**Software Requirements Specification**

**Project Name:** *Web Solution for Horticulture Crop specific Market, Storage, Value Addition, and Price*

**Course Code: INT220**

**Course Name: Server Side Scripting**

Prepared for

Continuous Assessment 3

Spring 2025



**Submitted by :-**

**Student Names and Registration Nos.**

1. **Chiluveru shiva prasad– (12304258)**
2. **Raj Kumar – (12311580)**
3. **Saurabh Vishwakarma – (12312583)**
4. **Giduturi Girivardhan – (12310977)**

**Table of Contents**

1. Introduction 3

1.1 Purpose 3

1.2 Scope 3

1.3 Definitions, Acronyms, and Abbreviations 4

1.4 References 4

1.5 Overview 5

2. General Description 5

2.1 Product Perspective 5

2.2 Product Functions 5

2.3 User Characteristics 6

2.4 General Constraints 6

2.5 Assumptions and Dependencies 7

3. Specific Requirements 7

3.1 External Interface Requirements 7

3.1.1 User Interfaces 7

3.1.2 Hardware Interfaces 8

3.1.3 Software Interfaces 8

3.1.4 Communications Interfaces 3

3.2 Functional Requirements 8

3.2.1 <Functional Requirement or Feature #1> 9

3.2.2 <Functional Requirement or Feature #2> 9

3.5 Non-Functional Requirements 9

3.5.1 Performance 9

3.5.2 Reliability 10

3.5.3 Availability 10

3.5.4 Security 10

3.5.5 Maintainability 10

3.5.6 Portability 10

3.7 Design Constraints 10

3.9 Other Requirements 10

4. Analysis Models………………………………………………………………………………………..11-13

4.1 Data Flow Diagrams (DFD) 11-13

5. Github link………………………………………………………………………………………..………….14

**6. DEPLOYED LINK………………………………………………………………………………………………...............................................14**

7. CLIENT APPROVAL PROOF……………..…………………………………………………………………N/A

**8. CLIENT LOCATION PROOF…………………………………………………………………………………………………………………………..N/A**

**9. TRANSACTION ID PROOF……………………………………………………………………………………........................................N/A**

**10. EMAIL ACKNOWLEDGEMENT…………………………………………………………………………………………………………….…….N/A**

**11. GST No…………………………………………………………………………………………………………............................................N/A**

A. Appendices…………………………………………………………………………………………………..14

A.1 Appendix 1………………………………………………………………………………………………..14-16

**1. INTRODUCTION**

**1.1 PURPOSE**

The purpose of this document is to describe the design, functionality, and technical structure of a web-based solution developed specifically for the horticultural sector. The system is intended to assist farmers, traders, and stakeholders by providing comprehensive information about crop-specific markets, storage facilities, value addition opportunities, and real-time price tracking. By leveraging web technologies like HTML, Tailwind CSS, JavaScript, PHP, and MySQL, this solution aims to simplify access to agricultural intelligence, increase profitability, and support better decision-making.

This Software Requirements Specification (SRS) outlines what the system will do, the design constraints, performance parameters, and interfaces to external systems. It will serve as a foundation for development, testing, and deployment.

**1.2 SCOPE**

This web application is a dynamic and responsive system that caters to the specific needs of horticultural crop stakeholders. The scope includes:

* **Crop-Specific Market Access:** Users can select a particular horticultural crop and view the latest market prices across regions.
* **Storage Finder:** A module that allows users to locate nearby storage facilities such as cold storages, warehouses, and pack houses.
* **Value Addition Resources:** The system provides educational materials such as videos, articles, and best practices for converting raw produce into value-added products (e.g., jams, juices, dried fruits).
* **Price Trends and Analytics:** Displays historical and seasonal price data using interactive graphs and visualizations.
* **User Dashboard:** Registered users (farmers, traders) can save preferences, access personalized suggestions, and manage their profiles.
* **Admin Dashboard:** Admins can update crop data, market trends, and manage users via a secure backend interface.

**1.3 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS**

| **Term** | **Definition** |
| --- | --- |
| **HTML** | HyperText Markup Language – Used to structure the content of web pages |
| **CSS** | Cascading Style Sheets – Used for styling HTML content |
| **Tailwind CSS** | A utility-first CSS framework used for building responsive and modern UI designs |
| **JS / JavaScript** | A scripting language used to add interactivity to web pages |
| **PHP** | Hypertext Preprocessor – A server-side scripting language used for backend logic |
| **MySQL** | An open-source relational database management system |
| **SRS** | Software Requirements Specification |
| **DFD** | Data Flow Diagram – A visual representation of data flow in the system |
| **UI** | User Interface |
| **UX** | User Experience |
|  |  |

**1.4 REFERENCES**

* [Tailwind CSS Documentation](https://tailwindcss.com/docs)
* [PHP Manual](https://www.php.net/manual/en/)
* [MySQL Documentation](https://dev.mysql.com/doc/)
* National Horticulture Board (NHB) – <https://nhb.gov.in>
* Agricultural Market Information System (Agmarknet) – <https://agmarknet.gov.in>
* FAO Value Addition Toolkit for Horticulture

**1.5 OVERVIEW**

This document is organized into several major sections to provide a detailed understanding of the system. Section 2 explains the general description and the product’s context. Section 3 outlines the specific functional and non-functional requirements. Section 4 includes analysis models such as Data Flow Diagrams. Sections 5 through 11 provide validation materials such as the GitHub repository, deployment link, client approval, and various proof documents. The Appendices provide supporting screenshots, diagrams, and additional resources.

**2. GENERAL DESCRIPTION**

**2.1 PRODUCT PERSPECTIVE**

This system is a **standalone web-based solution** but can be integrated with external agricultural data sources such as government APIs (e.g., Agmarknet, NHB) for enhanced functionality. It is developed as a centralized platform to help various horticulture stakeholders access relevant information easily and efficiently.

The system consists of multiple integrated modules:

* **Frontend** built using **HTML**, **Tailwind CSS**, and **JavaScript** provides a clean, modern, and responsive interface.
* **Backend** built using **PHP** handles server-side logic such as user authentication, data retrieval, and storage management.
* **Database** managed using **MySQL** stores all user data, crop price records, storage facility details, and historical trends.

**2.2 PRODUCT FUNCTIONS**

The main functions of the system include:

* **Crop Market Module**  
  Displays live market prices of selected horticultural crops from various markets across regions.
* **Storage Information Module**  
  Helps users find nearby cold storages and warehouses using location-based filtering.
* **Value Addition Knowledge Base**  
  Provides farmers with guides, videos, and articles to add value to their crops (e.g., drying, juicing, packaging).
* **Price Trend Analysis**  
  Shows historical pricing data and generates visualizations like line graphs and bar charts.
* **User Registration and Login**  
  Farmers, traders, and other stakeholders can create accounts and access personalized dashboards.
* **Admin Dashboard**  
  Admins can update crop databases, price feeds, and manage users securely.

**2.3 USER CHARACTERISTICS**

The primary users of this application include:

* **Farmers**  
  They may have limited digital literacy but are increasingly familiar with smartphones. The UI is designed to be simple and intuitive.
* **Traders and Buyers**  
  Require market data and trends to make informed purchasing decisions.
* **Storage Facility Managers**  
  Can register their storage units and keep information updated for others to discover.
* **Agri-extension Officers / NGOs**  
  Use the system to train farmers and disseminate value addition knowledge.
* **Researchers & Analysts**  
  Can use the platform to study market dynamics and farmer behavior over time.

**2.4 GENERAL CONSTRAINTS**

* **Network Dependency**  
  The application requires a stable internet connection to function effectively, especially for fetching live data and media content.
* **Device Compatibility**  
  The web app is optimized for major browsers (Chrome, Firefox, Edge) and supports both desktop and mobile screens. Very old devices may face performance issues.
* **Data Accuracy**  
  Since market data may be sourced manually or through third-party APIs, occasional discrepancies may arise.
* **Language Limitations**  
  The initial version is in English only. Support for regional languages can be considered in future versions.

**2.5 ASSUMPTIONS AND DEPENDENCIES**

* **Assumptions**
  + Users have basic familiarity with internet use and smartphones.
  + Admins will regularly update price and storage information in the backend.
  + Farmers and storage managers are willing to engage with the platform.
* **Dependencies**
  + Reliable database connectivity for real-time data access.
  + Access to third-party APIs for live crop prices (if integrated).
  + Hosting server with PHP and MySQL support.
  + SMS or email gateways (optional) for user notifications.

**3. SPECIFIC REQUIREMENTS**

**3.1 EXTERNAL INTERFACE REQUIREMENTS**

**3.1.1 User Interfaces**

* **Home Page**  
  Displays introduction, featured crops, key features, and navigation to all modules.
* **Login & Registration Pages**  
  Enables users (farmers, traders, admin) to sign up and log in with basic information.
* **Dashboard (User)**  
  Displays crop prices, value addition resources, suggested storage, and personal bookmarks.
* **Admin Panel**  
  Allows admin to:
  + Add/update/delete crop information
  + Upload price data
  + Manage storage facility entries
  + Manage users
* **Responsive UI**  
  Developed using **Tailwind CSS** to ensure the layout adjusts smoothly across mobile, tablet, and desktop devices.

**3.1.2 Hardware Interfaces**

* Client-side: Mobile phones, tablets, desktops with internet browsers.
* Server-side: Hosted on a PHP-supporting web server with MySQL database connectivity.

**3.1.3 Software Interfaces**

* **Frontend Technologies**: HTML, Tailwind CSS, JavaScript
* **Backend Technologies**: PHP
* **Database**: MySQL
* **Web Hosting**: Deployed using a LAMP stack (Linux, Apache, MySQL, PHP).

**3.1.4 Communications Interfaces**

* HTTP/HTTPS protocols for secure client-server communication.
* (Optional) Email/SMS API for alerting users about price changes or storage availability.

**3.2 FUNCTIONAL REQUIREMENTS**

**3.2.1 Crop Price Module**

* Users can select a crop and view the latest market prices.
* Filters by location and date range.
* Graphical trends using JavaScript (e.g., Chart.js).

**3.2.2 Storage Finder Module**

* Displays nearby storage units based on user location.
* Storage data includes type (cold storage, warehouse), capacity, contact info, and availability status.
* Admins can add/update storage listings.

**3.2.3 Value Addition Resource Center**

* Provides downloadable guides, videos, and tips for post-harvest value addition.
* Categories like drying, packaging, canning, etc.

**3.2.4 User Account Management**

* Registration, login, profile update, and password recovery.
* Save preferences like favorite crops or regions.

**3.2.5 Admin Management Module**

* Secure admin login.
* CRUD (Create, Read, Update, Delete) operations for crop data, prices, users, and storage.

**3.5 NON-FUNCTIONAL REQUIREMENTS**

**3.5.1 Performance**

* Fast load times (< 3 seconds for major pages).
* Capable of handling 100+ concurrent users without slowdown.

**3.5.2 Reliability**

* Error-handling in forms and backend processes.
* Fail-safe fallback if price APIs are down.

**3.5.3 Availability**

* 99% uptime required for server availability.
* Hosted on a reliable cloud-based platform.

**3.5.4 Security**

* Passwords stored in hashed form using bcrypt.
* Admin panel protected by session validation.
* SQL injection and XSS protection implemented.

**3.5.5 Maintainability**

* Clean codebase with modular components.
* Documentation provided for each module.

**3.5.6 Portability**

* Runs on any standard browser.
* Can be deployed on different servers with minimal changes.

**3.7 DESIGN CONSTRAINTS**

* Must use only specified technologies: HTML, Tailwind CSS, JS, PHP, MySQL.
* Should be optimized for mobile-first design.
* Data accuracy is dependent on admin or external API input.

**3.9 OTHER REQUIREMENTS**

* Should support multi-language content in future versions.
* A feedback form for farmers and traders.
* Backup functionality to export crop price and user data.

**4. ANALYSIS MODELS**

**4.1 DATA FLOW DIAGRAMS (DFD)**

Data Flow Diagrams are used to visually represent how data flows through the system and how inputs are transformed into outputs.

**Level 0 DFD (Context Level Diagram)**

This is the top-level diagram that shows the entire system as a single process and its interaction with external entities.

**Entities:**

* **Farmer/User**
* **Admin**
* **External Price APIs (optional)**

**Process:**

* Web System for Horticultural Crops

**Data Stores:**

* User Database
* Crop Price Database
* Storage Info Database
* Value Addition Resources

**Flow:**

* Users interact with the system to:
  + View crop prices
  + Search for storage
  + Access value addition resources
* Admin maintains all databases.
* External APIs provide live crop prices (if integrated).

**Level 1 DFD**

This breaks the main system down into sub-processes.

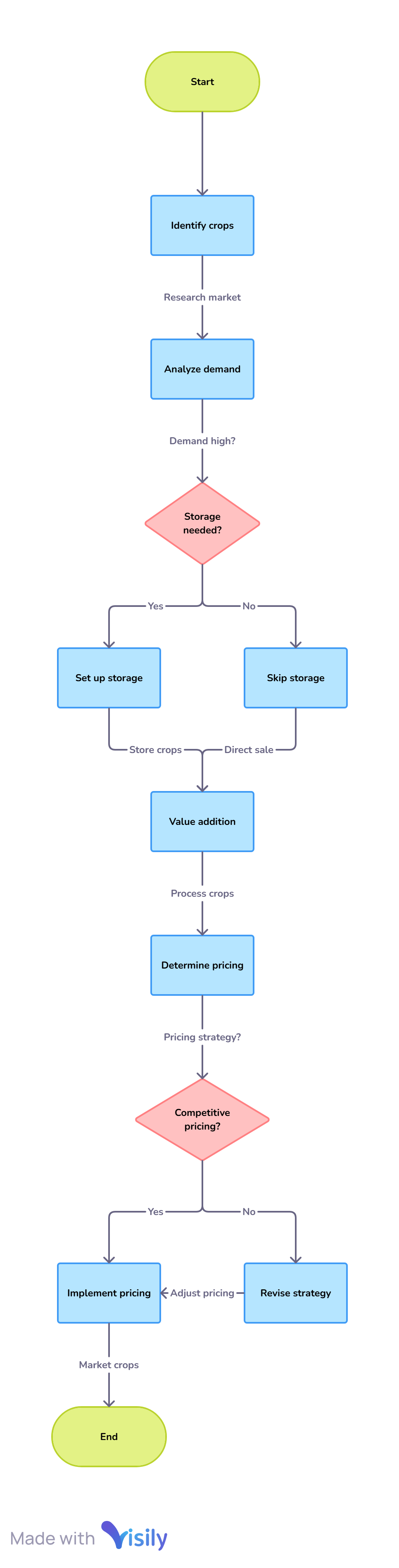
**Main Processes:**

1. **User Registration/Login**
   * Input: Username, password
   * Output: Access to personalized dashboard
2. **Crop Price Management**
   * Input: Crop name, region, date
   * Output: Latest price list and historical graph
   * Admin can update prices or fetch from API
3. **Storage Information System**
   * Input: User location or storage type
   * Output: List of available storage options
4. **Value Addition Resource Access**
   * Input: Selected crop or value addition category
   * Output: List of resources (PDFs, videos, tutorials)
5. **Admin Panel**
   * Input: Admin credentials
   * Output: Dashboard for data entry and updates

**Level 2 DFD (Example: Crop Price Management Subsystem)**

**Process:**

* User selects crop & location → System fetches from Price DB or API → Displays data → Optionally saves to user history.

****

**5. GITHUB LINK**

[**https://github.com/ShivChilu/Horti-connect**](https://github.com/ShivChilu/Horti-connect)

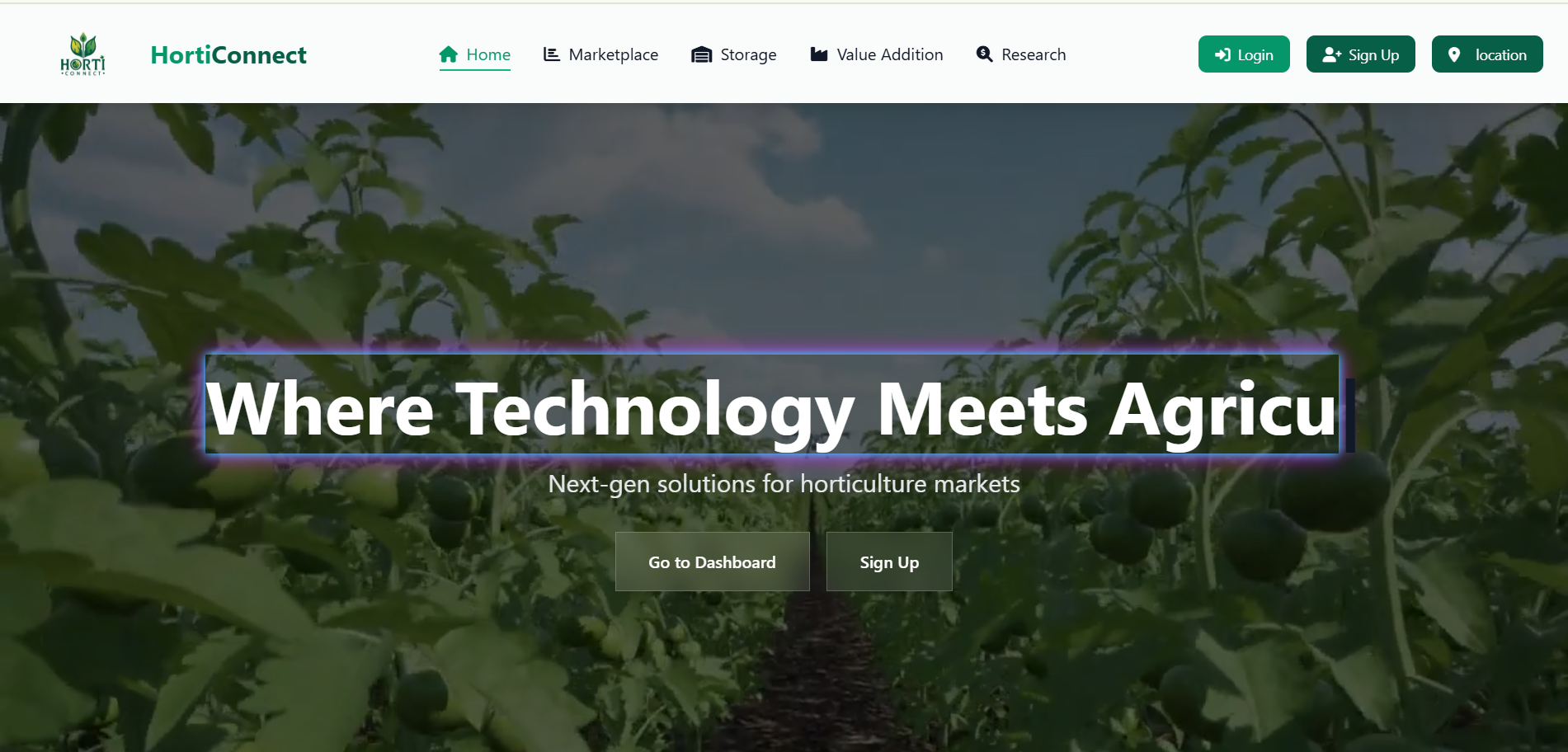
**6. DEPLOYED LINK**

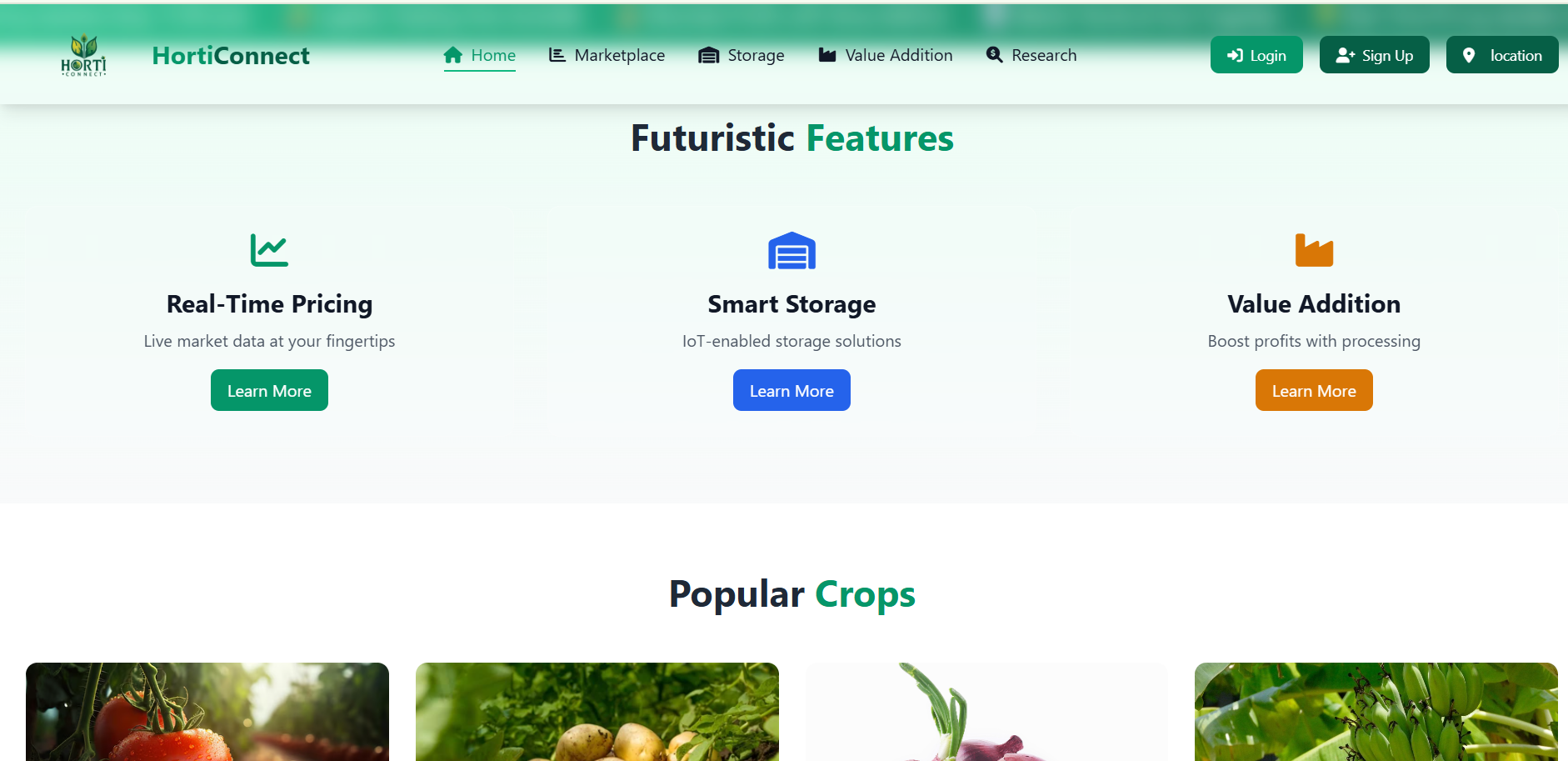
[**https://horti-connect.wuaze.com/**](https://horti-connect.wuaze.com/)

**A. APPENDICES**

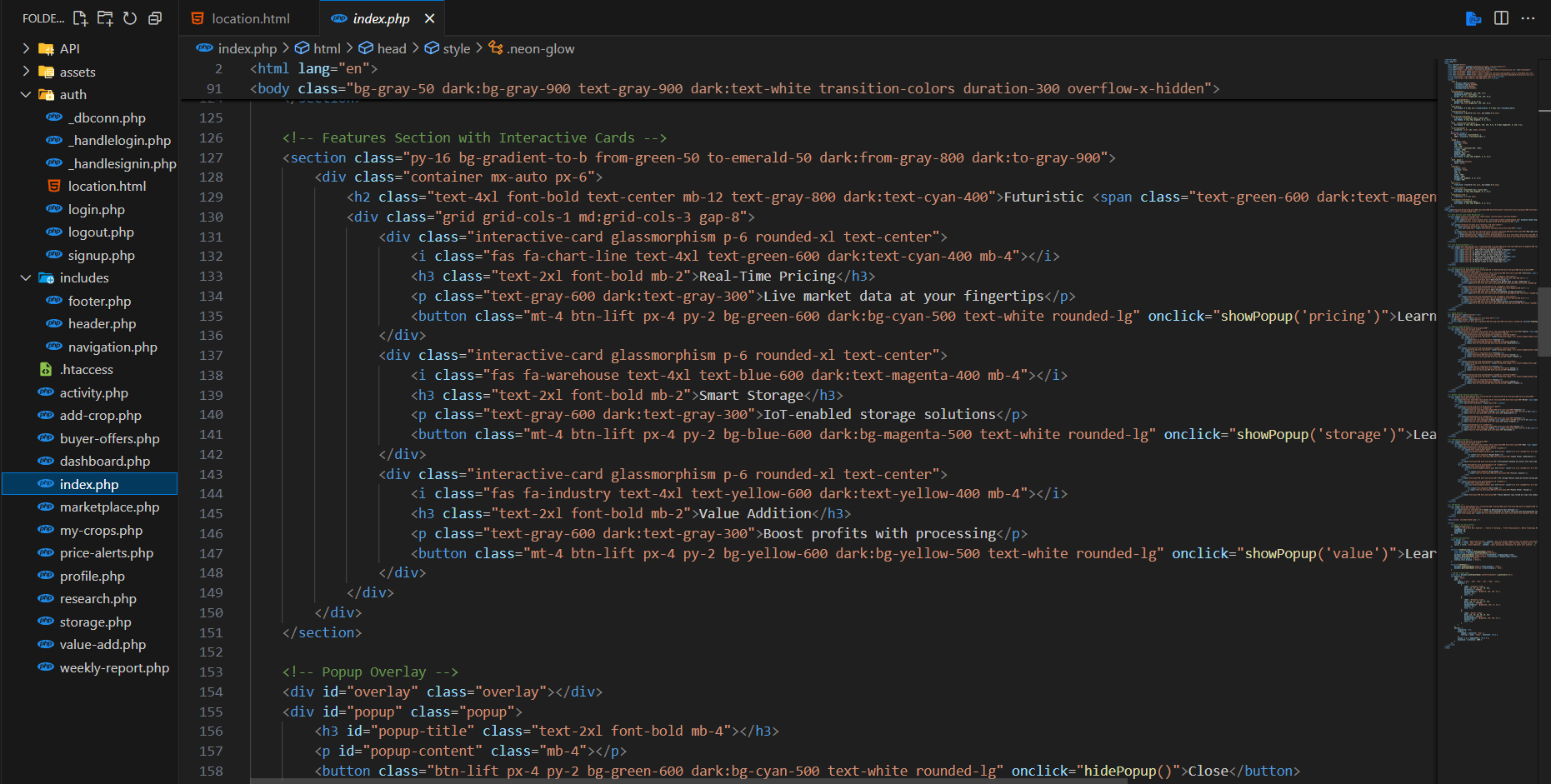
**A.1 APPENDIX 1**

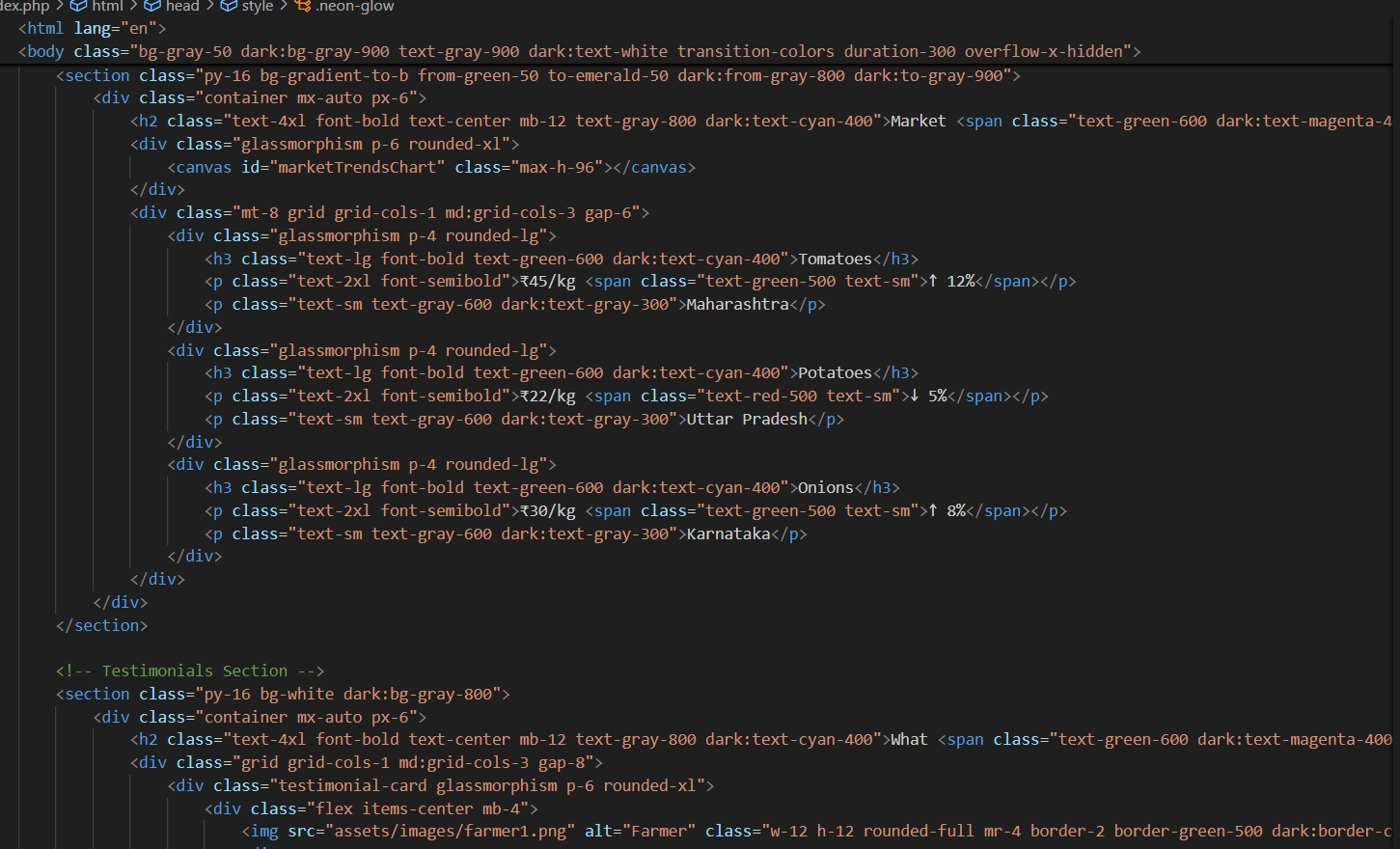
* **Screenshot of Web UI**

****

****

* **Code snippet samples**

****

****

**A.2 APPENDIX 2**

* **User feedback forms**
* **API integration documentation**