Xeno Shopify Data Ingestion & Insights Service

Technical Documentation

1. Assumptions

Business Assumptions

- **Multi-tenant SaaS Model:** Each Shopify store operates as an independent tenant with completely isolated data. This ensures data privacy.
- **Store Admin Access:** Users registering are assumed to have admin access to their Shopify store to generate API access tokens.
- **Data Sync Frequency:** Manual sync is sufficient for the MVP. Users can trigger data synchronization on-demand.
- **Single Currency:** The initial implementation assumes single currency per store.

Technical Assumptions

- **Shopify REST API:** Using version 2024-01 for wider compatibility.
- **PostgreSQL Database:** Chosen for ACID compliance, JSON support, and scalability.
- **JWT Authentication:** Stateless JWT-based authentication is used for the distributed architecture.
- **Development Store Limitations:** Note that Shopify development stores redact customer PII (name, email) in API responses.

Data Assumptions

- **Draft Orders as Orders:** Draft orders are treated as valid orders for revenue calculation.
- **Customer Spend Calculation:** Calculated by summing order totals linked to each customer.
- **Product Inventory:** Uses the first variant's inventory quantity when multiple variants exist.

2. High-Level Architecture

System Components

- **Client Layer:** Web browsers accessing via HTTPS.
- **Frontend Layer (Vercel):** **Next.js 14**, React 18, Tailwind CSS, Recharts.

- **Backend Layer (Render): Express.js server**, Middleware (CORS, JWT), Shopify Services.
- **Database Layer: Prisma ORM**, PostgreSQL (Prisma Postgres Cloud).
- **External Services:** Shopify REST API v2024-01.

REDFER PAGE 6 FOR ARCHITECTURE DAIGRAM

Component Responsibilities

Component	Responsibilities
Frontend	Handles UI rendering, user interactions, form validation, data visualization, and API communication.
Backend	Manages API endpoints, business logic, authentication, authorization, and Shopify integration.
Prisma ORM	Provides type-safe database queries, schema migrations, and data modeling.
PostgreSQL DB	Stores tenants, users, customers, orders, and products with tenant isolation.
Shopify API	Source of truth for customer, order, and product data.

3. APIs and Data Models

3.1 REST API Endpoints

Category	Method / Endpoint	Description
Authentication	**POST /api/auth/register**	Registers tenant/user. Requires store URL & Access Token.
	POST /api/auth/login	Authenticates user. Returns JWT token.
	GET /api/auth/me	Returns current authenticated user info.
Dashboard	**GET /api/dashboard/stats**	Returns totals, revenue charts, top customers, order status.
Data Sync	**POST /api/sync/all**	Triggers sync for Customers, Orders, and Products.
	GET /api/sync/logs	Returns history of sync operations.
Resources	GET /api/customers GET /api/orders GET /api/products	Returns paginated lists with search/filter capabilities.

	GET /api/[resource]/:id	Returns detailed information for specific ID.

3.2 Data Models

Model	**Key Fields & Details**
Tenant	UUID, Name, ShopifyStoreUrl (Unique), ShopifyAccessToken. References all other models for isolation.
User	UUID, Email (Unique), HashedPassword, Role (Admin/User), TenantId.
Customer	ShopifyCustomerId, Email, Name, OrdersCount, TotalSpent. *Constraint: Unique(ShopifyCustomerId + TenantId).*
Order	ShopifyOrderId, FinancialStatus, FulfillmentStatus, IsDraft, TotalPrice. *Constraint: Unique(ShopifyOrderId + TenantId).*
Product	ShopifyProductId, Title, InventoryQuantity, Status, ImageUrl. *Constraint: Unique(ShopifyProductId + TenantId).*
SyncLog	SyncType, Status, RecordsProcessed, ErrorMessage, Timestamps.

4. Next Steps to Productionize

Phase 1: Security and Reliability (2-4 weeks)

- **Rate Limiting:** Implement `express-rate-limit`.
- **Input Validation:** specific schemas using Zod/Joi.

Phase 2: Performance and Scalability (1-2 months)

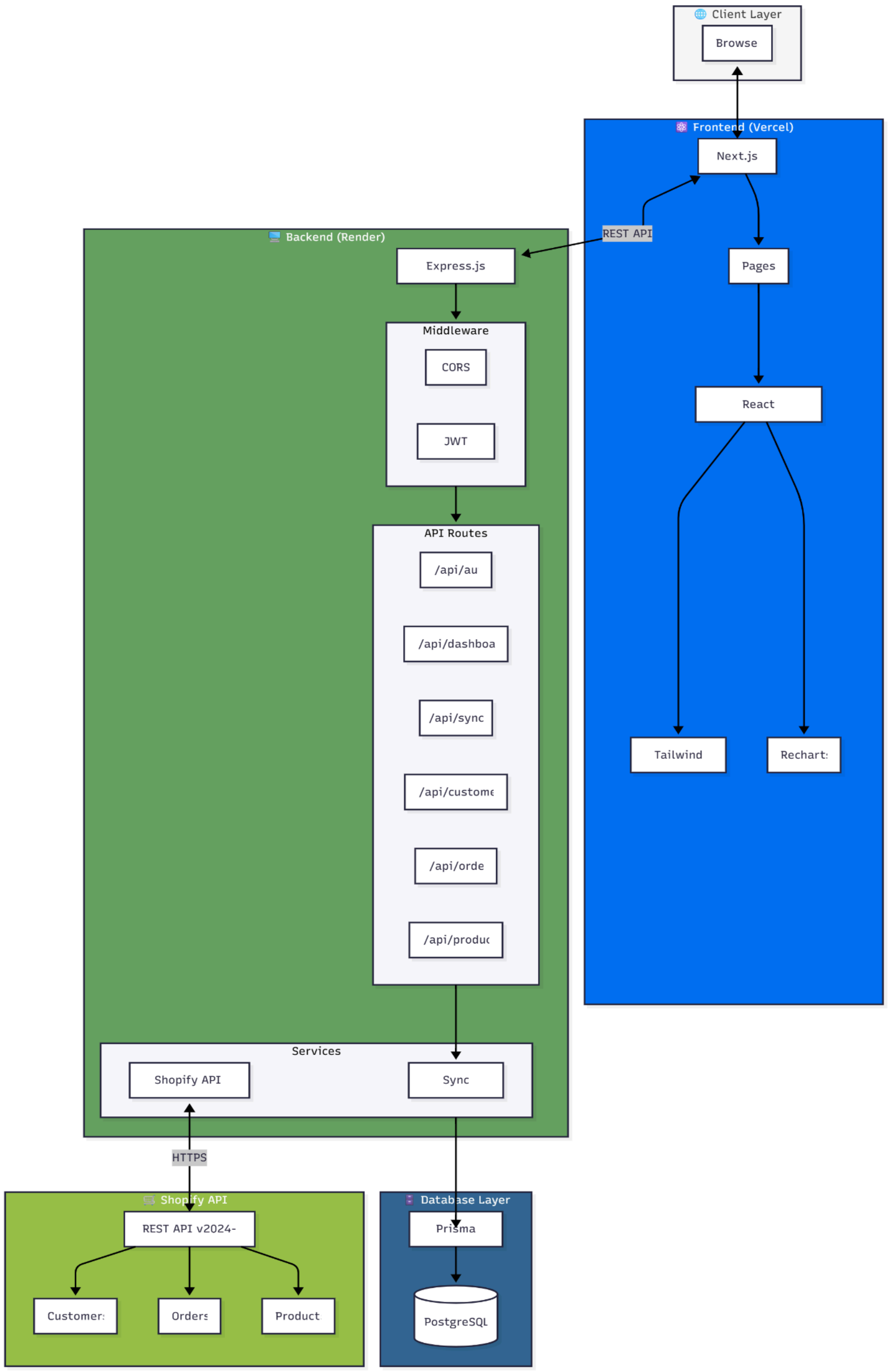
- **Caching Layer:** Redis for dashboard stats.
- **Background Jobs:** BullMQ for async sync operations.
- **Webhooks:** Replace polling with Shopify Webhooks.

Phase 3: AI-Powered Features (2-3 months)

- **RFM Analysis:** Auto-segmentation (Champions, At Risk).
- **Churn Prediction:** ML models for retention.
- **Smart Alerts:** Anomaly detection for orders/inventory.

Project Links:

Frontend: **https://xeno-demo-chi.vercel.app**
Backend: **https://xeno-demo.onrender.com**
GitHub: https://github.com/achennakeshavareddy1301/xeno-demo


Client Layer
Browse
Frontend (Vercel)
Next.js
Pages
React
Tailwind
Recharts
REST API
Backend (Render)
Express.js
Middleware
CORS
JWT
API Routes
/api/au
/api/dashboa
/api/sync
/api/custome
/api/orde
/api/produc
Services
Shopify API
Sync
HTTPS
Shopify API
REST API v2024-
Customers
Orders
Product
Database Layer
Prisma
PostgreSQL