**SparrowX Technical Specifications**

**Overview  
SparrowX is a multi-tenant SaaS platform designed for Jamaican package-forwarding companies. It provides an API-driven backend and a Next.js-based frontend portal for both customers and employees, with data isolation, role-based access control, and dynamic company branding.**

**1. System Architecture**

* **Frontend: Next.js application using Auth0 for authentication, TailwindCSS (Shadcn) for styling, and React Context for dynamic theming.**
* **Backend: Express.js API secured by Auth0 JWTs, organized into controllers, services, and repositories.**
* **Database: PostgreSQL accessed via Drizzle ORM; single database with shared schema, using company\_id for tenant isolation.**
* **Authentication: Auth0 Organizations and RBAC for user roles (Customer, Admin L1, Admin L2).**
* **Validation: Zod schemas on both frontend and backend for payload validation.**

**2. Multi-Tenant Strategy**

* **Data Isolation: Every table includes a company\_id foreign key; every query filters by the authenticated tenant’s ID.**
* **Auth0 Organizations: Each tenant is an Auth0 Organization for isolated login/branding. Tokens include company\_id and roles claims.**
* **Tenant Context: Middleware extracts company\_id from JWT and sets req.companyId for all routes.**

**3. Authentication & Authorization**

* **Sign-Up / Login:**
  + **Customers register via /api/companies/:companyId/customers.**
  + **Employees (L2) are invited via Auth0 Management API.**
* **Roles:**
  + **Customer: Access to their own packages, prealerts, invoices.**
  + **Admin L1: Customer and package management, bill generation, payments.**
  + **Admin L2: All L1 rights plus employee management, company settings, fee configuration.**
* **Middleware: checkJwt verifies JWT, checkRole(role) enforces RBAC.**

**4. Database Schema & Drizzle Migrations**

**Below is the consolidated, multi‑tenant database schema, updated from the old design. All tables include a company\_id foreign key (UUID) to enforce tenant isolation. Primary keys use UUIDs to simplify federation and security.**

**Core Tables and Field Descriptions**

**Below are all core tables with detailed descriptions for each field.**

**companies**

* **id (UUID PK): Unique identifier for each tenant.**
* **name (TEXT): Official company name.**
* **subdomain (TEXT): Unique subdomain for tenant portal.**
* **images (JSONB): Object storing sets of image URLs (e.g., {"logo": "...", "banner": "..."}).**
* **address (TEXT): Street address for the company.**
* **phone (TEXT): Contact phone number.**
* **locations (TEXT[]): Locations where users can pick up their packages.**
* **email (TEXT): Support or general contact email (unique).**
* **website (TEXT): Company website URL.**
* **bank\_info (TEXT): Banking details for payments.**
* **created\_at (TIMESTAMPTZ): Record creation timestamp.**
* **updated\_at (TIMESTAMPTZ): Last modification timestamp.**

**company\_assets**

* **id (UUID PK): Unique asset identifier.**
* **company\_id (UUID FK): References companies.id to tie asset to tenant.**
* **type (ENUM): Asset type, one of (logo, banner, favicon, small\_logo).**
* **url (TEXT): Public URL of the stored asset.**
* **created\_at (TIMESTAMPTZ): When the asset was uploaded.**

**users**

* **id (UUID PK): Unique identifier for all user accounts.**
* **company\_id (UUID FK): References tenant to which user belongs.**
* **email (TEXT): User login email (unique).**
* **password\_hash (TEXT): Hashed password (Auth0 stores credentials; this field can mirror if needed).**
* **role (ENUM): User role, one of (Customer, Admin\_L1, Admin\_L2).**
* **status (TEXT): Account status, e.g. (active, suspended).**
* **created\_at (TIMESTAMPTZ): When the user record was created.**
* **last\_login (TIMESTAMPTZ): Timestamp of most recent login.**

**customers**

* **id (UUID PK): Unique customer identifier.**
* **company\_id (UUID FK): Tenant reference.**
* **user\_id (UUID FK): Reference to corresponding users.id.**
* **first\_name, last\_name (TEXT): Customer’s name.**
* **trn (TEXT): Tax Registration Number (unique per company).**
* **pickup\_location (TEXT): Selected location for pickup (provided by the company).**
* **phone (TEXT): Contact phone number.**
* **created\_at (TIMESTAMPTZ): When the customer was onboarded.**
* **updated\_at (TIMESTAMPTZ): Last profile update.**

**employees**

* **id (UUID PK): Unique employee record identifier.**
* **company\_id (UUID FK): Tenant reference.**
* **user\_id (UUID FK): Corresponding users.id.**
* **first\_name, last\_name (TEXT): Employee’s name.**
* **email (TEXT): Work email (unique per tenant).**
* **role (ENUM): Employee role (Admin\_L1 or Admin\_L2).**
* **created\_at (TIMESTAMPTZ): Onboarding timestamp.**
* **updated\_at (TIMESTAMPTZ): Last update time.**

**prealerts**

* **id (UUID PK): Unique prealert identifier.**
* **company\_id (UUID FK): Tenant reference.**
* **customer\_id (UUID FK): Customer who created prealert.**
* **package\_id (UUID FK, optional): Linked package after arrival.**
* **description (TEXT): Customer’s package description.**
* **weight (NUMERIC): Package weight (lbs or kg).**
* **tracking\_number (TEXT): Carrier tracking identifier.**
* **cost (NUMERIC): Declared value or cost estimate.**
* **invoice\_file\_url (TEXT): URL to uploaded invoice document.**
* **filename (TEXT): Original filename of uploaded document.**
* **created\_at (TIMESTAMPTZ): When prealert was submitted.**

**packages**

* **id (UUID PK): Unique package identifier.**
* **company\_id (UUID FK): Tenant reference.**
* **customer\_id (UUID FK): Owner of the package.**
* **description (TEXT): Internal description.**
* **weight (NUMERIC): Actual weight.**
* **tracking\_number (TEXT): Carrier tracking info.**
* **status (TEXT): Current state (awaiting, in\_transit, delivered).**
* **customs\_cost (NUMERIC): Customs duties if any.**
* **invoice\_id (UUID FK, optional): Linked invoice.**
* **tags (TEXT[]): Package categories (e.g., fragile, oversize).**
* **source (ENUM 'manual','magaya'): Origin of the package data.**
* **magaya\_shipment\_id (TEXT): Identifier from Magaya system.**
* **created\_at (TIMESTAMPTZ): When package was entered into system.  
  \*\* (TIMESTAMPTZ): When package was entered into system.**

**invoices**

* **id (UUID PK): Invoice record identifier.**
* **company\_id (UUID FK): Tenant reference.**
* **customer\_id (UUID FK): Billed customer.**
* **status (TEXT): Invoice status (pending, paid, overdue).**
* **invoice\_number (TEXT): Human‑friendly invoice code.**
* **subtotal (NUMERIC): Sum of line items before tax.**
* **tax\_amount (NUMERIC): Applied tax.**
* **total\_amount (NUMERIC): Grand total.**
* **notes (TEXT): Optional comments.**
* **items (JSONB): Array of line‑item objects ({description, amount}).**
* **created\_at, updated\_at (TIMESTAMPTZ): Timestamps.**

**payments**

* **id (UUID PK): Payment identifier.**
* **company\_id (UUID FK): Tenant reference.**
* **invoice\_id (UUID FK): Invoice being paid.**
* **amount\_paid (NUMERIC): Amount recorded.**
* **payment\_date (TIMESTAMPTZ): When payment occurred.**
* **payment\_method (TEXT): Method (credit\_card, bank\_transfer).**
* **status (TEXT): completed, failed, pending.**
* **transaction\_ref (TEXT): External gateway reference.**
* **created\_at, updated\_at (TIMESTAMPTZ): Audit timestamps.**

**fees**

* **id (UUID PK): Fee definition ID.**
* **company\_id (UUID FK): Tenant reference.**
* **name (TEXT): Fee label (e.g., “Oversize Surcharge”).**
* **code (TEXT): Unique fee code.**
* **fee\_type (TEXT): Category (tax, service, customs).**
* **calculation\_method (ENUM): Calculation strategy (flat, percent, per\_weight, delayed, tiered, volume).**
* **amount (NUMERIC): Base rate or percentage.**
* **currency (TEXT): Currency code (default USD).**
* **applies\_to (TEXT[]): Tags determining applicability.**
* **description (TEXT): Detailed explanation.**
* **enabled (BOOLEAN): Active/inactive flag.**
* **created\_at, updated\_at (TIMESTAMPTZ): Timestamps.**

**package\_fees**

* **id (`UUID PK...**

**10. Package Data Integration with Magaya**

**To streamline package data ingestion and minimize manual entry, SparrowX integrates with Magaya’s Open API for automated synchronization of warehouse shipment information.**

**Overview**

* **Magaya’s XML-based Web Service API enables retrieval of shipment details, tracking updates, and invoice data directly from the warehouse system.**

**Integration Approach**

* **API Communication: Scheduled jobs poll Magaya endpoints for new or updated shipments, importing fields such as magaya\_shipment\_id, tracking\_number, weight, and status.**
* **Data Mapping: Magaya data structures are mapped to SparrowX’s packages schema, populating fields like description, weight, status, and customs\_cost.**
* **Scheduled Synchronization: A configurable cron schedule ensures package data is refreshed (e.g., every 15 minutes), with logs for audit and error tracking.**
* **Error Handling: Failures (e.g., network issues, schema mismatches) are captured in application logs and surfaced via the Super Admin System Logs page.**

**Database Schema Adjustments**

* **New Fields on packages:**
  + **source (ENUM 'manual','magaya'): Indicates data origin.**
  + **magaya\_shipment\_id (TEXT): Unique identifier from Magaya.**
* **Indexing: Add an index on magaya\_shipment\_id for efficient upserts and lookups.**

**User Interface Enhancements**

* **Data Source Indicator: In the Package List and Package Details views, display a badge (Imported from Magaya) when source = 'magaya'.**
* **Manual Override: Maintain “Add Package” and “Edit” buttons for manual entry or correction, ensuring flexibility when automated data is incomplete.**

**By incorporating Magaya integration, SparrowX achieves real-time accuracy of package records while retaining manual controls for exceptional cases.**

**11. UI Design: Admin Dashboards (L1 & L2)**

**This section describes the frontend design for both Admin L1 and Admin L2 dashboards, detailing components, layout conventions, and feature access.**

**11.1 General Layout & Conventions**

* **Responsive Layout: Uses a two-pane layout: a collapsible Sidebar on the left and a scrollable Main Content area on the right. On smaller screens, the sidebar collapses into a hamburger menu.**
* **Header Bar: Persistent across all admin pages, containing:**
  + **Company logo (from company\_assets)**
  + **Page title**
  + **Notification icon with badge for new alerts**
  + **User avatar dropdown (Profile, Settings, Logout)**
* **Theming: Follows ShadCN design tokens; primary and accent colors pulled from tenant branding. Components adhere to consistent spacing (p-4, m-4), typography (text-lg for headings, text-base for body), and rounded corners (rounded-lg).**
* **NavigationActive State: Active menu items in the sidebar use bold text and a left border accent.**
* **Data Fetching Indicators: Use a Skeleton or Spinner for tables and cards while loading.**

**11.2 Sidebar Menu Items**

* **Dashboard (L1 & L2)**
* **Customers (L1 & L2)**
* **Prealerts (L1 & L2)**
* **Packages (L1 & L2)**
* **Invoices (L1 & L2)**
* **Payments (L1 & L2)**
* **Reports (L1 & L2)**
* **Fees (L2 only)**
* **Employees (L2 only)**
* **Company Settings (L2 only)**

**11.3 Dashboard Overview Page**

**Components:**

* **Stat Cards: Horizontal cards showing key metrics (Total Packages, Pending Prealerts, Outstanding Invoices, Revenue This Month). Each card uses an icon, metric value, and small sparkline.**
* **Recent Activity Feed: Vertical list component showing latest actions (e.g., “Prealert #123 approved by Alice”, “Invoice #456 paid”). Uses List and Badge components for status.**
* **Alerts Banner: Dismissible alert at top for system-wide messages (e.g., integration errors).**

**11.4 Data Tables & Forms**

**Data Table Component:**

* **Columns: sortable and filterable headers, inline search box.**
* **Pagination controls at bottom.**
* **Action column with Menu (three-dot) containing row-specific actions.**
* **Implemented via DataTable, with props for columns, data, loading state.**

**Form Component:**

* **Use Form, FormField, FormItem, Input, Select, Textarea, Switch from ShadCN/ui.**
* **Validation UI: inline error messages beneath fields.**
* **Submit button with loading state and disabled until form is valid.**

**11.5 Feature Pages**

**11.5.1 Customers**

* **Table View: List of customers with columns (Name, Email, TRN, Phone, Status, Actions).**
* **Customer Details Drawer: Side panel opens on row click, showing profile info and edit button.**
* **Edit Customer Modal: Form prefilled with customer data.**

**11.5.2 Prealerts**

* **Table View: Columns (ID, Customer, Tracking, Weight, Status, Submitted At, Actions).**
* **Approval Workflow: Row action Approve or Reject triggers a confirmation dialog and updates status.**

**11.5.3 Packages**

* **Table View: Columns (ID, Tracking, Source, Status, Weight, Arrival Date, Actions).**
* **Details Page: Tabs (Overview, Fees, Invoices, Tracking History). Each tab uses a Card to display related data.**
* **Manual Entry: Add Package button opens modal form for manual entry.**

**11.5.4 Invoices & Payments**

* **Invoices Table: Columns (Invoice #, Customer, Amount, Status, Created At, Actions).**
* **Generate Invoice Modal (L1 & L2): Dropdown to select packages, auto-calculate totals, fee line items displayed in Table.**
* **Payments Page: List of payments with filter by date/customer.**

**11.5.5 Reports**

* **Report Builder: Select date range, metrics, and grouping. Uses DatePicker, Select, and a Button to run report.**
* **Chart Display: Use Recharts line/bar charts embedded in Cards.**

**11.5.6 Fees Management (L2)**

* **Fees Table: Columns (Name, Type, Calculation, Amount, Tags, Status, Actions).**
* **Fee Form Drawer: Sidebar drawer for create/edit, reusing form components.**
* **Bulk Operations: Checkbox select multiple fees to enable/disable.**

**11.5.7 Employee Management (L2)**

* **Employees Table: Columns (Name, Email, Role, Status, Last Login, Actions).**
* **Invite Employee Modal: Form to add new employee.**
* **Role Assignment: Inline dropdown in table to change role.**

**11.5.8 Company Settings (L2)**

* **Settings Tabs: Branding, Preferences, Magaya Integration.**
* **Branding Tab: Drag-and-drop file upload for logo, banner; preview thumbnails.**
* **Preferences Tab: Toggle switches for feature flags, exchange-rate input.**
* **Magaya Integration Tab: Display connection status, button to test API connection, scheduling control for sync interval.**

**11.6 Accessibility & Best Practices**

* **Keyboard Navigation: Ensure all interactive elements reachable via TAB.**
* **ARIA Labels: Provide aria-label on icons and buttons.**
* **Color Contrast: Adhere to WCAG AA for text and UI elements.**
* **Responsive Tables: Collapse columns or use horizontal scroll on small screens.**

**11.7 Reusable Component Library**

* **Button: Variants (primary, secondary, danger), sizes.**
* **Input/Select/Textarea: Standardized spacing and error state styling.**
* **Modal/Drawer: Title, close icon, footer actions.**
* **Table/DataGrid: Unified props interface for pagination, sorting.**
* **Card: CardTitle, CardContent, CardFooter helpers.**

**This detailed UI blueprint will guide the Next.js + ShadCN implementation, ensuring consistency, accessibility, and alignment with backend capabilities.**