                onSuccess={onSuccess}
                onCancel={onCancel}
                style={style}/>
        );
    }
}
```

8. MobilePay

I. Otp.js

```
import React, { Component } from 'react'
import '../../styles/PhoneBill.css';

class Otp extends Component {

    constructor(props) {
        super(props)

        this.state = {
           pin: " ",
           mobile : " ",
           amount: " ",
        }
    }
    handleMobileNo = (event) => {
        this.setState({
            mobile: event.target.value
        })
    }

    handleAmount = (event) => {
        this.setState({
            amount: event.target.value
        })
    }

    handlePin = (event) => {
        this.setState({
```

```
            pin: event.target.value
        })
    }

    // handleSubmit = (event) => {
    //     alert(`Your Pin :  ${this.state.pin}`)
    //     event.preventDefault()
    // }
    postData = async (e) => {
        e.preventDefault();
        const {mobile, amount, pin} = this.state;

        const res2 = await fetch("http://localhost:3100/api/mobilepay/sendData", {
            method: "post",
            headers : {
                'Content-Type': 'application/json',
                'Accept': 'application/json'
            },
            body: JSON.stringify({
                phone : mobile,
                amount : amount,
                pin : pin,
            })
        })

        const data = await res2.json();
        if(res2.status === 404 || !data){
            window.alert("Data Not added");
            console.log("Data Not added");
        }else{
            window.alert("Data added");
            console.log("Data added");
```

```
        }
    }


    render() {
        return (
            <div>
                <form method='POST'  className='billCard mpay-col-1'
onSubmit={this.postData}>
                    <h2 className='formTitle'>Enter OTP</h2>
                    <div>
                        <label className='label'>Enter Mobile Number</label>
                        <input type='tel' id='mobile' value={this.state.mobile}
name='mobile' required placeholder='enter number' onChange={this.handleMobileNo}
size='30' minLength='12' maxLength='12' />
                    </div>
                    <div>
                        <label>Charging Amount</label>
                        <input type='text' id='amount' name='amount'
value={this.state.amount} required placeholder='enter amount'
onChange={this.handleAmount} size='30' />
                    </div>
                    <div>
                        <label>Pin Number</label>
                        <input type='text' id='pin' name='pin' value={this.state.pin}
required placeholder='enter pin number' size='30' onChange={this.handlePin}
minLength='6' maxLength='6' />
                    </div>
                    <div>
                        <label />
                        <button className='billbutton
                    </div>
```

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

export default Otp;
```

II. Billing.js

```
import React, { Component } from 'react'
import '../../styles/PhoneBill.css';

class Billing extends Component {

    constructor(props) {
        super(props)

        this.state = {
           mobile : " ",
           amount: " ",
        }
    }

    handleMobileNo = (event) => {
        this.setState({
            mobile: event.target.value
        })
    }
```

```
handlePin = (event) => {
    this.setState({
        pin: event.target.value
    })
}

handleAmount = (event) => {
    this.setState({
        amount: event.target.value
    })
}

handleSubmit = (event) => {
    alert(`Mobile No : ${this.state.mobile} & Amount : ${this.state.amount}`)
    event.preventDefault()
}

postData = async (e) => {
    e.preventDefault();
    const {mobile, amount} = this.state;

    const res = await fetch("http://localhost:5100/api", {
        method: "post",
        headers : {
            'Content-Type': 'application/json',
            'Accept': 'application/json'
        },
        body: JSON.stringify({
            phone : mobile,
            amount : amount,
            pin : amount,
        })
```

```
        })

        const res2 = await fetch("http://localhost:5100/api/sendOTP", {
            method: "post",
            headers : {
                'Content-Type': 'application/json',
                'Accept': 'application/json'
            },
            body: JSON.stringify({
                phone : mobile,
                amount : amount,
                pin : amount,
            })
        })

        const data = await res.json();
        if(res.status === 404 || !data){
            window.alert("Data Not added");
            console.log("Data Not added");
        }else{
            window.alert("Data added");
            console.log("Data added");
        }
    }

    render() {
        return (
            <div>
                <form method='POST' className='billCard mpay-col-2'
onSubmit={this.postData}>
                    <h2 className='formTitle'>Mobile Pay</h2>
                    <div>
```

```
                        <label className='label'>Enter Mobile Number</label>
                        <input type='tel' id='mobile' value={this.state.mobile}
name='mobile' required placeholder='enter number' onChange={this.handleMobileNo}
size='30' minLength='12' maxLength='12' />
                    </div>
                    <div>
                        <label>Charging Amount</label>
                        <input type='text' id='amount' name='amount'
value={this.state.amount} required placeholder='enter amount'
onChange={this.handleAmount} size='30' />
                    </div>
                    {/* <div>
                        <label>Pin Number</label>
                        <input type='text' id='pin' name='pin' value={this.state.pin}
required placeholder='enter pin number' size='30' onChange={this.handlePin}
minLength='6' maxLength='6' />
                    </div> */}
                    <div>
                        <label />
                        <button className='billbutton next' type='submit' >Get
OTP</button>
                    </div>
                </form>
            </div>
        )
    }
}

export default Billing
```

III. MobilePayMain.js

```
import React from 'react';
import Billing from './Billing';
import Otp from './Otp';



function MobilePayMain() {
  return (
    <div className='mpay-row'>
        <Billing className='mpay-col-2'/>
        <Otp className='mpay-col-1'/>
    </div>
  );
}

export default MobilePayMain;
otp' type='submit'>Pay Bill</button>
```