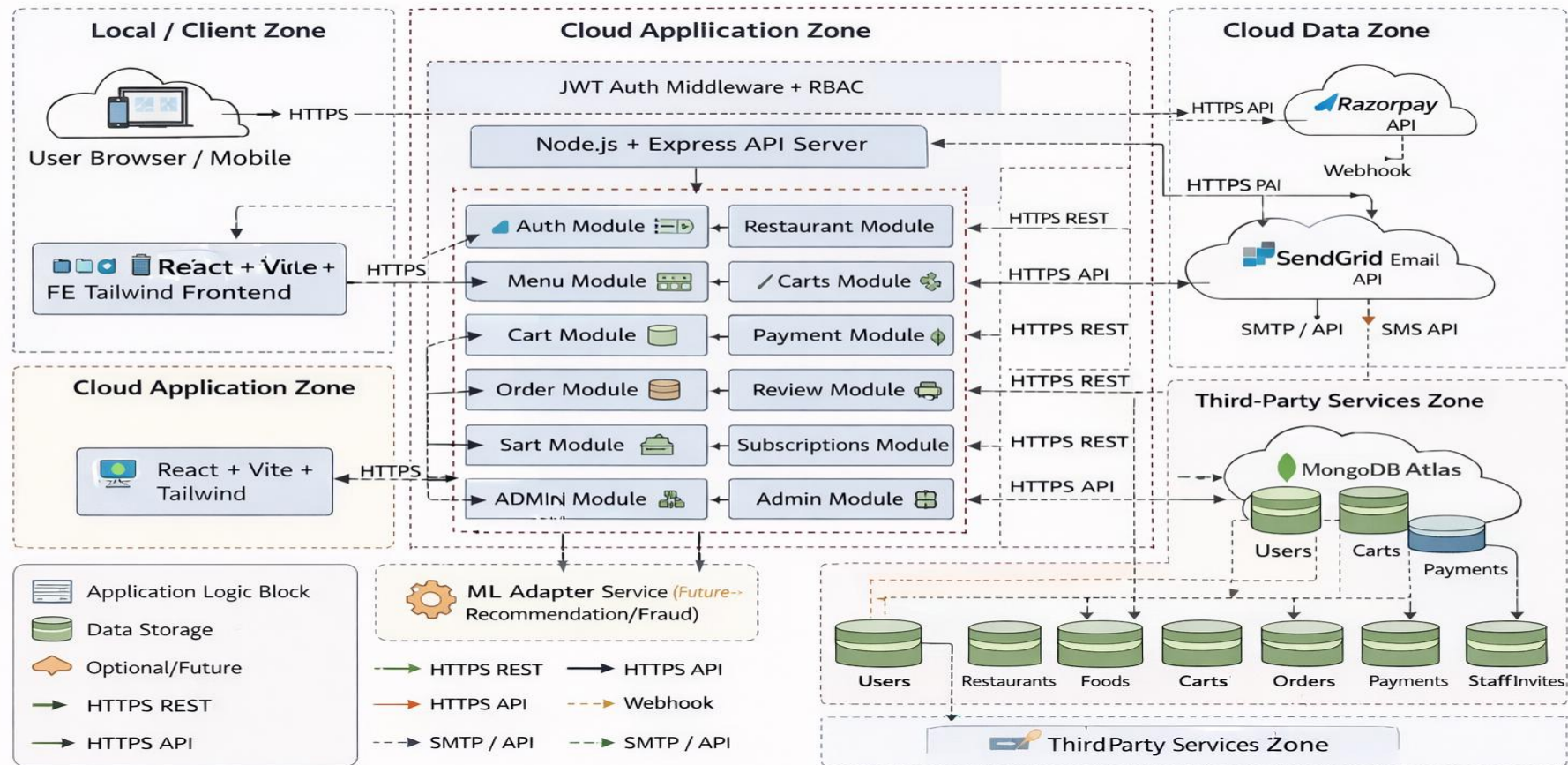


**Project Design Phase-II**  
**Technology Stack (Architecture & Stack)**

Date	20 February 2026
Team ID	TVIP2026TMIDS41611
Project Name	OrderOnTheGo: Your On-Demand Food Ordering Solution
Maximum Marks	4 Marks

**Technical Architecture:**

# Order on the Go – Technical Architecture



**Table-1 : Components & Technologies:**

S.No	Component	Description	Technology
1.	User Interface	Web interface for customers, restaurant owners, staff, and admin dashboards	React, Vite, Tailwind CSS, Axios, HTML5, CSS3, JavaScript
2.	Application Logic-1	Core business APIs: authentication, menu, cart, order lifecycle, reviews	Node.js, Express.js, JavaScript, REST APIs
3.	Application Logic-2	Access control and workflow enforcement: JWT auth, role-based access, subscription limits	JWT, Express middleware, Mongoose validation/hooks
4.	Application Logic-3	Payments and communication orchestration: payment create/verify, email/SMS verification	Razorpay SDK/API, SendGrid API, Twilio API
5.	Database	Operational data model for users, restaurants, foods, carts, orders, payments, reviews, subscriptions	MongoDB (NoSQL document DB), Mongoose ODM
6.	Cloud Database	Managed cloud database for production persistence and backup	MongoDB Atlas
7.	File Storage	User/restaurant profile image handling and file uploads	Local filesystem (uploads/profiles), optional cloud object storage (recommended: IBM COS / S3 / Azure Blob)
8.	External API-1	Online payment order creation, signature verification, transaction status	Razorpay Payment Gateway API
9.	External API-2	Customer communication for account verification and notifications	SendGrid Email API, Twilio SMS API
10.	Machine Learning Model	Currently not implemented; future option for recommendations, fraud/risk scoring, demand prediction	Recommended: Python ML service (scikit-learn/TensorFlow) exposed via REST, or cloud ML endpoint
11.	Infrastructure (Server / Cloud)	<b>Local:</b> React dev server + Node/Express + local/Atlas DB. <b>Cloud:</b> frontend static hosting + API server + managed DB + third-party integrations	Local VM/dev machine; Cloud options: Kubernetes, App Service/Cloud Foundry equivalent, Docker, managed MongoDB Atlas

**Table-2: Application Characteristics:**

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Full-stack open-source web architecture for rapid delivery and maintenance	React, Vite, Tailwind CSS, Node.js, Express.js, Mongoose
2.	Security Implementations	Token-based auth, RBAC, credential hashing, verification flows, API protection baseline	JWT, bcrypt password hashing, role-based authorization middleware, CORS policy, HTTPS (production), environment secrets, OTP/email verification
3.	Scalable Architecture	Modular 3-tier architecture (UI, API, DB), domain-based controllers/routes, horizontal API scaling path	React SPA + Express modular monolith + MongoDB Atlas; containerization-ready with Docker/Kubernetes
4.	Availability	Production-ready by deploying multiple API instances and managed DB with replication/backups	Load balancer + multi-instance Node API, MongoDB Atlas replica set, process manager/restart policy (PM2/Docker), health checks
5.	Performance	Fast API response through indexed queries, pagination, lean payloads, and static asset optimization	MongoDB indexes, pagination/filtering, efficient Mongoose queries, Vite optimized build, CDN for static assets/images (recommended), Redis cache (recommended)

**References:**

<https://c4model.com/>

<https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/>

<https://www.ibm.com/cloud/architecture>

<https://aws.amazon.com/architecture>

<https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d>