FINAL PROJECT PROPOSAL

GitHUb Link: https://github.com/singhal6183/nailart_studio

YouTube Link: https://youtu.be/OEG3_zxhFVE

1-person Web API Spring Boot Project:

Project Participant: Ankita Aggarwal

Title: Nail Art Studio

Executive Summary:

This application will store information for Nail Art Studios, Employees working with studios and Customers. It will be able to add/ delete/ and edit information for Studios, Employees, and Customers.

Project Requirements:

- **Database design which contains at least 3 entities and 3 tables**

- Nailart_studio
- Employee
- Customer

- **Contains all CRUD operations (Create, Read, Update & Delete)**

- **Each entity should have at least one CRUD operation**

- Nailart_studio - Create, Read, Update, Delete
- Employee - Create, Read
- Customer - Create, Read

- **One or more entities need to have all 4 CRUD operations (Create, Read, Update & Delete).**

- Nailart_studio - Create, Read, Update, Delete

- **Contains at least 1 one-to-many relationship**

nailart_studio and employees - Read

- **Contains at least 1 many-to-many relationship with one or more CRUD operations on this relationship**

nailart_studio_customer - Read

- **REST Web API UI Used to test all CRUD operations**

Advanced Rest Client (ARC)

Database Entity Relationship Diagram (ERD):

Entity Details

● Nailart studio: Has a unique Nailart studio's ID, studio name, address, city, state, zip and phone number.

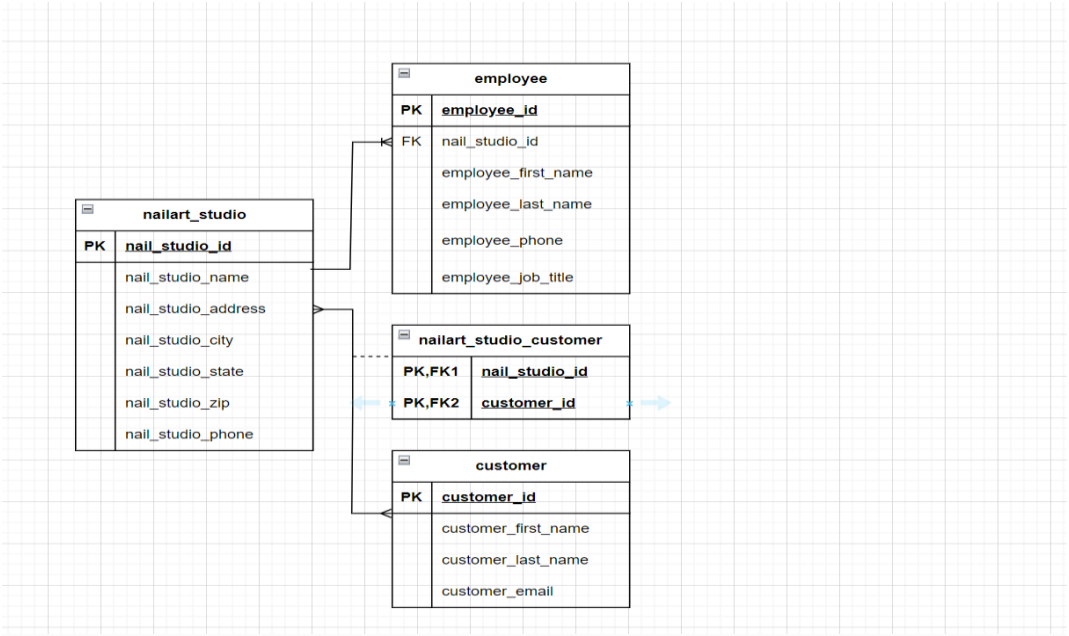
● Employee: Has a unique employee ID, first name, last name, phone number and job title.

● Customer: Has unique customer ID, first name, last name and email address.

Relationship Information

● **nailart_studio** and **employee** tables have a one to many relationship since a nailart studio can have many employees. The nailart studio is referenced in the employee table by using the **nail_studio_id** as a foreign key.

● A nailart studio can have many customers, and a customer can take services from several nailart studio, hence, the **nailart_studio** and **customer** tables have a many to many relationship and are referenced using the **nailart_studio_customer** join table.



Features:

- Create new NailArtStudio (POST on Nail Art Studio)
- Browse all NailArtStudios (GET on Studios) – without listing all customers and employees.
- Browse specific NailArtStudios (GET on Studio specified) – with listing all customers and employees.
- Add new Employee to a NailArtStudio (POST on Employee with NailArtStudio specified)
- Add new Customer to a NailArtStudio (POST on Customer with NailArtStudio specified)
- GET all Customers
- GET all Employees
- Browse all customers by NailArtStudio (GET on Customers with NailArtStudio specified)
- Browse Employee by NailArtStudio (GET on Employees with NailArtStudio specified)
- Browse a customer associated with specific customer id.
- Delete NailArtStudio (DELETE) – deletes all associated employees, without deleting all customers.
- Update NailArtStudio (PUT on a specific MassageStudio).