Assignment: Shopify Order Management Interface

Objective:

Design and develop an interface that integrates Shopify to fetch order data, implements role-based authorization, updates delivery status via Shiprocket API, and provides analytical insights with charts on the homepage.

Core Functionalities:

1. Shopify Integration & Order Fetching

- Authenticate and connect with the client's Shopify store.
- Fetch all new orders with fulfillment status "Pending".
- Store order data in an AWS database.
- Update orders hourly using the Shiprocket API for tracking.

2. Role-Based Authorization

- Implement authentication and authorization for different user roles:
    - **Bfast Admin** - Full access (Home, Pending Orders, Order Data, Track, Manifest, Channel, User & Client Creation)
    - **Bfast Executive** - Access to Home, Pending Orders, Order Data, and Track/Manifest.
    - **Client Admin** - Access to Home, Pending Orders, Order Data, and Track/Manifest for their own store.
    - **Client Executive** - Access to Home, Order Data, and Track/Manifest for their own store.
- Use JWT or OAuth-based authentication systems.

3. Data Storage & Updates in AWS Database

- Maintain order data in AWS with real-time tracking updates from Shiprocket.

4. Home Page Dashboard

- **Charts & Analytics:**
    - Total Shipments - Status-wise breakdown (Delivered, RTO, In Process, NDR, Lost)
    - India Map visualization with Order Distribution (Time Period Filter)

5. Pending Orders Management

- Display orders with fulfillment status "Pending" or "In-Process".
- Option to update shipment details (Dimensions, Weight, Transport Mode).
- Bulk upload via CSV for mass updates.
- Download Pending / In Process Orders in csv Format

6. AWB Assignment & Order Update

- Bfast Admin can upload a CSV to assign AWB numbers to multiple orders at once.
- Once AWB is assigned, the order moves from "Pending Orders" to "Order Data".

7. Order Data Page

- Displays all orders in a table format.
- Filter bar for searching based on criteria like status, date, courier, etc.
- Includes all order details for easy reference.

Note - Only Client Specific Data will be visible to client_admin and client_executive

8. Public Shipment Tracking Page

- Route: `/track/{awb}`
- Fetch shipment details from Shiprocket API.
- Display order progress with an improved UI/UX.
- Include brand logo on the tracking page.

Routes & Access Control:

Route	Access
`/add-client`	Bfast Admin
`/add-user`	Client Admin, Bfast Admin
`/channels`	Bfast Admin
`/manifest-orders`	Bfast Admin
`/login`	Client Admin, Bfast Admin, Bfast Executive, Client Executive
`/home`	All Users

`/pending-orders`	All Users
`/order-data`	All Users
`/track/{awb}`	Public

Database Schema: Order Table

```
{
  "client_id": "string",
  "shopify_store_id": "string",
  "created_at": "datetime",
  "order_id": "string",
  "fulfillment_status": "string", // Pending, In-Process, Completed
  "pickup_date": "datetime",
  "shipping_details": {
    "name": "string",
    "phone_1": "string",
    "phone_2": "string",
    "email": "string",
    "address": "string",
    "pincode": "string",
    "city": "string",
    "state": "string"
    "shipping_method": "string" // Express, Surface
    "payment_mode": "string", // COD, Prepaid
    "amount": "float",
  },
  "product_details": {
    "category": "string",
    "product_name": "string",
    "quantity": "integer",
    "dimensions": ["float", "float", "float"], // L, B, H in cm
    "weight": "float", // in Kg
  },
  "courier": "string",
  "awb": "string",
  "delivery_status": "string", // Delivered, In Transit, RTO, etc.
  "last_scan_location": "string",
  "last_timestamp": "datetime",
  "last_remark": "string"
}
```

Deliverables:

1. **Front-End (React, TypeScript)**
    - UI components for Dashboard, Order Tables, Tracking Page
    - Charts and Maps for order analytics
    - Role-based UI rendering
2. **Back-End (FastAPI, Node.js)**
    - API endpoints for fetching, updating, and managing orders
    - Authentication and authorization middleware
    - Scheduled task for hourly Shiprocket updates
3. **Database (AWS RDS)**
    - Structured schema as defined above
4. **Deployment & Hosting (AWS)**
    - Hosting using AWS
    - CI/CD pipeline for automated deployments

Evaluation Criteria:

- API Efficiency & Data Handling
- UI/UX Quality
- Security & Role-Based Access Implementation
- AWS Cloud Optimization
- Overall System Performance
