Technical Documentation

Leads Management System

Version - [1.0]

Date - 3rd April 2025

Prepared by : Saoud Ahmed

© OnTimeits.com

Background

Purpose

The Leads Managements System is a full stack application designed to store , display and manage leads efficiently. The frontend built with ReactJS provides an interactive user interface, allowing users to submit , display and manage lead information. The backend leverages Java Spring Boot with Restful APIs for handling data and its retrieval from database

Technology Stack

- **Frontend**: ReactJS, TailwindCSS, JavaScript
- **Backend**: Java, Spring Boot, Spring Data JPA (Hibernate)
- **Database**: H2 (In-Memory Database)
- **Dependencies and Tools**: React Hooks, Fetch API
- **Deployment** : Backend (render.com) , Frontend (vercel)

Setup Instructions

- **Frontend**
  a. Clone the Github Repository
  b. Move to leadsmgmt Directory
  c. Open Directory inside an IDE / VS code
  d. Run “npm install” inside terminal
  e. Run “npm run dev” to run frontend locally
- **Backend**
  - Open Project on IntelliJ Community Edition
  - Navigate to ./src/main/java/com/example/leads
  - Open file “ LeadsBackendApplication.java “
  - Press the Play/Run icon on the top right
  - Or
  - Right click on the file and Click on Run application
  - (Ensure Dependencies are Installed and Working)

Module Requirements

Functional Requirements

- Submit lead data via frontend form to backend API
- Retrieve all stored leads in the database
- Delete leads by UUID
- Input Validation on fields present inside form
- Pagination Support in table view of data

Non-Functional Requirements

- Response UI with easy to navigate form design
- Graceful error handling with appropriate feedback
- Code maintainability and Readability

Module/Feature Implementation Details - Code Documentation

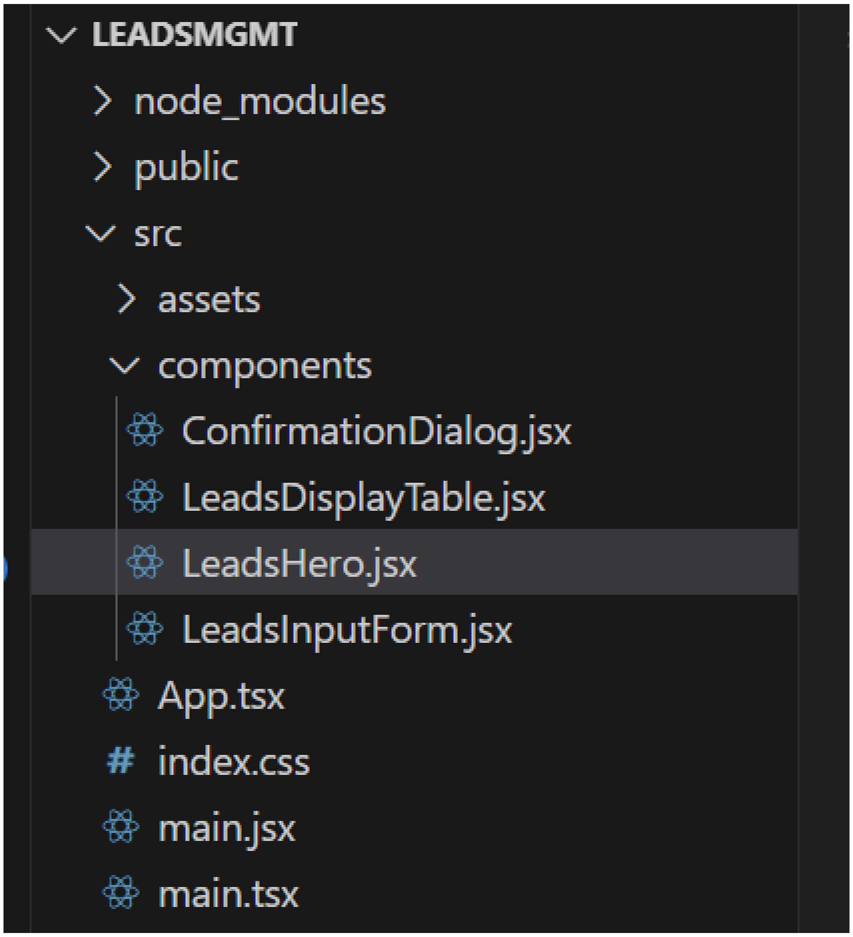
Function/Class Descriptions

- **Frontend**
  - App (App.tsx)
    - Entry point rendering only LeadsHero Component
  - LeadsHero (LeadsHero.jsx)
    - Combines LeadsInputForm and LeadsDisplayTable components, applying Flexbox layout to present the UI clearly
  - LeadsInputForm (LeadsInputForm.jsx)
    - Communicates with backendAPI
    - Handles insertion and deletion of leads
  - LeadsDisplayTable (LeadsDisplayTable.jsx)
    - Fetches and displays leads from the backend API in a paginated table format
  - ConfirmationDialog (ConfirmationDialog.jsx)
    - Reusable Dialog Modal for confirming actions
- **Backend**
  - Main Application (LeadsBackendApplication.java)
    - Entry Point for the Spring Boot application
  - REST Controller (LeadController.java)
    - Manages HTTP Endpoints (GET, POST, DELETE) for leads
  - Service Layer (LeadService.java)
    - Contains Business Logic for CRUD operations interfacing with the repository layer
  - Repository (LeadRepository.java)
    - Leverages Hibernate by extending JPA operations for database
  - Entity Model (Lead.java)
    - Defines database schema and object mapping for leads

Folder Structure Explanation

- Frontend/
  - Components/
    - LeadsInputForm.jsx
    - LeadsHero.jsx
    - LeadsDisplayTable.jsx

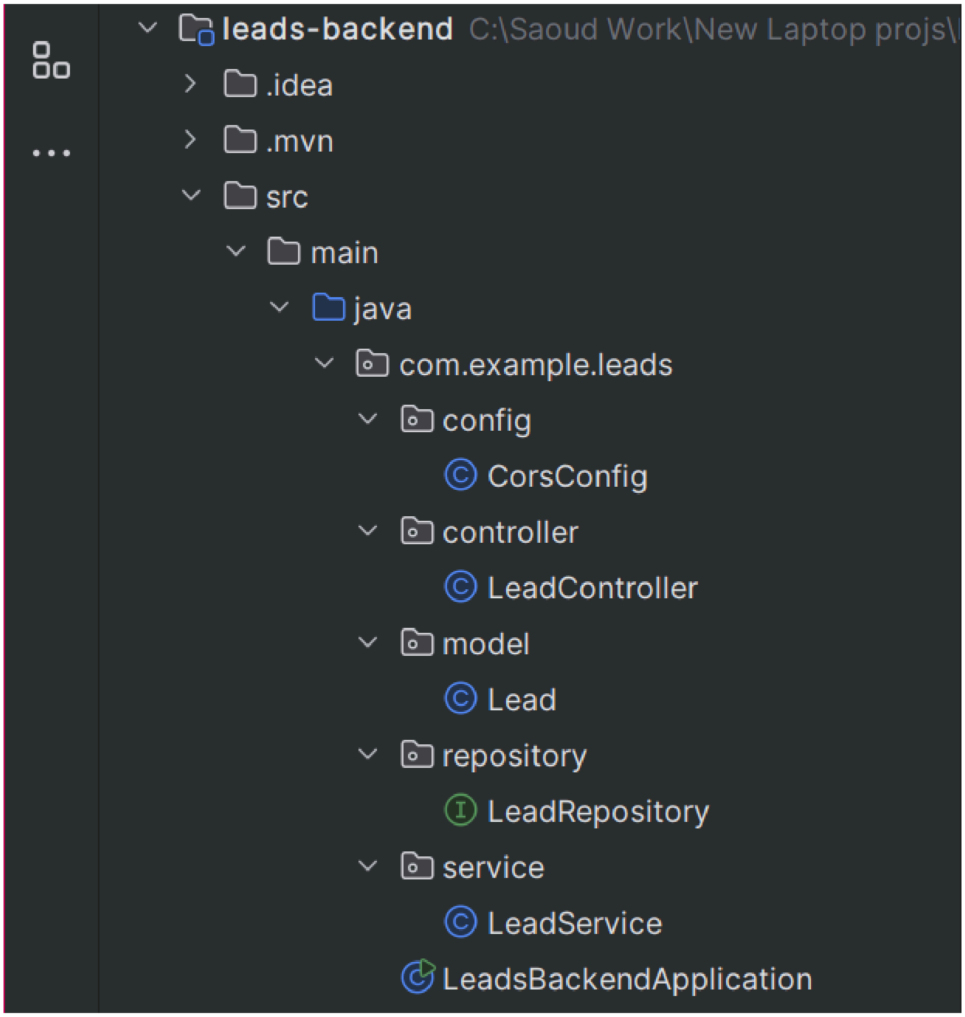
        - ConfirmationDialog.jsx
    - App.tsx
    - public
    - styles
    - assets



```
LEADSMGMT
  > node_modules
  > public
  v src
    > assets
    v components
      ConfirmationDialog.jsx
      LeadsDisplayTable.jsx
      LeadsHero.jsx
      LeadsInputForm.jsx
    App.tsx
    index.css
    main.jsx
    main.tsx
```

- Backend/
    - src/
        - main/
            - java/
                - com/
                    - example/
                        - leads/
                            - config/
                                - CorsConfig.java
                            - controller/

    - LeadController.java
- model/
    - Lead.java
- repository/
    - LeadRepository.java
- service/
    - LeadRepository.java
- LeadsBackendApplication.java



```
leads-backend C:\Saoud Work\New Laptop projs\
  .idea
  .mvn
  src
    main
      java
        com.example.leads
          config
            CorsConfig
          controller
            LeadController
          model
            Lead
          repository
            LeadRepository
          service
            LeadService
          LeadsBackendApplication
```

Module Name - Database Schema Documentation

Schema Description

Column	Type	Description
id	UUID	Primary key, auto-generated UUID
full_name	VARCHAR	Lead’s full name (Required)
email	VARCHAR	Email Address (Unique, Required)
phone	VARCHAR	Contact Information
company_name	VARCHAR	Company Name
notes	TEXT	Additional notes
created_at	TIMESTAMP	Timestamp when lead is created

Module API Contracts - API Documentation

Endpoints & Methods

- GET /leads ({getLeads})
    - Retrieves all records from the database
- POST /leads ({submitLead(data)})
    - Saves a new lead into the database
- DELETE /leads/{id}
    - Deletes a lead based on UUID entered

Request/Response Formats

- GET /leads ({getLeads})
    - Response
        - [
        - {
        - "id": "0151-159125-21512151",
        - "fullName": "Saoud Ahmed",
        - "email": "saoud@ontimeits.com",
        - "phone": "1234567890",
        - "companyName": "Ontime its.",
        - "notes": "Interview",

        - "createdAt": "2025-03-07T13:31:59"
        - }
        -]
- POST /leads ({submitLead(data)})
    - {
    - "fullName": "Saoud Ahmed",
    - "email": "saoud@ontimeits.com",
    - "phone": "1234567890",
    - "companyName": "Ontime its.",
    - "notes": "Interview",
    - }
    -]
- DELETE /leads/{id}
    - String Response returned " Lead Deleted "

Testing/Error Handling

Common API Errors:

- Localhost:5173 cannot resolve, something went wrong
- Incorrect Setting of CORS
- Attempting to delete non existing lead
- Frontend - Backend Connection errors

Deployment Guide

Deployment Instructions

- Local Deployment
    - Open frontend on VScode
    - Enter "npm install" in terminal
    - Once completed , enter "npm run dev"
    - Switch to intelliJ
    - Navigate to ./src/main/java/com/example/leads
    - Run intellij compiler
    - Both frontend and backend deployed

Time Estimation

Task	Your Estimated Time	My Estimated Time	Actual Time
Setup React & Install Dependencies	1 hour	<1 hour	1.5 hours
Build Form & Validation	2 hours	1-2 hours	1 hour
Create API Endpoints (Java)	3-4 hours	3-4 hours	3 hours
Implement Database Model	2 hours	1-2 hours	1 hour
Connect Frontend to Backend	2-3 hours	1-2 hours	3 hours
Display Leads in Table/List	2 hours	1 hour	1 hour
Testing & Debugging	2 hours	2-3 hours	2 hours
Documentation Writing	1-2 hours	2-3 hours	1.5 hours

Links

- Github : https://github.com/AnnedithB/LMSCompleteProject
- Frontend Github : https://github.com/AnnedithB/LeadsManagementSystem
- Backend Github : https://github.com/AnnedithB/LeadsBackend
- Vercel : http://leads-management-system-tau.vercel.app/