**README.md**

# FoodLink AI 🌱🥗

\*\*FoodLink AI\*\* is a hackathon project focused on \*\*reducing food waste\*\* by connecting surplus food providers (restaurants, bakeries, grocery stores) with communities, NGOs, and buyers using \*\*AI-powered matching\*\*. The platform works as a \*\*web and mobile PWA\*\*, includes real-time dashboards, micro-lessons, and optional monetization via \*\*IntaSend\*\*.

---

## Features

- Snap & list surplus food in 30 seconds

- AI-powered matching by distance, freshness, dietary needs

- Reserve & pickup securely

- Impact dashboard: meals saved, partners, revenue

- Mobile-friendly Progressive Web App (PWA)

- Offline-first support with Service Worker

- Demo mode with mock data included

---

## Tech Stack

\*\*Frontend:\*\* HTML5, CSS, JavaScript

\*\*Backend:\*\* Python (FastAPI), MySQL

\*\*PWA:\*\* Service Worker, manifest.webmanifest

\*\*AI & Low-code Tools:\*\* Cursor AI, MetaGPT X, Superbase, Lovable Dev

\*\*Payments:\*\* IntaSend

\*\*Deployment:\*\* Vercel, Heroku (optional)

---

### 1. Clone the Repository

```bash

git clone https://github.com/yourusername/foodlink-ai.git

cd foodlink-ai

**2. Frontend Setup (Demo Mode)**

1. Open frontend/index.html in **VS Code Live Server** or any static server.
2. Navigate to /login.html to sign in with demo credentials.
3. Go to /insight.html to view AI-powered mock items (Chicken Rice Boxes, Bakery Packs, Vegetable Mix Boxes) and impact charts.

**3. Backend Setup (Optional for real API)**

1. Create a **MySQL database** and run answer.sql to create tables.
2. Create a Python virtual environment:

python -m venv .venv

source .venv/bin/activate # Windows: .venv\Scripts\activate

1. Install dependencies:

pip install fastapi uvicorn mysql-connector-python python-jose PyJWT

1. Set environment variables:

export DB\_HOST=localhost

export DB\_USER=root

export DB\_PASS=yourpassword

export DB\_NAME=foodlink

export PAYSTACK\_SECRET=your\_paystack\_key

export SUPABASE\_JWKS=your\_supabase\_jwks

1. Run the backend server:

uvicorn backend.main:app --reload --port 8000

1. Update frontend window.API\_BASE to point to your backend:

window.API\_BASE = "http://localhost:8000";

**4. Deployment**

**Option 1: Vercel**

1. Push repo to GitHub.
2. Connect Vercel and deploy the frontend/ folder.
3. Enable static site deployment and PWA support.

**Option 2: Heroku (for backend)**

1. Push backend folder to GitHub.
2. Create a new Heroku app.
3. Add environment variables via dashboard or CLI.
4. Deploy with GitHub integration or git push heroku main.

**5. GitHub Repository**

[GitHub Repo](https://github.com/yourusername/foodlink-ai?utm_source=chatgpt.com)  
https://github.com/Jsews/hackathon2.git

**6. Deployment Link**

Live Demo Link  
*(Replace with your actual deployed URL)*

**7. Pitch Deck**

https://www.canva.com/design/DAGxt0-C9SU/npNLexSIkpwGqCljCNeFqQ/edit?utm\_content=DAGxt0-C9SU&utm\_campaign=designshare&utm\_medium=link2&utm\_source=sharebutton

**9. File Structure**

foodlink-ai/

│

├── frontend/

│ ├── index.html

│ ├── insight.html

│ ├── login.html

│ ├── style.css

│ ├── script.js

│ ├── sw.js

│ └── manifest.webmanifest

│

├── backend/

│ └── main.py

│

├── answer.sql

└── README.md

**10. Credits**

* Developed by-
* JANICE SEWAVA
* ELVIS KESSY
* ISHENGOMA KAKWEZI
* Hackathon Project: Zero Hunger / Food Sustainability
* Stock images: Unsplash
* Tools: FastAPI, Supabase, Paystack, Cursor AI, MetaGPT X, Lovable Dev

**11. Notes & Tips**

* Use **demo mode** for immediate web/mobile preview.
* Connect **Supabase Auth** for real login functionality.
* Enable **HTTPS** in production for secure payments and service worker.
* Optional: Add **animated impact charts** and more food categories for visual appeal.