Project Design Phase
Solution Architecture

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

Solution Architecture:

Solution Architecture (OrderOnTheGo)

- OrderOnTheGo follows a layered, role-based web architecture:
- Client Layer → Frontend Layer → Backend API Layer → Data & Integration Layer.
- Client Layer: four actors use the same app with different permissions — USER, RESTAURANT, STAFF, ADMIN.

Core Layers

- Frontend (React + Vite + Tailwind): single-page app with role-specific dashboards and flows; communicates with backend via secure REST calls.
- Backend (Node.js + Express): modular APIs for auth, restaurants, menu/foods, cart, orders, payments, reviews, staff invites, subscriptions, verification, and admin analytics.
- Data Layer (MongoDB + Mongoose): persistent storage for users, restaurants, foods, carts, orders, payments, reviews, staff invites, subscriptions, and status requests.
- External Services: Razorpay (payments), SendGrid (email), Twilio (SMS), and image upload storage.

How Data Flows

- Users browse menus and place orders through the frontend; backend validates identity/role, processes business rules, and updates MongoDB.
- Payments go through Razorpay, then backend verifies signatures and updates payment/order states.
- Verification and notifications are handled asynchronously via SendGrid/Twilio.
- Admin analytics aggregate operational data (orders, revenue, subscriptions) from MongoDB for dashboards**.**

Architecture Strengths

- Clear separation of concerns, reusable API modules, centralized security (JWT + RBAC), and easy scalability by expanding backend services or splitting modules into microservices later.

Solution Architecture Diagram:

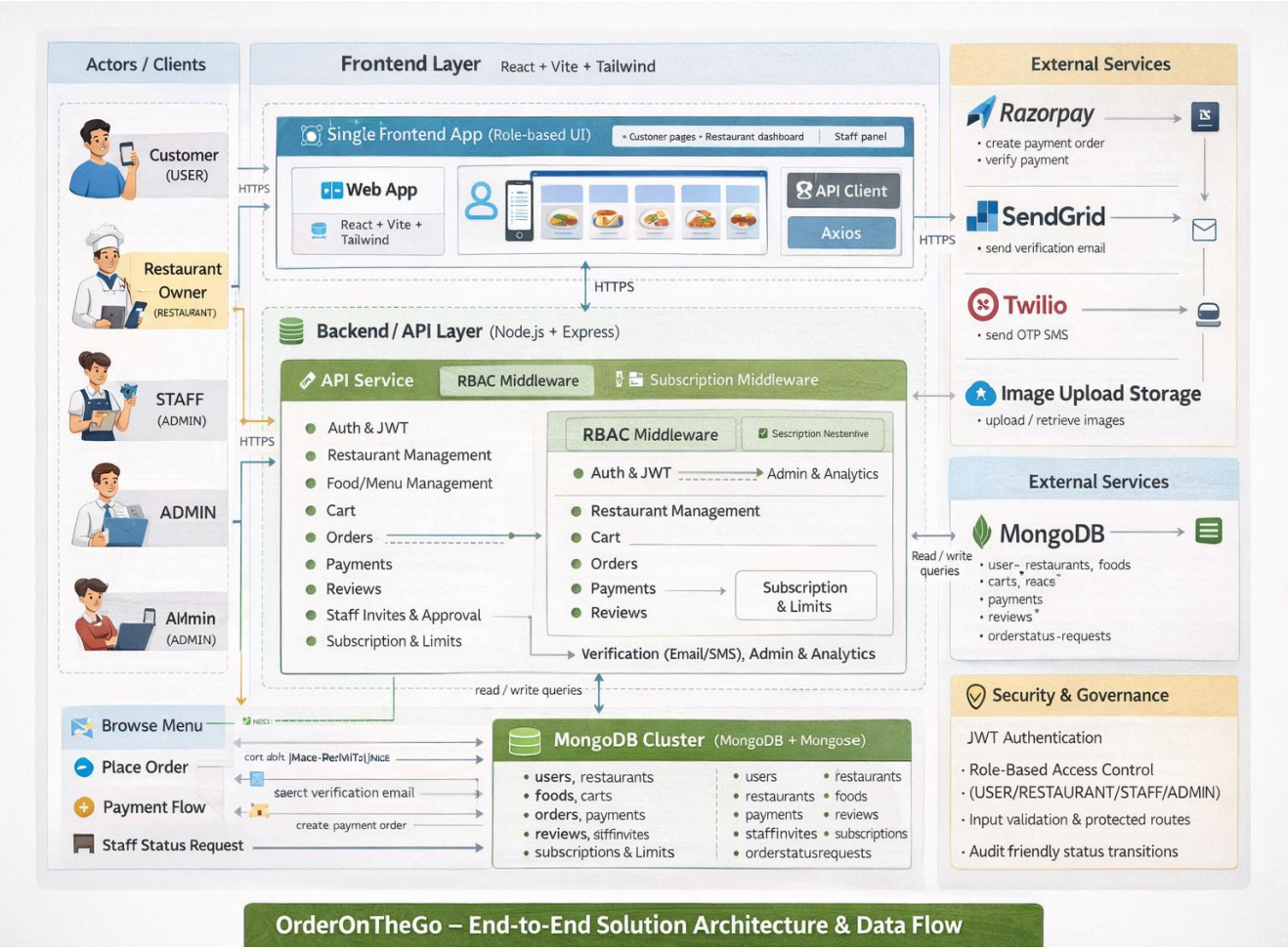

Figure 1: Architecture and data flow of OrderOnTheGo

Reference: https://aws.amazon.com/blogs/industries/voice-applications-in-clinical-research-powered-by-ai-on-aws-part-1-architecture-and-design-considerations/