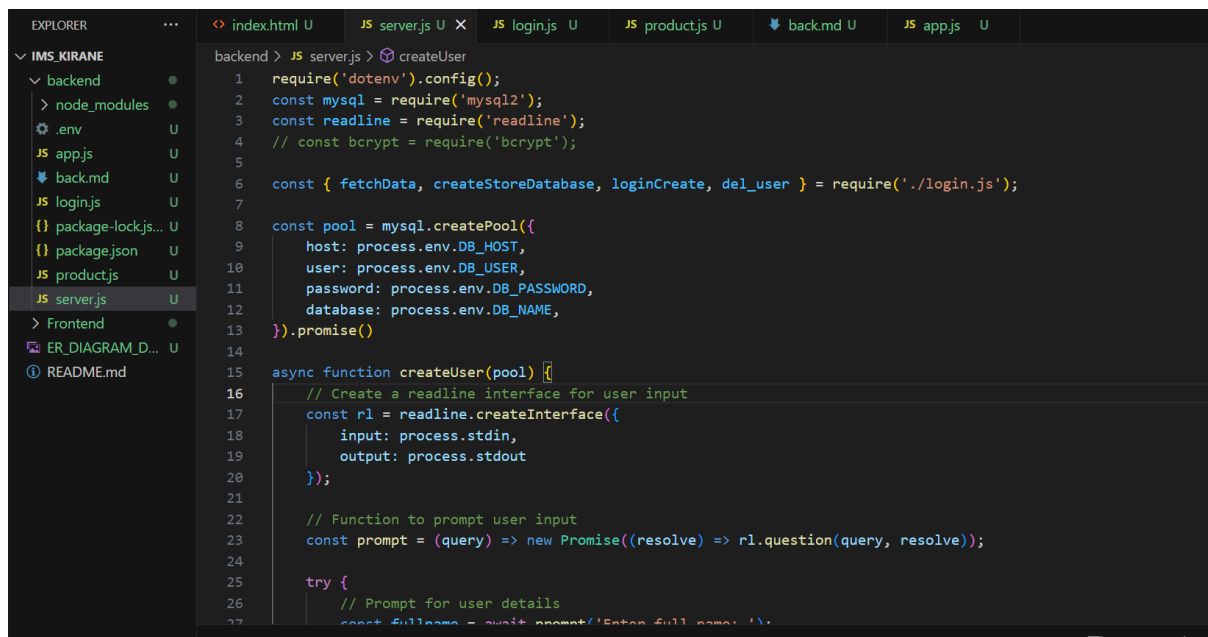


**Department of Computer Science and Engineering****B.Tech (CSE)-5th Semester-Aug-Dec2024****UE22CS341A – Software Engineering****IMPLEMENTATION DOCUMENT****KIRANA STORE MANAGEMENT SYSTEM****Team #: _****Deliverable 2: Implementation details**

PES1UG22CS478	Rishith P
PES1UG22CS443	Prathik S Hanji

Implementation Of Plan: Inventory Management System (IMS) for Kirana Stores

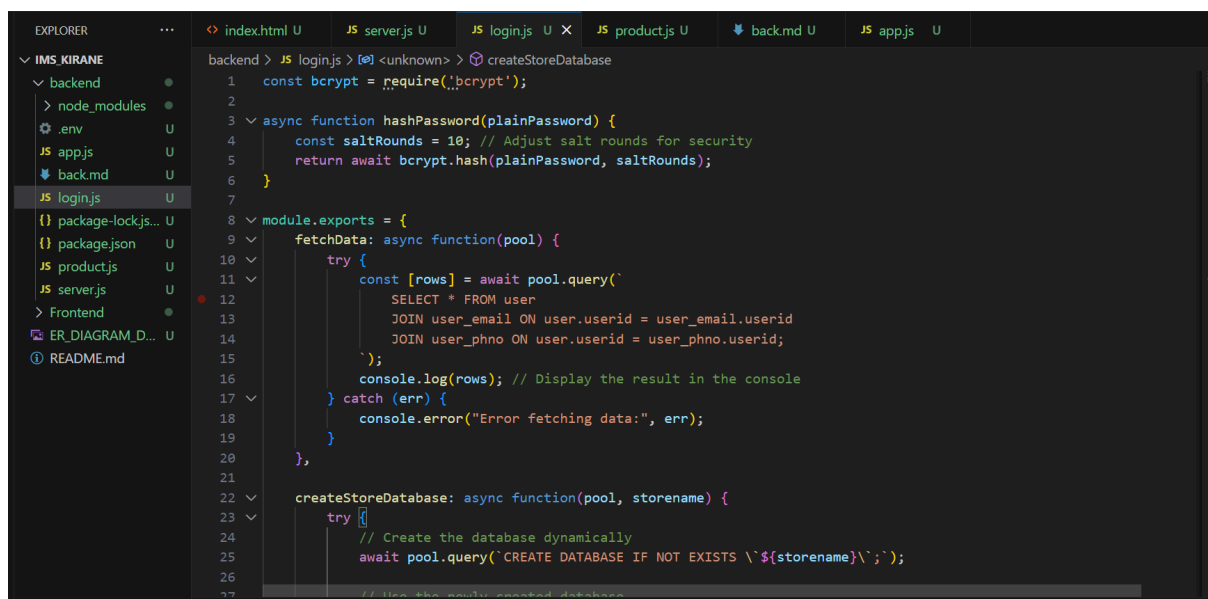


```
EXPLORER  ...  < index.html U  JS server.js U X  JS login.js U  JS product.js U  back.md U  JS app.js U

IMS_KIRANE
├── backend
│   ├── node_modules
│   ├── .env
│   ├── app.js
│   ├── back.md
│   ├── login.js
│   ├── package-lock.js...
│   ├── package.json
│   ├── product.js
│   └── server.js
├── Frontend
├── ER_DIAGRAM_D...
└── README.md

backend > JS server.js > create user
1  require('dotenv').config();
2  const mysql = require('mysql2');
3  const readline = require('readline');
4  // const bcrypt = require('bcrypt');
5
6  const { fetchData, createStoreDatabase, loginCreate, del_user } = require('./login.js');
7
8  const pool = mysql.createPool({
9      host: process.env.DB_HOST,
10     user: process.env.DB_USER,
11     password: process.env.DB_PASSWORD,
12     database: process.env.DB_NAME,
13 }).promise()
14
15 async function createuser(pool) {
16     // Create a readline interface for user input
17     const rl = readline.createInterface({
18         input: process.stdin,
19         output: process.stdout
20     });
21
22     // Function to prompt user input
23     const prompt = (query) => new Promise((resolve) => rl.question(query, resolve));
24
25     try {
26         // Prompt for user details
27         const fullname = await prompt('Enter full name: ');
```

server.js

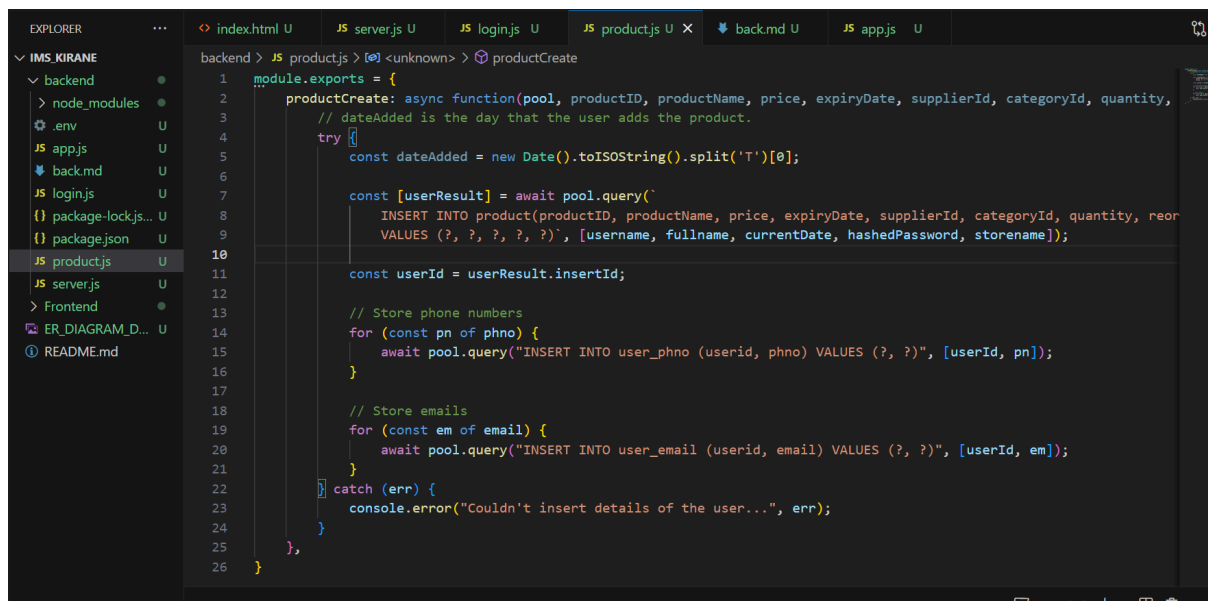


```
EXPLORER  ...  < index.html U  JS server.js U  JS login.js U X  JS product.js U  back.md U  JS app.js U

IMS_KIRANE
├── backend
│   ├── node_modules
│   ├── .env
│   ├── app.js
│   ├── back.md
│   ├── login.js
│   ├── package-lock.js...
│   ├── package.json
│   ├── product.js
│   ├── server.js
│   └── Frontend
├── ER_DIAGRAM_D...
└── README.md

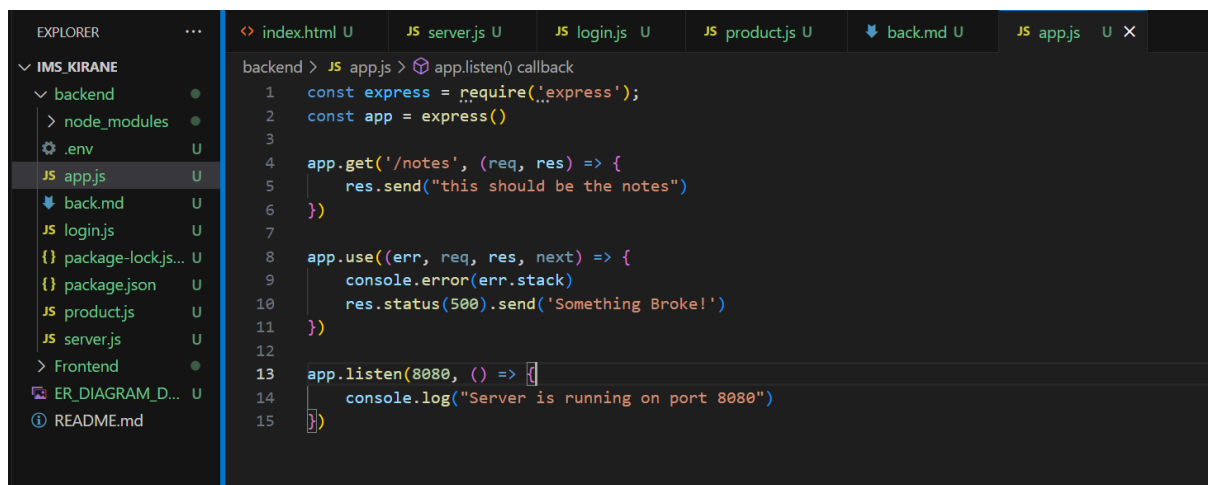
backend > JS login.js > <unknown> > createStoreDatabase
1  const bcrypt = require('bcrypt');
2
3  async function hashPassword(plainPassword) {
4      const saltRounds = 10; // Adjust salt rounds for security
5      return await bcrypt.hash(plainPassword, saltRounds);
6  }
7
8  module.exports = {
9      fetchData: async function(pool) {
10         try {
11             const [rows] = await pool.query(`
12                 SELECT * FROM user
13                 JOIN user_email ON user.userid = user_email.userid
14                 JOIN user_phno ON user.userid = user_phno.userid;
15             `);
16             console.log(rows); // Display the result in the console
17         } catch (err) {
18             console.error("Error fetching data:", err);
19         }
20     },
21
22     createStoreDatabase: async function(pool, storename) {
23         try {
24             // Create the database dynamically
25             await pool.query(`CREATE DATABASE IF NOT EXISTS \`${storename}\``);
26         }
27     }
28 }
```

login.js



```
1 module.exports = {
2   productCreate: async function(pool, productID, productName, price, expiryDate, supplierId, categoryId, quantity,
3     // dateAdded is the day that the user adds the product.
4     try {
5       const dateAdded = new Date().toISOString().split('T')[0];
6
7       const [userResult] = await pool.query(`
8         INSERT INTO product(productID, productName, price, expiryDate, supplierId, categoryId, quantity, reor
9         VALUES (?, ?, ?, ?, ?), [username, fullname, currentDate, hashedPassword, storename]);
10
11       const userId = userResult.insertId;
12
13       // Store phone numbers
14       for (const pn of phno) {
15         await pool.query("INSERT INTO user_phno (userid, phno) VALUES (?, ?)", [userId, pn]);
16       }
17
18       // Store emails
19       for (const em of email) {
20         await pool.query("INSERT INTO user_email (userid, email) VALUES (?, ?)", [userId, em]);
21       }
22     } catch (err) {
23       console.error("Couldn't insert details of the user...", err);
24     }
25   },
26 }
```

product.js



```
1 const express = require('express');
2 const app = express()
3
4 app.get('/notes', (req, res) => {
5   res.send("this should be the notes")
6 })
7
8 app.use((err, req, res, next) => {
9   console.error(err.stack)
10   res.status(500).send('Something Broke!')
11 })
12
13 app.listen(8080, () => {
14   console.log("Server is running on port 8080")
15 })
```

app.js