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
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
index.html U | server.js U | login.js U | product.js U | back.md U | app.js U
backend > server.js > createUser
```

```
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
```

server.js

```
EXPLORER
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
index.html U | server.js U | login.js U | product.js U | back.md U | app.js U
backend > 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
```

login.js

```
EXPLORER
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

index.html U | server.js U | login.js U | product.js U | back.md U | app.js U
backend > product.js > <unknown> > productCreate

module.exports = {
    productCreate: async function(pool, productID, productName, price, expiryDate, supplierId, categoryId, quantity,
        // dateAdded is the day that the user adds the product.
        try {
            const dateAdded = new Date().toISOString().split('T')[0];

            const [userResult] = await pool.query(`
                INSERT INTO product(productID, productName, price, expiryDate, supplierId, categoryId, quantity, reor
                VALUES (?, ?, ?, ?, ?)`, [username, fullname, currentDate, hashedPassword, storename]);

            const userId = userResult.insertId;

            // Store phone numbers
            for (const pn of phno) {
                await pool.query("INSERT INTO user_phno (userid, phno) VALUES (?, ?)", [userId, pn]);
            }

            // Store emails
            for (const em of email) {
                await pool.query("INSERT INTO user_email (userid, email) VALUES (?, ?)", [userId, em]);
            }
        } catch (err) {
            console.error("Couldn't insert details of the user...", err);
        }
    },
}
```

product.js

```
EXPLORER
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

index.html U | server.js U | login.js U | product.js U | back.md U | app.js U
backend > app.js > app.listen() callback

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

app.get('/notes', (req, res) => {
    res.send("this should be the notes")
})

app.use((err, req, res, next) => {
    console.error(err.stack)
    res.status(500).send('Something Broke!')
})

app.listen(8080, () => {
    console.log("Server is running on port 8080")
})
```

app.js