# MONTH 4 Project Title:-CRM Software

### **2** Client Overview:

A business client needs a web-based **Customer Relationship Management (CRM)** tool to manage leads, customers, tasks, and sales pipelines. The goal is to increase team productivity and customer engagement via a centralized CRM platform.

# **©** Project Objective:

To develop a secure, scalable full-stack CRM system using:

• Backend: Java (Spring Boot)

• Frontend: React.js

• Database: MySQL

• Authentication: JWT + Spring Security

The system must allow users (Admins and Sales Reps) to manage customers, leads, tasks, and sales, with role-based access and user authentication.

### Functional Modules

#### 1. 👗 User Management & Authentication

# API Endpoints:

- POST /api/register Register new user (Admin or Sales)
- POST /api/login Login, return JWT token
- GET /api/users/me Get logged-in user profile

#### User Details:

- Full Name
- Email
- Password (encrypted)
- Role: Admin / Sales
- CreatedAt

# Security:

- Use **JWT** for login tokens
- Use **Spring Security** for role-based API restrictions

• Frontend stores token in localStorage

### 2. **L** Customer Management

### **✓** API Endpoints:

- GET /api/customers
- GET /api/customers/{id}
- POST /api/customers
- PUT /api/customers/{id}
- DELETE /api/customers/{id}

#### Customer Data:

- Name
- Email
- Phone
- Company
- Address
- Assigned Sales Rep
- Notes

### 3. <a>\bar{\text{\text{Cart}}}\] Lead Management</a>

### API Endpoints:

- GET /api/leads
- POST /api/leads
- PUT /api/leads/{id} Update lead info or status
- DELETE /api/leads/{id}

### Lead Data:

- Name
- Contact Info
- Source (Referral, Ads, Web)
- Status (New, Contacted, Converted, Lost)
- Assigned Sales Rep

# 4. 🔋 Task Management

# **✓** API Endpoints:

- GET /api/tasks
- POST /api/tasks
- PUT /api/tasks/{id}
- DELETE /api/tasks/{id}

### Task Data:

- Title
- Description
- Due Date
- Priority
- Assigned To (User ID)
- Status (Open, In Progress, Completed)

### 5. ales Management

# API Endpoints:

- GET /api/sales
- POST /api/sales
- GET /api/sales/{id}
- PUT /api/sales/{id}

#### Sales Data:

- Customer ID
- Amount
- Status (Proposal, Negotiation, Closed-Won, Closed-Lost)
- Date
- Assigned Sales Rep

# Database Schema (MySQL)

#### Tables:

users

- customers
- leads
- tasks
- sales

#### **Relationships:**

- A user (sales rep) can be assigned to leads, customers, tasks, and sales.
- Sales must be linked to customers.

# Security

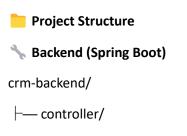
- JWT authentication system
- Spring Security for role-based authorization
- React route protection via React Router and token check

### **API Documentation**

Use Swagger/OpenAPI for auto-generated docs
 Accessible at http://localhost:8080/swagger-ui.html

#### Frontend (React.js)

- Login / Registration pages
- **Dashboard:** Overview of sales/leads/tasks
- Customers Page (list, add, edit, delete)
- Leads Page (filter by status, assign to sales rep)
- Tasks Page (my tasks, assign task, mark done)
- Sales Page (track deals, update status)
- Role-based rendering (Admin vs Sales Rep views)



— service/

├— repository/
├— model/
├— security/
├— dto/
└— application.properties
☑ Frontend (React)

crm-frontend/
├— src/
│ ├— components/
│ ├— pages/
│ ├— services/ (Axios API calls)
│ ├— utils/

### m Project Timeline (4 Weeks)

#### Week Backend

#### **Frontend**

- 1 User Auth + JWT + DB setup Login/Register Pages
- 2 CRUD for Customers & Leads Customer & Lead Pages
- 3 Tasks & Sales Modules Task & Sales Pages
- 4 Swagger, Pagination, Tests Dashboard, UX, Final Polish

#### **Deliverables**

- Complete source code (frontend + backend)
- Postman Collection for API testing
- Swagger API docs
- Fully running full-stack CRM system
- GitHub repository (optional)

### Testing

• Backend: Unit tests with JUnit + Mockito

- API Testing: **Postman** or Swagger UI
- Frontend: Manual testing (Jest optional)

# Optional Enhancements

- Notification system for tasks
- Email sending (for follow-ups)
- Export reports (PDF, CSV)
- Dark mode toggle
- Dashboard charts (Sales stats)