CRM Application – Jewel Management
Developer Documentation

1. Overview

The Jewel Management CRM Application is designed to manage customer relationships, inventory, sales, and service requests for jewelry businesses. It provides a unified platform for tracking customers, managing jewelry items, processing orders, and handling after-sales services. This documentation is for developers who are building, extending, or maintaining the application.

2. Architecture

Frontend: React (or Lightning Web Components if Salesforce)
Backend: Node.js / Apex classes
Database: MySQL / Salesforce Objects
APIs: RESTful APIs for integration
Authentication: OAuth 2.0 / Salesforce login
Deployment: GitHub → CI/CD → Cloud

3. Modules

Customer Management: Maintain customer profiles.
Jewelry Inventory: Track items.
Sales & Orders: Record orders.
Service & Repairs: Manage repair requests.
Reports: Generate analytics.

4. Data Model

Core Objects/Tables: Customer, Item, Order, OrderItem, ServiceRequest.
Relationships:
Customer 1-n Order
Order n-n Item (via OrderItem)

5. API Endpoints (Examples)

GET /api/customers – Retrieve all customers
POST /api/customers – Add a new customer
GET /api/items – Retrieve inventory items
POST /api/orders – Create a new order
PUT /api/service-requests/:id – Update service request status

6. Developer Setup

1. Clone repository: git clone ...
2. Install dependencies: npm install
3. Configure environment variables: .env
4. Run locally: npm start

5. Salesforce org setup: Deploy Apex classes, configure custom objects, assign permissions

7. Deployment

Push changes to GitHub main branch.
CI/CD pipeline runs tests and deploys to staging.
Manual or automatic deployment to production.

8. Extending the Application

Add new custom objects or fields.
Integrate with payment gateways.
Build Lightning or React components for new UI features.
Use webhooks to sync with third-party systems.

9. Testing

Unit Tests: Jest / Apex test classes
Integration Tests: Postman / Newman
UI Tests: Cypress or Selenium

10. Security

Ensure sensitive data is encrypted at rest.
Enforce field-level and object-level security.
Audit logs for every update.

11. Contribution Guidelines

Fork the repository
Create a feature branch: feature/
Commit with conventional messages
Submit a Pull Request

12. License

Specify your license (MIT, Apache 2.0, etc.)