**Employee Management**

**Objective**

The main objective of this project was to create a thorough Full Stack Application using Spring Boot & MySQL as the back-end and React as the front-end.

Summary:

* Develop a **RESTful API** using **Spring Boot and MySQL for the back-end**, tested it with **Postman**, and integrated it into a **React front-end using axios**.
* **CREATE, READ, UPDATE and DELETE (CRUD)** appliction.
* This repository contains the **back-end** of a **Full Stack** **personal project**, which is **responsible for storing employee information** and is implemented using the **Spring Boot Java Framework** which serves as the **web framework** for the **back-end**.
* This application allows users to **view, add, remove, and edit** individuals within their management system

**Tables:**

**m\_organizations**

id int(10) PK

short\_code varchar(2)

address varchar(100)

city varchar(50)

country varchar(50)

state varchar(50)

pincode varchar(10)

logo\_id int(10) FK to **m\_system\_files**

created\_date datetime

modified\_date datetime

deleted\_date datetime

**m\_system\_files**

id int(10) PK

original\_file\_name varchar(50)

file\_name varchar(50)

bucketname varchar(50)

s3\_url varchar(150)

created\_date datetime

**m\_departments**

id int(10) PK

organization\_id int(10) FK to **m\_oraganization**

name varchar(50)

created\_date datetime

modified\_date datetime

deleted\_date datetime

**m\_designations**

id int(10) PK

name varchar(50)

is\_active bit(1) default 1

created\_date datetime

**t\_employees**

id int(10) PK

emp\_code varchar(15)

first\_name varchar(50)

last\_name varchar(50)

full\_name varchar(100)

email\_id varchar(50)

phone varchar(15)

doj date

organization\_id int(10) FK

designation\_id int(10) FK **m\_designations**

image\_id int(10) FK **m\_system\_files**

**t\_employee\_communication**

id int(10) PK

employee\_id int(10) FK **t\_employees**

is\_permanent bit(1)

address varchar(100)

city varchar(50)

country varchar(50)

state varchar(50)

pincode varchar(10)

**t\_past\_experiences**

id int(10) PK

employee\_id int(10) FK **t\_employees**

company\_name varchar(100)

start\_date date

end\_date date

designation int(10) FK **m\_designations**

responsibilities tinytext null

1. Implement API to fetch organizations

Method: GET

Endpoint: /api/organizations

?searchStr=<name%>

&page=1

&pageSize=<10|20|30>

Response:

{

“data”: [

{

Id:

name:

city,

country

state

pincode…

}

]

}

1. Implement API to fetch departments

Method: get

Endpoint: /api/departments

?searchStr=<name%>

&organizationId=<>

&page=1

&pageSize=<10|20|30>

Response:

{

“data”: [

{

Id:

name:

organizationId:

organizationName:

}

]

}

1. Implement API to fetch designations

Method: GET

Endpoint: /api/designations

?searchStr=<name%>

&page=1

&pageSize=<10|20|30>

Response:

{

“data”: [

{

Id:

name:

}

]

}

1. Implement API to fetch employee list

Method: GET

Endpoint: /api/employees

?searchStr=<%name% | code%>

&organizationId=<>

&designationId=<>

&doj=<>

&page=1

&pageSize=<10|20|30>

Response:

{

Total: ,

Page:,

pageSize: ,

“data”: [

{

id

empCode

firstName

lastName

fullName

emailId

phone

doj

organizationId

organizationName

designationId

designationName

imageUrl:

}

]

}

1. Implement API to fetch employee details based on ID

Method: GET

Endpoint: /api/employees/<employeeId>

Response:

{

“data”: {

id

empCode

firstName

lastName

fullName

emailId

phone

doj

organizationId

organizationName

designationId

designationName

imageUrl:

communications: [

{}

],

pastExperiences: [

{}

]

}

}

1. Implement API to create employee

Method: POST

Endpoint: /api/employees

request:

{

firstName

lastName

emailId

phone

doj

organizationId

designationId

imageFile:

communications: [

{}

],

pastExperiences: [

{}

]

}

1. Geneate emp code 8 digit

Ex: <org shortcode>+autoincrement of that organization total count of emp

CG000001

Image should be uploaded to S3 so that the s3 url will be generated

Save the s3 url image in system\_file table

Communications will be saved in communication table

Past exp will be saved in past\_exp table

The master data of employee will be saved in t\_employee table.

1. API to edit the employee details

Method: PUT

Endpoint: /api/employees/<employeeId>