Fogon Inventory Management System (IMS) – User Manual:

https://github.com/Landy2233/Fogon-Inventory-Management-System.git

1. Introduction

The Fogon Inventory Management System (IMS) is a mobile application designed to help restaurant staff efficiently manage inventory, submit item requests, and approve or deny stock movements. The system supports two user roles:

- **Manager** – Full CRUD access, approval/denial of requests.
- **Cook/Employee** – Can request items and view request status.

This user manual guides you through installation, setup, and usage of the application.

---

2. System Requirements

For Development & Running the App

- **Git**
- **Python 3.10+**
- **Node.js + npm + npx**
- **Expo CLI**
- **Xcode (Mac only, for iOS emulator)**
- **Expo Go App (mobile)**

---

3. Installation Instructions

3.1 Install Required Tools (First Time Only)

Follow the installation and version‑check steps for Git, Python, Node.js, npm, npx, Expo, and Xcode (macOS). Install Expo Go on your mobile phone.

---

4. Downloading the Project

```
git clone https://github.com/Landy2233/Fogon-Inventory-Management-System.git
```

```
cd Fogon-Inventory-Management-System
```

5. Backend Setup

5.1 Create Virtual Environment

macOS/Linux:

```
python3 -m venv venv
source venv/bin/activate
```

Windows:

```
python -m venv venv
venv\bin\activate
```

5.2 Install Dependencies

```
pip install --upgrade pip
pip install -r requirements.txt
```

5.3 Initialize the Database (First Time Only)

```
flask --app app init-db
```

5.4 Run the Backend

```
python app.py
```

Backend runs at:

- http://localhost:5001
- http://YOUR_WIFI_IP:5001 (required for mobile access)

Keep this terminal running.

6. Start the Mobile App (Expo)

Open a **new terminal window**.

6.1 Install Mobile Dependencies

```
cd Fogon-Inventory-Management-System/FogonIMSMobile
npm install
```

6.2 Configure Backend URL

Open:
`FogonIMSMobile/src/api/client.ts`

Set your Wi-Fi IP:

```
const DEV_LAN = "http://YOUR_WIFI_IP:5001/api";
```

Find your IP:

- **Windows**: `ipconfig`
- **macOS**: `ifconfig`

6.3 Start Expo

```
npx expo start
```

6.4 Open the App

- **iOS Simulator (Mac)** → Press **i**
- **Expo Go App (iPhone/Android)** → Scan QR code

Ensure phone and laptop are on the **same WiFi network**.

---

7. Using the App

7.1 Logging In

You will be prompted to:

- **Log in**, or
- **Create an account**

Demo users (manager & cook) are included for testing. You may also create your own manager or cooker account.

8. User Roles & Functionality

Cook/Employee

- Submit inventory requests
- View request status (pending/approved/denied)
- View available items

Manager

- View all inventory items
- Create, edit, and delete stock items
- View item request list
- Approve or deny requests

9. Common Workflows

Cook Requests an Item

1. Log in as Cook
2. Select the item from the inventory
3. Submit a request for quantity needed
4. Wait for approval/denial

Manager Approves or Denies Requests

1. Log in as Manager
2. Open the "Requests" tab
3. Review pending requests
4. Select **Approve** or **Deny**
5. Inventory updates automatically when approved

Manager CRUD Operations

- Add new items to stock
- Update item details (quantity, name, category)
- Remove items from stock

10. Troubleshooting

App won’t connect to backend

- Ensure the backend is running at `http://YOUR_WIFI_IP:5001`.
- Ensure both devices are on the same WiFi.
- Recheck `client.ts` DEV_LAN URL.

Expo Go app stuck loading

- Close Expo Go > reopen > rescan QR.
- Restart Expo: `ctrl + c` → `npx expo start`.

Database issues

Reinitialize:

```
flask --app app init-db
```

11. Filing a Bug Report

To report a bug:

1. Go to the GitHub repository.
2. Open the **Issues** tab.
3. Click **New Issue**.
4. Include:
    - Clear description of the bug
    - Steps to reproduce
    - Expected vs actual behavior
    - Screenshots (if helpful)
    - Your device/platform & OS

This ensures maintainers can replicate and resolve the problem efficiently.