SDE Intern Assignment – Erino
Contact Management System

Name: Pratham Asrani

Email: prathamasrani.cs@gmail.com

Phone: +91 9145495032

Demontration Video Link | Codebase Link
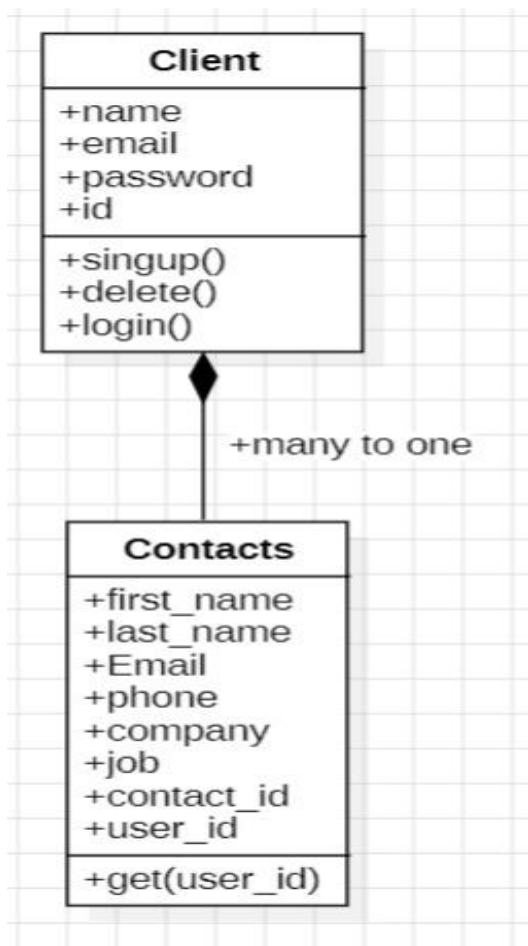
This module is designed in a 3 tier architecture!

Level 1: Database

Level 2: Backend

Level 3: Frontend

Class diagram:

CLASS DIAGRAM SUMMARY

Client: this entity is strong, and have attributes (name, email, password, and id) along with that we have defined some methods in backend for this entity

Method1 signup: creates new client

Method2 delete: remove the client from db

Method3 login: authenticate client present in db or not.

Contacts: this is a weak entity depends on client using composition (means if client A is deleted then all the contacts associated with client A is deleted). Attributes are shown in figure.

Method1 get(user_id): returns all the contacts associated with user_id

Method2 post(user_id, contact): creates new contact

Method3: put(user_id, contact, contact_id): edits contact

Method4: delete(contact_id): removes contacts

Level 1: Database

Technologies: PostgresSQL

Q. Why chosen PostgresSQL?

ans. The fields for storing data is fixed, so to store such data we require relational DB.

In market, currently PostgresSQL is the mostly used relational db along with that it implements the ACID properties for handling simulataneous transactions even in 'Erino' it is used that's why I chose this as a db!

Note: I request to please open class diagram for better understanding of below Level 1

we have two tables here

first table is client

CREATE TABLE IF NOT EXISTS client (

   id serial primary key,

   name varchar(255),

   email varchar(255) unique,

   password varchar(255)

);

second table is contacts, and it's dependent on client

create table if not exists contacts(

   contact_id serial primary key,

   user_id int not null,

   first_name varchar(55) not null,

   last_name varchar(55) not null,

   Email varchar(255) UNIQUE,

   phone varchar(15) not null,

   company varchar(255),

   job varchar(255),

   foreign key (user_id) references users(id) on delete cascade

);

Note: The contacts are weak entity in our system, if user deletes it's account all the contacts associated with it will also be deleted.

Level 2: Backend

Technologies: NodeJs, ExpressJs (specified)

Created Rest api by considering SOLID principles therefore new routes can easily be added or system code is flexible to expand! to see the required libararies I insist to check package.json

Flow:

Please refer Server.js

1. rest app is created using express, and cors is configured for all origins, methods, headers.

2. json objects parsing is allowed

3. the route are added for contact and users and controllers are defined with in the route files itself!

Level 3: Frontend

Technologies: ReactJs, localStorage, ContextAPI.

* ReactJs: I am proficient in it that's why I chose this!

* localStoarge: to store the authentocation token so that if user reloads the website the token will not lost! and user will remain signin

* contaxtApi: to keep user_id and pages field global so that data related to that user_id will be fetched from any component and pages will teel how many tuples will be extracted from the database.

Functionalities

User

1. User sign in

2. User sign up

3. User delete

Contant

1. add

2. delete

3. edit

4. update

Features

1. Readable-- Written In Clean Format

2. Understandable-- Comments Provided

3. Able To Share Resources With Another Websites (Cors Enabled)

4. Maintainable (Using Single Responsibility Principle)

5. Extendanable (Using Open For Extension And Close For Modification Principle)

6. Secure: Enable Security Against DDoS Attack, And More Can Be Enabled Using Helmet In Nodejs

Respected hiring team (**Akansha Srivastava**), I am available to join from 25th of November 2024, because my semester is ended. I am vey keen to work with your team and either contribute in your CRM project or in any other project where the need is required!
Thanking you for shortlisting my profile

Best Regards,
Pratham Asrani