MODULE 4: CUSTOMER MANAGEMENT (ACCOUNTS RECEIVABLE)

this is a large projecct ..my company is building ...this is my part ..given to me , read well , I don,t want to slow them down ..

Frontend Responsibility:
You are responsible for building all frontend pages for the Customer Management (AR) module using React + Ionic. and material ui for styling ..keep the styling simple and professional so it will not look different from the other modules the other frontend devs are building The backend team will supply APIs later. Your task is to create reusable, API -ready UI components that can be plugged into the core system.

Folder Structure (Deliverable)
customer-ar/
- pages/
- components/
- services/
- hooks/
- types/
- routes.ts
- index.ts

Pages to Build

1. Customer List Page
Route: /customers
Purpose: View all customers
Features: - Search customers - Filter by status - Customer table (name, code, credit limit, balance, status) - Actions: View customer, View statement

2. Create / Edit Customer Page/ delete customer / view customer details
Routes: - /customers/new - /customers/:id/edit
Fields: - Customer name - Customer type - Credit limit - Payment terms - Status

3. Customer Detail (Overview)
Route: /customers/:id
Sections: - Customer profile card - Summary cards (Invoices, Outstanding, Paid) - Tabs: Invoices, Payments, Ledger, Statement

4. Customer Invoices Page
Route: /customers/:id/invoices
Table: - Invoice number - Date - Amount - Balance - Status

5. Invoice Detail Page
Route: /invoices/:invoiceId
Displays: - Invoice header - Line items - Payment allocations - Remaining balance

6. Receipts & Allocation Page
Route: /customers/:id/receipts

7. Customer Ledger Page

Route: /customers/:id/ledger
Table: - Date - Reference - Debit - Credit - Balance Includes date filters.

8. Customer Statement Page

Route: /customers/:id/statement
Features: - Date range selector - Statement preview - Export to PDF

Reusable Components

- CustomerCard - SummaryCard - DataTable - StatusBadge - DateRangePicker - LoadingSkeleton - EmptyState

Services (API Layer)

- customer.service.ts - invoice.service.ts - receipt.service.ts
All API calls must be abstracted here.

Types

- customer.ts - invoice.ts - receipt.ts - ledger.ts

Development Strategy

1. Build all pages ..don,t use mock data , when api arrives i will integrate it so it can display live data where required.
2. Ensure clean routing and reusable components.
3. When APIs arrive, i will connect via services only.

Important Rules

- No business logic on frontend - No accounting calculations - Read-only invoices and receipts - Backend controls validation and posting

Final Deliverable

A plug-and-play folder: customer-ar/ that can be integrated directly into the main project.