International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 04 | April – 2025 DOI: 10.55041/ISJEM02770
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Development of Web Based Plant Nursery Management

Prof. Nayana Gawali[1], Shravan Chinchkar[2], Kiran Dhanawade[3], Siddhi Diwadkar[4], Akshay Kalbhor[5]

[1,2,3,4,5]Department of Computer Engineering,
Pune District Education Association's College of Engineering, Manjari Bk., Hadapsar, Pune, Maharashtra, India – 412307

Abstract - The Plant Commerce Platform is a modern web application designed to make the buying and selling of plant- related products easier, more organized, and accessible online. Traditional methods often involve scattered tools, manual processes, and limited reach for individual sellers. This platform brings everything into one place, allowing multiple vendors to register, manage their listings, and track orders seamlessly. It is built using a full-stack architecture with Next.js for both the buyer and seller interfaces, PostgreSQL as the database, and Prisma as the ORM. TailwindCSS is used to create a clean, responsive design, while deployment is handled through platforms like Vercel and AWS. The system also integrates with Shopify to allow real-time syncing of product inventory for sellers. By solving key challenges such as poor discoverability, disconnected systems, and inefficient workflows, the Plant Commerce Platform offers a practicaland scalable solution for taking plant product commerce online in a more structured and modern way.

***Key Words*:** Plant commerce, web application, Next.js, Prisma, PostgreSQL, Shopify integration, multi-vendor platform, e-commerce.

1. INTRODUCTION

The rise of digital commerce has revolutionized various sectors, yet the horticultural industry continues to face unique challenges that standard e-commerce platforms often fail to address. Selling plant-related products online—such as live plants, pots, fertilizers, and gardening tools—requires specialized handling, real-time inventory updates, seasonal availability tracking, and product care integration. Most traditional platforms lack the flexibility and features needed to support these specific requirements, resulting in operational inefficiencies and limited customer engagement.

This project introduces a web-based multi-vendor plant commerce platform that aims to bridge this digital gap. Designed to support multiple sellers and improve the discoverability of plant products, the platform enables vendors to register, upload inventory, manage orders, and sync products via Shopify integration. Buyers, on the other hand, benefit from a responsive, user-friendly interface where they can browse, search, and purchase a wide range of products tailored to their needs.

Built using a full-stack architecture with Next.js, PostgreSQL, Prisma, and TailwindCSS, the system provides a scalable and modern foundation. Deployment is handled through Vercel and AWS for optimized performance. This study evaluates the effectiveness of a specialized e-commerce solution in supporting small- scale plant sellers and enhancing the digital experience for plant buyers in India's growing online market.

2. BODY OF PAPER

The Web-Based Plant Nursery Management System is designed to create a smooth and efficient online marketplace for plant sales. It bridges the gap between traditional nurseries and modern e-commerce, making it easier for plant lovers to buy what they need while helping nursery owners manage their online businesses effortlessly.

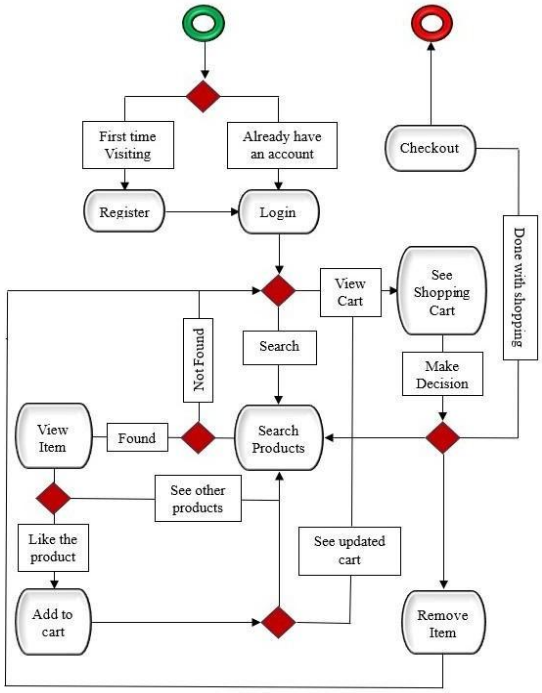

Figure 2.1 System Architecture

To ensure the platform is scalable, high-performing, and easy to maintain, we've structured it using a mono-repo architecture powered by Turbo-repo. This keeps the codebase organized, promotes reusability, and speeds up development.

2.1 Mono-Repo Structure with Turbo-Repo

Instead of managing multiple repositories separately, we chose a mono-repo structure, meaning all parts of our system—frontend, backend, and shared utilities—live within a single repository. This setup has several benefits: **Code Reusability:** Shared UI components, utilities, and business logic keep the system consistent while reducing duplicate code.

Synchronized Dependencies: Ensures all parts of the system use the same library versions, preventing compatibility issues.

Optimized Build Performance: Thanks to Turbo-repo's caching mechanism, only changed files are rebuilt, making development and deployment much faster.

Streamlined Collaboration: Developers can work across the entire system seamlessly, improving productivity and consistency.

2.2 Frontend Implementation with Next.js

The frontend consists of two separates Next.js applications, each catering to a specific user group:

Customer-Facing Application

The customer-facing application is designed to be fast, user-friendly, and visually appealing, offering a seamless experience for users browsing and purchasing plants. It leverages Server-Side Rendering (SSR) to enhance SEO, making it easier for potential buyers to discover products through search engines. Static Generation is used for content that changes infrequently, such as plant care guides and blog posts, ensuring quick load times. Additionally, dynamic client-side interactions enable smooth updates to the shopping cart and checkout process. The application features a fully responsive design, providing an optimal user experience across both desktop and mobile devices.

Seller Admin Portal

The seller admin portal is tailored specifically for nursery owners, enabling them to efficiently manage their online business through a centralized and intuitive interface. It features secure authentication with role-based access control to ensure only authorized users can access sensitive data. The portal provides a comprehensive dashboard that delivers real-time insights into sales performance, inventory levels, and order statuses. To streamline product management, it includes bulk editing tools that allow sellers to update multiple listings simultaneously. Additionally, the portal supports custom product attributes unique to plants, such as detailed care instructions, growth stages, and climate suitability, empowering nursery owners to present their offerings with clarity and precision.

2.3 Backend System with Express.js

The backend, powered by Express.js, is the heart of the system, handling all the heavy lifting—business logic, orderprocessing, payments, and third-party integrations. It consists of three key layers:

API Layer

The API layer of the system serves as the communication bridge between the frontend and backend, providing well- structured RESTful APIs that handle data exchange efficiently. These APIs are documented using OpenAPI/Swagger, making them easy to understand, test, and integrate for both internal and external developers. To ensure stability and long-term maintainability, the system supports API versioning, allowing new features to be introduced without disrupting existing services. Additionally, the API layer incorporates essential security measures such as rate limiting and input validation to prevent misuse, ensure data integrity, and safeguard against common web vulnerabilities.

Business Logic Layer

The Business Logic Layer serves as the core of the backend, handling all the essential operations that ensure smooth functioning of the platform. It manages order processing while taking into account plant-specific conditions such as growth cycles and seasonal availability, ensuring that only suitable plants are available for purchase during the right time of year. This layer also synchronizes inventory in real-time across multiple nurseries, preventing overselling and maintaining accurate stock levels. Additionally, it calculates commissions and automates seller payouts according to predefined business rules, ensuring fair and transparent transactions. To enhance user experience, the system also handles automated notifications, keeping both customers and nursery owners informed about order updates, shipping status, and other important events throughout the transaction lifecycle.

Integration Layer

The system integrates with various third-party services to enhance its core functionalities. It connects with Shopify APIs to manage essential e-commerce operations, ensuring smooth product listing, inventory syncing, and order processing. For handling payments, it integrates secure and reliable gateways like Stripe and PayPal, providing customers with multiple transaction options. Additionally, the platform interfaces with shipping carrier APIs to automatically generate tracking labels and offer real-time delivery updates. To keep both customers and sellers informed, it also leverages email and SMS services for timely notifications regarding orders, shipments, and other important activities.

2.4 Database Design with PostgreSQL

To handle the complexities of selling live plants online, we've structured the database using PostgreSQL with a schema optimized for plant sales.

Inventory Management

The inventory management system is designed to maintain real-time stock levels across multiple nursery locations while tracking batch-wise propagation and sourcing to support accurate lifecycle management of plants. It incorporates inventory aging mechanisms to prioritize the sale of older or more perishable stock, reducing waste and ensuring plant freshness. Additionally, restocking alerts are triggered based on sales trends, helping nurseries stay well-stocked.

Order System

The order system complements this by supporting complex order states that account for factors such as plant readiness and seasonal availability. It ensures appropriate shipping conditions to protect delicate plants in transit and validates customer growing zones to recommend plants suitable for their climate. Furthermore, the platform offers options for personalized gift messaging and care instructions during checkout, enhancing the overall customer experience.

3. PROPOSED SYSTEM MODEL AND IMPLEMENTATION:

The Web-Based Plant Nursery Management System was developed to tackle the unique challenges nurseries face when moving their operations online. Selling live plants comes with its own set of complexities—like managing perishable inventory, accommodating seasonal growth, and ensuring plants are shipped safely. Many traditional nurseries lack the tools to handle these issues digitally, which is why our platform serves as a centralized, multi- seller marketplace that not only helps nursery owners list and manage their products but also makes it easier for customers to discover and buy plants from multiple trusted sources in one place.

3.1. Multi-Seller Marketplace Functionality

Seller Onboarding:

Getting started as a seller is quick and user-friendly. Nursery owners can register, go through a simple verification process, and customize their store with their own branding. We also support bulk imports of products from existing catalogs, so sellers don't have to start from scratch. Each nursery operates under a clear commission and payout structure.

Inventory Isolation:

Every nursery manages its own product listings and stock levels, but the customer browsing experience remains seamless. A shared taxonomy ensures that categories and filters work consistently across the site, and the platform can suggest complementary products from different sellers—helping both sellers and buyers. Search results are shown in a unified way, but customers can still see which nursery each product comes from.

Order Fulfillment:

Each seller gets access to their own dashboard where they can manage incoming orders. The system helps streamline logistics by automatically generating shipping labels and tracking updates. Both sellers and customers are notified when a plant is ready to ship, and there's a built-in messaging feature for sharing special care instructions or addressing customer queries.

3.2. Agile Development and Quality Assurance

We've followed an agile development process to make sure the platform grows and improves with real-world feedback.

Development Lifecycle:

Work is done in two-week sprints, allowing us to push updates and new features regularly. Daily stand-up meetings help the team stay on track, and feedback sessions with nursery owners ensure that we're building something that truly works for them.

Testing Strategy:

To keep things reliable, we've built a multi-layered testing process. We use unit tests to check plant-specific logic, integration tests to make sure our APIs and database interactions are solid, and user acceptance testing to get feedback directly from real nursery operators.

3.3. Deployment and Infrastructure

The system is hosted on a cloud infrastructure that's built to scale and handle peak traffic without breaking a sweat. **Frontend Hosting (Vercel):**

The frontend is deployed using Vercel, which offers global edge hosting for faster load times. It comes with built-in SSL for secure connections and supports instant preview deployments, so changes can be tested before going live.

Backend Hosting (AWS & Render):

The backend services are containerized with Docker and hosted on platforms like AWS and Render, allowing us to automatically scale resources based on demand. The database, powered by PostgreSQL, is managed with built-in backups, recovery features, and optimized read performance.

CI/CD Pipeline:

Our deployment process is fully automated. Every code change runs through a pipeline that checks for bugs, code

quality, and test coverage. Deployments happen with zero downtime, and once live, we monitor system performance and errors in real-time to catch any issues early.

4. FUTURE SCOPE

4.1 Plant Growth Track:

This means that there would be a mechanism to monitor and track the growth and health of an individual plant or set of plants within the nursery.

1) Data Gathering: Incorporating sensors (temperature, humidity, soil moisture, light), manual measurement, or image analysis to collect data based on plant growth.

2) Tracking Growth Stages: Capturing and modeling the different stages of growth in plants.

3) Monitoring Health: Early detection of any potential problem like diseases, pests, or insufficient nutrients by means of data analysis and warning.

4) Analysis of Performance: Comparing rates of growth and yields across differing conditions (e.g., different soil mixes, watering schedules, nursery locations).

5) Inventory Integration: Linking growth information with inventory levels in order to streamline stock control and sales prediction.

4.2 Subscription-Based Garden Services:

To offer customers repeat services as per the expertise and infrastructure of the nursery.

1) Plant Subscription Boxes: Delivering individually handpicked bunches of plants (seasonal, themed, special specifications) to subscribers at fixed time intervals.

2) Gardening Advice & Support Subscriptions: Offering ongoing advice, tips, and trouble-shooting solutions via web sites, consultations, or classes.

3) Maintenance Subscriptions: Performing services like pruning, pest management, fertilizing, or tidying up the garden seasonally for subscribers.

4) Customized Planting Plans: Designing and supplying customized planting plans and supporting plants for the garden of subscribers.

5) Upselling and Cross-selling: Supplying additional products (fertilizers, accessories, tools) or services to subscribers.

6) Loyalty Programs: Offering special perks to long-time subscribers.

4.3 Cloud-Based Data Analysis and Reporting:

This involves utilizing cloud technology to store, process, and analyze the data being generated through the nursery management system and providing valuable insights and reports.

1) Centralized Data Storage: Keeping all of the nursery data (inventory data, sales data, customer information, plant growth data) secure in the cloud.

2) Real-time Dashboards: Easily available and interactive dashboards for the managers to monitor significant key performance indicators (KPIs) such as sales trends, inventory level, plant health indicators, and customer interaction.

3) Customizable Reports: Generating various reports on demand or on a schedule, as per given requirements (e.g., sales report, inventory report, plant health report).

4) Advanced Analytics: Applying data mining and machine learning techniques to find patterns, predict trends, optimize resource utilization, and improve decision-making.

5) Accessibility and Collaboration: Allowing permitted staff to remotely access reports and data through an internet connection, facilitating collaboration.

6) Integration with Other Systems: Integration with e-commerce websites, accounting software, or other business applications to provide an end-to-end overview of operations.

5. CONCLUSIONS

The Web-Based Plant Nursery Management System was built with a clear purpose: to help traditional plant nurseries embrace the digital world without the usual struggles. Selling plants online isn't the same as selling regular products—it comes with its own set of challenges like managing perishable inventory, accounting for seasonal availability, and making sure plants survive shipping. Many nurseries don't have the technical resources to handle all this, which is where our system comes in. It provides an all-in-one solution that makes life easier for both nursery owners and plant lovers. At the heart of the platform is a multi-seller marketplace that allows different nurseries to sell their plants under one roof, while still maintaining their individual identities. Sellers can easily register, upload their products, and manage orders through a personalized dashboard. Customers, on the other hand, can browse through a wide variety of plants from trusted sellers, get recommendations based on their local climate, and enjoy a smooth shopping experience. On the technical side, the system uses modern tools like Next.js for the frontend and Express.js for the backend. We chose a mono-repo architecture with Turbo-repo to make the development process more organized and efficient. This helps in sharing code across projects, keeping dependencies consistent, and reducing build times. The backend handles all the business logic, including user authentication, order management, inventory tracking, and integrations with payment and shipping services. What really sets this platform apart is how it's tailored for live plant sales. Plants aren't like books or gadgets—they grow, change with the seasons, and need specific care. So, we included features like growth tracking, climate compatibility using USDA zones, and detailed plant care instructions. Customers even get post-purchase care tips to help their plants thrive, and sellers are notified when a plant is ready to ship based on its growth stage. The system also manages real-time inventory across multiple

nursery locations, tracks plant batches, and alerts sellers when it's time to restock. Orders go through various stages, accounting for things like plant readiness and special shipping requirements. All of this is backed by a carefully structured PostgreSQL database that supports complex relationships and real-time updates. To ensure everything works smoothly, we followed an agile development process with regular updates, testing at every level, and feedback from real nursery owners. The platform is deployed on cloud infrastructure with Vercel for the frontend and AWS/Render for the backend, giving it the flexibility to scale as traffic grows. Our CI/CD pipeline ensures that every update goes live without downtime, and any issues are quickly caught through monitoring.

In the end, this project is more than just a technical achievement—it's a real solution to a real-world problem. It opens up new opportunities for small and medium- sized nurseries, giving them a way to grow their business online. For customers, it makes discovering, buying, and caring for plants easier and more enjoyable. With room to grow and evolve, this system is ready to support the future of plant shopping in the digital age.

5. REFERENCES

[1] Next.js Documentation. (2024). The React Framework for Production. Retrieved from https://nextjs.org/docs

[2] Express.js. (2024). Fast, unopinionated, minimalist web framework for Node.js. Retrieved from https://expressjs.com/

[3] TurboRepo. (2024). Monorepo Build System. Vercel. Retrieved from https://turbo.build/repo/docs

[4] PostgreSQL Global Development Group. (2024). PostgreSQL 16 Documentation. Retrieved from https://www.postgresql.org/docs/

[5] Stripe. (2024). Online Payment Processing for Internet Businesses. Retrieved from https://stripe.com/docs

[6] Shopify API Reference. (2024). Shopify Admin API Documentation. Retrieved from https://shopify.dev/docs/api

[7] USDA Plant Hardiness Zone Map. (2024). Agricultural Research Service, U.S. Department of Agriculture. Retrieved from https://planthardiness.ars.usda.gov/

[8] WCAG 2.1. (2018). Web Content Accessibility Guidelines. World Wide Web Consortium (W3C). Retrieved from https://www.w3.org/TR/WCAG21/

[9] Schwaber, K., & Sutherland, J. (2020). The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game. Retrieved from https://scrumguides.org

[10] MDN Web Docs. (2024). Web development documentation for HTML, CSS, and JavaScript. Mozilla.