WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal Homepage: www.ijrpr.com ISSN 2582-7421

Farm Cart: A Web-Based System for Farm Products and Agricultural Services

Manoj S Jadav[1], Sneha Bagale[2], Shashank K[3], Shreyaa Vishwadnya[4], Saima Anjum[5]

[1]Dept. of Computer Science and Engineering, GMIT, Davanagere, Karnataka, India 4GM24CS407
[2]Dept. of Computer Science and Engineering, GMIT, Davanagere, Karnataka, India 4GM23CS089
[3]Dept. of Computer Science and Engineering, GMIT, Davanagere, Karnataka, India 4GM24CS408
[4]Dept. of Computer Science and Engineering, GMIT, Davanagere, Karnataka, India 4GM23CS086
[5] (Guide), Assistant Professor, Department of Computer Science and Engineering, GMIT, Davanagere, Karnataka, India

DOI : https://doi.org/10.55248/gengpi.07.1026.34043

ABSTRACT—

Farm Cart is a web-based platform that brings the purchase of agricultural inputs and the booking of tractor services into a single application for farmers. It addresses inefficiencies of traditional agricultural commerce, namely dependence on local shops and middlemen, delays in finding tractor services, and little transparency in price and stock. The platform provides a product catalogue with category filtering, keyword search, price sorting and live stock availability; a cart and checkout module with delivery-charge calculation, order history and cancellation; a tractor and spot-mechanic booking module with a transparent fare estimate; a service-provider facility through which tractor owners list their machines; a booking tracking screen showing operator details and a service OTP; and an admin console for managing products, orders, tractors and bookings. The system follows a three-tier architecture: a React front end, a RESTful application server built with Node.js and Express.js and secured with JSON Web Tokens, and a MongoDB database with a local JSON-file fallback store. The back-end modules were checked through functional test scenarios covering authentication, access control, ordering with stock control, booking and administration, and all scenarios passed. Voice assistance and regional-language support are identified as planned extensions and are not part of the present prototype. No field trial with farmers has been carried out.

***Keywords*—Agriculture, E-Commerce, Tractor Booking, Farm Machinery Services, Service Providers, Order Management, Live Tracking, Web Application, React, Node.js, MongoDB, Rural Digital Services.**

I. INTRODUCTION

Agriculture is one of the most important sectors in rural development, but many farmers still face difficulty in purchasing agricultural products, booking farm machinery and accessing services quickly. In many villages, farmers depend on local shops or middlemen for seeds, fertilizers, pesticides and equipment, which may increase cost and reduce convenience.

This work presents a web-based agriculture support platform through which farmers can browse seeds, fertilizers, pesticides, bio-nutrients and farming tools, check stock availability and place orders. A major feature is the tractor booking service. Farmers who need a tractor for ploughing, sowing, harvesting, field levelling or transportation, or a mechanic for an on-spot tractor breakdown, can book the service for their village and see an estimated fare before confirming. Tractor owners can register their machines on the platform so that farmers are connected with local service providers.

To keep the platform usable by farmers with basic digital knowledge, the interface uses a single navigation bar, category chips, icons and rate-card style fare summaries. Voice guidance and regional-language support are planned but are not part of the current prototype (Section IX).

A. Problem Statement

Farmers face difficulty in buying agricultural products and getting tractor services on time. Many depend on local shops and middlemen, which increases cost and reduces product availability. Small farmers often do not own tractors and face delays in finding tractor services when needed. Existing agricultural websites are often not user-friendly for people with limited digital knowledge. A web-based system is therefore needed where farmers can easily buy products, book tractors, list tractors for hire and follow the status of their orders and bookings from one place.

II. OBJECTIVES

- To develop an online platform for purchasing agricultural products with categories, keyword search, price sorting, product details, stock availability, cart, checkout (UPI, cash on delivery or card/net-banking option), order tracking, purchase history and order cancellation.
- To provide tractor booking services for farming activities with a transparent fare estimate.
- To connect farmers with tractor owners by letting owners register their tractors as service providers.
- To support on-spot tractor breakdown requests and to show booking status, operator details and a service OTP to the farmer.
- To give administrators a console for managing products, orders, tractors and bookings, with role-based access control.

III. RELATED WORKS

Digital tractor-hire platforms have been studied in India and Nigeria by Daum et al. [1]. They note that tractor owners are often reluctant to serve smallholder farmers because of high transaction costs, that start-ups and manufacturers have built ICT applications to reduce these costs, and that farmers in practice still relied largely on booking agents and phone calls rather than smartphone apps. Their findings stress the need for ICT literacy and a supportive environment, which motivates the simple, guided interface used in Farm Cart. At policy level, India's Digital Agriculture Mission, approved in September 2024 as an umbrella scheme for digital public infrastructure in agriculture [2], shows the attention given to digital services for farmers.

Most such work addresses either machinery hire or input e-commerce as a separate service. Farm Cart combines an agricultural input marketplace, tractor and spot-mechanic booking, service-provider registration, booking tracking and an administrator console in one system built from widely used open technologies.

IV. SYSTEM REQUIREMENTS

The software and tools used are summarised in Table I. For development, an Intel Core i5 or above processor, at least 8 GB RAM, a 256 GB SSD, a stable internet connection and a mobile phone (for testing the responsive farmer interface) are recommended.

TABLE I. SOFTWARE REQUIREMENTS

Category	**Technology**
Operating system	Windows 10/11; Ubuntu Linux (optional, deployment)
Front end	React 19, Vite, Tailwind CSS 4, React Router 7, Axios, Lucide icons
Back end	Node.js, Express.js 5, Mongoose, JWT (jsonwebtoken), bcryptjs, CORS, dotenv; REST APIs
Database	MongoDB; local JSON-file store as fallback
Tools	Visual Studio Code, Git and GitHub
Browsers	Google Chrome and other modern browsers

V. PROPOSED METHODOLOGY

The methodology (Fig. 1) has five steps.

1) Requirement analysis. The problems farmers face in buying seeds, fertilizers, pesticides and tools and in arranging tractors are analysed, and requirements for product purchase, tractor booking, service-provider support, order and booking tracking and administration are identified.

2) System design and module development. The system is designed as a web platform with separate modules for registration and login, product marketplace, cart and checkout, tractor booking, tractor registration, order and booking history, live tracking and admin management. The interface is kept simple for farmers with basic digital knowledge.

3) Database design and service integration. The database stores users, product details with stock, tractors, bookings and orders. Tractor owners register their machines so that farmers can find available machinery. A persistence adapter lets the same models run on MongoDB or on a local JSON store.

4) Order, booking and admin workflow implementation. Stock checking and deduction at ordering, stock restoration on cancellation, fare estimation, operator assignment and status updates are implemented in the REST API and exposed through the front end.

5) Testing and implementation. Registration and login, access control, product browsing, ordering, tractor booking, tractor registration and administration are tested through functional scenarios, after which the system is deployed.

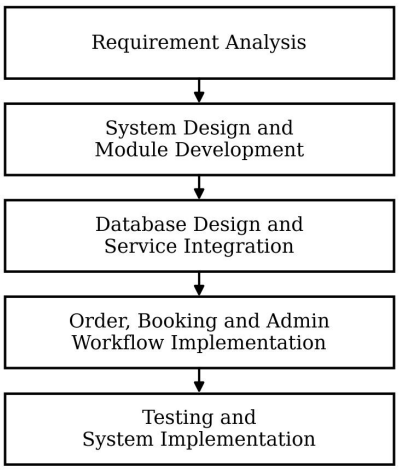


Fig. 1. Flowchart of the web-based agriculture support platform.

VI. SYSTEM DESIGN

The architecture (Fig. 2) has a frontend, a backend and a database.

Users. A farmer or customer browses products, places orders, books tractor or spot-mechanic services, tracks bookings and manages a profile. A tractor owner (service provider) registers and manages tractors. An admin manages products, orders, tractors and bookings.

Frontend. The user-interaction layer runs in a web browser and includes the product marketplace with category chips, search and sorting, cart drawer and checkout, tractor booking with a fare summary, tractor listing, order and booking history, a live tracking page, a profile page and the admin dashboard. It is built with React and styled with Tailwind CSS; the cart and login state are kept in React context.

Backend. The Node.js and Express.js server handles business logic. It provides authentication and authorization using JWT [4], with passwords hashed using bcrypt [5], and REST [3] services for products, orders, tractors, bookings and users. Middleware separates logged-in users from admin-only routes.

Database. MongoDB stores data in five collections (Table II). Payment details are stored inside each order. If a MongoDB connection cannot be established, the server switches to a JSON-file store behind the same model interface so that the application keeps running.

TABLE II. DATA MODEL

Collection	**Main fields**
Users	name, email, phone, password (hashed), role (farmer/admin), village, district, isTractorOwner
Products	name, category, price, stock, unit, brand, description, rating
Tractors	owner, tractor name, brand, HP, registration number, implements, rate per hour/acre, village, availability
Bookings	user, service type, farm location, date, time slot, estimated cost, payment method, status, assigned operator, service OTP
Orders	user, order items, shipping address, payment method and result, items/delivery/total price, order status

Business rules. An order is accepted only if every item has enough stock; stock is reduced when the order is saved and restored if the order is cancelled before shipping. Delivery is free when the item total is ₹500 or more and ₹50 otherwise. A tractor fare is the selected rate (per hour or per acre) multiplied by the work size. A booking cannot be cancelled once the service is in progress or completed. A four-digit service OTP is generated for every booking for the farmer to share with the operator on arrival.

Working flow. The user opens the web application; the frontend sends requests to the backend APIs; the backend validates the token and role, and processes the request; the backend reads or writes the database; and the result is displayed through the frontend.

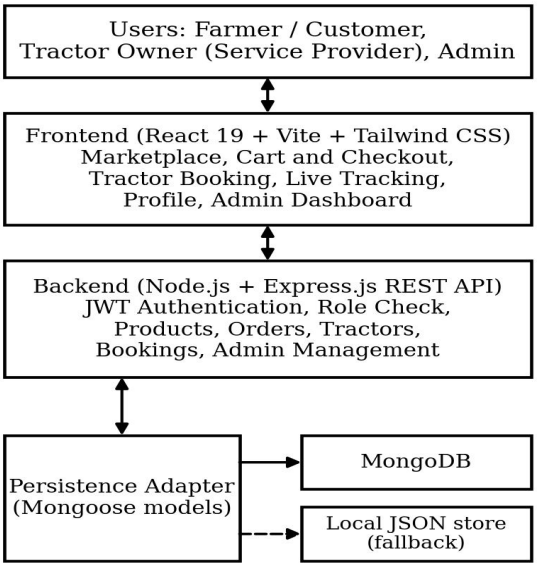


Fig. 2. Architecture of the Farm Cart web application.

VII. RESULTS AND DISCUSSION

The prototype was checked through functional test scenarios rather than a statistical benchmark (Table III). The scenarios were run against the REST API with the server using its local JSON persistence store, with demonstration farmer and admin accounts and seeded data (12 products in five categories and demonstration tractors). Each scenario was run once and the observed response was compared with the expected one.

TABLE III. FUNCTIONAL TESTING SUMMARY

Module	**Scenario checked**	**Result**
Login	Registration and login; wrong password rejected; duplicate e-mail rejected	Pass
Access control	No token rejected; farmer blocked from admin routes; admin allowed	Pass
Products	List, category filter, keyword search, price sorting	Pass
Orders	Place order; price and delivery charge; stock deducted; over-stock and empty order rejected; cancel restores stock	Pass
Tractor booking	Create booking; missing location rejected; cancel; farmer blocked from assigning operator	Pass
Service providers	Register tractor; list available tractors	Pass
Admin	Assign operator to booking; view all orders; update order status	Pass

The front-end pages were opened in a headless browser to capture the screenshots in Section VIII; no automated user-interface tests were written. No controlled usability study or field trial with farmers is claimed. The expected outcomes of the system are: easier purchase of agricultural products with less dependence on middlemen; quick online tractor booking for ploughing, sowing, harvesting and transportation; wider availability of tractor services through registered owners; clearer prices through the fare summary; and a simple interface that encourages digital adoption among farmers. These outcomes have not yet been measured.

Limitations. The prototype depends on internet connectivity, which may be unstable in rural areas. Online payment is simulated: no payment gateway is integrated and the transaction reference is generated by the application. The live tracking screen animates a simulated route and does not use real

GPS data. In the current prototype, a new booking is attached to the selected owner's details or to a default operator and can be reassigned by the admin. Role selection at registration, the default token secret and the absence of rate limiting must be hardened before real-world deployment. The tests were run on the JSON store, and no load testing or MongoDB performance measurement has been done. Voice assistance and multi-language support are not yet implemented.

VIII. SYSTEM INTERFACE

The interface follows a common navigation structure so that the marketplace, tractor booking, tractor listing, bookings, live tracking, orders, profile and admin console are reachable from the same application (Figs. 3–6). The screenshots show the working interface and are not a controlled usability experiment.

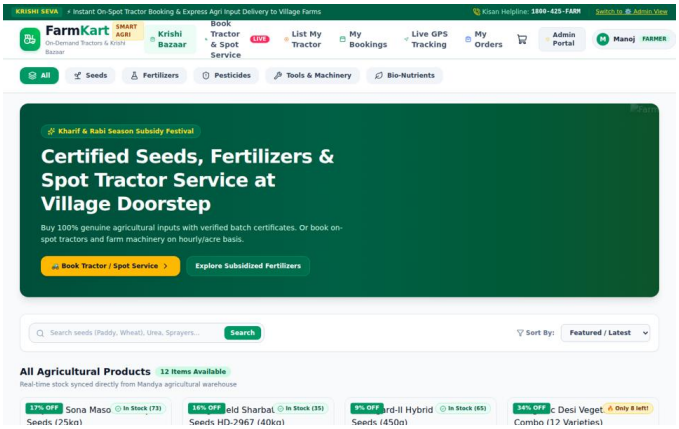


Fig. 3. Home Page – Farm Cart marketplace with product categories, search and sorting.

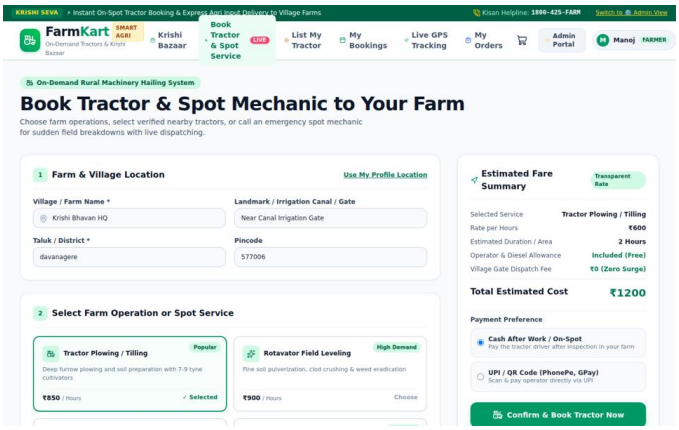


Fig. 4. Booking Page – tractor and spot-mechanic booking with fare summary.

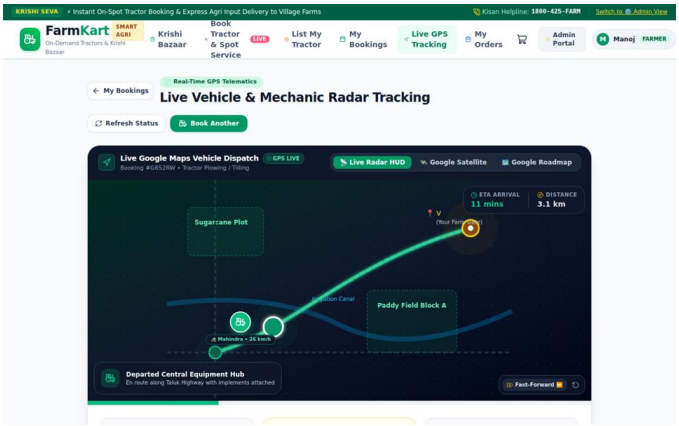


Fig. 5. Tracking Page – simulated live vehicle tracking for a booking.

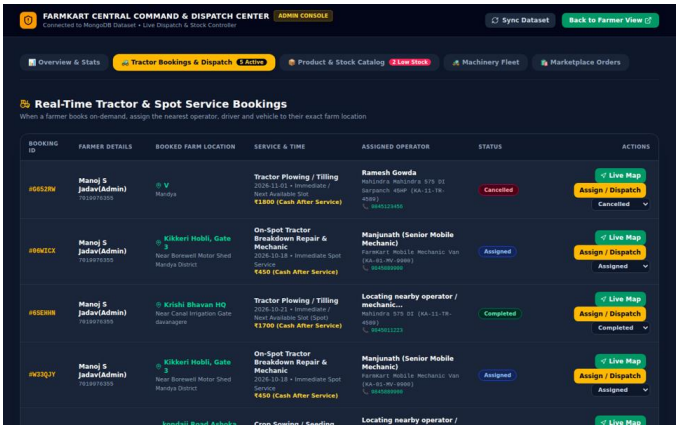


Fig. 6. Admin Console – management of bookings, products, tractors and orders.

IX. CONCLUSION AND FUTURE WORK

Farm Cart combines an agricultural product marketplace, tractor booking with registered service providers, booking tracking and an administrator console in one web platform for farmers. The three-tier design with React, Node.js, Express.js and MongoDB keeps the modules separate so that they can be extended independently, and the functional tests of the back end passed for all scenarios in Table III.

Future work should first evaluate the system with farmers through usability studies and field trials. Planned extensions are a voice assistant using the Web Speech API and regional-language support (for example Kannada and Hindi) through language files, integration of a real payment gateway, real-time GPS tracking of tractors, SMS notifications, product reviews and an approval workflow for tractor owners, offline-tolerant operation, and stronger security and load testing before real-world deployment.

REFERENCES

[1] T. Daum, R. Villalba, O. Anidi, S. M. Mayienga, S. Gupta, and R. Birner, “Uber for tractors? Opportunities and challenges of digital tools for tractor hire in India and Nigeria,” World Dev., vol. 144, 2021.

[2] Ministry of Agriculture & Farmers Welfare, Govt. of India, “Digital Agriculture Mission: Tech for Transforming Farmers’ Lives,” Press Information Bureau, Sep. 2024. [Online]. Available: https://static.pib.gov.in/WriteReadData/specificdocs/documents/2024/sep/ doc202494386501.pdf

[3] R. T. Fielding, “Architectural styles and the design of network-based software architectures,” Ph.D. dissertation, Univ. California, Irvine, CA, USA, 2000.

[4] M. Jones, J. Bradley, and N. Sakimura, “JSON Web Token (JWT),” IETF, RFC 7519, May 2015.

[5] N. Provos and D. Mazières, “A future-adaptable password scheme,” in Proc. USENIX Annu. Tech. Conf., FREENIX Track, Monterey, CA, USA, 1999, pp. 81–91.