Contact Management System

Create an application that will maintain the contact details of the user, the application must allow user to perform following operations:-

1. Create Profile
2. Login
3. View Profile
4. Edit Profile
5. Delete Profile
6. Add Contacts
7. View Contacts
8. Delete Contacts
9. Search Contacts by name
10. Logout

Maintain the Profile and Contacts in the database so that a user who logged in from a particular UserID and Password could able to manage his/her profile and contact details. Create the Back-End services so that any Front-End technologies could able to use them through API calls.

Database Tables

Two tables in the back-end

1. Profile: It will have columns to store UserID, Name, Password, DateOfBirth and PhoneNumber
2. Contacts: It will maintain multiple contact details for every profile which will have columns to store ContactID, ContactName, ContactNumber and UserIDReference

Note: UserID and ContactID will be auto-generated when profiles and contacts are stored respectively.

Business Requirement

1. Create Profile: When user wants to create a new account this operation will create a new profile.
2. Login: When user wants to login using the UserID and password this operation will authenticate user

All the below operations happen only for a user who has logged in hence the UserId is always sent to the back end

3. View Profile: User must able to see his/her profile
4. Edit Profile: User must able to update his/her password
5. Delete Profile: User must able to delete his/her profile
6. Add Contacts: User will add contact name and contact number
7. View Contacts: User will be able to see his/her contacts
8. Delete Contacts: User will be able to delete the particular contact
9. Search Contacts by Name: User will be able to search the contacts by entering name
10. Logout: User will be able to logout from the application, you don't have to back-end logic for this as the session will be handled at the Front-End side

As you will implement the business requirement independent of the Front-End technologies you will create RESTful Services and call these services from Front-End UI, You will call the business layer operations from the RESTful services.

You will create Proper layered architecture with Controller, Service and DAO

Business Requirement Operations

Below are the Interfaces that you will implement in the business layer

1. ProfileService
2. ContactService

Note: Adhere to the interface as it has all the operations user will perform.

1. ProfileService

```java
public interface ProfileService {
    //create profile for a user
    public Profile store(Profile profile);

    //get the profile based on the userId
    public Profile getProfile(int userId);

    //delete the profile based on userId
    public void deleteProfile(int userId);

    //update the password of a particular userId
    public Profile updatePassword(int userId, String password);

    //authenticate the user based on userId and password
    public Profile login(int userId, String password);
}
```

2. ContactService

```java
public interface ContactService {
    //add contact for the existing userId
    public Contact addContact(Contact contact);

    //get contacts matching by name for a particular userId
    public List<Contact> searchContactByName(int userId, String name);

    //delete the contact only for a particular userId
    public void deleteContact(int contactId, int userid);

    //get contacts of a particular userId
    public List<Contact> getMyContacts(int userId);
}
```

Note: Profile and Contact are entities which are mapped to the database tables, ensure you will have a sequence generation in both the entity and also correct association relationship of Profile and Contact

Implement these two interfaces in the Service Layer and Design the DAO's required for these operations which are called from the service layer, the RESTful API's will call the service layer operations whenever user wants to perform any operations, below are the ***RESTful URI's*** you create which you will test in Postman client as the Front-End technology can be any language like Angular, React, Java, C# and so on.

1) Get Contact API: */profile/{userId}/contacts*

2) Search Contact by Name API: */profile/{userId}/{name}*
3) Store Profile API: */createProfile*
4) Authenticate API: */login/{userId}/{password}*
5) Get Profile API: */profile/{userId}*
6) Update Password API: */updateProfile/{userId}/{password}*
7) Delete Profile API: */profile/{userId}*
8) Delete Contact API: */profile/{userId}/delete/{contactId}*
9) Add Contact API: */profile/{userId}/addContact*

Note: All the above API's must able to return the response in the JSON format, each operation would return success or failure responses.

Example

1. If user tries to login successfully the application responds his/her details or else error as Sorry UserId or Password is wrong
2. If user tries to see his/her contacts, the application responds with all the contacts belonging to him/her or error as Sorry no contacts