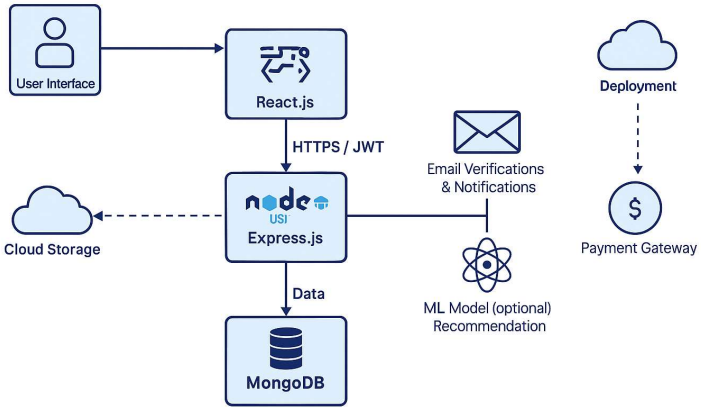
Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	17 June 2025
Team ID	LTVIP2025TMID58635
Project Name	ShopSmart - eCommerce Grocery Web app
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2



S.No.	Appressorst	Imtern1tionacst	Recensal
1	AmsonLIteage	Coud-Hosting Vercel / AWS	Local Server

Table-1: Components & Technologies

S.No	Component	Description	Technology
1	User Interface	Web UI where users interact: browse, login, add to cart etc.	HTML, CSS, JavaScript, React.js
2	Application Logic-1	Backend server handling authentication, cart, orders, etc.	Node.js, Express.js
3	Application Logic-2	Role-based access control (admin/user)	Middleware in Express.js, JWT authentication
4	Application Logic-3	Admin dashboard and product management logic	Node.js, Express.js, React.js
5	Database	Data storage for users, products, orders, wishlist/cart	MongoDB (NoSQL)
6	Cloud Database	Managed cloud database	MongoDB Atlas
7	File Storage	Product images and static assets	Cloud storage like AWS S3 / local filesystem
8	External API-1	Payment gateway integration (planned future)	Razorpay / Stripe API
9	External API-2	Email service for verification and notifications	Nodemailer / SendGrid API
10	Machine Learning Model	(Optional / future) product recommendation	Custom ML model hosted separately
11	Infrastructure (Server / Cloud)	Deployment environment	Local during dev; Vercel / Render / AWS EC2 in production

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology
1	Open-Source Frameworks	Using open-source frameworks for frontend and backend	React.js, Node.js, Express.js, MongoDB
2	Security Implementations	JWT authentication, hashed passwords, HTTPS, validation, OWASP practices	bcrypt, JWT, HTTPS, Helmet.js, CORS
3	Scalable Architecture	Separated frontend and backend, stateless APIs, scalable NoSQL DB	MERN stack, Docker/Kubernetes for scaling
4	Availability	Deploy on distributed cloud infrastructure; CDN for assets	Cloud hosting + CDN (e.g., Cloudflare)
5	Performance	API optimization, lazy loading, caching product images, pagination	Redis (caching, optional), CDN, React lazy loading

---

✅ Highlights:

Frontend: React.js

Backend: Node.js + Express.js

Database: MongoDB (cloud-hosted with MongoDB Atlas)

Security: JWT, bcrypt, Helmet.js

Deployment: Netlify

Planned future: Payment gateway & ML recommendations