**PROJECT DOCUMENTATION**

***Store Manager: Keep track on the inventory***

**1.Introduction:**

**Project Title:** Store Manager: Keep track on the inventory

**Team id:** NM2025TMID30068

**Team Leader:** SANJANA M & [202400138@sigc.edu](mailto:202400138@sigc.edu)

**Team Members:**

- AISWARIYA S & [202400019@sigc.edu](mailto:202400019@sigc.edu)

-HARINI K & [202400608@sigc.edu](mailto:202400608@sigc.edu)

-RENUGA DEVI I & [202400943@sigc.edu](mailto:202400943@sigc.edu)

**2.Project Overview:**

**Purpose:** The purpose of this project is to manage store inventory digitally and efficiently. It helps reduce manual errors and provides real-time stock tracking and alerts. This ensures smooth store operations and better product availability.

**Features:**

-Inventory Management

-Real-time Stock Tracking.

-Low Stock Alerts

-User Authentication & Roles

-Search & Filter Products

**3.Architecture:**

**Frontend:** React.js / Angular / Vue (or simple HTML, CSS, JS)

**Backend:** Node.js (Express), Django, or Spring Boot

**Database:** MongoDB stores user data, project information, applications, and chat messages, MySQL / PostgreSQL

# **4. Setup Instructions**

* **Prerequisites:**
  + Node.js
  + npm or Yarn
  + MongoDB
  + Code Editor (VSCode / WebStorm etc.)
  + Terminal / Command Line access
  + JavaScript / Node.js / React
* **Installation Steps:**

 **Clone the Repository**

Begin by cloning the repository to your local machine:

git clone https://github.com/larissaperinoto/store-manager.git

 **Navigate to the Project Directory**

Change into the project directory:

cd store-manager/

 **Install Dependencies**

Install the necessary dependencies using npm:

npm install

 **Set Up the Database**

Log in to your MySQL database:

mysql -u <your-username> -p

Run the migration and seed scripts to set up the database:

source migration.sql;

source seed.sql;

 **Start the Server**

Start the server using nodemon:

npm run debug

The application will be running locally.

 **Access the Application**

Open your browser and navigate to:

[http://localhost:3000](http://localhost:3000/)

This will take you to the application's homepage.

# **5. Folder Structure store-manager/**

# ├── app/

# │ ├── Controllers/

# │ ├── Models/

# │ ├── Routes/

# │ └── Services/

# ├── config/

# │ ├── database.js

# │ └── server.js

# ├── public/

# │ └── assets/

# ├── tests/

# │ ├── unit/

# │ └── integration/

# ├── .gitignore

# ├── README.md

# ├── package.json

# └── server.js

***Folder Description:***

* **app/**: Contains the core application logic.
  + **Controllers/**: Handles incoming requests and returns responses.
  + **Models/**: Defines the data structure and interacts with the database.
  + **Routes/**: Defines the API endpoints and maps them to controllers.
  + **Services/**: Contains business logic and interacts with models.
* **config/**: Stores configuration files.
  + **database.js**: Database connection settings.
  + **server.js**: Server configuration and middleware setup.
* **public/**: Holds static assets like images, stylesheets, and scripts.
  + **assets/**: Subfolder for organizing static files.
* **tests/**: Contains test files.
  + **unit/**: Unit tests for individual components.
  + **integration/**: Integration tests for API endpoints.
* **.gitignore**: Specifies files and directories to be ignored by Git.
* **README.md**: Project documentation and setup instructions.
* **package.json**: Manages project dependencies and scripts.
* **server.js**: Entry point for the application, setting up the server and middleware.

**6. Running the Application**

***Frontend***

cd client

npm start

***Backend***

cd server npm start

**Access:** Visit <http://localhost:3000/>

# **7. API Documentation**

1. **User:**

***Register a New User***

POST /api/user/register

***Login***

POST /api/user/login

**2. Projects Management**

***Create a New Project***

POST /api/projects/create

***Get Project by ID***

GET /api/projects/:id

**3. Applications**

***Apply for a Project***

POST /api/apply

**4. Inventory Management**

***Add New Product***

POST /api/products

***Update Product Details***

PUT /api/products/:id

***Delete Product***

DELETE /api/products/:id

***Get Product Details***

GET /api/products/:id

**5. Chats**

***Send a Message***

POST /api/chat/send

***Get Chat History***

GET /api/chat/:userId

# **8. Authentication**

* JWT-based authentication for secure login
* Middleware protects private routes

# **9. User Interface**

* Landing Page
* Freelancer Dashboard
* Inventory Management
* Order Processing
* Customer Management
* Reports & Analytics

# **10. Testing**

* Manual testing
* Tools: Jest, React Testing Library, Mocha, Chai, Supertest, GitHub Actions, Jenkins

**11.Screenshots or Demo**

**[https://drive.google.com/file/d/1jZYWzkF1KQZ3mzGwWIm0Q9HvYMo9xsYL/view?usp=drivesdk](https://drive.google.com/file/d/1mRYTNTKYDzgKPrygpH9JnphKlOdI4W7b/view?usp=drivesdk)**

**12.Known Issues**

1. Inaccurate Demand Forecasting

2. Stock Discrepancies

3. Manual Data Entry Errors

4. Lack of Real-Time Inventory Visibility

**13.Future Enhancements**

1.AI-Driven Demand Forecasting

2.Smart Shelves with RFID Technology

3.Automated Self-Checkout System

4.Enhanced Customer Insights with AI Analytics

5.Headless Commerce Architecture