**Employee-CRUD**

* Prerequisites:
  1. List the prerequisites and dependencies required to run the application. This may include:
  2. Java Development Kit (JDK)
  3. Spring Boot
  4. MySQL
  5. IntelliJ IDEA
  6. Postman (for testing the APIs)
* Database Setup
  + Run The sql commands one by one Employee Sql.sql (Location:\CRUD\Employee Sql.sql)
  + Extract the CRUD.zip file and import the extracted folder in Intellij
  + Go to Intellij , expand CRUD folder then the src -> main-> resources -> select application.properties-
    - Change spring.datasource.password=(put your sql server password here)
* Import,Building and Running the Application:
  + Extract the CRUD.zip file and import the extracted folder in Intellij
  + Let Intellij load the files and them go for the build (or use this mvn spring-boot:run).
  + After its build is completed expand CRUD folder then the src -> main->java -> com.example.CRUD then you will be able to see CrudApplication.
  + Right click on CrudApplication and select Run
  + Now the Application is running , we can can test our APi using Postman
* Testing the APIs:
  + Open Postman, Sign in with any google account . Then import the EmployeeCrud.postman\_collection.json . And then test all API’s
* Api Documentation

1. FindAll

* + - Description: Retrieve a list of all employees.
    - Request Method: GET
    - Endpoint: /employees/findAll
    - Response: List of all employees.

2. FindByFirstName

* + - Description: Retrieve employees by their first name.
    - Request Method: GET
    - Endpoint: /employees/findByFirstName
    - Response: Employees matching the provided first name.

3. FindById

* + - Description: Retrieve an employee by their ID.
    - Request Method: GET
    - Endpoint: /employees/findById?id={id}
    - Response: Employee details based on the provided ID.

4. Delete Employee

* + - Description: Delete an employee by their ID.
    - Request Method: DELETE

Endpoint: employees/delete?id={id}

Response: Success message or error message.

5. Create Employee

* + - Description: Create a new employee.
    - Request Method: POST
    - Endpoint: / employees/create?firstName={ firstName }&lastName={ lastName }&phoneNumber={ phoneNumber }&address={ address }&dateOfBirth={ dateOfBirth}
    - Request Body: JSON with employee details (e.g., first name, last name, phone number, address, date of birth).
    - Response: Success message or error message.

6. Update Employee

* + - Description: Update an existing employee's details.
    - Request Method: PUT
    - Endpoint: employees/update?existingId={ existingId }&updatedFirstName={ updatedFirstName }&updatedLastName={ updatedLastName }&updatedPhoneNumber={ updatedPhoneNumber }&updatedAddress={ updatedAddress }&updatedDateOfBirth={ updatedDateOfBirth}
    - Request Parameters: existingId (employee ID), updatedFirstName, updatedLastName, updatedPhoneNumber, updatedAddress, updatedDateOfBirth.
    - Response: Success message or error message.