Farmers Tool Belt – A Web-Based Agricultural Recommendation System for Malawi

1. Project Purpose & Target Users

The Farmers Tool Belt is a web app built to assist Malawian farmers in choosing the right crops, seed varieties, fertilizers, and suppliers based on their district. It serves all 28 districts and helps verified suppliers connect with local farmers. The app simplifies agricultural decisions by providing tailored, location-based recommendations.

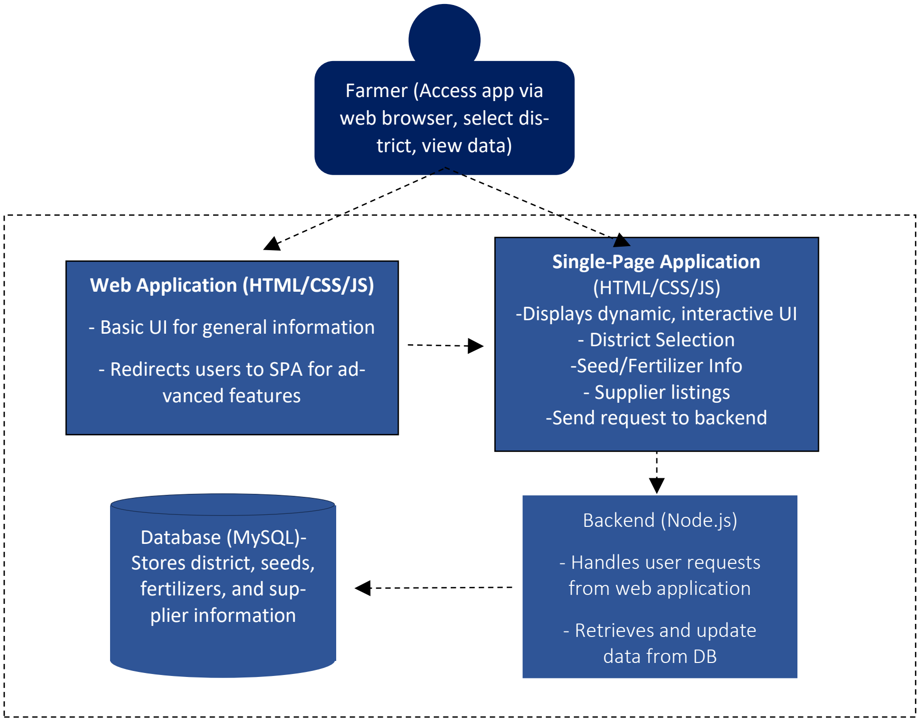
2. Technologies Used

Component	Technology
Frontend	HTML, CSS, JavaScript
Backend	Node.js with Express
Database	MySQL (local)
Tools	Visual Studio Code, GitHub

The frontend is responsive and mobile-friendly. The backend connects to a structured MySQL database to return accurate data

3. System Architecture

High-Level Architecture (Container Diagram) Farmers Tool Belt


Farmer (Access app via web browser, select district, view data)
Web Application (HTML/CSS/JS)
- Basic UI for general information
- Redirects users to SPA for advanced features
Single-Page Application (HTML/CSS/JS)
-Displays dynamic, interactive UI
- District Selection
-Seed/Fertilizer Info
- Supplier listings
-Send request to backend
Database (MySQL)- Stores district, seeds, fertilizers, and supplier information
Backend (Node.js)
- Handles user requests from web application
- Retrieves and update data from DB

The system follows a three-layer model:

- Frontend: A single-page interface with dropdowns and buttons to view tailored crop recommendations.
- Backend API: Built with Express and handles requests using two main routes.
- Database: A normalized MySQL structure linking districts, crops, fertilizers, and suppliers.

4. Development Process

- JavaScript modules manage section transitions and user input.
- Data is fetched from the backend and displayed dynamically using fetch API.
- SQL queries were designed to join multiple tables and return filtered results.
- CSS styling ensures a clean interface that adjusts across screen sizes

5. Feedback Implementation

- **Screencast Missing in Phase 2**: A new screencast has been created and added under 03-final_video/.
- Code remains unchanged since architecture and flow were approved in the previous phase.

6. Key Challenges

- Designing dynamic interactions while ensuring accessibility and responsiveness.
- Managing SQL queries for nested data (district → crop → variety → fertilizers/suppliers).
- Debugging database joins to ensure accurate district-based filtering.

7. Lessons Learned

- Gained experience in connecting frontend and backend using REST APIs.
- Learned how to manage async data and user feedback using fetch() and Promises.
- Improved understanding of database structure and how to support scalable queries.