**Simple Inventory Management System Documentation**

**1. Overview**

This document describes the implementation of a Simple Inventory Management System. The system allows users to add items to the inventory, view the list of items, and update item quantities. The application is built using Angular for the front end and PHP with MySQL for the back end.

**2. Environment Setup**

**2.1 Prerequisites**

* **Operating System:** Windows/Mac/Linux
* **Web Server:** Apache or Nginx
* **Database:** MySQL
* **Backend:** PHP (Version 7.4 or later)
* **Frontend:** Angular (Version 12 or later)
* **Package Manager:** Node.js (for Angular)
* **Browser:** Chrome, Firefox, or any modern browser

**2.2 Installation Steps**

1. **Install Apache and MySQL:**
   * Install XAMPP (Windows/Mac) or LAMP stack (Linux) to set up Apache and MySQL.
2. **Set Up MySQL Database:**
   * Create a new MySQL database named inventory\_db.
   * Import the provided inventory\_db.sql file to create the necessary tables.
3. **Clone the Repository:**
   * Clone the project repository to your local machine.
4. **Set Up Angular Project:**
   * Navigate to the frontend directory.
   * Run npm install to install the necessary dependencies.
   * Run ng serve to start the Angular development server.
5. **Configure Backend:**
   * Place the backend directory inside your web server's root directory (e.g., htdocs for XAMPP).
   * Update the database configuration in config.php with your MySQL credentials.
6. **Run the Application:**
   * Access the Angular frontend via http://localhost:4200.
   * The PHP backend will be accessed through http://localhost/backend/.

**3. Database Structure**

**3.1 Users Table**

| **Field** | **Type** | **Description** |
| --- | --- | --- |
| id | INT | Primary Key, Auto Increment |
| username | VARCHAR(255) | Username for the user |
| password | VARCHAR(255) | Hashed password for the user |
| role | ENUM | Role of the user ('admin', 'user') |

**3.2 Items Table**

| **Field** | **Type** | **Description** |
| --- | --- | --- |
| id | INT | Primary Key, Auto Increment |
| name | VARCHAR(255) | Name of the item |
| description | TEXT | Description of the item |
| quantity | INT | Quantity available in inventory |
| created\_at | TIMESTAMP | Timestamp when the item was created |

**3.3 Inventory\_Logs Table**

| **Field** | **Type** | **Description** |
| --- | --- | --- |
| id | INT | Primary Key, Auto Increment |
| item\_id | INT | Foreign Key referencing Items |
| user\_id | INT | Foreign Key referencing Users |
| change\_type | ENUM | Type of change ('add', 'remove') |
| quantity\_change | INT | Amount added/removed from stock |
| timestamp | TIMESTAMP | Timestamp of the action |

**4. API Endpoints**

**4.1 Add Item to Inventory**

* **Endpoint:** /backend/addItem.php
* **Method:** POST
* **Request Body:**

json

Copy code

{

"name": "Item Name",

"description": "Item Description",

"quantity": 10

}

* **Response:**

json

Copy code

{

"success": true,

"message": "Item added successfully."

}

**4.2 Update Item Quantity**

* **Endpoint:** /backend/updateQuantity.php
* **Method:** POST
* **Request Body:**

json

Copy code

{

"item\_id": 1,

"change\_type": "add",

"quantity\_change": 5,

"user\_id": 1

}

* **Response:**

json

Copy code

{

"success": true,

"message": "Quantity updated successfully."

}

**4.3 Retrieve Items**

* **Endpoint:** /backend/getItems.php
* **Method:** GET
* **Response:**

json

Copy code

{

"items": [

{

"id": 1,

"name": "Item1",

"description": "Sample item",

"quantity": 15

},

...

]

}

**5. Angular Front-End Features**

**5.1 Item Form**

* **Description:** Allows the user to add a new item to the inventory.
* **Validation:** Ensures that all fields (name, description, quantity) are filled and that quantity is a positive integer.
* **Action:** Submits a POST request to the /backend/addItem.php endpoint.

**5.2 Item List**

* **Description:** Displays a list of all items in the inventory.
* **Columns:** Item name, description, and current quantity.
* **Action:** Includes an "Update Quantity" button next to each item.

**5.3 Update Quantity**

* **Description:** Allows the user to update the quantity of an item.
* **Form Fields:** Dropdown to select 'add' or 'remove' and an input for the quantity change.
* **Action:** Submits a POST request to the /backend/updateQuantity.php endpoint.

**6. Testing and Debugging**

**6.1 Testing the API**

* Use Postman or a similar tool to test the API endpoints independently.
* Verify that adding, updating, and retrieving items work as expected.

**6.2 Debugging Tips**

* **Frontend Errors:** Use Chrome DevTools to debug Angular errors.
* **Backend Errors:** Check the Apache error log for PHP errors. Ensure that error reporting is enabled in php.ini.

**7. Conclusion**

The Simple Inventory Management System provides a basic example of a CRUD application using Angular and PHP. This documentation should help you understand the system's structure, how to set it up, and how to interact with its components.