System Design Document

Cafe Employee Manager

1. User Information

Created / Update by	Creation / Updated date
Ahamed Faizal	Monday 26th August 2024

2. Overview

Application will manage cafes and employees associated with them, providing functionalities to add, edit, and delete both cafes and employees. It will include a frontend web application for user interaction and a backend API to manage the data.

3. Architecture

3.1 Frontend

Technology Stack: React, TypeScript, Redux (for state management).
Libraries: Tanstack Query, and Material-UI.
Components:

- **Cafe Shop List:** List all the cafe shops.
- **Employee List:** List all the employees.
- **Add/Edit cafe shop:** Ability to add/edit cafe shop.
- **Add/Edit Employee:** Ability to add/edit employees.
- **View Employee:** Ability to view specific employee details.
- **View Cafe Shop:** Ability to view specific cafe shop details.

3.2 Backend

Technology Stack: Node.js with Express.
Responsibilities:
Endpoints:

- **GET /api/cafes/getAllCafeShop:** Retrieve all cafe shop data.
- **GET /api/cafes/getCafeShopDetailsById:** Retrieve specific cafe shop data by id.
- **POST /api/cafes/submitShopCafe:** Create/Update a cafe shop details.
- **DELETE /api/employees/deleteShopCafeById:** Delete cafe shop by id.
- **GET /api/employees/getAllEmployee:** Retrieve all employees data.
- **GET /api/employees/getEmployeeDetailsById:** Retrieve specific employee data by id.

- **POST /api/employees/submitEmployee:** Create/Update employee details.
- **DELETE /api/employees/deleteEmployeeById:** Delete an employee by id.

3.3 Database

Technology Stack: MySQL

4. Components & Data Flow

4.1 CafeShopPage

- Users can view all the cafe shops which have been created.

4.2 ViewCafeShop

- Users can view specific cafe shop details.

4.3 AddCafeShop

- Users add and update specific cafe shops.

4.4 EmployeePage

- Users can view all the employees which have been created.

4.5 ViewEmployee

- Users can view specific employee details.

4.6 AddEmployee

- Users add and update employees.

5. Data Model

5.1 Employees

- **id:** String (UIXXXXXXX)
- **name:** String
- **email_address:** String (Email format)
- **phone_number:** String (Starts with 9 or 8, 8 digits)
- **gender:** String (M/F)
- **cafe_shop_id:** UUID (Foreign Key to Cafes table)
- **job_start_date:** Date
- **date_created:** DateTime
- **date_updated:** DateTime

5.2 Cafes

- **id:** UUID
- **name:** String
- **description:** String
- **location:** String
- **date_created:** DateTime

- **date_updated:** DateTime

5.3 Relationships

- **One-to-Many:** One cafe shop can have many employees.
- **Constraints:** An employee cannot work at more than one cafe shop.
- **Cascade Delete:** When a cafe shop is deleted then all the connected employee entries also will be deleted.

6. API Endpoints

6.1 GET /api/cafes/getAllCafeShop

- **Query Parameters:** NA
- **Response:** List of cafe shops.

6.2 GET /api/employees/getAllEmployee

- **Query Parameters:** NA
- **Response:** List of employees.

6.3 GET /api/cafes/getCafeShopDetailsById

- **Query Parameters:** id
- **Response:** Cafe shop data.

6.4 GET /api/employees/getEmployeeDetailsById

- **Query Parameters:** id
- **Response:** Employee data.

6.5 POST /api/cafes/submitShopCafe

- **Body:** Cafe shop data.
- **Response:** Status of creation / updation.

6.6 POST /api/employees/submitEmployee

- **Body:** Employee data.
- **Response:** Status of creation / updation.

6.7 DELETE /api/cafes/deleteShopCafeById

- **Query Parameters:** id.
- **Response:** Status of deletion.

6.8 DELETE /api/employees/deleteEmployeeById

- **Query Parameters:** id.
- **Response:** Status of deletion.

6.9 GET /api/cafes/getAllCafeShopOptions

- **Query Parameters:** NA
- **Response:** List of cafe shops (Only id and name)

7. Deployment

7.1 Hosting:

- **Frontend:** Use a Vercel, Github for deployment.
- **Backend:** Heroku (Node.js) for deployment.
- **Database:** Godaddy for mysql