



Sri Lanka Institute of Information Technology

Agri product online purchasing platform

Distributed Systems (SE3020)

Assignment 2: REST API

Group Number: 2022S1_REG_WD_14

Group Members

Registration No	Student Name
IT20022310	W G K Lakshan
IT20076566	De Silva G K S
IT20124830	Sampath H W A
IT19963020	Bandara E M L P

Table of Contents

01.	Introduction.....	3
02.	Technologies.....	3
03.	High Level Architectural Diagram	4
04.	Main Workflow of the System.....	5
05.	Functionalities.....	13
5.1	Farmer.....	13
5.2	Buyer.....	18
5.3	Payment Gateway	19
	Appendix.....	21
	Back End (server side implementation)	21
	Front-End (client side implementation)	55

01. Introduction

“Ceylon agri pvt.ltd” is an Agri product online purchasing platform which you can purchase any Agri item through the internet very easily. The main purpose of the system is to deliver an appropriate way to do their online transactions without wasting their time in shops. Anyone and easily login to the system and purchase the items and do the payment without any complicated.

These systems mainly contain two user types

- Farmer: The Person who adds and store items in the site
- Buyer: The Person who purchase items through the site

Customers can login to the system and after the successful login they can get the list of items and through the list they can select items as their wish. Selected items are collected in the cart with the total bill payment. After getting the total bill payment user can proceed with the payment. They can choose a preferred payment method. Two payment types are available in the site

- Pay by Credit Card
- Pay By Mobile Bill

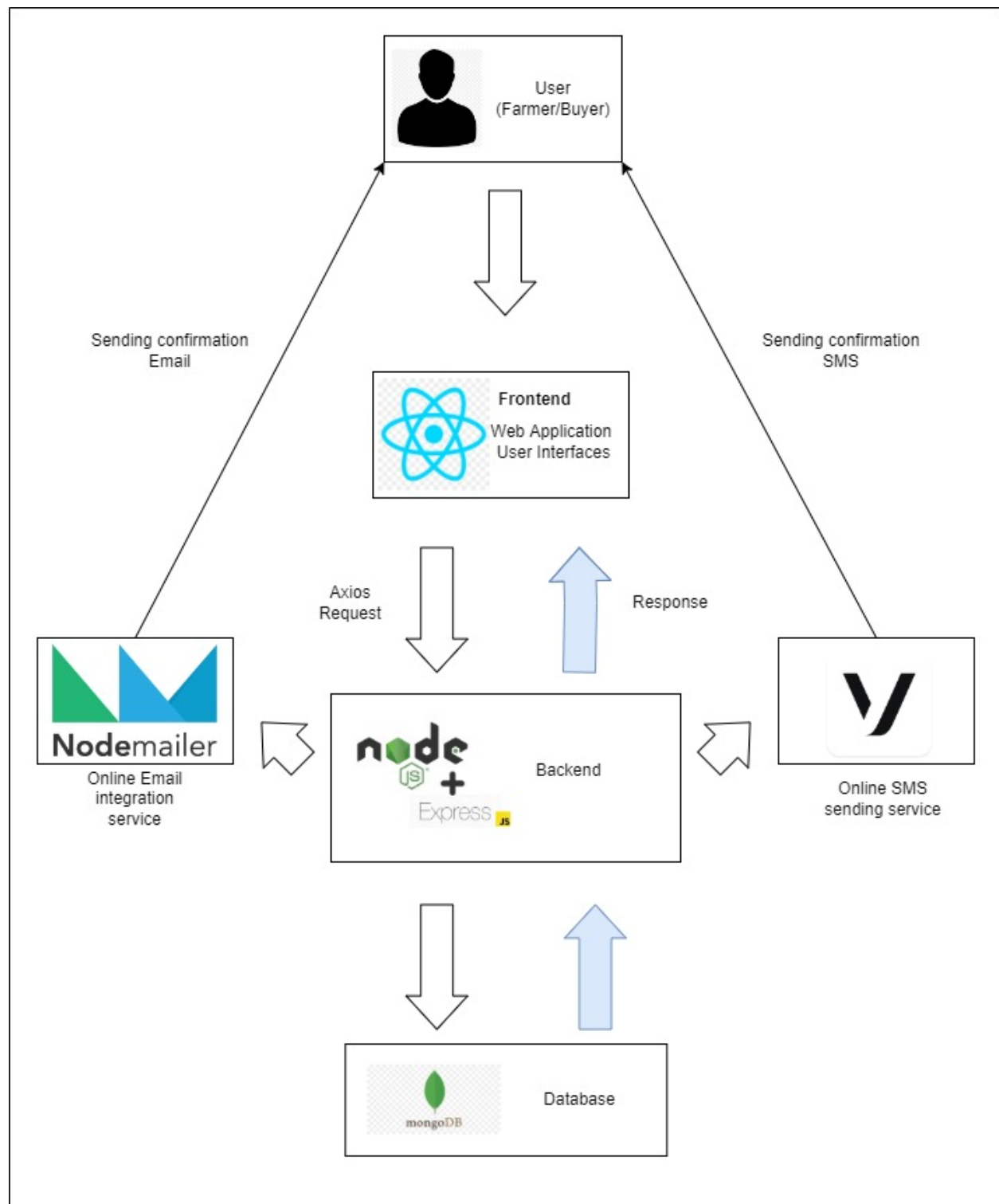
One the payment is finished; user can request to deliver the items to their designed address. After the success payment user get their payment confirmation via e-mail. Farmer has the access to view the list of the items in the store and execute the CRUD operations such as Add items, Update items, delete items and retrieve the items form the database.

02. Technologies

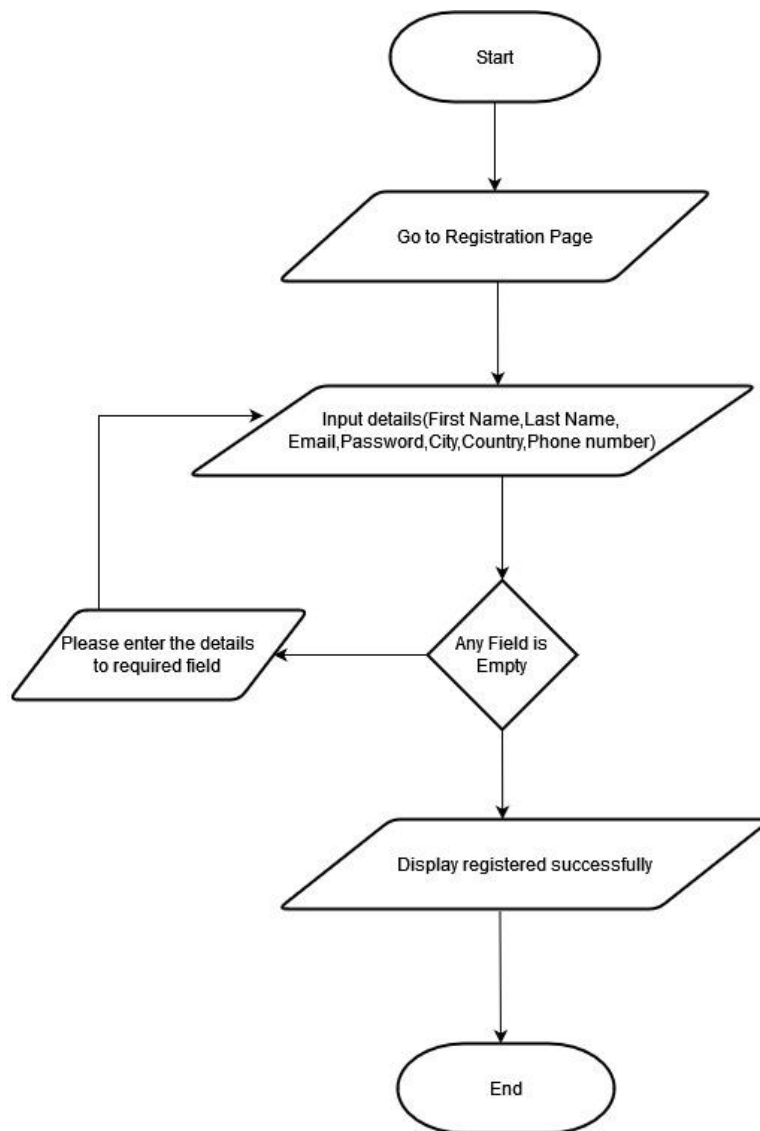
Following are the technologies that we used for developing the Online Shopping Site.

- ReactJS: Frontend Framework
- ExpressJS & NodeJS: Backend Framework
- MongoDB: Database
- Vonage: Online mobile service for SMS sending service
- Nodemailer: Online Email service to confirm the payment

03. High Level Architectural Diagram

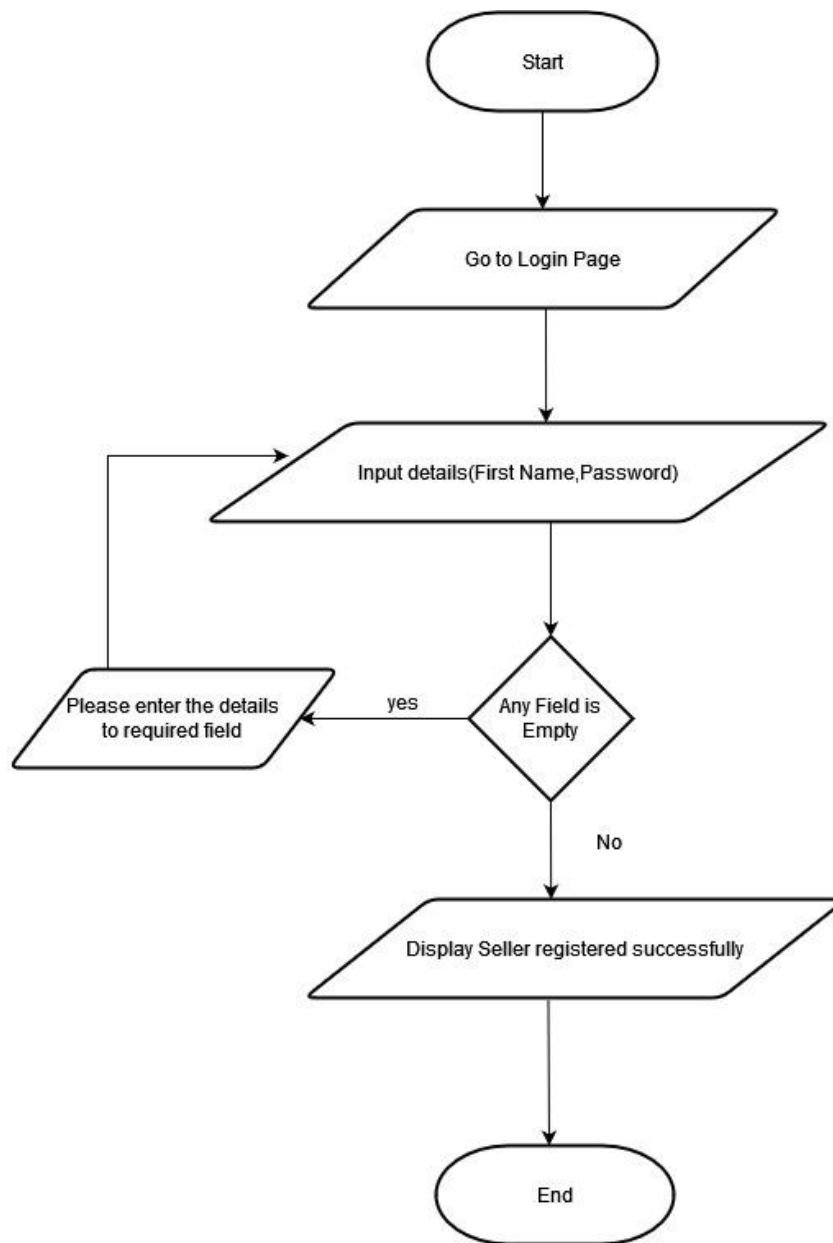


04. Main Workflow of the System



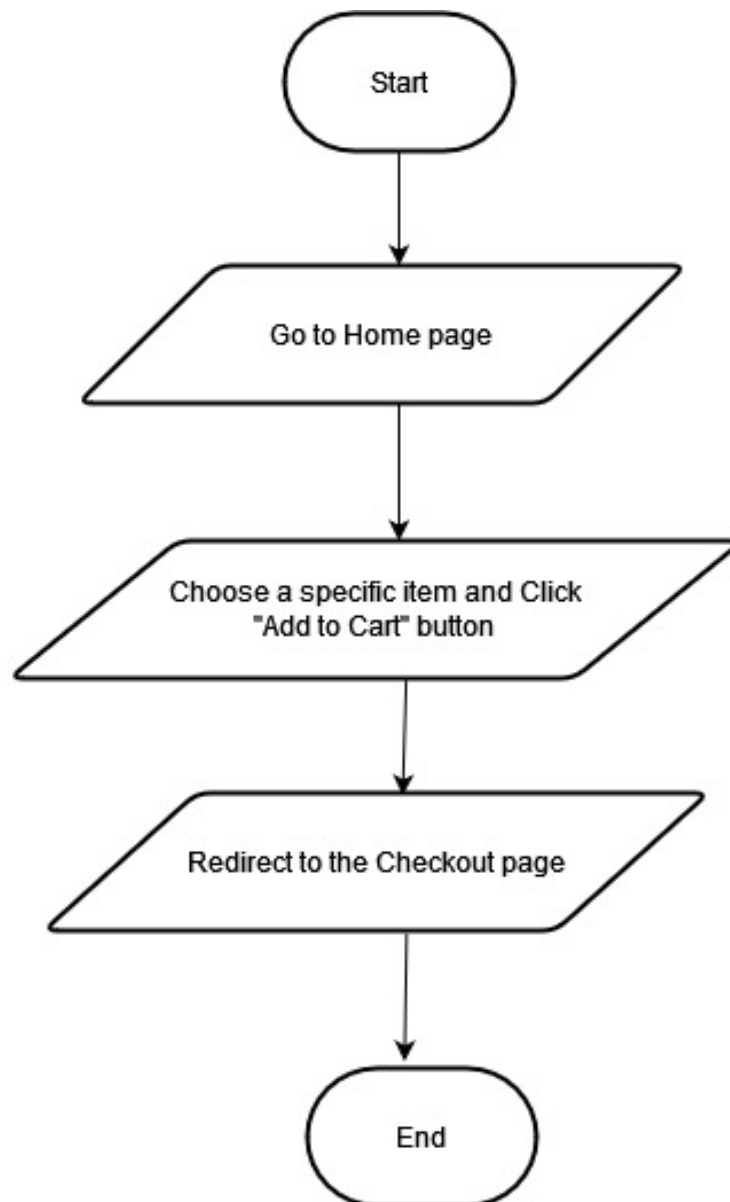
User Registration:

There are 2 main user roles. They are Farmer and Buyer. According to the user role, user can perform separate task.



User login:

When logging to the system user has to enter the Name and Password. After the successful login, user can do their task according to the user role.

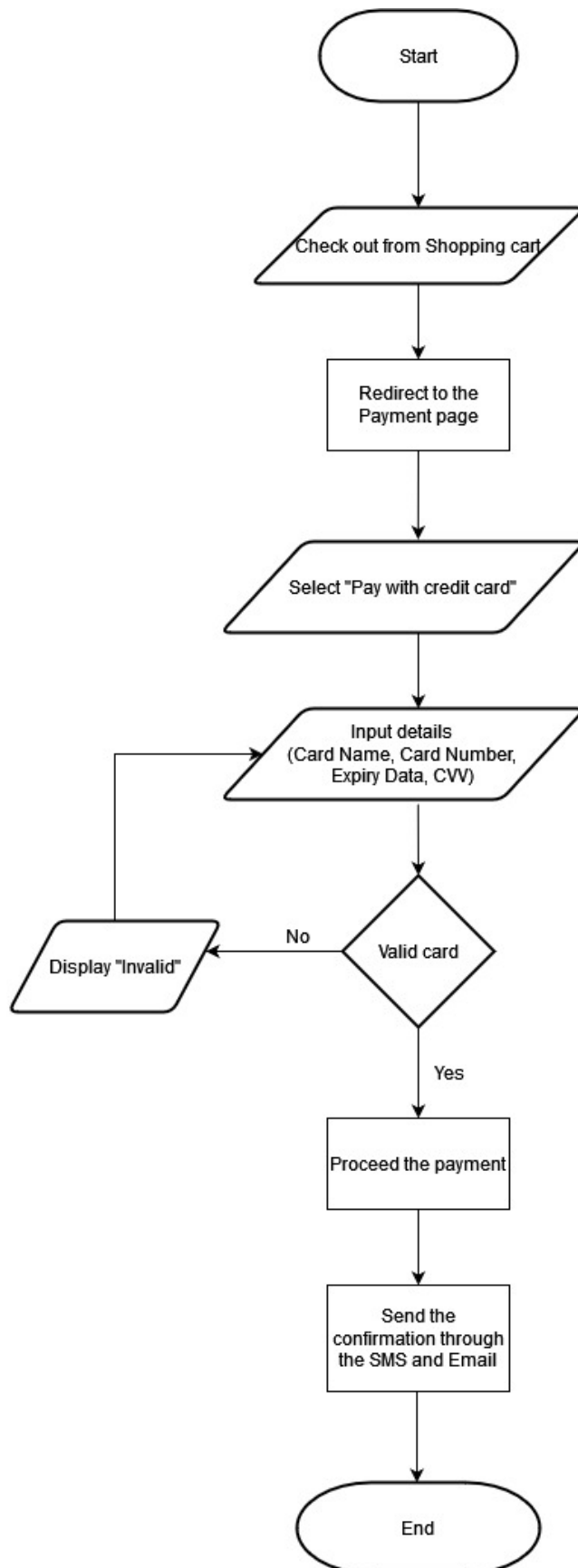


Add to cart:

When user logging to the system as a Buyer, he/she can buy several items. Simply user can click the “Add to cart” button then it will be added to the shopping cart. When user is in the shopping cart, user can remove that added items as well as user can shop more items.

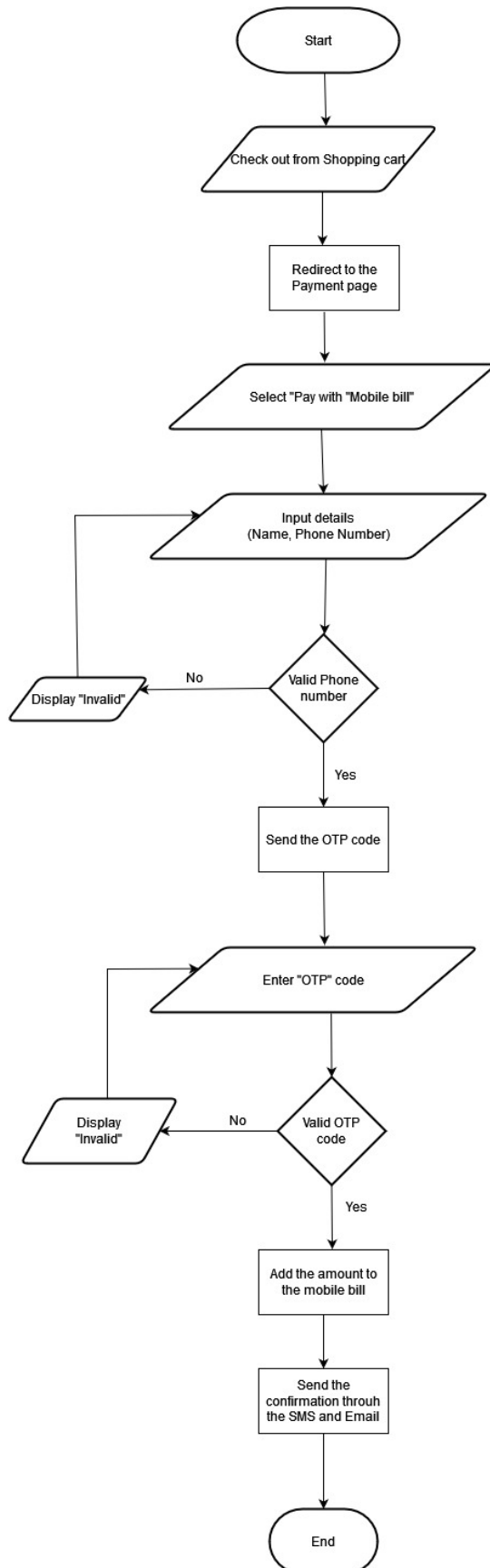
There are 2 types of payment methods.

1. Pay with Credit card
2. Pay with Mobile bill



Pay with Credit card:

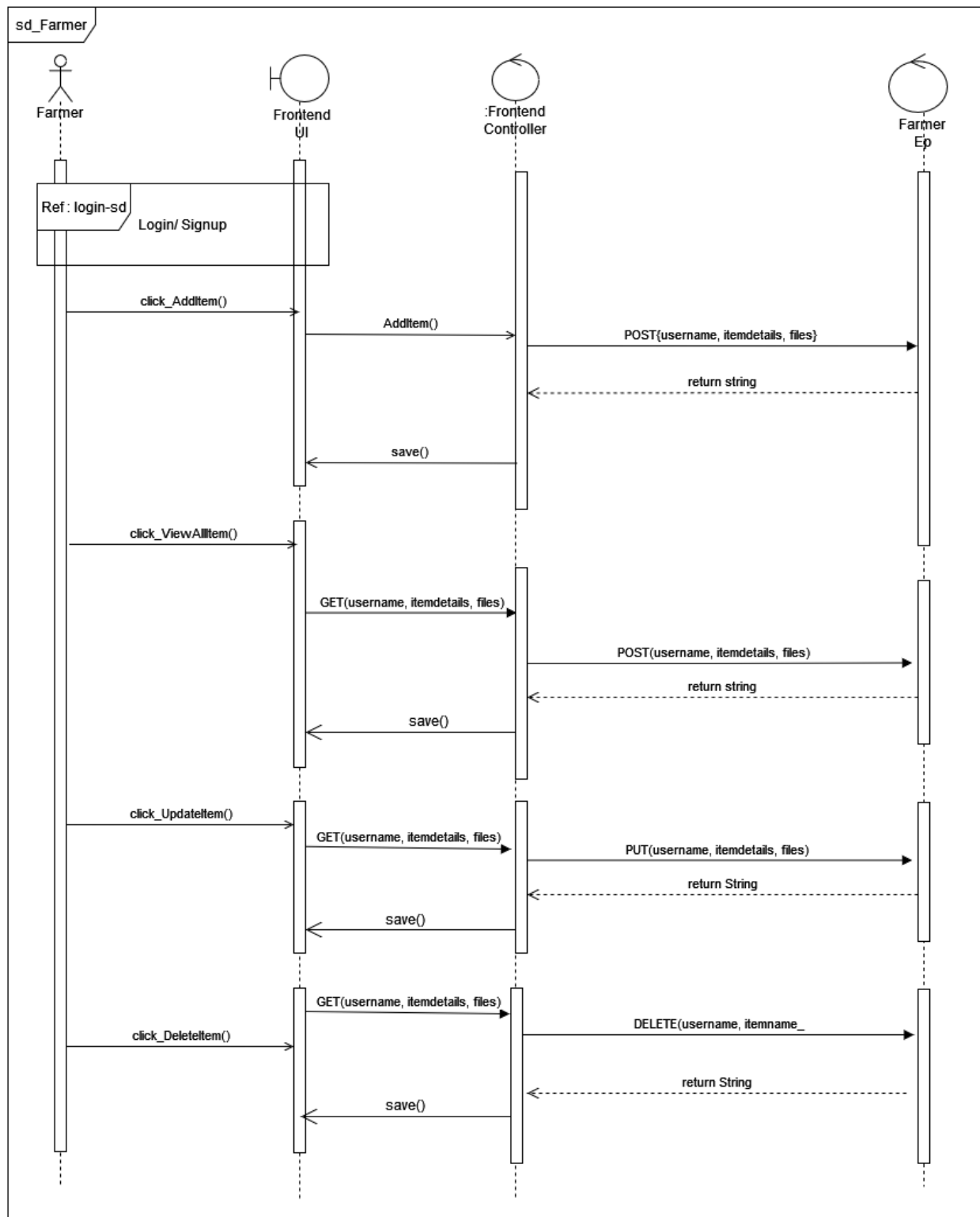
A dummy payment gateway has been implemented for this part. When user clicks “Pay with Credit card”, then user has to enter the Card name, Card number, Expiry date and CVV. Then it will validate the card number. After the validation then the amount will deduct from the dummy personal account and credited to the dummy business account. Then it will send the confirmation via SMS and email.



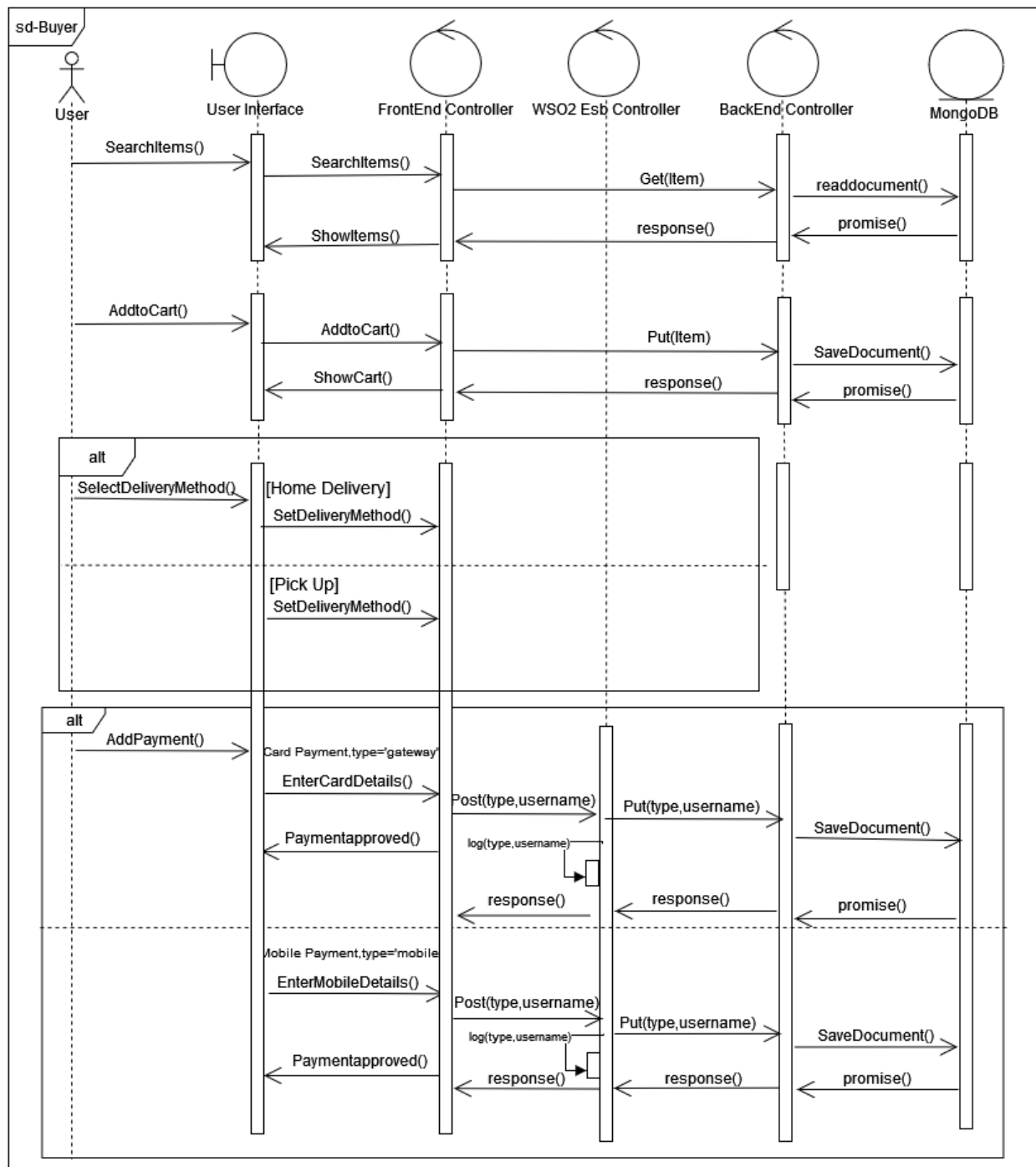
Request Delivery service:

After the payment process, system will request the Delivery Service. User can accept the request by giving Name, Address and Phone number or User can declined the service.

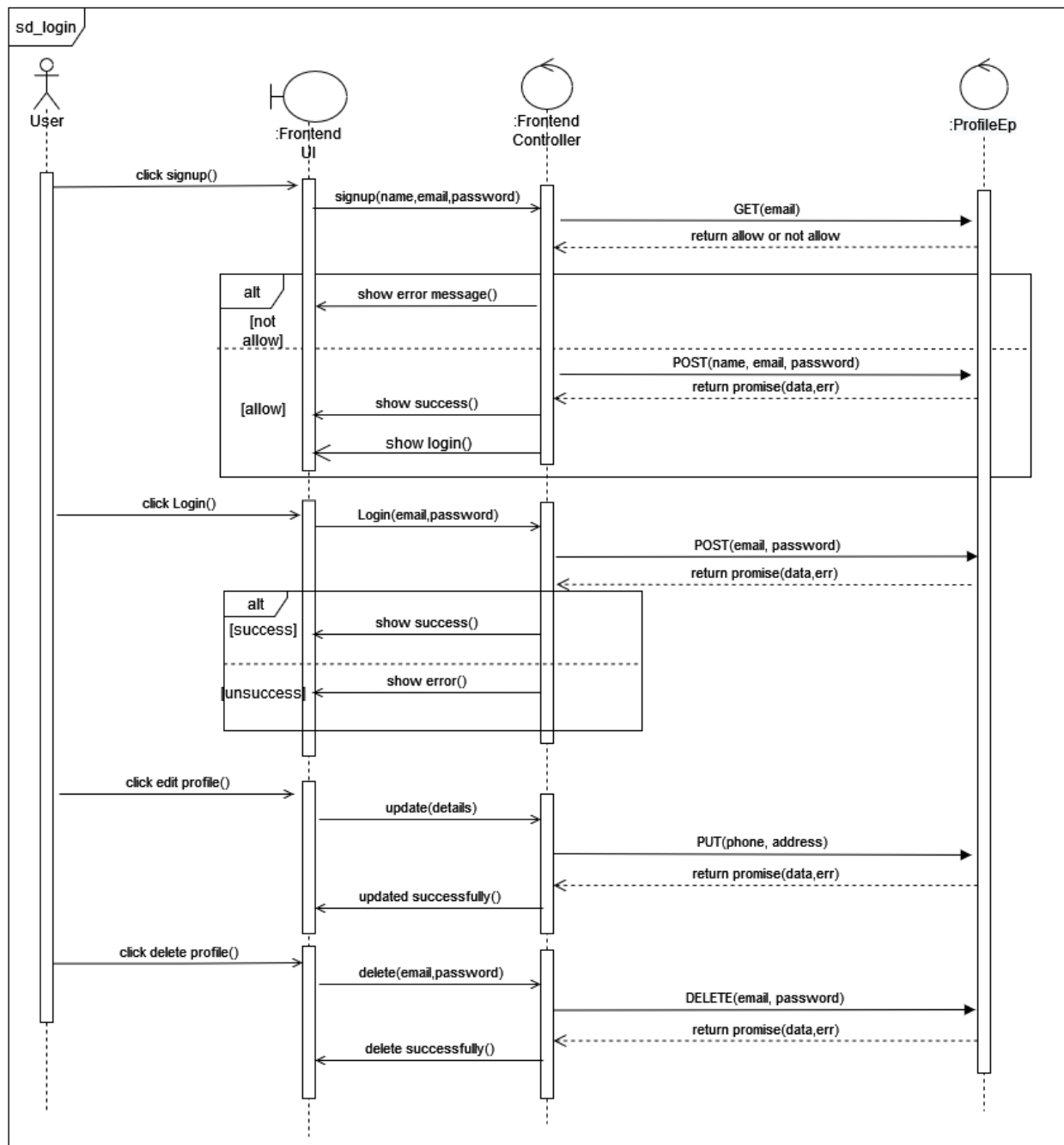
Service Diagram – Farmer



Service Diagram – Buyer



Service Diagram – login



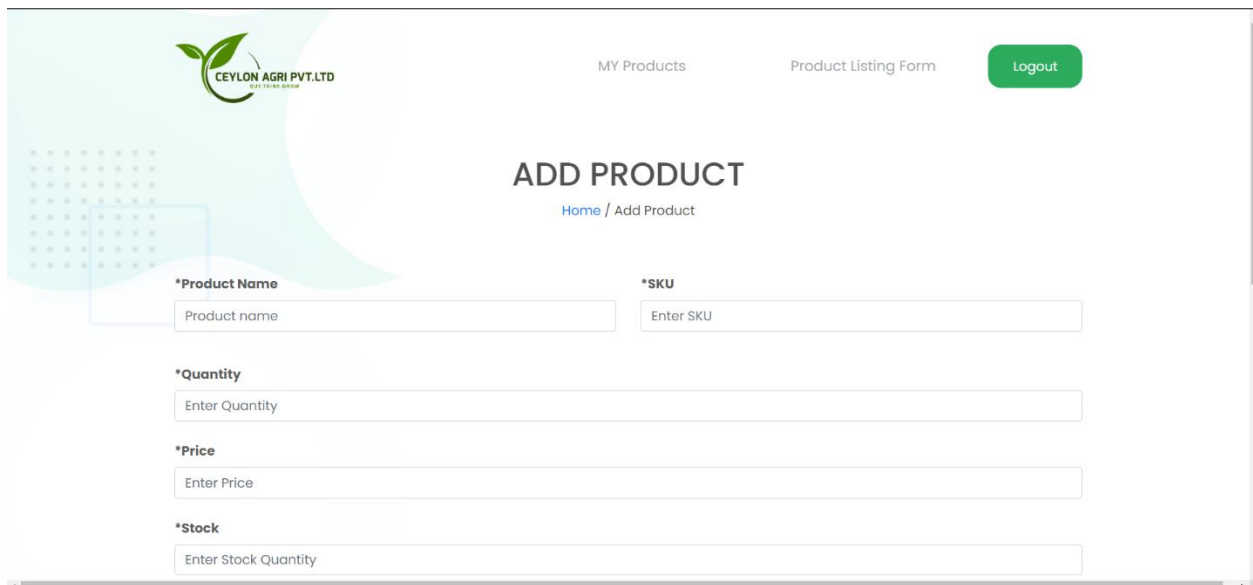
05. Functionalities

This system is a online purchasing platform where the farmers can manage their seller items and the buyers can buy the items. We have taken our site as an example of Simple Agri product online purchasing platform

5.1 Farmer

The farmer is like the producer of this system. They are the ones responsible for all manipulations of items. Manipulations include adding items, updating them or deleting them. Once an item is added, it can be viewed in the list of added items where there will be options to update or delete a selected one.

Add item01




The screenshot displays the 'ADD PRODUCT' interface of the Simple Agri system. The header includes the 'CEYLON AGRI PVT.LTD' logo, 'MY Products' and 'Product Listing Form' links, and a 'Logout' button. The main heading is 'ADD PRODUCT' with a breadcrumb trail 'Home / Add Product'. The form contains five input fields: '*Product Name' (labeled 'Product name'), '*SKU' (labeled 'Enter SKU'), '*Quantity' (labeled 'Enter Quantity'), '*Price' (labeled 'Enter Price'), and '*Stock' (labeled 'Enter Stock Quantity').

Add item02

Enter Product Description

Drag 'n' drop your image file here, or click to select files

File Details

 Add Product

©2022 **Ceylon Agri Pvt.Ltd.**, All Rights Reserved

[Back to Top](#) [Terms of Use](#) [Privacy](#)

List Item View



MY Products

Product Listing Form

Logout

MY PRODUCT LISTING

[Home](#) / [Add Product](#)





EDIT PRODUCT

[Home](#) / [My Products](#) / [Edit Product](#)

*Product Name

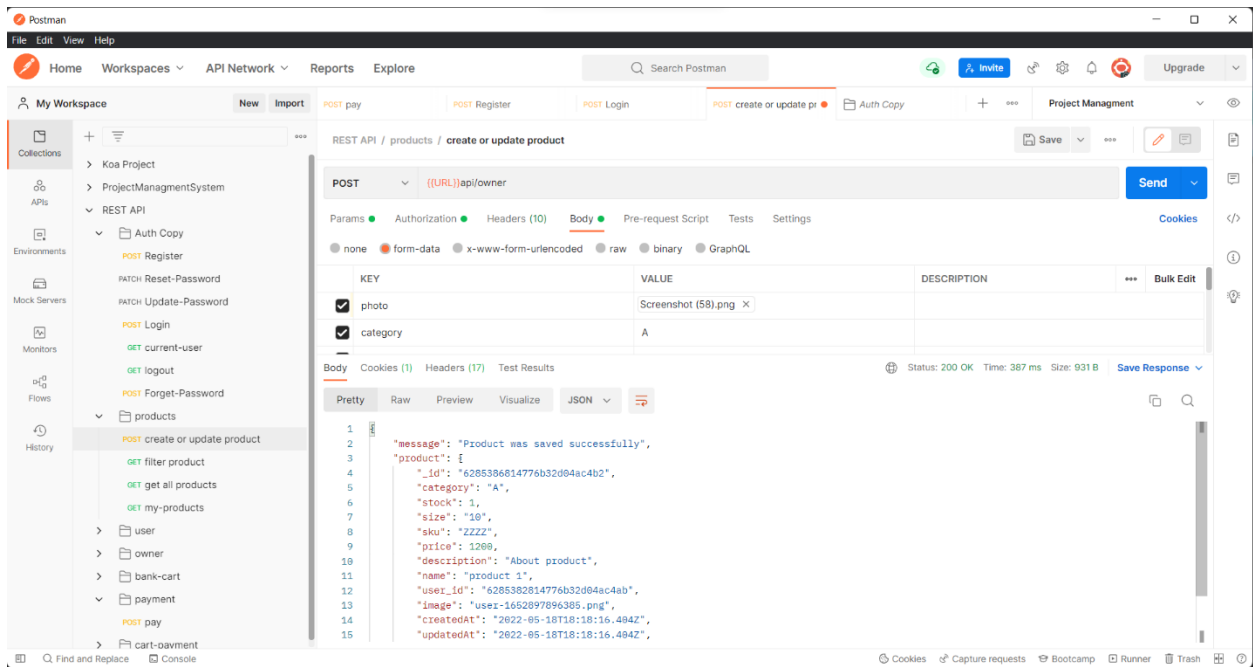
*SKU

*Quantity

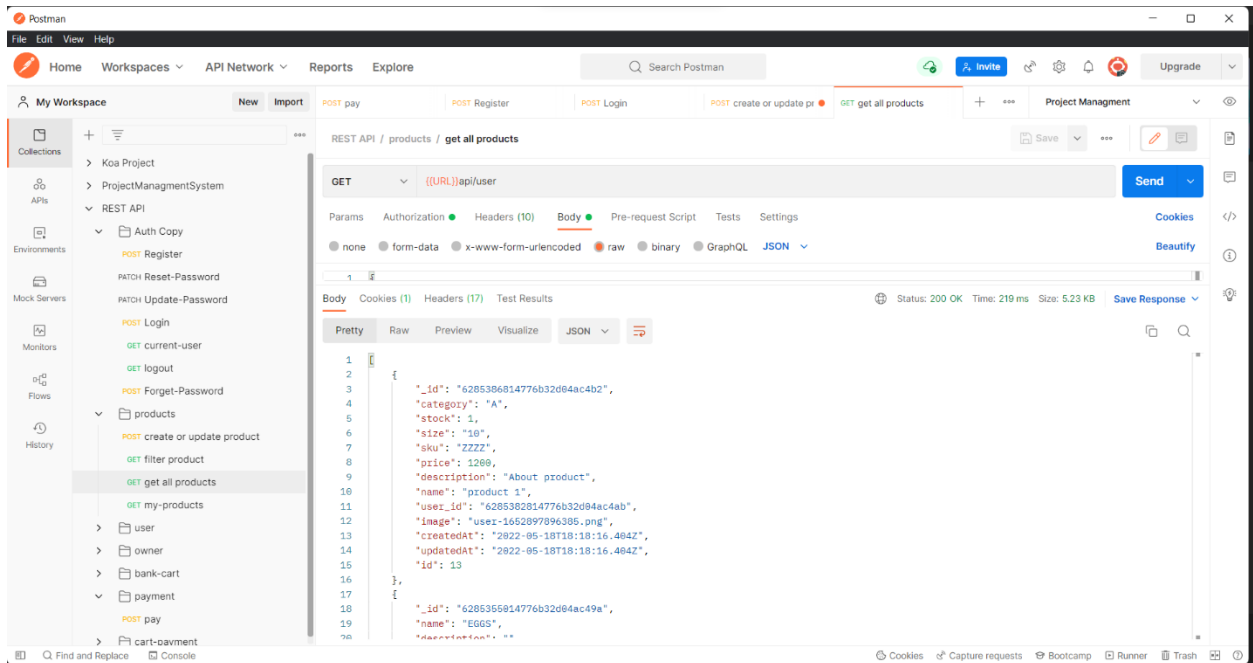
*Price

*Stock

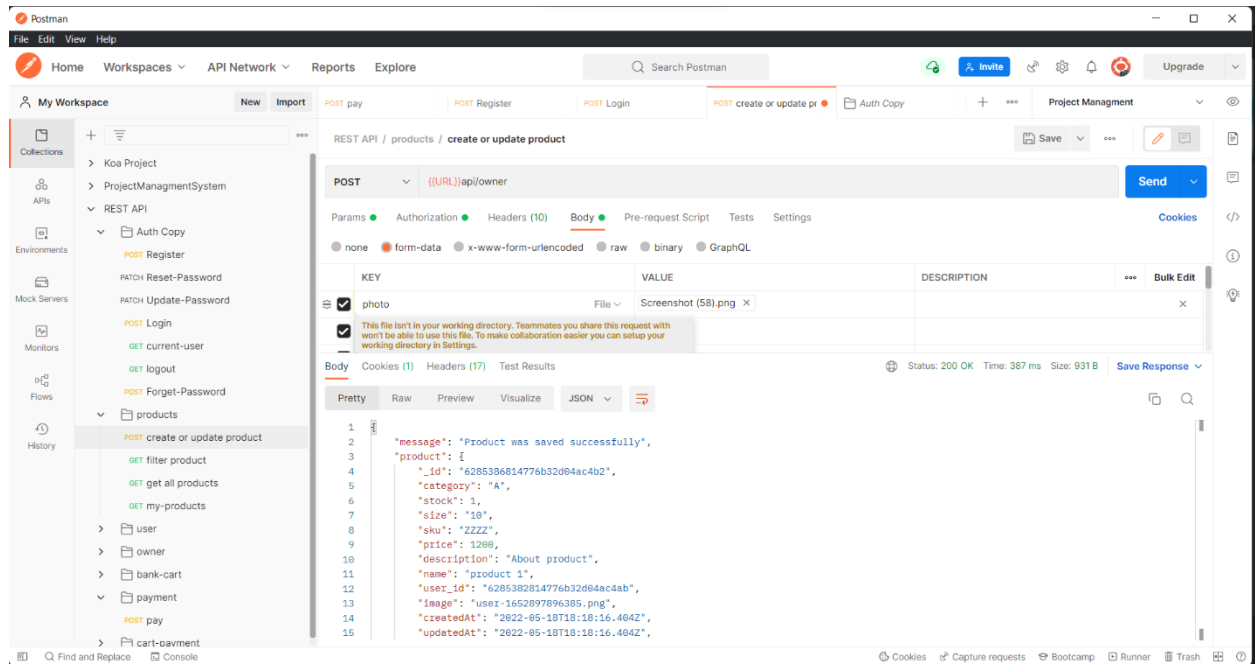
Data inserted successfully.



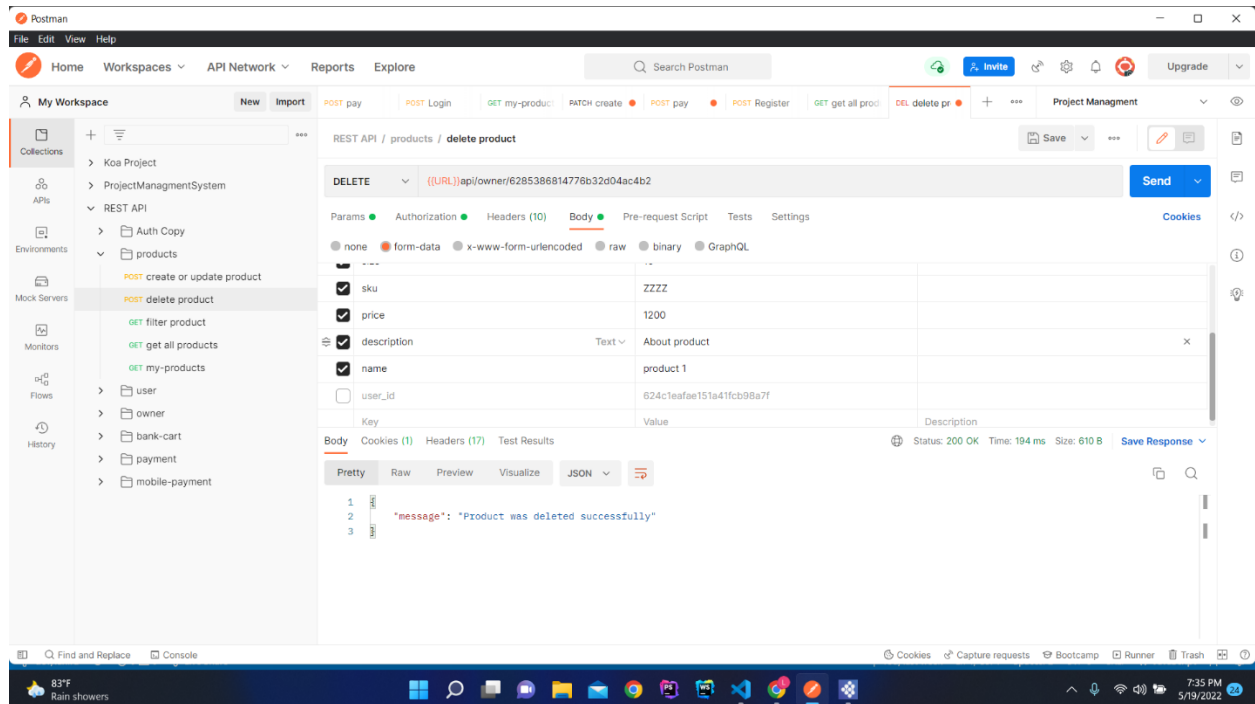
Data Retrieved successfully



Data Updated Successfully



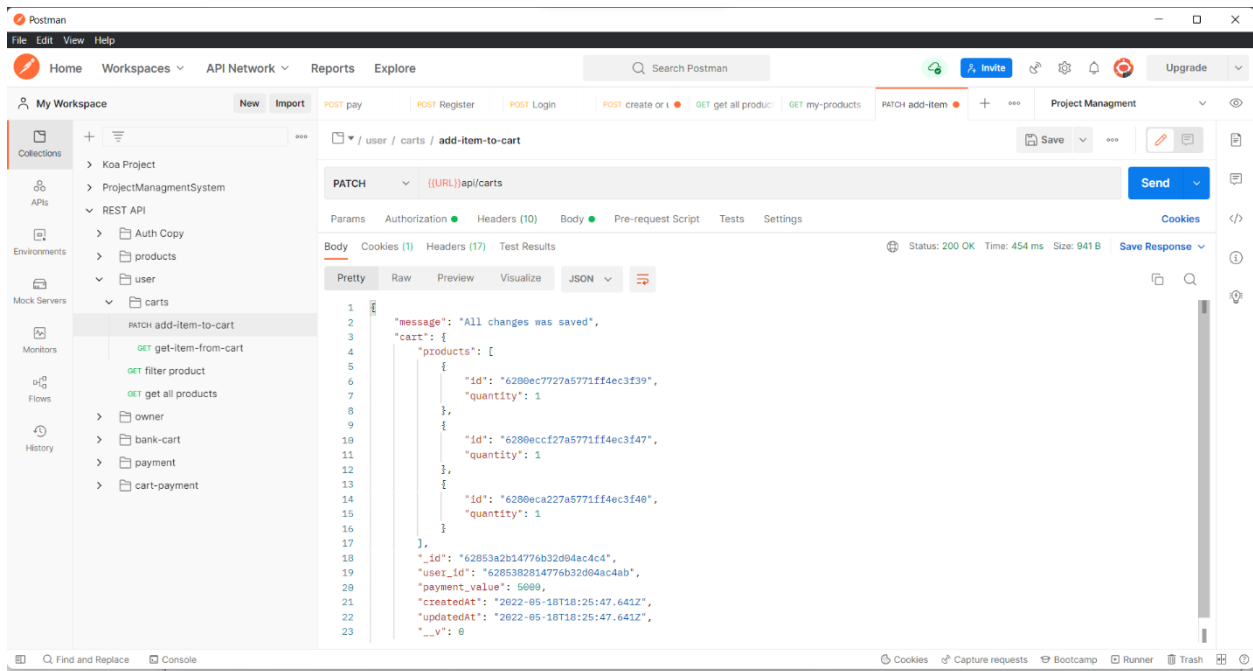
Data Deleted successfully



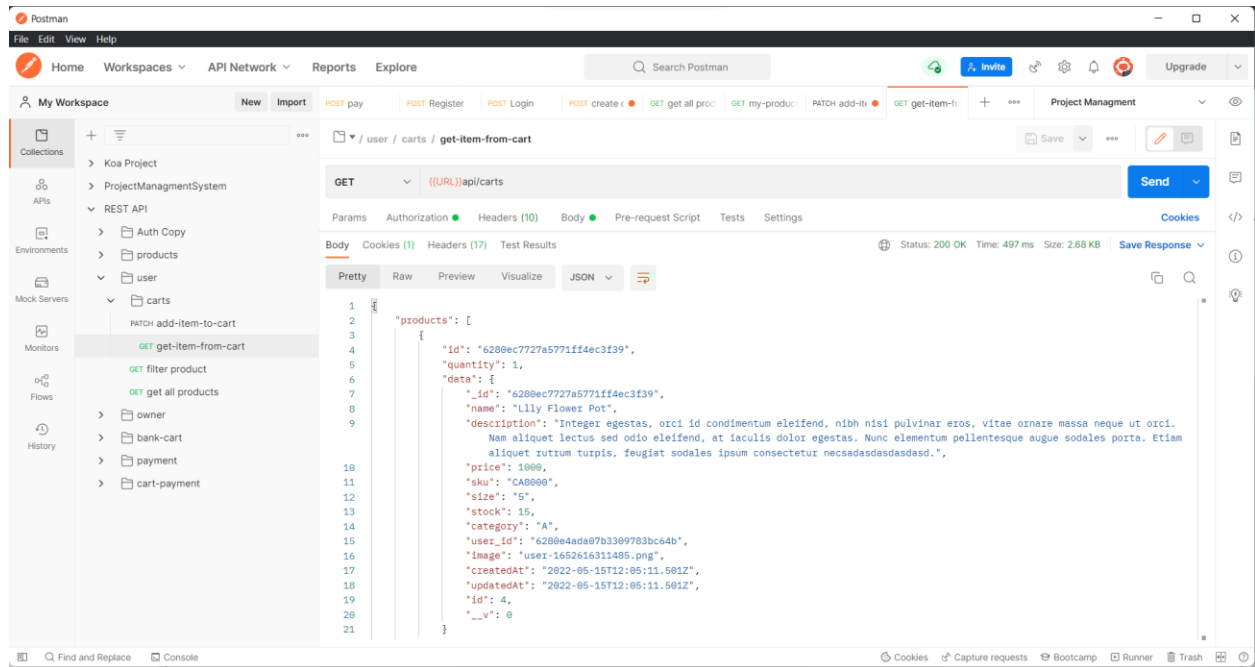
5.2 Buyer

Buyer is capable of browsing the items added by the farmers and then they can add the items they want to buy and the quantity they need to the cart. The users are also capable of viewing the items they have added to the cart, updating the cart and removing items added from the cart. Next the buyers can proceed to make the necessary payments by adding the card details.

Adding, Updating, and Deleting data from the cart

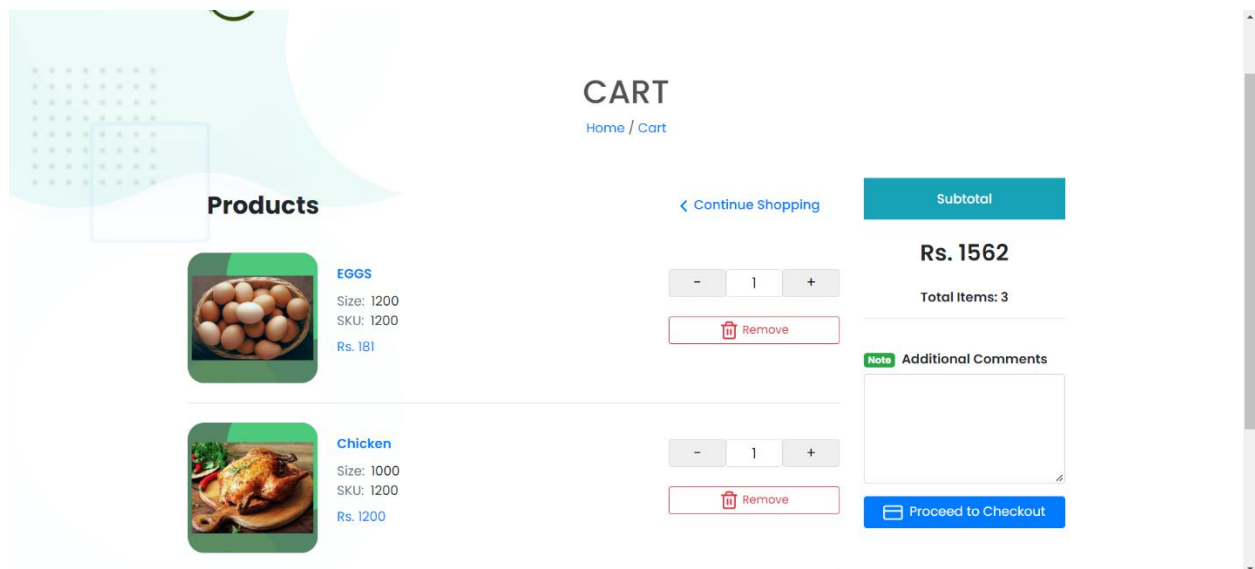


Retrieving data from the cart



5.3 Payment Gateway

Payment checkout interface



Appendix

Back End (server side implementation)

Server.js (main-services)

```
const mongoose = require('mongoose');
const dotenv = require('dotenv');
const cors = require("cors")

dotenv.config({ path: './config.env' });

const app = require('./app');
app.use(cors({
  origin: ["http://localhost:3000"],
  credentials: true
}));

const port = process.env.PORT || 3000;
const server = app.listen(port, () => {
  console.log(`App running on port ${port}...`);
});

const URL = process.env.DATABASE;

mongoose
  .connect(URL, {
    useNewUrlParser: true,
    useCreateIndex: true,
    useFindAndModify: false,
    useUnifiedTopology: true
  })
  .then(() => console.log('DB connection successful!'));
```

Server.Js (cart-payment-service)

```
const express = require("express");
const mongoose = require("mongoose");
const cookieParser = require("cookie-parser");
const cors = require("cors");
const dotenv = require("dotenv");
require("dotenv").config();

/*-----Set Up Server-----*/
-----*/
const app = express();
app.use(express.json());
app.use(cookieParser());
app.use(cors({
  origin: ["http://localhost:3000"],
  credentials: true
}));
dotenv.config({ path: './config.env' });

const PORT = process.env.PORT || 5000;
app.listen(PORT, () => {
  console.log(`Server start on port : ${PORT}`)
})

/*-----connect to mongoDB-----*/
-----*/
//Type 01
const URL= process.env.DATABASE;

mongoose
  .connect(URL,{
    useNewUrlParser: true,
    useCreateIndex: true,
    useFindAndModify: false,
    useUnifiedTopology: true
  })
  .then(() => console.log('DB connection successful!'));

/*-----Create Routes-----*/
-----*/
```

```
// Employee manager routes
app.use("/cart-payment", require("./Routes/GetwayRouter"));
```

Server.js (mobile-payment-service)

```
const express = require("express");
const mongoose = require("mongoose");
const cookieParser = require("cookie-parser");
const cors = require("cors")
const dotenv = require("dotenv");
require("dotenv").config();

/*-----Set Up Server-----*/
const app = express();
app.use(express.json());
app.use(cookieParser());
app.use(cors({
  origin: ["http://localhost:3000"],
  credentials: true
}));
dotenv.config({ path: './config.env' });

const PORT = process.env.PORT || 5000;
app.listen(PORT, () => {
  console.log(`Server start on port : ${PORT}`)
})

/*-----connect to mongoDB-----*/
//Type 01
const URL= process.env.DATABASE;

mongoose
```

```

    .connect(URL,{
      useUrlParser: true,
      useCreateIndex: true,
      useFindAndModify: false,
      useUnifiedTopology: true
    })
    .then(() => console.log('DB connection successful!'));

/*-----Create Routes-----
-----*/

// Employee manager routes
// app.use("/payment", require("../Routes/GetwayRouter"));

```

main-services

Controllers

authController.js

```

const User = require('../Models/userModel');
// const Group = require('../Models/groupModel');
const catchAsync = require('../Utils/catchAsync');
const jwt = require('jsonwebtoken');
const AppError = require('../Utils/appError');
const { promisify } = require('util');
const sendEmail = require('../Utils/email');
const crypto = require('crypto');

/**Auth controllers */

```



```

//ClientRegistration new user
exports.signup = catchAsync(async (req, res, next) => {
  const newUser = await User.create(req.body)
  createSendToken(newUser, 200, res);
});

//Login user
exports.login = catchAsync(async (req, res, next) => {
  const { email, password } = req.body;

  if (!email || !password) {
    return next(new AppError('Please provide email and password!', 400));
  }
  //Check if user exists && password is correct
  const user = await User.findOne({ email }).select('+password');

  if (!user || !(await user.correctPassword(password, user.password))) {
    return next(new AppError('Incorrect email or password', 401));
  }

  // // If everything ok, send token to client
  createSendToken(user, 200, res);
});

//forgot user password
exports.forgotPassword = catchAsync(async (req, res, next) => {
  // Get user based on POSTed email
  const user = await User.findOne({ email: req.body.email });
  if (!user) {
    return next(new AppError('There is no user with email address.', 404));
  }

  //Generate the random reset token
  const resetToken = user.createPasswordResetToken();
  await user.save({ validateBeforeSave: false });

  // Send it to user's email
  const resetURL = `${req.protocol}://${req.get('host')}/reset-
password/${resetToken}`;

```

```

    const message = `Forgot your password? Submit a PATCH request with your new
password and passwordConfirm to: ${resetURL}.\nIf you didn't forget your
password, please ignore this email!`;

    try {
      await sendEmail({
        email: req.body.email,
        subject: 'Your password reset token (valid for 10 min)',
        message
      });

      res.status(200).json({
        status: 'success',
        message: 'Email has been sent'
      });
    } catch (err) {
      user.passwordResetToken = undefined;
      user.passwordResetExpires = undefined;
      await user.save({ validateBeforeSave: false });

      return next(
        new AppError('There was an error sending the email. Try again later!'),
        500
      );
    }
  });
});

//Reset password
exports.resetPassword = catchAsync(async (req, res, next) => {
  // Get user based on the token
  const hashedToken = crypto
    .createHash('sha256')
    .update(req.params.token)
    .digest('hex');

  const user = await User.findOne({ passwordResetToken: hashedToken,
passwordResetExpires: { $gt: Date.now() } });

  //If token has not expired, and there is user, set the new password
  if (!user) {
    return next(new AppError('Token is invalid or has expired', 400));
  }
  user.password = req.body.password;
  user.passwordConfirm = req.body.passwordConfirm;

```

```

    user.passwordResetToken = undefined;
    user.passwordResetExpires = undefined;
    await user.save();

    // Log the user in, send JWT
    createSendToken(user, 200, res);
  });

//update password
exports.updatePassword = catchAsync(async (req, res, next) => {
  // Get user from collection
  const user = await User.findById(req.user.id).select('+password');

  //Check if POSTed current password is correct
  if (!(await user.correctPassword(req.body.old_password, user.password))) {
    return next(new AppError('Your current password is wrong.', 401));
  }

  // If so, update password
  user.password = req.body.password;
  user.passwordConfirm = req.body.passwordConfirm;
  await user.save();
  // User.findByIdAndUpdate will NOT work as intended!

  //Log user in, send JWT
  createSendToken(user, 200, res);
});

//get current user
exports.currentUser = catchAsync(async (req, res, next) => {
  res.status(200).json({
    status: 'success',
    data: req.user
  });
});

//logout user
exports.logout = catchAsync(async (req, res, next) => {
  res.cookie("jwt", "loggedout", {
    expires: new Date(Date.now() + 10 * 1000),
    httpOnly: true
  });
  res.status(200).json({

```

```

        status: 'success'
    })
});

/**End of the modules controllers */

/**Below functions are some middlewares */

//created token
const signToken = id => {
    return jwt.sign({ id }, process.env.JWT_SECRET, {
        expiresIn: process.env.JWT_EXPIRES_IN
    });
};

//crate new token and send it
const createSendToken = (user, statusCode, res) => {
    const token = signToken(user._id);
    const cookieOptions = {
        expires: new Date(Date.now() + 900000),
        // maxAge: new Date(
        //   Date.now() + process.env.JWT_COOKIE_EXPIRES_IN * 24 * 60 * 60 * 1000
        // ),
        httpOnly: true
    };
    if (process.env.NODE_ENV === 'production') cookieOptions.secure = true;

    res.cookie('jwt', token, cookieOptions);

    // Remove password from output
    user.password = undefined;

    res.status(statusCode).json({
        status: 'success',
        token,

```

```

    data: {
      user
    }
  });
};

//protected routes
exports.protect = catchAsync(async (req, res, next) => {
  // Getting token and check of it's there
  let token;
  if (
    req.headers.authorization &&
    req.headers.authorization.startsWith('Bearer')
  ) {
    token = req.headers.authorization.split(' ')[1];
  }

  if (!token) {
    return next(
      new AppError('You are not logged in! Please log in to get access.', 401)
    );
  }

  //Verification token. This will return promise. So, I just used promisify
  inbuilt method
  const decoded = await promisify(jwt.verify)(token, process.env.JWT_SECRET);

  // Check if user still exists
  const currentUser = await User.findById(decoded.id);
  if (!currentUser) {
    return next(
      new AppError(
        'The user belonging to this token does no longer exist.',
        401
      )
    );
  }

  //Check if user changed password after the token was issued
  if (currentUser.changedPasswordAfter(decoded.iat)) {
    return next(
      new AppError('User recently changed password! Please log in again.', 401)
    );
  }
}

```

```

    // GRANT ACCESS TO PROTECTED ROUTE AND SET USER AND GROUP ID GLOBALLY
    req.user = currentUser;
    // req.group = await Group.findById(currentUser.groupID)
    next();
  });

  //Give permission each protected route to access
  exports.restrictTo = (...roles) => {
    return (req, res, next) => {
      if (!roles.includes(req.user.role)) {
        return next(
          new AppError('You do not have permission to perform this action', 403)
        );
      }

      next();
    };
  };

  /**end of the middlewares */

```

cartController.js

```

const Cart = require("../Models/Cart");
const Product = require("../Models/productModel");
const OrderValidator = require("../validators/orderValidator");

// exports.addToCart = async (req, res) => {
//   try {
//     // validate the products in the cart
//     // use the validator used to validate order products
//     const validatedOrder = await OrderValidator.ValidateOrderProducts(req,
res);
//     console.log(validatedOrder)
//     let cart = await Cart.findOne({ user_id: req.user._id });
//
//     if (!cart) {
//       // if-the user does not have a cart
//       cart = new Cart({
//         user_id: req.user._id,
//         products: validatedOrder.products,

```

```

//      payment_value: validatedOrder.payment_value,
//    });
//  } else {
//    // if the user already have a cart
//    cart.products = [...cart.products, ...validatedOrder.products];
//    cart.payment_value = cart.payment_value + validatedOrder.payment_value;
//  }
//  const result = await cart.save();
//
//  // if cart save fail
//  if (result && result.error) return res.status(400).json(result);
//
//  return res
//    .status(200)
//    .json({ message: "All changes was saved", cart: result._doc });
//  } catch (error) {
//    console.error(error);
//    return res.status(400).json({ message: "Unexpected error" });
//  }
// };

// get cart of the logged-in user
exports.getCart = async (req, res) => {
  try {
    const cart = await Cart.findOne({ user_id: req.user._id });
    // if the cart exist and if the cart has products, get the products from the
    // cart
    if (cart && cart.products.length > 0) {
      for (let index = 0; index < cart.products.length; index++) {
        try {
          // add all the product data
          cart.products[index].data = await Product.findById(
            cart.products[index].id
          );
        } catch (error) {
          console.log(error);
        }
      }
    }
    return res.status(200).json(cart ? cart : {});
  } catch (error) {
    console.error(error);
    return res.status(400).json({ message: "User cart not found" });
  }
};

```

```

// add, update or delete a product in cart
exports.storeToCart = async (req, res) => {
  try {
    let validatedCart = {};
    // validate the products in the cart if it has products only
    if (
      req.body.products &&
      Array.isArray(req.body.products) &&
      req.body.products.length > 0
    ) {

      validatedCart = await OrderValidator.ValidateOrderProducts(req, res);
      console.log(validatedCart);
    }
    else {
      // no cart products means that user has deleted all the products from the
      // we can allow it because the cart can be empty
      validatedCart.products = [];
      validatedCart.payment_value = 0;
    }

    if(!validatedCart.products){

  }else {
    // get cart details
    let cart = await Cart.findOne({ user_id: req.user._id });
    if (!cart) {
      // if-the user does not have a cart
      cart = new Cart({
        user_id: req.user._id,
        products: validatedCart.products,
        payment_value: validatedCart.payment_value,
      });
    } else {
      // if the user already have a cart
      cart.products = validatedCart.products;
      cart.payment_value = validatedCart.payment_value;
    }
    // save cart
    const result = await cart.save();
    if (result && result.error) return res.status(400).json(result);
    return res
  }
}

```



```

        .status(200)
        .json({ message: "All changes was saved", cart: result._doc });
    }

    } catch (error) {
        console.error(error);
        return res.status(400).json({ message: "Invalid cart details" });
    }
};

```

productController.js

```

const Product = require("../Models/productModel");
const Filters = require("../Utils/filters");
const User = require("../Models/userModel");
const FileUpload = require("../Utils/fileUpload");
const multer = require("multer");

const multerStorage = FileUpload.setPath('public/img/product')
const multerFilter = FileUpload.FileTypeFilter('image')

const upload = multer({
    storage: multerStorage ,
    fileFilter: multerFilter
});

exports.uploadProductPhoto = upload.single('photo');

// list all the products
exports.listProducts = async (req, res) => {
    try {
        const Respond = new Filters(Product.find(),
req.query).filter().sort().limitFields().paginate();
        const products = await Respond.query;
        return res.status(200).json(products);
    } catch (error) {
        console.error(error);
        return res.status(400).json({ message: "Error when retrieving products" });
    }
};

```

```

// list all the products
exports.listMyProducts = async (req, res) => {
  try {
    console.log(req.user._id)
    let userID = req.user._id
    const Respond = new Filters(Product.find({user_id:userID}),
req.query).filter().sort().limitFields().paginate();
    const products = await Respond.query;
    return res.status(200).json(products);
  } catch (error) {
    console.error(error);
    return res.status(400).json({ message: "Error when retrieving products" });
  }
};

// find product based on id
exports.findProduct = async (req, res) => {
  try {
    let product = await Product.findById(req.params.id);
    if (product) return res.status(200).json(product);
    else return res.status(400).json({ message: "Product not found" });
  } catch (error) {
    console.error(error);
    return res.status(400).json(error);
  }
};

// create / update a product
exports.saveProduct = async (req, res) => {
  try {
    req.body.user_id = req.user._id
    req.body.image = req.file.filename
    console.log(req.file)
    const new_product = await Product.create(req.body)
    return res.status(200).json({
      message: "Product was saved successfully",
      product: new_product._doc,
    });
  } catch (error) {
    console.error(error);
    return res.status(400).json({ message: error.message});
  }
};

```

```

exports.updateProduct = async (req, res) => {
  try {
    const updated_product = await Product.findByIdAndUpdate(req.params.id,
req.body)
    return res.status(200).json({
      message: "Product was updated successfully",
      product: updated_product._doc,
    });

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

// delete a product
exports.deleteProduct = async (req, res) => {
  try {
    await Product.findByIdAndDelete(req.params.id);
    return res
      .status(200)
      .json({ message: "Product was deleted successfully" });
  } catch (error) {
    console.error(error);
    return res.status(400).json(error);
  }
};

```

UserController.js

```

const User = require('../Models/userModel');
const catchAsync = require('../Utils/catchAsync');
const AppError = require('../Utils/appError');
const Filters = require('../Utils/filters');
const FileUpload = require('../Utils/fileUpload');
const multer = require('multer');

/**Upload a profile picture */

```

```
const multerStorage = FileUpload.setPath('public/img/users')
const multerFilter = FileUpload.FileTypeFilter('image')

const upload = multer({
  storage: multerStorage ,
  fileFilter: multerFilter
});

exports.uploadUserPhoto = upload.single('photo');

/**All users apis */

//update user
exports.updateMe = catchAsync(async (req, res, next) => {

  // Filtered out unwanted fields names that are not allowed to be updated
  const filteredBody = filterObj(req.body, 'name', 'email');

  //If update photo it will updated
  if (req.file) filteredBody.photo = req.file.filename;

  // Update user document
  const updatedUser = await User.findByIdAndUpdate(req.user.id, filteredBody, {
    new: true,
    runValidators: true
  });

  res.status(200).json({
    status: 'success',
    data: {
      user: updatedUser
    }
  });
});

//delete my account
exports.deleteMe = catchAsync(async (req, res, next) => {
  await User.findByIdAndUpdate(req.user.id, { active: false });
});
```

```

    res.status(204).json({
      status: 'success',
      data: null
    });
  });

  /**End of the user apis */

  /**Some conditions */

  //filter and return column that needed to be updated
  const filterObj = (obj, ...allowedFields) => {
    const newObj = {};
    Object.keys(obj).forEach(e1 => {
      if (allowedFields.includes(e1)) newObj[e1] = obj[e1];
    });
    return newObj;
  };
};

```

Models

BankCartModel.js

```

const { model, Schema } = require("mongoose");

const bankCartSchema = new Schema(
  {
    card_no: { type: Number, required: [true, "Please enter a card no"], },
    user_id: { type: String, required: true },
    card_cvc: { type: Number, required: [true, "Please enter a card cvc"], },
    card_holder_name: { type: String, required: [true, "Please enter a card holder name"], },
    balance: { type: Number, default: 10000 },
  },
  { timestamps: true }
);

module.exports = model("Bank_Cart", bankCartSchema);

```

Cart.js

```
const { model, Schema } = require("mongoose");

const cartSchema = new Schema({
  user_id: { type: String, required: true },
  products: { type: Array, required: true },
  payment_value: { type: Number, required: false },
}, { timestamps: true });

module.exports = model("cart", cartSchema);
```

productModel.js

```
const { model, Schema } = require("mongoose");
const autoIncrement = require("mongoose-auto-increment");
const mongoose = require("mongoose");

const productSchema = new Schema(
  {
    name: {
      type: String,
      required: [true, "Please enter a valid product name"],
    },
    image: {
      type: String,
      required: [true, "Please enter a valid product name"],
    },
    description: {
      type: String,
    },
    category: {
      type: String,
      enum: ['A', 'B', 'C', 'D'],
      required: [true, "Please enter a valid product category"],
    },
    price: {
      type: Number,
      required: [true, "Please enter a valid product price"],
    },
    sku: {
      type: String,
      required: [true, "Please enter a valid product sku"],
    },
    size: {
```

```

        type: String,
        required: [true, "Please enter a valid product size"],
      },
      stock: {
        type: Number,
        required: [true, "Please enter a valid product name"],
      },
      user_id: {
        type: String,
        required: [true, "Please enter a user id"],
      },
    },
    { timestamps: true }
  );
  autoIncrement.initialize(mongoose.connection);

  productSchema.plugin(autoIncrement.plugin, {
    model: "product", // collection or table name in which you want to apply auto
    increment
    field: "id", // field of model which you want to auto increment
    startAt: 0, // start your auto increment value from 1
    incrementBy: 1, // incremented by 1
  });
  productSchema.index({ name: "text", description: "text" });

  module.exports = model("product", productSchema);

```

userModel.js

```

const mongoose = require('mongoose');
const validator = require('validator');
const bcrypt = require('bcryptjs');
const crypto = require('crypto');

const userSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, 'Please tell us your name!']
  },
  email: {
    type: String,
    required: [true, 'Please provide your email'],
    unique: true,
    lowercase: true,

```

```

        validate: [validator.isEmail, 'Please provide a valid email']
    },
    photo: {
        type: String,
        default: 'default.png'
    },
    role: {
        type: String,
        enum: ['buyer', 'owner'],
        default: 'buyer'
    },
    password: {
        type: String,
        required: [true, 'Please provide a password'],
        minlength: 8,
        select: false
    },
    passwordConfirm: {
        type: String,
        required: [true, 'Please confirm your password'],
        validate: {
            // This only works on CREATE and SAVE!!!
            validator: function (el) {
                return el === this.password;
            },
            message: 'Passwords are not the same!'
        }
    },
    passwordChangedAt: Date,
    passwordResetToken: String,
    passwordResetExpires: Date,
});

```

```

/**Middleware */

```

```

//This middleware performs encrypt password to save the database
userSchema.pre('save', async function (next) {
    // Only run this function if password was actually modified
    if (!this.isModified('password')) return next();

    // Hash the password with cost of 12

```



```

    this.password = await bcrypt.hash(this.password, 12);

    // Delete passwordConfirm field
    this.passwordConfirm = undefined;
    next();
  });

  // //use this query to filter all users whose active status only true
  // userSchema.pre(/^find/, function (next) {
  //   // this points to the current query
  //   this.find({ active: { $ne: false } });
  //   next();
  // });

  //If user change password passwordChangedAt column will update. otherwise no.
  userSchema.pre('save', function (next) {
    if (!this.isModified('password') || this.isNew) return next();

    this.passwordChangedAt = Date.now() - 1000;
    next();
  });
  /**End of the middleware */

  //Check passwords is correct or incorrect
  userSchema.methods.correctPassword = async function (candidatePassword,
  userPassword) {
    return await bcrypt.compare(candidatePassword, userPassword);
  };

  //return true or false by checking change password time and jwt token issued
  time. JWTTimestamp will return time that token was issued
  userSchema.methods.changedPasswordAfter = function (JWTTimestamp) {
    if (this.passwordChangedAt) {
      const changedTimestamp = parseInt(this.passwordChangedAt.getTime() /
1000, 10);

      return JWTTimestamp < changedTimestamp;
    }

    // False means NOT changed
    return false;
  };

```

```

//create token for reset password
userSchema.methods.createPasswordResetToken = function () {
  const resetToken = crypto.randomBytes(32).toString('hex');

  this.passwordResetToken =
crypto.createHash('sha256').update(resetToken).digest('hex');

  console.log({ resetToken }, this.passwordResetToken);

  //set expiration time for rest password token
  this.passwordResetExpires = Date.now() + 10 * 60 * 1000;

  return resetToken;
};

const User = mongoose.model('User', userSchema);
module.exports = User;

```

Routes

authRoutes.js

```

const express = require('express');
const authController = require('../Controllers/authController');
const userController = require('../Controllers/userController');
const router = express.Router();

router.post('/signup',userController.uploadUserPhoto, authController.signup);
router.post('/signing', authController.login);
router.post('/forget-password', authController.forgotPassword);
router.patch('/reset-password/:token', authController.resetPassword);
router.patch('/update-password', authController.protect,
authController.updatePassword);
router.get('/current-user', authController.protect, authController.currentUser);
router.get('/logout', authController.protect, authController.logout);

module.exports = router;

```

bankCartRoute.js

```
const express = require('express');
const bankCardController = require('../Controllers/banckCardController');
const authController = require("../Controllers/authController");
const router = express.Router();

router.post('/', authController.protect, bankCardController.save);
router.delete('/:id', authController.protect, bankCardController.removeCart);

module.exports = router;
```

cartRoutes.js

```
const express = require('express');
const cartController = require('../Controllers/cartController');
const authController = require('../Controllers/authController');
const router = express.Router();

//This api-resource route for update and delete specific student
router.route('/')
  .get(authController.protect, cartController.getCart)
  // .post(authController.protect, cartController.addToCart)
  .patch(authController.protect, cartController.storeToCart);

// //This api-resource route for update and delete specific student
// router.route('/:id')
//   .get(authController.protect, authController.restrictTo('owner'),
// productController.findProduct)
//   .patch(authController.protect, authController.restrictTo('owner'),
// productController.updateProduct)
//   .delete(authController.protect, authController.restrictTo('owner'),
// productController.deleteProduct)

module.exports = router;
```

ownerRoute.js

```
const express = require('express');
const productController = require('../Controllers/productController');
const authController = require('../Controllers/authController');
const userController = require("../Controllers/userController");
const router = express.Router();

//This api-resource route for update and delete specific student
router.route('/')
  .get(authController.protect, authController.restrictTo('owner'),
productController.listMyProducts)
  .post(authController.protect, authController.restrictTo('owner'),
productController.uploadProductPhoto, productController.saveProduct);

//This api-resource route for update and delete specific student
router.route('/:id')
  .get(authController.protect, productController.findProduct)
  .patch(authController.protect, authController.restrictTo('owner'),
productController.updateProduct)
  .delete(authController.protect, authController.restrictTo('owner'),
productController.deleteProduct)

module.exports = router;
```

userRoute.js

```
const express = require('express');
const productController = require('../Controllers/productController');
```

```
const authController = require('../Controllers/authController');
const userController = require("../Controllers/userController");
const router = express.Router();

//This api-resource route for update and delete specific student
router.route('/')
  .get(authController.protect, productController.listProducts)

module.exports = router;
```

Utils

appError.js

```
class AppError extends Error {
  constructor(message, statusCode) {
    super(message);

    this.statusCode = statusCode;
    this.status = `${statusCode}`.startsWith('4') ? 'fail' : 'error';
    this.isOperational = true;

    Error.captureStackTrace(this, this.constructor);
  }
}

module.exports = AppError;
```

catchAsync.js

```
module.exports = fn => {
  return (req, res, next) => {
    fn(req, res, next).catch(next);
  };
}
```

```
};  
};
```

email.js

```
const nodemailer = require('nodemailer');  
  
const sendEmail = async options => {  
  // 1) Create a transporter  
  const transporter = nodemailer.createTransport({  
    host: process.env.EMAIL_HOST,  
    port: process.env.EMAIL_PORT,  
    auth: {  
      user: process.env.EMAIL_USERNAME,  
      pass: process.env.EMAIL_PASSWORD  
    }  
  });  
  
  // 2) Define the email options  
  const mailOptions = {  
    from: 'Kushnara Siriwardhana <kushnara@jonas.io>',  
    to: options.email,  
    subject: options.subject,  
    text: options.message  
    // html:  
  };  
  
  // 3) Actually send the email  
  await transporter.sendMail(mailOptions);  
};
```

```
module.exports = sendEmail;
```

fileUpload.js

```
const multer = require('multer');
const AppError = require('../Utils/appError');

//Set path file to save
exports.setPath = (path) => multer.diskStorage({

  destination: (req, file, next) => {
    console.log('ava')
    console.log(file)
    next(null, `${path}`);
  },
  filename: (req, file, next) => {
    const ext = file.mimetype.split('/')[1];
    next(null, `user-${Date.now()}.${ext}`);
  }
});

//Filter file type
exports.FileTypeFilter = (type) => (req, file, next) => {
  console.log('ava')
  console.log(file)
  if (file.mimetype.startsWith(`${type}`)) {
    next(null, true);
  } else {
    next(new AppError(`Not an ${type}! Please upload only images.`, 400), false);
  }
};
```

filters.js

```
class Filters {
  constructor(query, queryString) {
    this.query = query;
    this.queryString = queryString;
  }

  filter() {
    const queryObj = { ...this.queryString };
    const excludedFields = ['page', 'sort', 'limit', 'fields'];
    excludedFields.forEach(el => delete queryObj[el]);

    // 1B) Advanced filtering
    let queryStr = JSON.stringify(queryObj);
    queryStr = queryStr.replace(/\b(gte|gt|lte|lt)\b/g, match => `>${match}<`);

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

    return this;
  }

  sort() {
    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;
  }

  limitFields() {
    if (this.queryString.fields) {
      const fields = this.queryString.fields.split(',').join(' ');
      this.query = this.query.select(fields);
    } else {
      this.query = this.query.select('-__v');
    }

    return this;
  }

  paginate() {
```



```

    const page = this.queryString.page * 1 || 1;
    const limit = this.queryString.limit * 1 || 100;
    const skip = (page - 1) * limit;

    this.query = this.query.skip(skip).limit(limit);

    return this;
  }
}
module.exports = Filters;

```

updateColunFilter.js

```

//filter and return column that needed to be updated
exports.filterObj = (obj, ...allowedFields) => {
  const newObj = {};
  Object.keys(obj).forEach(el => {
    if (allowedFields.includes(el)) newObj[el] = obj[el];
  });
  return newObj;
};

```

app.js

```

const express = require('express');
const morgan = require('morgan');
const rateLimit = require('express-rate-limit');
const helmet = require('helmet');
const mongoSanitize = require('express-mongo-sanitize');
const xss = require('xss-clean');
const hpp = require('hpp');

const AppError = require('./Utils/appError');

const authRouter = require('./Routes/authRoutes');
const userRoute = require('./Routes/userRoute');
const ownerRoute = require('./Routes/ownerRoute');
const cartRoutes = require('./Routes/cartRoutes');

```

```
const bankCartRoute = require('./Routes/bankCartRoute');
const cors = require("cors");
const path = require("path");

const app = express();
app.use(cors({
  origin:["http://localhost:3000"],
  credentials:true
}));
//GLOBAL MIDDLEWARES

//This is used to save image
app.use('/', express.static(path.join(__dirname, '/public')));

// Set security HTTP headers
app.use(helmet());

// Development logging
if (process.env.NODE_ENV === 'development') {
  app.use(morgan('dev'));
}

// Limit requests from same API
const limiter = rateLimit({
  max: 100,
  windowMs: 60 * 60 * 1000,
  message: 'Too many requests from this IP, please try again in an hour!'
});
app.use('/api', limiter);

// Body parser, reading data from body into req.body
app.use(express.json({ limit: '10kb' }));

// Data sanitization against NoSQL query injection
app.use(mongoSanitize());

// Data sanitization against XSS
app.use(xss());

// Prevent parameter pollution
app.use(
  hpp({
    whitelist: [
      'duration',
```

```

        'ratingsQuantity',
        'ratingsAverage',
        'maxGroupSize',
        'difficulty',
        'price'
    ]
  })
);

// Serving static files
app.use(express.static(`${__dirname}/public`));

// 3) ROUTES
// app.use('/api/v1/tours', tourRouter);
app.use('/', authRouter);
app.use('/api/user', userRoute);
app.use('/api/owner', ownerRoute);
app.use('/api/carts', cartRoutes);
app.use('/api/bank-carts', bankCartRoute);

app.all('*', (req, res, next) => {
  next(new AppError(`Can't find ${req.originalUrl} on this server!`, 404));
});

// app.use(globalErrorHandler);

module.exports = app;

```

config.cnv

NODE_ENV=development

PORT=5000

DATABASE=mongodb+srv://Admin:hash99lahi20@hotelsobana.z6log.mongodb.net/project_management?retryWrites=true&w=majority

JWT_SECRET=<:C<}x6Ta%qw7HYRe/qV7QQ9jZt9RAV#\Nw]P{)ZL?~#@K"3N

JWT_EXPIRES_IN=90d

JWT_COOKIE_EXPIRES_IN=90d

EMAIL_USERNAME=d2b7b7951de2ef

EMAIL_PASSWORD=2f106c82c5e463

EMAIL_HOST=smtplib.mailtrap.io

EMAIL_PORT=25

package.json

```
{
  "name": "project-managment",
  "version": "1.0.0",
  "description": "Third year first semester final project",
  "main": "app.js",
  "scripts": {
    "serve": "nodemon server.js",
    "start:prod": "NODE_ENV=production nodemon server.js",
    "debug": "ndb server.js"
  },
  "author": "",
  "license": "",
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "cors": "^2.8.5",
    "dotenv": "^7.0.0",
    "express": "^4.16.4",
    "express-mongo-sanitize": "^1.3.2",
    "express-rate-limit": "^3.5.0",
    "helmet": "^3.16.0",
    "hpp": "^0.2.2",
    "jsonwebtoken": "^8.5.1",
    "mongoose": "^5.5.2",
    "mongoose-auto-increment": "^5.0.1",
    "morgan": "^1.9.1",
    "multer": "^1.4.1",
    "nodemailer": "^6.1.1",
  }
}
```

```
    "nodemon": "^2.0.15",
    "slugify": "^1.3.4",
    "validator": "^10.11.0",
    "xss-clean": "^0.1.1"
  },
  "devDependencies": {
    "eslint": "^5.16.0",
    "eslint-config-airbnb": "^17.1.0",
    "eslint-config-prettier": "^4.1.0",
    "eslint-plugin-import": "^2.17.2",
    "eslint-plugin-jsx-a11y": "^6.2.1",
    "eslint-plugin-node": "^8.0.1",
    "eslint-plugin-prettier": "^3.0.1",
    "eslint-plugin-react": "^7.12.4",
    "prettier": "^1.17.0"
  },
  "engines": {
    "node": ">=10.0.0"
  }
}
```

server.js

```
const mongoose = require('mongoose');
const dotenv = require('dotenv');
const cors = require("cors")

dotenv.config({ path: './config.env' });

const app = require('./app');
app.use(cors({
  origin: ["http://localhost:3000"],
  credentials: true
}));

const port = process.env.PORT || 3000;
const server = app.listen(port, () => {
  console.log(`App running on port ${port}...`);
});
```

```
const URL = process.env.DATABASE;

mongoose
  .connect(URL, {
    useNewUrlParser: true,
    useCreateIndex: true,
    useFindAndModify: false,
    useUnifiedTopology: true
  })
  .then(() => console.log('DB connection successful!'));
```

Front-End (client side implementation)

Apis

auth.js

```
import api from "../axios";

// eslint-disable-next-line import/no-anonymous-default-export
export default {
  login: (payload) => api.post(`/signing`, payload),
  register: (payload) => api.post(`/signup`, payload),
  currentUser: () => api.get(`/current-user`),
  logout: () => api.get(`/logout`),
}
```

buyer.js

```
import api from "../axios";
const resource = '/api/user';

export default {
  listPrduct: () => api.get(`${resource}`)
}
```

Owner.js

```
import api from "../axios";
const resource = '/api/owner';

export default {
  listPrduct: () => api.get(`${resource}`),
  createProduct: (payload) => api.post(`${resource}`, payload),
  getProduct: (id) => api.get(`${resource}/${id}`),
}
```

axios.js

```
import axios from "axios";

const instance = axios.create({
  baseURL: 'http://localhost:5000'
})

if(localStorage.getItem('JWT')){
  instance.defaults.headers.common['Authorization'] = `Bearer ${localStorage.getItem('JWT')}`;
}

export default instance;
```

Context

AuthContext.js

```
import React,{createContext, useContext, useEffect, useState} from 'react'
import auth from "../apis/modules/auth";

const AuthContext = createContext();

function AuthContextProvider(props) {
  const [loggedIn, setloggedIn] = useState({});

  async function getLogged(){
    try{
      const loggedInRes = await auth.currentUser();
      setloggedIn(loggedInRes.data.data);
    }catch (error){
      setloggedIn(null)
    }
  }

  useEffect(() => {
    getLogged();
  }, [])

  return <AuthContext.Provider value={{loggedIn, getLogged}}>
    {props.children}
  </AuthContext.Provider>
}
```



```
    </AuthContext.Provider>
  }

  export default AuthContext;
  export {AuthContextProvider}
```

landingPage.js

```
import React, {useContext} from "react";
import './landingPagestyle.css';
import {Link} from 'react-router-dom';
import Footer from "../layouts/footer";
import Header from "../layouts/header";
import AuthContext from "../context/AuthContext";

export default function LandingPage(){

  const { loggedIn } = useContext(AuthContext);

  const redirectPage = ()=>{
    if(loggedIn === null){
      window.location = '/login'
    }else if(loggedIn.role === 'owner'){
      window.location = '/homeowner'
    }else {
      window.location = '/homeclient'
    }
  }

  return (
    <>
    <head>
    <meta charSet="UTF-8" />
    <meta httpEquiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Landing Page</title>
    </head>
    <body>
    <main>
    <Header/>
```

```

        
        <div class="showcase-area">
            <div class="container">
                <div class="left">
                    <div class="big-title">
                        <h1>Welcome To Ceylon Agri</h1>
                        <h1>Start Exploring Now.</h1>
                    </div>
                    <p class="text">
                        Lorem ipsum dolor sit amet, consectetur adipisicing elit.
                        Delectus eius distinctio odit, magni magnam qui ex perferendis
                        vitae!Lorem ipsum dolor sit amet, consectetur adipisicing elit.
                        Delectus eius distinctio odit, magni magnam qui ex perferendis
                        vitae!
                    </p>
                    <div class="cta">

                        <Link><li href="#" onClick={redirectToPage} class="btn2">Get
started</li></Link>

                    </div>
                </div>

                <div class="right">
                    
                </div>
            <row></row>
        </div>
    </div>
</main>
<Footer/>
<script src="https://kit.fontawesome.com/a81368914c.js"></script>
<script src="./app.js"></script>
</body>
</>
)
}

```

footer.js

```
import React from "react";
import './footer.css';

export default function Footer(){

    return (
        <center>
<footer>
        <div class="wrapper">
            <small>&copy;2022 <strong>Ceylon Agri Pvt.Ltd</strong>, All Rights
Reserved</small>
            <nav class="footer-nav">
                <a href="#">Back to Top</a>
                <a href="#">Terms of Use</a>
                <a href="#">Privacy</a>
            </nav>
        </div>
    </footer>
    </center>
    )
}
```

header.js

```
import React, { useContext } from "react";
import { Link, Route } from "react-router-dom";
import AuthContext from '../context/AuthContext';
import auth from '../apis/modules/auth';
import { useCart } from "react-use-cart";

export default function Header() {
    const { loggedIn } = useContext(AuthContext);
    const { totalItems } = useCart();

    const logout = async () => {
        await auth.logout();
        localStorage.clear();
        window.location = '/login'
    }

    return (
        <div class="big-wrapper">
```

```

        <header>
            
            <div class="container">
                <div class="logo">
                    <Link to="/"></Link>
                </div>

                <div class="links">
                    <ul>

                        {/*unauthorized user*/}
                        {
                            loggedIn === null && (<
                                <Link to="/login"><li><a >Home</a></li></Link>
                                <Link to="/login"><li><a>Products</a></li></Link>
                                <Link to="/"><li><a>Testimonials</a></li></Link>
                                <Link to="/login"><li class="btn2">Sign in</li></Link>
                            </>)
                        }

                        {/*owner routes*/}
                        {
                            loggedIn !== null && loggedIn.role === 'owner' && (<
                                <Link to="/homeowner"><li><a>MY Products</a></li></Link>
                                <Link to="/add-product"><li><a>Product Listing
Form</a></li></Link>
                                <Link to="/login"><li class="btn2"
onClick={logout}>Logout</li></Link>
                            </>)
                        }

                        {/*client routes*/}
                        {
                            loggedIn !== null && loggedIn.role === 'buyer' && (<
                                <Link to="/homeclient"><li><a>Home</a></li></Link>
                                <Link to="/homeclient"><li><a>Products</a></li></Link>
                                {/* <Link to="/cart" style={{ textDecoration: "none"
}}><li><svg xmlns="http://www.w3.org/2000/svg" width="30" height="30"
fill="currentColor" class="bi bi-cart-fill" viewBox="0 0 16 16"><path d="M0
1.5A.5.5 0 0 1 .5 1H2a.5.5 0 0 1 .485.379L2.89 3H14.5a.5.5 0 0 1 .491.592l-1.5

```

```

8A.5.5 0 0 1 13 12H4a.5.5 0 0 1-.491-.408L2.01 3.607 1.61 2H.5a.5.5 0 0 1-.5-
.5zM5 12a2 2 0 1 0 0 4 2 2 0 0 0 0-4zm7 0a2 2 0 1 0 0 4 2 2 0 0 0 0-4zm-7 1a1 1 0
1 1 0 2 1 1 0 0 1 0-2zm7 0a1 1 0 1 1 0 2 1 1 0 0 1 0-2z" /></svg><span style={{
marginBottom: "10px" }}>{totalItems === 0 ? '' : String(totalItems)[0] !== 0 ?
totalItems : totalItems.replace(/^0+/, '')}</span></li></Link> */}
    <Link to="/cart" style={{ textDecoration: "none" }}><li
class="nav-item px-3 text-uppercase mb-0 position-relative d-lg-flex">
    <div id="cart" class="d-none"></div>
    <a href="/store/cart.shtml" class="cart position-relative d-
inline-flex" aria-label="View your shopping cart">
    <i class="fas fa fa-shopping-cart fa-lg"></i>
    <span class="cart-basket d-flex align-items-center justify-
content-center">{totalItems}</span>
    </a>
    </li></Link>
    <Link to="/login"><li class="btn2"
onClick={logout}>Logout</li></Link>
    </>)
  }
</ul>
</div>
</div>
</header>
</div>
)
}

```

Views

Login.jsx

```

import React, { useState } from 'react';
import './login.css';
import auth from "../../apis/modules/auth";
import { SigningForm } from '../../validations/index'
import { Formik, Form, Field } from 'formik'

export function Login() {

  // const [email, setEmail] = useState("");
  // const [password, setPassword] = useState("");
  const [error, setError] = useState("");

```

```

const login = async (data) => {
  try {
    let payload = {
      email: data.email,
      password: data.password
    }
    let respond = await auth.login(payload)
    console.log(respond.data.data.user.role)
    localStorage.setItem('JWT', respond.data.token)
    if(respond.data.data.user.role === 'owner'){
      window.location = '/homeowner'
    }else {
      window.location = '/homeclient'
    }

  } catch (e) {
    setError('Your user name or password is incorrect')
  }
}

return (
  <>
    <head>
      <meta charset="UTF-8" />
      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
      <meta httpEquiv="X-UA-Compatible" content="ie=edge" />
      <title>Login Page</title>
      <link
href="https://fonts.googleapis.com/css?family=Karla:400,700&display=swap"
rel="stylesheet" />
      <link rel="stylesheet"
href="https://cdn.materialdesignicons.com/4.8.95/css/materialdesignicons.min.css"
/>
      <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
/>
    </head>

    <body style={{ paddingTop: '3em' }}>
      <main class="d-flex align-items-center min-vh-100 py-3 py-md-0">
        <div class="container" >
          <div class="card login-card">
            <div class="row no-gutters">
              <div class="col-md-5">

```

```

        
      </div>
      <div class="col-md-7">
        <div class="card-body">
          <div class="logo">
            <a href="/"></a>
          </div>
          <p class="login-card-description">Sign into your account</p>
          <Formik
            initialValues={{
              email: '',
              password: ''
            }}
            validationSchema={SigningForm}
            onSubmit={values => {
              login(values)
            }}
          >
            {({ errors, touched }) => (
              <Form>
                <div>
                  <Field type="email" name="email" id="email"
class="form-control" placeholder="Email Address" />
                  {errors.email && touched.email ? <p id={"login-
error"} class="text-danger">{errors.email}</p> : null}
                </div>
                <div>
                  <Field type="password" name="password" id="password"
class="form-control" placeholder="Password" />
                  {errors.password && touched.password ? <p id={"login-
error"} class="text-danger">{errors.password}</p> : null}
                  {error ? <p id={"login-error"} class="text-
danger">{error}</p> : null}
                </div>
                <button type="submit" class="btn btn-block login-btn
mb-4">Login</button>
              </Form>
            )}
          </Formik>
          {/* eslint-disable-next-line jsx-a11y/anchor-is-valid */}
          <a href="#" class="forgot-password-link">Forgot password?</a>

```

```

                <p class="login-card-footer-text">Don't have an account? <a
href="/register" class="text-reset">Register here</a></p>
            </div>
        </div>
    </div>
</div>
</div>
</main>
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"></scri
pt>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></sc
ript>
    </body>
</>

)
}

```

clientRegistration.jsx

```

import React, {useState} from "react";
import './login.css';
import auth from "../../apis/modules/auth";
import {SignupSchema} from "../../validations";
import {Field, Form, Formik} from "formik";

export default function ClientRegistration(){
    const [error, setError] = useState("");

    const register = async (data)=>{
        try{
            const payload = {
                name:data.name,
                email:data.email,
                password:data.password, passwordConfirm:data.passwordConfirm
            }
            await auth.register(payload)
            window.location = '/login'
        }catch (e){
            setError('Your email is already exists!!')
        }
    }
}

```



```

    }

    return(
      <>
        <head>
          <meta charSet="UTF-8" />
          <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
          <meta httpEquiv="X-UA-Compatible" content="ie=edge" />
          <title>Register Page</title>
          <link
href="https://fonts.googleapis.com/css?family=Karla:400,700&display=swap"
rel="stylesheet" />
          <link rel="stylesheet"
href="https://cdn.materialdesignicons.com/4.8.95/css/materialdesignicons.min.css"
/>
          <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
/>
        </head>

        <body style={{paddingTop: '3em'}}>
          <main class="d-flex align-items-center min-vh-100 py-3 py-md-0">
            <div class="container">
              <div class="card login-card">
                <div class="row no-gutters">
                  <div class="col-md-5">
                    
                  </div>
                  <div class="col-md-7">
                    <div class="card-body">
                      <div class="logo">
                        <a href="/"></a>
                      </div>
                      <p class="login-card-description">Signup to
your account</p>

                      <Formik
                        initialValues={{
                          name: '',
                          email: '',
                          password: '',
                          passwordConfirm: ''
                        }}

```

```

validationSchema={SignupSchema}
onSubmit={values => {
  register(values)
}}
>
(({ errors, touched }) => (
  <Form>
    <div>
      <Field type="text"
name="name" id="name" class="form-control"
placeholder="Name"/>
{errors.name && touched.name
?
      <p id={"login-error"}
className="text-danger">{errors.name}</p> : null}
    </div>
    <div>
      <Field type="email"
name="email" id="email" class="form-control" placeholder="Email address" />
{errors.email &&
touched.email ? <p id={"login-error"} class="text-danger">{errors.email}</p> :
null}
    </div>
    <div>
      <Field type="password"
name="password" id="password" class="form-control" placeholder="Password" />
{errors.password &&
touched.password ? <p id={"login-error"} class="text-
danger">{errors.password}</p> : null}
    </div>
    <div>
      <Field type="password"
name="passwordConfirm" id="passwordConfirm" class="form-control"
placeholder="Confirm
Password"/>
{errors.passwordConfirm &&
touched.passwordConfirm ?
      <p id={"login-error"}
className="text-danger">{errors.passwordConfirm}</p> : null}
      {error ? <p id={"login-
error"} class="text-danger">{error}</p> : null}
    </div>
    <button type="submit" class="btn
btn-block login-btn mb-4">Register</button>

```

```

        </Form>
      )}
    </Formik>
    <p class="login-card-footer-text">Already
have an account? <a href="/login" class="text-reset">Login here</a></p>
    </div>
  </div>
</div>
</div>
</div>
</div>
</main>
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"></scri
pt>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></sc
ript>
</body>
</>
)
}

```

ownerRegistration.jsx

```

import React, {useState} from "react";
import './login.css';
import auth from "../../apis/modules/auth";
import {SignupSchema} from "../../validations";
import {Field, Form, Formik} from "formik";

export default function OwnerRegistration(){
  const [error, setError] = useState("");

  const register = async (data)=>{
    try{
      const payload = {
        name:data.name,
        email:data.email,
        password:data.password, passwordConfirm:data.passwordConfirm,

```

```

        role: 'owner'
      }
      await auth.register(payload)
      window.location = '/login'
    } catch (e) {
      setError('Your email is already exists!!')
    }
  }

  return(
    <>
      <head>
        <meta charSet="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
        <meta httpEquiv="X-UA-Compatible" content="ie=edge" />
        <title>Register Page</title>
        <link
href="https://fonts.googleapis.com/css?family=Karla:400,700&display=swap"
rel="stylesheet" />
        <link rel="stylesheet"
href="https://cdn.materialdesignicons.com/4.8.95/css/materialdesignicons.min.css"
/>
        <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
/>
      </head>

      <body style={{paddingTop: '3em'}}>
        <main class="d-flex align-items-center min-vh-100 py-3 py-md-0">
          <div class="container">
            <div class="card login-card">
              <div class="row no-gutters">
                <div class="col-md-5">
                  
                </div>
                <div class="col-md-7">
                  <div class="card-body">
                    <div class="logo">
                      <a href="/"></a>
                    </div>
                    <p class="login-card-description">Signup to
your account</p>

```

```

<Formik
  initialValues={{
    name: '',
    email: '',
    password: '',
    passwordConfirm: ''
  }}
  validationSchema={SignupSchema}
  onSubmit={values => {
    register(values)
  }}
>
  ({ errors, touched }) => (
    <Form>
      <div>
        <Field type="text"
name="name" id="name" class="form-control"
placeholder="Name" />
        {errors.name && touched.name
?
          <p id={"login-error"}
className="text-danger">{errors.name}</p> : null}
        </div>
        <div>
          <Field type="email"
name="email" id="email" class="form-control" placeholder="Email address" />
          {errors.email &&
touched.email ? <p id={"login-error"} class="text-danger">{errors.email}</p> :
null}
          </div>
          <div>
            <Field type="password"
name="password" id="password" class="form-control" placeholder="Password" />
            {errors.password &&
touched.password ? <p id={"login-error"} class="text-
danger">{errors.password}</p> : null}
            </div>
            <div>
              <Field type="password"
name="passwordConfirm" id="passwordConfirm" class="form-control"
placeholder="Confirm
Password" />
              {errors.passwordConfirm &&
touched.passwordConfirm ?

```

```

                                <p id={"login-error"}
className="text-danger">{errors.passwordConfirm}</p> : null}
                                {error ? <p id={"login-
error"} class="text-danger">{error}</p> : null}
                                </div>

                                <button type="submit" class="btn
btn-block login-btn mb-4">Register</button>
                                </Form>
                                )}
                                </Formik>
                                <p class="login-card-footer-text">Already
have an account? <a href="/login" class="text-reset">Login here</a></p>
                                </div>
                                </div>
                                </div>
                                </div>
                                </div>
                                </main>
                                <script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
                                <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"></scri
pt>
                                <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></sc
ript>
                                </body>
                                </>

                                )
}

```

cart.jsx

```
import React, { useState } from 'react';
import { Link } from 'react-router-dom';
import Footer from '../../layouts/footer';
import Header from '../../layouts/header';
import { useCart } from 'react-use-cart';
import './cart.css'

const Cart = () => {
  const [qty, setQty] = useState('')
  const {
    items,
    totalItems,
    isEmpty,
    totalUniqueItems,
    cartTotal,
    updateItemQuantity,
    removeItem,
    emptyCart
  } = useCart();

  if (isEmpty) return (
    <>
      <Header />
      
      <div class="container" >
        <div className='shopc'>
          <h1>CART</h1>
          <p><Link to="/">Home</Link> / Cart</p>
        </div>
      </div>
      <div class="container">
        <div class="col-sm-12 empty-cart-cls text-center">
          </* <div style={{padding: '20px', color: '#515151'}}><svg
xmlns="http://www.w3.org/2000/svg" width="80" height="80" fill="currentColor"
class="bi bi-cart" viewBox="0 0 16 16"><path d="M0 1.5A.5.5 0 0 1 .5 1H2a.5.5 0 0
1 .485.379L2.89 3H14.5a.5.5 0 0 1 .491.592l-1.5 8A.5.5 0 0 1 13 12H4a.5.5 0 0 1-
.491-.408L2.01 3.607 1.61 2H.5a.5.5 0 0 1-.5-.5zM3.102 4l1.313 7h8.171l1.313-
7H3.102zM5 12a2 2 0 1 0 0 4 2 2 0 0 0 0-4zM7 0a2 2 0 1 0 0 4 2 2 0 0 0 0-4zM-7"
*/></div>
          <p>Your cart is empty</p>
        </div>
      </div>
    </>
  )
}
```

```

1a1 1 0 1 1 0 2 1 1 0 0 1 0-2zm7 0a1 1 0 1 1 0 2 1 1 0 0 1 0-2z"/></svg></div>
*/}

    <div className='crti'>
        <div style={{ padding: '20px', color: '#515151' }}></img></div>
        </div>
        <h4><strong style={{ lineHeight: '30px', color: '#515151'
}}>Your Cart is Empty</strong></h4>
        <h5 style={{ color: '#515151', paddingBottom: '25px' }}>Add
something to make me happy :)</h5>
        <Link to="/homeclient"><div className='btn2' >Continue
Shopping</div></Link>
    </div>
</div>
<Footer />
</>
);
return (
    <div>
        <Header />
        

        <div class="container" >
            <div className='shopc'>
                <h1>CART</h1>
                <p><Link to="/">Home</Link> / <Link
to="/cart">Cart</Link></p>
            </div>
            <div class="container pb-5 mt-n2 mt-md-n3">
                <div class="row">
                    <div class="col-xl-9 col-md-8">
                        <h2 class="h6 d-flex flex-wrap justify-content-
between align-items-center px-4 py-3"><span style={{ fontWeight: "bold",
fontSize: "30px", fontFamily: "Poppins" }}>Products</span>
                        <Link to="/homeclient"><a class="font-size-sm"
href="/homeclient"><svg xmlns="http://www.w3.org/2000/svg" width="24" height="24"
viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2" stroke-
linecap="round" stroke-linejoin="round" class="feather feather-chevron-left"
style={{ fontWeight: "bold", fontSize: "30px", fontFamily: "Poppins" }}><polyline
points="15 18 9 12 15 6"></polyline></svg>Continue Shopping</a></Link></h2>
                        {items.map((item, index) => {
                            return (
                                <div class="d-sm-flex justify-content-between
my-4 pb-4 border-bottom" key={index}>

```



```

<div class="media d-block d-sm-flex text-center text-sm-left">
    <a class="cart-item-thumb mx-auto mr-sm-4" href="#">
        <div class="cardc">
            <div class="imgBxc">
                <Link
to={` /itemview/${item._id}`}><img src={"http://localhost:5000/img/product/" +
item.image} /></Link>
            </div>
        </div></a>
        <div class="media-body pt-3">
            <Link
to={` /itemview/${item._id}`}><h3 class="product-card-title font-weight-semibold border-0 pb-0">{item.name}</h3></Link>
            <div class="font-size-sm"><span
class="text-muted mr-2">Size:</span>{item.size}</div>
            {/* <div class="font-size-sm"><span class="text-muted mr-2">Color:</span>Black</div> */}
            <div class="font-size-sm"><span
class="text-muted mr-2">SKU:</span>{item.sku}</div>
            <div class="font-size-lg text-primary pt-2">Rs. {item.price}</div>
        </div>
    </div>

    <div class="pt-2 pt-sm-0 pl-sm-3 mx-auto mx-sm-0 text-center text-sm-left" style={{ width: "auto" }}>
        {/* <div class="form-group mb-2"
style={{ textAlign: "center" }}>
            <label for="quantity1">Qty:
{item.quantity}</label>
            <input class="form-control form-control-sm" type="number" id="quantity1" onChange={(e) => setQty(e.target.value)}
min={0} />
        </div> */}
        {/* <button class="btn btn-outline-secondary btn-sm btn-block mb-2" type="button" onClick={() =>
updateItemQuantity(item.id, qty)}>
            <svg
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
fill="none" stroke="currentColor" stroke-width="2" stroke-linecap="round" stroke-linejoin="round" class="feather feather-refresh-cw mr-1">

```

```

17 10"></polyline>
14"></polyline>
14.85-3.36L23 10M1 1414.64 4.36A9 9 0 0 0 20.49 15"></path>
</svg>Update Cart</button> */}
<div className='plum'>
  <input className='crt'
type="button" onclick="decrementValue()" value="-" />
  <input className='crt'
type="text" name="quantity" value="1" maxLength="2" max="10" size="1" id="number"
/>
  <input className='crt'
type="button" onclick="incrementValue()" value="+" />
</div>
<br/>
<button class="btn btn-outline-danger
btn-sm btn-block mb-2" type="button" onClick={() => removeItem(item.id)}>
  <svg
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
fill="none" stroke="currentColor" stroke-width="2" stroke-linecap="round" stroke-
linejoin="round" class="feather feather-trash-2 mr-1">
    <polyline points="3 6 5 6 21
6"></polyline>
    <path d="M19 6v14a2 2 0 0 1-2
2H7a2 2 0 0 1-2-2V6m3 0V4a2 2 0 0 1 2-2h4a2 2 0 0 1 2 2v2"></path>
    <line x1="10" y1="11" x2="10"
y2="17"></line>
    <line x1="14" y1="11" x2="14"
y2="17"></line>
  </svg>Remove</button>
</div>
</div>
)
}}
</div>
<div class="col-xl-3 col-md-4 pt-3 pt-md-0">
  <h2 class="h6 px-4 py-3 bg-info text-center" style={{
color: 'white' }}>Subtotal</h2>
  {/* <div class="h3 font-weight-semibold text-center
py-3">Cart ({totalUniqueItems}) Total Items ({totalItems}) </div> */}

```

```

        <div class="h3 font-weight-semibold text-center py-3">Rs. {cartTotal}</div>
        <h3 class="h6 pt-1 font-weight-semibold" style={{
"textAlign": "center" }}>Total Items: {totalItems}</h3>
        <hr />
        <h3 class="h6 pt-4 font-weight-semibold"><span
class="badge badge-success mr-2">Note</span>Additional Comments</h3>
        <textarea class="form-control mb-3" id="order-
comments" rows="5"></textarea>
        <Link to="/cardpay"><a class="btn btn-primary btn-
block" href="#">
            <svg xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" fill="none" stroke="currentColor"
stroke-width="2" stroke-linecap="round" stroke-linejoin="round" class="feather
feather-credit-card mr-2">
                <rect x="1" y="4" width="22" height="16"
rx="2" ry="2"></rect>
                <line x1="1" y1="10" x2="23" y2="10"></line>
            </svg>Proceed to Checkout</a></Link>
        </div>
    </div>
</div>
</div>
</div>
</div>
    <Footer />
</div>
);
};

export default Cart;

```

carts.js

```

function increment() {
    document.getElementById('demoInput').stepUp();
}
function decrement() {
    document.getElementById('demoInput').stepDown();
}

```

home.jsx

```
import React, { useState, useEffect } from 'react';
```



```

: Rs. {item.price}</p>

    <div class="size">
      <h3>Size :</h3>
      <span>{item.size}</span>
    </div>
    <div class="color" style={{ padding:
      "5px" }}>

      <h3>SKU : {item.sku}</h3>
    </div>
    {/* <div className='btn btn-light'
href="#" onClick={() => addItem(item)}>Add to Cart</div> */}
    <Link
to={` /itemview/${item._id}`}>View</Link>
  </div>
</>
</div>
)
}}
</div>

</div>
</div>
<Footer />
</div>
)
}

export default ClientHome;

```

cardpay.jsx

```
import React from 'react';
import { Link } from 'react-router-dom';
import Footer from '../../layouts/footer';
import Header from '../../layouts/header';
import './cardpay.css'

const Cardpay = () => {
  return (
    <div>
      <Header/>
      
```

```

        <div class="container" >
        <div className='shopc'>
            <h1>PAYMENT</h1>
            <p><Link to="/">Home</Link> / <Link to="/cart">Cart</Link> / <Link
to="/cardpay">Payment</Link></p>
        </div>

        <link href="https://fonts.googleapis.com/css?family=Josefin+Sans"
rel="stylesheet"/>
<script src="https://code.jquery.com/jquery-3.1.1.min.js" integrity="sha256-
hVVnYaiADRT02PzUGmuLJr8BLUSjGIZsDYGmIJLv2b8=" crossorigin="anonymous"></script>
<script src="https://code.jquery.com/ui/1.12.1/jquery-ui.min.js"></script>
<script src="https://use.fontawesome.com/aa95071b26.js"></script>

<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.4/angular.min.js"></scri
pt>
<div id="bodycp">
    <div class="out-wrap">
        <h1 class="text-center">Enter Card Details</h1>
        <div id="payment" ng-app="myApp" ng-controller="myCtrl">
            <div class="details">
                <div class="input-wrap four-num">
                    <label>Card Number</label>
                    <input maxlength="4" ng-model="f4" type="text"/>
                    <input maxlength="4" ng-model="s4" type="text"/>
                    <input maxlength="4" ng-model="t4" type="text"/>
                    <input maxlength="4" ng-model="l4" type="text"/>
                </div>
                <div class="input-wrap">
                    <label>Card Holder's Name </label>
                    <input ng-model="hname" type="text"/>
                </div>
                <div class="halfbox">
                    <div class="input-wrap">
                        <label>Valid Till</label>
                        <div class="twin">
                            <input maxlength="2" ng-model="edm" type="text"/>
                            <input maxlength="2" ng-model="edy" type="text"/>
                        </div>
                    </div>
                </div>

                <div class="input-wrap">
                    <label>CVV</label>

```

```
<input maxlength="3" ng-model="cvv" type="text"/>
</div>
</div>
</div>
<div class="card">
  <div>
    <div class="clayout">
      <div class="c-front">
        <p class="chiplogo">
          <span class="chip">
            <b></b></span>
          <span class="logo">
            <b class="master"></b>
          </span>
        </p>
        <p class="cnum">
          <span ng-bind="f4">XXXX</span> - <span ng-bind="s4">XXXX</span> - 
<span ng-bind="t4">XXXX</span> - <span ng-bind="l4">XXXX</span>
        </p>
        <p class="cname"><span>Card Holder</span><label ng-
bind="hname">PRAVEEN KUMAR GORAKALA</label></p>
        <p class="cvtr"><span>Valid Till</span><label ng-
bind="edm">01</label> / <label ng-bind="edy">28</label></p>
      </div>
      <div class="c-back">
        <p class="topc"><span>136589420</span><span>sddv5</span></p>
        <p class="backit"></p>
        <p class="cvv"><span>CVV</span><b ng-bind="cvv"></b></p>
      <p class="chiplogo">
        <span class="logo">
          <b class="master"></b>
        </span>
      </p>
    </div>
  </div>
</div>
</div>
</div>
</div>
</div>
</div>
```

```
    );  
  };  
</script>  
  
</script>  
export default Cardpay;  
Home.jsx
```

```
import React from 'react';  
import axios from "../../apis/axios";  
import product from "../../apis/modules/product";  
  
const Home = () => {  
  
  const creatGroup = async()=>{  
    let data = {  
      category:"A",  
      stock:"1",  
      size:10,  
      sku:"ZZZZ",  
      price:1200,  
      description:"About product",  
      name:"product 1"  
    }  
  
    let respond = await product.createProduct(data)  
  }  
  return (  
    <div>  
      Hello Home  
      <button type="submit" onClick={(e)=>{creatGroup()}}>create  
group</button>  
    </div>  
  );  
};  
  
export default Home;
```


itemView.jsx

```
import React, { useEffect, useState } from 'react';
import { Link, useParams } from 'react-router-dom';
import Footer from '../../layouts/footer';
import Header from '../../layouts/header';
import owner from '../../apis/modules/owner'
import { useCart } from 'react-use-cart'
import './itemview.css'

const Itemview = () => {
  const { id } = useParams();
  const [item, setItem] = useState([]);
  const { addItem, inCart, items } = useCart();

  const getProduct = async () => {
    try {
      const arr = await owner.getProduct(id)
      setItem(arr.data)
      // setItems(arr.data)
    } catch {
      setItem(null)
    }
  }

  const directToCart = () => {
    window.location = "/cart"
  }

  useEffect(() => {
    getProduct()
  }, [id])

  return (
    <div>
      <Header />
```

```

        <div class="container" >
            <div className='shopc'>
                <h1>PRODUCT DETAILS</h1>
                <p><Link to="/">Home</Link> / <Link
to="/homeclient">Products</Link> / {item.sku}</p>
            </div>
            <div class="container">
                <div class="product-content product-wrap clearfix product-
deatil">
                    <div class="row">
                        <div class="col-md-5 col-sm-12 col-xs-12">
                            <div class="product-image">
                                <div id="myCarousel-2" class="carousel
slide">
                                    <div class="carousel-inner">
                                        <div class="cardiv">
                                            <>
                                                <div class="imgBx">
                                                    <img
src={"http://localhost:5000/img/product/" + item.image} />
                                                </div>
                                            </>
                                        </div>
                                    </div>
                                    <a class="left carousel-control"
href="#myCarousel-2" data-slide="prev"> <span class="glyphicon glyphicon-chevron-
left"></span> </a>
                                    <a class="right carousel-control"
href="#myCarousel-2" data-slide="next"> <span class="glyphicon glyphicon-chevron-
right"></span> </a>
                                </div>
                            </div>
                        </div>
                        <div class="col-md-6 col-md-offset-1 col-sm-12 col-
xs-12">
                            <h2 class="name">
                                {item.name}
                                {/* <small>Product by Ceylon Agri
Pvt.Ltd</small> */}
                            </h2>

```

```

        <hr />
        <h3 class="price-container">
            Rs. {item.price}
        </h3>
        <div class="certified">
            <ul>
                <li>
                    <a style={{ fontSize: '15px' }}>SKU
CODE<span>{item.sku}</span></a>
                </li>
                <li>
                    <a style={{ fontSize: '15px' }}>ITEM
SIZE<span>{item.size}</span></a>
                </li>
            </ul>
        </div>
        <hr />
        <div class="description description-tabs">
            <ul id="myTab" class="nav nav-pills">
                <li class="active" style={{ color:
'white' }}><a class="btn btn-secondary">Product Description </a></li>
            </ul>
            <div class="detail" >
                <br />
                <p style={{ color: 'gray' }}>
                    {item.description}
                </p>
            </div>
        </div>
        <hr />
        <div class="rowbt" key={item.name}>
            <Link to="/cart">
                <button class="btn btn-success btn-lg"
onClick={() => addItem(item)}>Add to Cart</button>
            </Link>
        </div>
    </div>
</div>
</div>
</div>
</div>
<Footer />
</div>
);

```

```
};
<script>

</script>
export default Itemview;
```

owner – home.jsx

```
import React, { useState, useEffect } from 'react';
import '../client/home/chome.css'
import { Link } from 'react-router-dom';
import Footer from '../layouts/footer';
import Header from '../layouts/header';
import owner from '../apis/modules/owner'

const OwnerHome = () => {

  const [products, setProducts] = useState([]);

  const listProduct = async () => {
    try {
      const productsArr = await owner.listPrduct()
      setProducts(productsArr.data)
    } catch {
      setProducts(null)
    }
  }

  useEffect(() => {
    listProduct()
  }, [])

  return (
    <div>
      <Header />
      

      <div class="container" >
        <div className='shopc'>
          <h1>MY PRODUCT LISTING</h1>
          <p><Link to="/">Home</Link> / Add Product</p>
```

```

        </div>
        <div className='c2'>
          <div class='rowp'>
            {products.map(item => {
              return (
                <div class="cardp">
                  <>
                    <div class="imgBx">
                      <img
src={"http://localhost:5000/img/product/" + item.image} />
                    </div><div class="contentBx">
                      <h2>{item.name}</h2>
                      <p style={{ color: 'ffffff' }}>Price
: {item.price}</p>

                      <div class="size">
                        <h3>Size :</h3>
                        <span>{item.size}</span>
                      </div>
                      <div class="color" style={{ padding:
"5px" }}>

                        <h3>SKU : {item.sku}</h3>
                      </div>
                      <Link to="/edit-product">Edit</Link>
                    </div>
                  </>
                </div>
              )
            })}
          </div>
        </div>
      </div>
      <Footer />
    </div>
  )
}

export default OwnerHome;

```

editProduct.jsx

```

import React, { useState, useEffect } from 'react';
import '../client/home/chome.css'
import { Link } from 'react-router-dom';
import Footer from '../layouts/footer';

```

```

import Header from '../../layouts/header';

const EditProduct = () => {

  return (
    <div>
      <Header />
      

      <div class="container" >
        <div className='shopc'>
          <h1>EDIT PRODUCT</h1>
          <p><Link to="/">Home</Link> / <Link to="/homeowner">My
Products</Link> / Edit Product</p>
        </div>
        <form>

          <div className="row mb-4">
            <div className="col">
              <div className="form-group">
                <label style={{ fontWeight: 'bold',
color: '#5D5D5D' }} className="form-label">*Product Name</label>
                <input type="text" name="name" id="name"
class="form-control" placeholder="Product name" />

              </div>
            </div>
            <div className="col">
              <div className="form-outline">
                <label style={{ fontWeight: 'bold',
color: '#5D5D5D' }}
className="form-label">*SKU</label>
                <input type="text" id="sku" name="sku"
className="form-control"
placeholder="Enter SKU" />

              </div>
            </div>
          </div>
          <div className="form-outline mb-4">
            <label style={{ fontWeight: 'bold', color:
'#5D5D5D' }}
className="form-label">*Quantity</label>
            <input type="text" id="size" name="size"
className="form-control"

```

```

placeholder="Enter Quantity" />

</div>
<div className="form-outline mb-4">
  <label style={{ fontWeight: 'bold', color:
'#5D5D5D' }} className="form-label">*Price</label>
  <input type="number" id="price" name="price"
className="form-control" placeholder="Enter Price" />

</div>
<div className="form-outline mb-4">
  <label style={{ fontWeight: 'bold', color:
'#5D5D5D' }} className="form-label">*Stock</label>
  <input type="number" id="stock" name="stock"
className="form-control" placeholder="Enter Stock Quantity" />

</div>
<div className="form-outline mb-4">
  <label style={{ fontWeight: 'bold', color:
'#5D5D5D' }} className="form-label">*Description</label>
  <textarea className="form-control"
id="description" name="description" rows="5" placeholder="Enter Product
Description"

  ></textarea >

</div>
<div >
  <input />
  <p>Drag 'n' drop your image file here, or click
to select files</p>
</div>

<h4>File Details</h4>
<ul>{}</ul>
<div className='d-flex justify-content-center'>
  <center>
    <button style={{marginRight:
'10px'}}type="submit" className="btn btn-primary"><svg
xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor"
class="bi bi-cloud-download" viewBox="0 0 16 16">
      <path d="M4.406 1.342A5.53 5.53 0 0 1 8 0c2.69 0
4.923 2 5.166 4.579C14.758 4.804 16 6.137 16 7.773 16 9.569 14.502 11 12.687
11H10a.5.5 0 0 1 0-1h2.688C13.979 10 15 8.988 15 7.773c0-1.216-1.02-2.228-2.313-
2.228h-.5v-.5C12.188 2.825 10.328 1 8 1a4.53 4.53 0 0 0-2.941 1.1c-.757.652-1.153
1.438-1.153 2.055v.448l-.445.049C2.064 4.805 1 5.952 1 7.318 1 8.785 2.23 10

```

```

3.781 10H6a.5.5 0 0 1 0 1H3.781C1.708 11 0 9.366 0 7.318c0-1.763 1.266-3.223
2.942-3.593.143-.863.698-1.723 1.464-2.383z"/>
      <path d="M7.646 15.854a.5.5 0 0 0 .708 0l3-3a.5.5
0 0 0-.708-.708l8.5 14.293V5.5a.5.5 0 0 0-1 0v8.793l-2.146-2.147a.5.5 0 0 0-
.708.708l3 3z"/>
    </svg> Update</button>

    <button type="submit" className="btn btn-
danger"><svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-trash" viewBox="0 0 16 16">
      <path d="M5.5 5.5A.5.5 0 0 1 6 6v6a.5.5 0 0 1-1
0V6a.5.5 0 0 1 .5-.5zm2.5 0a.5.5 0 0 1 .5.5v6a.5.5 0 0 1-1 1-1 0V6a.5.5 0 0 1 .5-
.5zm3 .5a.5.5 0 0 0-1 0v6a.5.5 0 0 1 0V6z"/>
      <path fill-rule="evenodd" d="M14.5 3a1 1 0 0 1-1
1H13v9a2 2 0 0 1-2 2H5a2 2 0 0 1-2 2V4h-.5a1 1 0 0 1-1-1V2a1 1 0 0 1 1-1H6a1 1 0
0 1 1-1h2a1 1 0 0 1 1 1h3.5a1 1 0 0 1 1 1v1zM4.118 4 4.059V13a1 1 0 0 0 1 1h6a1
1 0 0 0 1-1V4.059L11.882 4H4.118zM2.5 3V2h11v1h-11z"/>
    </svg> Delete </button>
  </center>
</div>

  <br />
</form>

</div>
<Footer />
</div>
)
}

export default EditProduct;

```

addProduct.jsx

```

import React, { useMemo, useState } from 'react';
import { Link } from 'react-router-dom';
import Footer from '../../layouts/footer';
import Header from '../../layouts/header';
import owner from '../../apis/modules/owner'
import { useDropzone } from "react-dropzone";
import { Formik, Form, Field } from 'formik'
import { ProductSchema } from "../../validations";

```



```

const Home_Owner = () => {
  // eslint-disable-next-line react-hooks/exhaustive-deps
  const baseStyle = {
    flex: 1,
    display: "flex",
    flexDirection: "column",
    alignItems: "center",
    padding: "90px",
    borderWidth: 2,
    borderRadius: 2,
    borderColor: "#A9A9B0",
    borderStyle: "dashed",
    marginBottom: "20px",
    backgroundColor: "#ffffff",
    color: "default",
    outline: "none",
    transition: "border .24s ease-in-out",
  };
  // eslint-disable-next-line react-hooks/exhaustive-deps
  const activeStyle = {
    borderColor: "#2196f3",
  };
  // eslint-disable-next-line react-hooks/exhaustive-deps
  const acceptStyle = {
    borderColor: "#00e676",
  };
  // eslint-disable-next-line react-hooks/exhaustive-deps
  const rejectStyle = {
    borderColor: "#ff1744",
  };

  const [files, setFiles] = useState([]);
  const [description, setDescription] = useState('');
  const [category, setCategory] = useState('A');

  //This is used to drag and drop image
  const { acceptedFiles, getRootProps, getInputProps, isDragActive,
isDragAccept, isDragReject, open } = useDropzone({
    onDrop: (acceptedFiles) => {
      setFiles(
        acceptedFiles.map((file) =>
          Object.assign(file, {
            preview: URL.createObjectURL(file),
          })
        )
      );
    }
  });

```

```

    )
  );
},
});

//This is used style drag and drop image
const style = useMemo(
  () => ({
    ...baseStyle,
    ...(isDragActive ? activeStyle : {}),
    ...(isDragAccept ? acceptStyle : {}),
    ...(isDragReject ? rejectStyle : {}),
  }),
  [baseStyle, isDragActive, activeStyle, isDragAccept, acceptStyle,
isDragReject, rejectStyle]
);
const filepath = acceptedFiles.map((file) => (
  <li key={file.name}>
    {file.name} - {file.size} bytes
  </li>
));

//Save product
const addProduct = async (product) => {
  try {
    console.log(acceptedFiles[0])
    let formdata = new FormData();
    formdata.append("photo", acceptedFiles[0]);
    formdata.append("name", product.name);
    formdata.append("description", description);
    formdata.append("price", product.price);
    formdata.append("sku", product.sku);
    formdata.append("size", product.size);
    formdata.append("stock", product.stock);
    formdata.append("category", category);

    await owner.createProduct(formdata)

    window.location = '/homeowner'
  } catch (e) {
  }
}
}

```

```

return (
  <>
    <Header />
    
    <div className='container'>
      <div className='shopc'>
        <h1>ADD PRODUCT</h1>
        <p><Link to="/">Home</Link> / Add Product</p>
      </div>
      <Formik initialValues={{
        stock: '',
        size: '',
        category: '',
        price: '',
        sku: '',
        description: '',
        name: '',
      }}
        validationSchema={ProductSchema}
        onSubmit={values => {
          addProduct(values)
        }}
      >
        <{({ errors, touched }) => (
          <Form>
            <div className="row mb-4">
              <div className="col">
                <div className="form-group">
                  <label style={{ fontWeight: 'bold',
color: '#5D5D5D' }} className="form-label">*Product Name</label>
                  <Field type="text" name="name" id="name"
class="form-control" placeholder="Product name" />
                  {errors.name && touched.name ? <p
id={"login-error"} class="text-danger mt-1">{errors.name}</p> : null}
                </div>
              </div>
              <div className="col">
                <div className="form-outline">
                  <label style={{ fontWeight: 'bold',
color: '#5D5D5D' }}
                    className="form-label">*SKU</label>
                  <Field type="text" id="sku" name="sku"
className="form-control"
                    placeholder="Enter SKU" />

```

```

{errors.sku && touched.sku ?
  <p id={"login-error"}
className="text-danger mt-1">{errors.sku}</p> : null}
</div>
</div>
</div>
<div className="form-outline mb-4">
  <label style={{ fontWeight: 'bold', color:
'#5D5D5D' }}
    className="form-label">*Quantity</label>
  <Field type="text" id="size" name="size"
className="form-control"
    placeholder="Enter Quantity" />
  {errors.size && touched.size ?
    <p id={"login-error"} className="text-danger
mt-1">{errors.size}</p> : null}
  </div>
  <div className="form-outline mb-4">
    <label style={{ fontWeight: 'bold', color:
'#5D5D5D' }} className="form-label">*Price</label>
    <Field type="number" id="price" name="price"
className="form-control" placeholder="Enter Price" />
    {errors.price && touched.price ?
      <p id={"login-error"} className="text-danger
mt-1">{errors.price}</p> : null}
    </div>
    <div className="form-outline mb-4">
      <label style={{ fontWeight: 'bold', color:
'#5D5D5D' }} className="form-label">*Stock</label>
      <Field type="number" id="stock" name="stock"
className="form-control" placeholder="Enter Stock Quantity" />
      {errors.stock && touched.stock ?
        <p id={"login-error"} className="text-danger
mt-1">{errors.stock}</p> : null}
      </div>
      <div className="form-outline mb-4">
        <label style={{ fontWeight: 'bold', color:
'#5D5D5D' }} className="form-label">*Description</label>
        <textarea className="form-control"
id="description" name="description" rows="5" placeholder="Enter Product
Description"
          onChange={(e) => {
setDesctription(e.target.value) }}></textarea>
        </div>

```

```

        <div hidden={filepath.length > 0} {...getRootProps({
style }})>
            <input {...getInputProps()} />
            <p>Drag 'n' drop your image file here, or click
to select files</p>
        </div>

        <h4>File Details</h4>
        <ul>{filepath}</ul>
        <center>
            <button type="submit" className="btn btn-
primary"><svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-cloud-plus-fill" viewBox="0 0 16 16">
                <path d="M8 2a5.53 5.53 0 0 0-3.594 1.342c-
.766.66-1.321 1.52-1.464 2.383C1.266 6.095 0 7.555 0 9.318 0 11.366 1.708 13
3.781 13h8.906C14.502 13 16 11.57 16 9.773c0-1.636-1.242-2.969-2.834-3.194C12.923
3.999 10.69 2 8 2zm.5 4v1.5H10a.5.5 0 0 1 0 1H8.5V10a.5.5 0 0 1-1 0V8.5H6a.5.5 0
0 1 0-1h1.5V6a.5.5 0 0 1 1 0z"/>
            </svg> Add Product</button>
        </center>
        <br />
    </Form>
    )}
</Formik>
</div>
<Footer />
</>
);
};

export default Home_Owner;

```

Test.js

```

import React from 'react';
import { Formik, Form, Field } from 'formik';
import * as Yup from 'yup';

const SignupSchema = Yup.object().shape({
  firstName: Yup.string()
    .min(2, 'Too Short!')
    .max(50, 'Too Long!')

```

```

        .required('Required'),
    lastName: Yup.string()
        .min(2, 'Too Short!')
        .max(50, 'Too Long!')
        .required('Required'),
    email: Yup.string().email('Invalid email').required('Required'),
  });

export const ValidationSchemaExample = () => (
  <div>
    <h1>Signup</h1>
    <Formik>
      initialValues={{
        firstName: '',
        lastName: '',
        email: '',
      }}
      validationSchema={SignupSchema}
      onSubmit={values => {
        // same shape as initial values
        console.log(values);
      }}
    >
    {({ errors, touched }) => (
      <Form>
        <Field name="firstName" />
        {errors.firstName && touched.firstName ? (
          <div>{errors.firstName}</div>
        ) : null}
        <Field name="lastName" />
        {errors.lastName && touched.lastName ? (
          <div>{errors.lastName}</div>
        ) : null}
        <Field name="email" type="email" />
        {errors.email && touched.email ? <div>{errors.email}</div> :
null}

        <button type="submit">Submit</button>
      </Form>
    )}
  </Formik>
</div>
);

```

App.js

```
import React from "react";
import Router from "../Router"
import axios from "axios";
import { AuthContextProvider } from "../src/context/AuthContext";
import { CartProvider } from "react-use-cart";

axios.defaults.withCredentials = true;

function App() {
  return (
    <CartProvider>
      <AuthContextProvider>
        <Router />
      </AuthContextProvider>
    </CartProvider>
  );
}

export default App;
```

index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

ReactDOM.render(
  <React.StrictMode>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
```

```

gg0yR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQU0hcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous"></link>
  <link href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-
awesome.min.css" rel="stylesheet" integrity="sha384-
wvfXpqpZZVQGK6TAh5PVlGOfQNHSoD2xbE+QkPxCAF1NEevoEH3Sl0sibVcOQVnN"
crossorigin="anonymous"></link>
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no"></meta>
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
  <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-
U02eT0CpHqdsSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy60rQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
  <App />
</React.StrictMode>,
document.querySelector('#root')
);

```

Router.jsx

```

import React, { useContext } from 'react';
import LandingPage from "../landing_page/landingPage"
import { Login } from "../views/auth/login"
import ClientRegistration from "../views/auth/clientRegistration"
import OwnerRegistration from "../views/auth/ownerRegistration"
import { ValidationSchemaExample } from '../views/Test'
import {
  BrowserRouter as Router,
  Route,
} from "react-router-dom";
import ClientHome from '../views/client/home/home';
import AddProduct from '../views/owner/home/addProduct';
import OwnerHome from '../views/owner/home/home';

```



```

import AuthContext from "../context/AuthContext";
import Cart from '../views/client/cart/cart';
import Cardpay from '../views/client/payment/cardpay';
import Itemview from '../views/itemview/itemview';
import EditProduct from '../views/owner/home/editProduct';

const Routers = () => {
  const { loggedIn } = useContext(AuthContext);

  return (
    <Router>

      <Route exact path="/login"><Login /></Route>
      <Route exact path="/register"><ClientRegistration /></Route>
      <Route exact path="/owner-register"><OwnerRegistration /></Route>
      <Route exact path="/" component={LandingPage} />

      {
        loggedIn !== null && (<>
          <Route exact path="/homeclient"><ClientHome /></Route>
          <Route exact path="/homeowner"><OwnerHome /></Route>
          <Route exact path="/add-product"><AddProduct /></Route>
          <Route exact path="/itemview/:id"><Itemview /></Route>
          <Route exact path="/cart"><Cart /></Route>
          <Route exact path="/edit-product"><EditProduct /></Route>
        </>)
      }

      <Route exact path="/test"><ValidationSchemaExample /></Route>

    </Router>
  );
};

export default Routers;

```

package.json

```

{
  "name": "frontend",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.16.3",
    "@testing-library/react": "^12.1.4",

```

```
"@testing-library/user-event": "^13.5.0",
"axios": "^0.27.2",
"formik": "^2.2.9",
"js-cookie": "^3.0.1",
"react": "^17.0.2",
"react-dom": "^17.0.2",
"react-dropzone": "^14.2.1",
"react-redux": "^7.2.4",
"react-router-dom": "^5.2.0",
"react-scripts": "4.0.3",
"react-use-cart": "^1.13.0",
"redux": "^4.1.0",
"redux-thunk": "^2.3.0",
"validator": "^13.6.0",
"web-vitals": "^1.1.2",
"yup": "^0.32.11"
},
"scripts": {
  "serve": "react-scripts start",
  "build": "react-scripts build",
  "test": "react-scripts test",
  "eject": "react-scripts eject"
},
"eslintConfig": {
  "extends": [
    "react-app",
    "react-app/jest"
  ]
},
"browserslist": {
  "production": [
    ">0.2%",
    "not dead",
    "not op_mini all"
  ],
  "development": [
    "last 1 chrome version",
    "last 1 firefox version",
    "last 1 safari version"
  ]
}
}
```

- END -