Training Center Management API

This project is a Spring Boot RESTful application that allows users to create, save, and retrieve information about training centers. The application uses an inbuilt H2 file database for persisting data.

# Features

* Create Training Center: Create and save information about a new training center.
* Get Training Centers: Retrieve a list of all stored training centers.
* Validations: Mandatory fields, email, phone, and size validations are handled using annotations.
* H2 Database: Inbuilt file-based H2 database to store training center information.

# Table of Contents

1. Technologies Used
2. Installation
3. Database Configuration
4. Running the Application
5. API Documentation
6. Validation Rules
7. Error Handling
8. Project Structure

# Technologies Used

* Java 17
* Spring Boot 3.3.4
* H2 Database (file-based)
* Maven
* Spring Data JPA

# Installation

Follow the steps below to set up and run the project locally:

## Prerequisites

* Java 17 or above installed.
* Maven installed.

## Clone the Repository

git clone <repository-url>  
cd <repository-directory>

# Database Configuration

The project uses H2 file-based database. No additional configuration is needed for H2, as it is embedded. The database will be created in the file system, and you can access it from the H2 console.

## H2 Database Configuration (in application.properties)

spring.datasource.url=jdbc:h2:file:./data/testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=password

spring.jpa.hibernate.ddl-auto = update

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

# Running the Application

To run the application:

1. Build the application using Maven:

mvn clean install

2. Run the application:

mvn spring-boot:run

By default, the application will start on port 8080. You can access the APIs using a tool like Postman or curl.

# API Documentation

## Create Training Center (POST)

URL: /api/training-centers  
Method: POST  
Content-Type: application/json  
Description: This API endpoint creates a new training center and saves it in the database.

### Request Body:

{  
 "centerName": "ABC Training Center",  
 "centerCode": "TC1234567890",  
 "address": {  
 "detailedAddress": "123, Main Road",  
 "city": "Mumbai",  
 "state": "Maharashtra",  
 "pincode": "400001"  
 },  
 "studentCapacity": 150,  
 "coursesOffered": ["Java", "Spring Boot", "React"],  
 "contactEmail": "info@abctraining.com",  
 "contactPhone": "9876543210"  
}

### Response:

Success: Returns the saved training center information.  
Validation Errors: Returns appropriate error messages for validation failures.

## Get Training Centers (GET)

URL: /api/training-centers  
Method: GET  
Content-Type: application/json  
Description: This API endpoint retrieves all stored training centers.

# Validation Rules

* CenterName: Must be provided and be less than 40 characters.
* CenterCode: Must be exactly 12 characters (alphanumeric).
* Address: Must have a valid detailed address, city, state, and pincode.
* ContactPhone: Must be a valid 10-digit phone number.
* ContactEmail (if present): Must be a valid email address.
* CreatedOn: Automatically generated by the server.

# Error Handling

The application includes an ExceptionHandler to manage validation and other exceptions gracefully. On validation failure, the user will receive a descriptive error message.

# Project Structure

training-center-api/  
 ├── src/  
 │ ├── main/  
 │ │ ├── java/  
 │ │ │ └── com.example.demo/  
 │ │ │ ├── controllers/ # API Controllers  
 │ │ │ ├── services/ # Service Layer  
 │ │ │ ├── repositories/ # Repository for Database Interaction  
 │ │ │ ├── entities/ # Entity Classes  
 │ │ │ └── customExceptions/ # Custom Exception

Handling   
 │ │ └── resources/  
 │ │ ├── application.properties # Application Configuration  
 │ │ └── data.sql # Initial data for H2 (if needed)  
 └── pom.xml # Maven Dependencies