Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	26 June 2025
Team ID	LTVIP2025TMID53236
Project Name	shopsmart: your digital grocery store experience
Maximum Marks	4 Marks

Technical Architecture

Below is the architecture summary for GreenCart:
- Users interact with the platform via a responsive Web UI.
- Application logic is handled through Node.js and Express.js backend.
- MongoDB Atlas is used as the cloud database.
- The app is hosted using Netlify (frontend) and Render or Railway (backend).
- Authentication uses JWT for secure user sessions.
- Product images and media are stored locally or on a cloud file storage platform (e.g. Cloudinary or Render).
- Integrated third-party services include Razorpay for payments.

Table-1: Components & Technologies

S.No	Component	Description	Technology
1	User Interface	Web UI to browse, search, and order groceries	HTML, CSS, JavaScript, React.js
2	Application Logic-1	Backend logic for product listings, cart, order processing	Node.js, Express.js
3	Application Logic-2	Payment processing and invoice generation	Razorpay SDK
4	Application Logic-3	Admin panel for product and order management	Node.js, Express.js, React Admin UI
5	Database	NoSQL document database for users, orders, products	MongoDB
6	Cloud Database	Hosted version of MongoDB	MongoDB Atlas
7	File Storage	Product images, media	Cloudinary / Local FileSystem
8	External API-1	Payment gateway integration	Razorpay API
9	External API-2	Delivery Pincode	India Post API /

		Check	Map APIs
10	Machine Learning Model	(Not used currently)	Object Recognition Model
11	Infrastructure	Deployment on Cloud for both frontend and backend	Netlify (Frontend), Render (Backend)

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology
1	Open-Source Frameworks	Frontend and Backend built on open-source libraries	React.js, Node.js, Express.js, MongoDB
2	Security Implementations	JWT for user authentication, secure APIs, encrypted passwords (bcrypt)	JWT, Bcrypt, Helmet, CORS
3	Scalable Architecture	Separated frontend-backend, cloud deployment allows scaling	Netlify, Render, MongoDB Atlas
4	Availability	Cloud hosting ensures uptime; load balancing is handled by platform	Netlify (edge), Render's scaling services
5	Performance	Caching at MongoDB level, optimized queries, React fast rendering	MongoDB, React, Lazy Loading, CDN (Vercel)

References

- https://c4model.com/

- https://www.vercel.com/

- https://render.com/

- https://www.mongodb.com/cloud/atlas

- https://razorpay.com/docs/api/