

# **UE20CS352 Project**

## **Object Oriented Analysis & Design with Java**

### **Mini LinkedIn**

**Hita Juneja PES1UG20CS645**

**Ishita Bharadwaj PES1UG20CS648**

**Meghana N PES1UG20CS663**

**Policharla Sai Sailaja PES1UG20CS671**

#### **Section K**

**25th April, 2023.**

#### **Description**

This is a Java Spring project based on designing a LinkedIn application using Object Oriented Approach and implemented it using JAVA that supports MVC Architectural Pattern. Project entails a full-stack web application where the user may register and log in to the service where they can create themselves a professional portfolio. User can invite other users to their contact network and praise their skills.

Project is built with Java Spring framework, Thymeleaf, HTML and Bootstrap CSS.

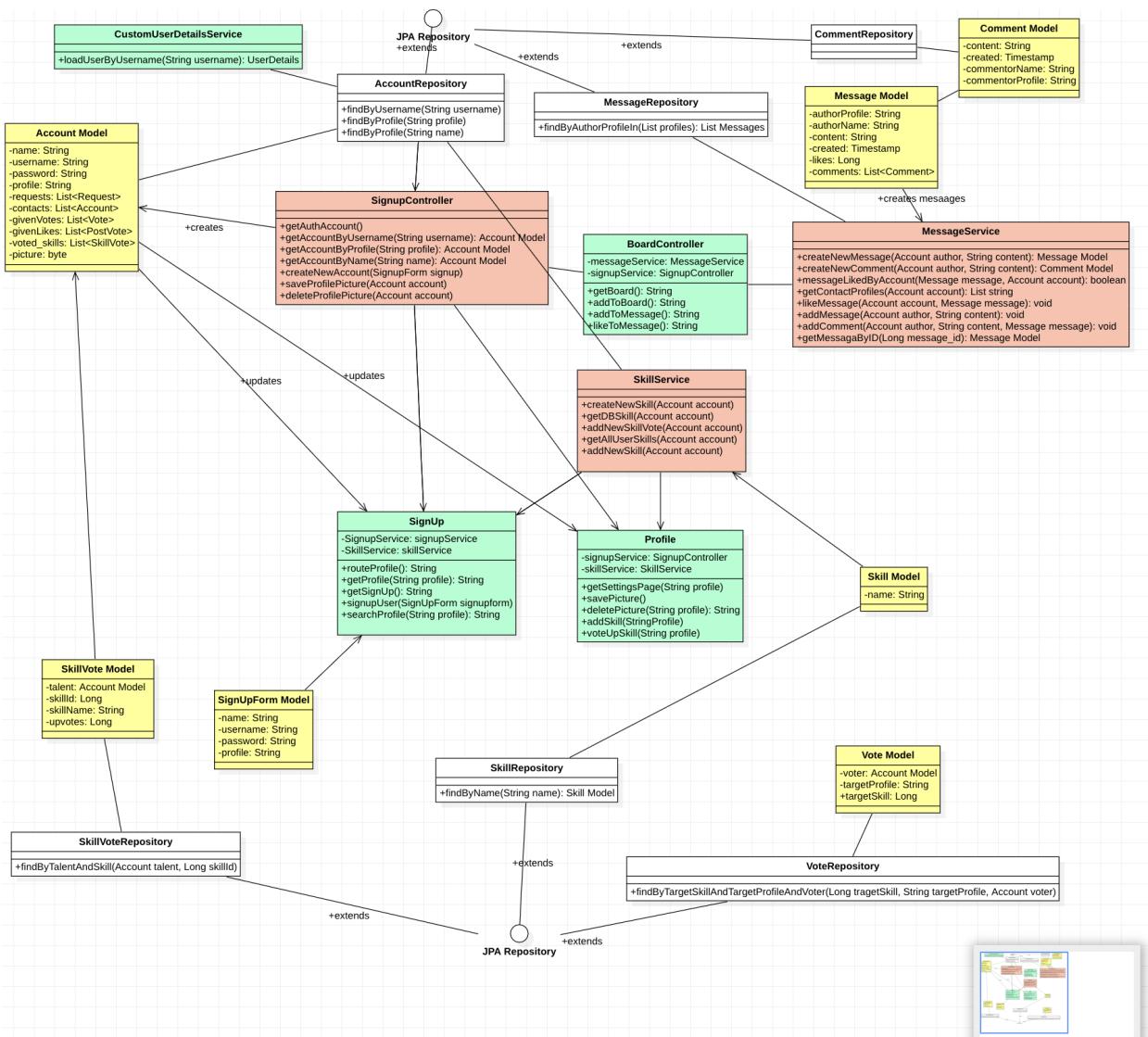
It uses H2 in-memory database with a SQL dialect to insert, fetch, update messages, posts, user accounts, comments, skills, skill votes and likes.

#### **Service functionalities**

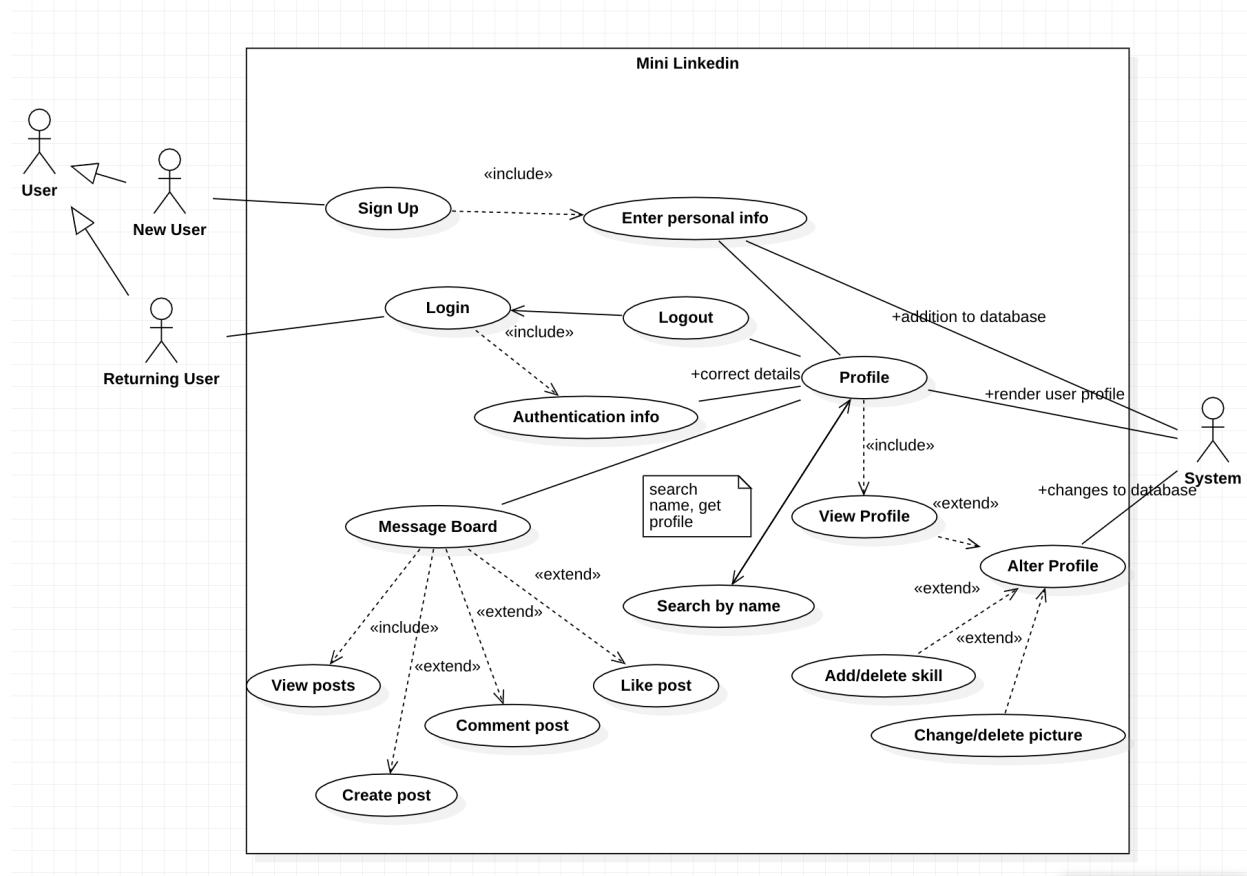
- Users may register to the service and log in and out to the service
- User has their own profile where they may add a profile picture and list skills for other users to see - like a small resume
- Logged in user can vote any one other user's skill. User's skill can be voted only once by the same user.
- Logged in user may ask another user to become their contact from this user's profile page. A request is sent and when this request is accepted these two users become contacts.
- Unlogged user may only view profiles and search them by users full name.

- Logged in user can post content in the message board. Content will be show to all contacts the user has.
- Contacts may comment and like each others' posts.

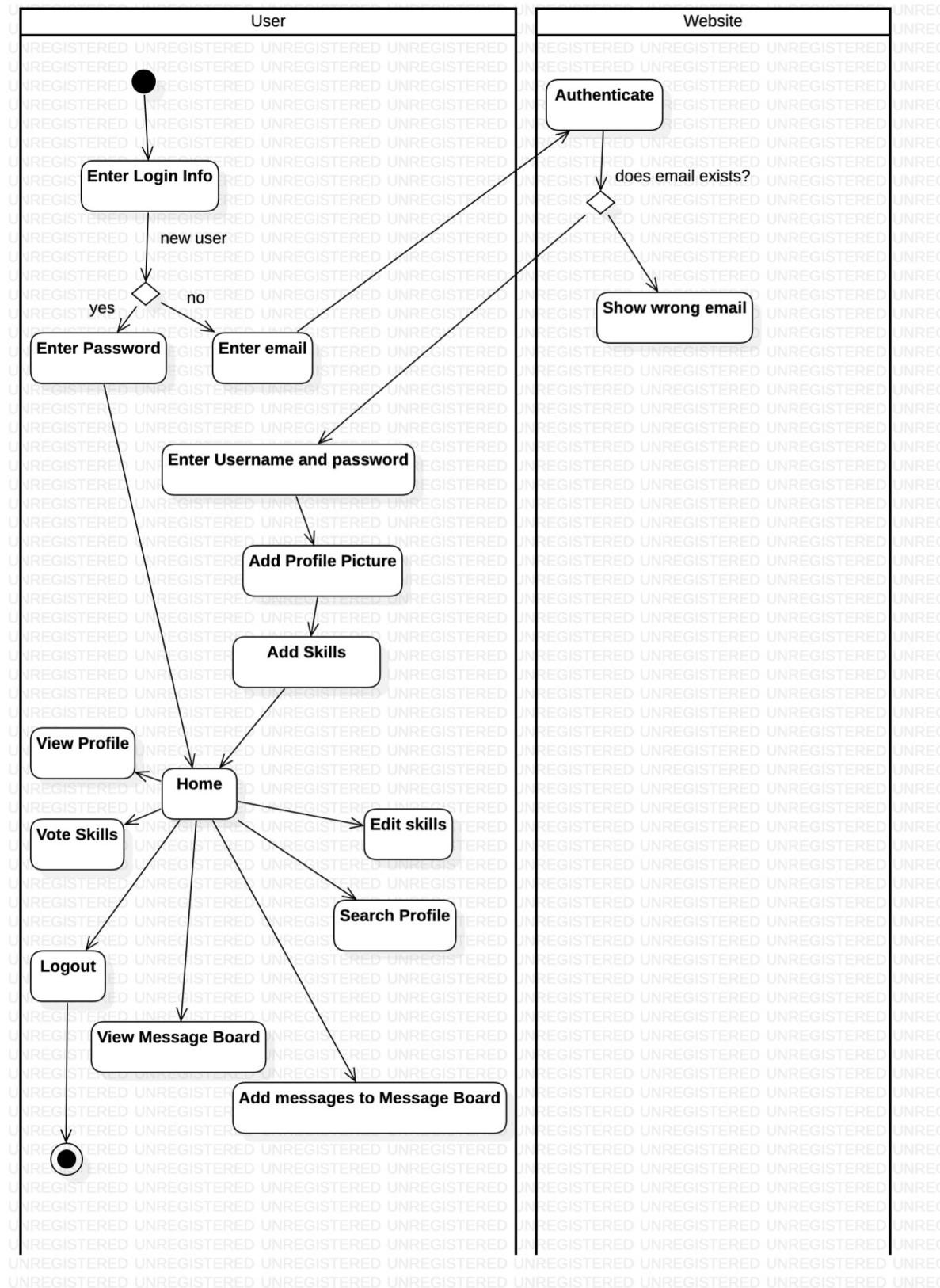
## Class Diagram



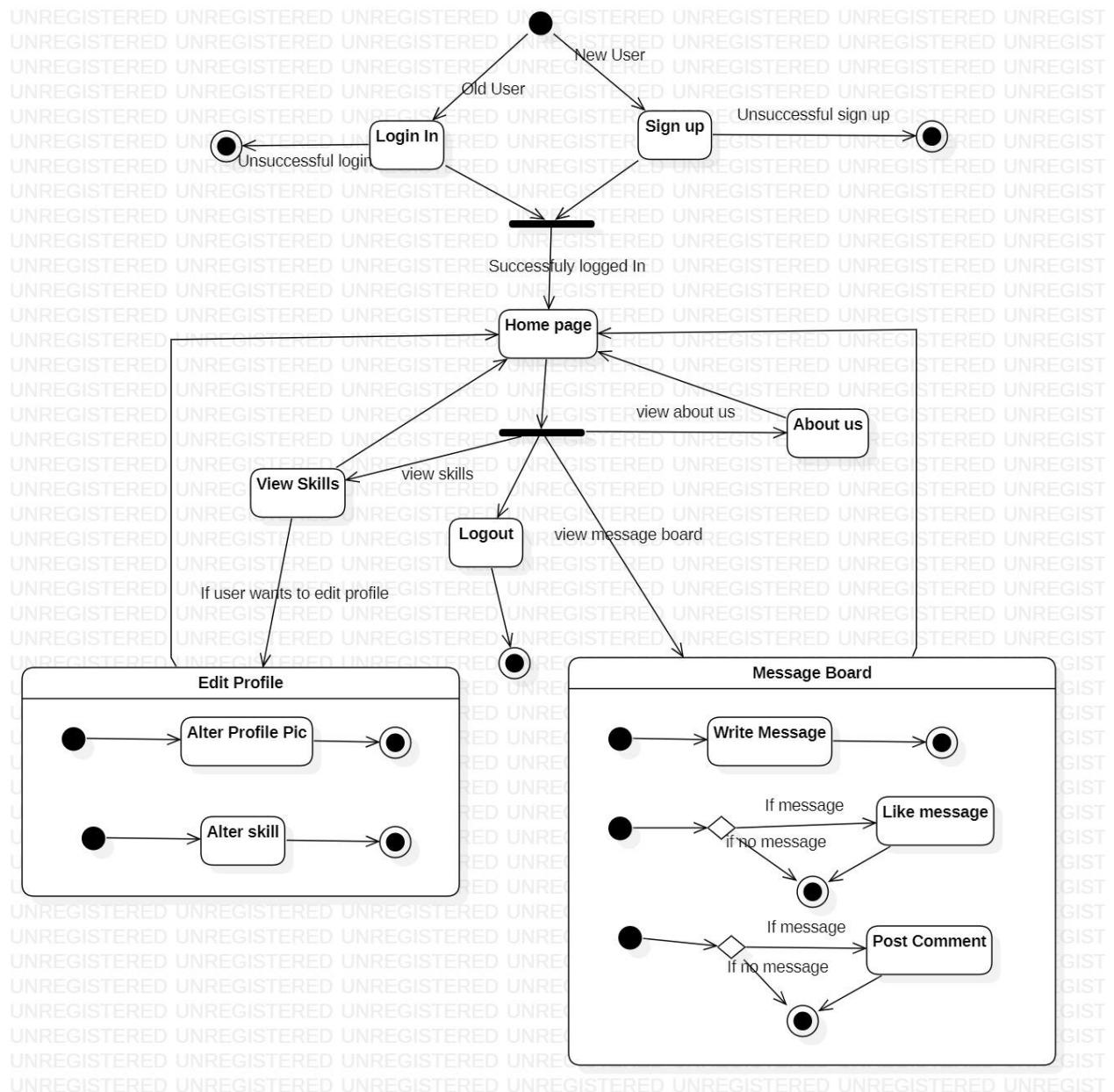
## Use Case



## Activity Diagram



## State Diagram



## Design Pattern

### Repository Pattern

In this project, the JPA repository class contains all persistence-related code but no business logic. It provides methods to persist, update, and remove an entity or methods that instantiate and execute specific queries. The goal of this pattern is to separate persistence-related code(Repository classes) from business code(Service and Controller classes) and to improve the reusability of persistence-related code. It also makes business code easier to read and write. It enables us to generate repositories easily for the various entities. Repository interface defines a set of methods for standard write operations, such as save, delete, and read operations. Skill Repository, SkillVote Repository, Account Repository, Message Repository and Comment Repository all extend the JPA repository.

## Design Principles

### Interface Segregation Principle

Interface segregation principle (ISP) states that no code should be forced to depend on methods it does not use. ISP splits interfaces that are very large into smaller and more specific ones so that clients will only have to know about the methods that are of interest. This is why we have separate interfaces that implement on the classes that are related to them. These include Repository, SkillVote Repository, Account Repository, Message Repository and Comment Repository.

### Single Responsibility Principle

The single-responsibility principle (SRP) states that "A module should be responsible to one, and only one, actor."

As shown in class diagram, since we have used MVC architecture pattern, there is only 1 model entity class, 1 controller class, 1 view/UI class for a particular use case. There aren't 2 or more classes that execute the same functionality. Every class is assigned a single responsibility.

### Indirection

The Indirection pattern is another way to reduce coupling by creating an intermediary class (or any kind of component) between two classes. Indirection introduces a layer between components in order to reduce coupling. This is shown by our service classes, MessageService, UserDetailsService, SkillService that act as an intermediary layer between their respective model and controller classes and reduce coupling.

### High Cohesion

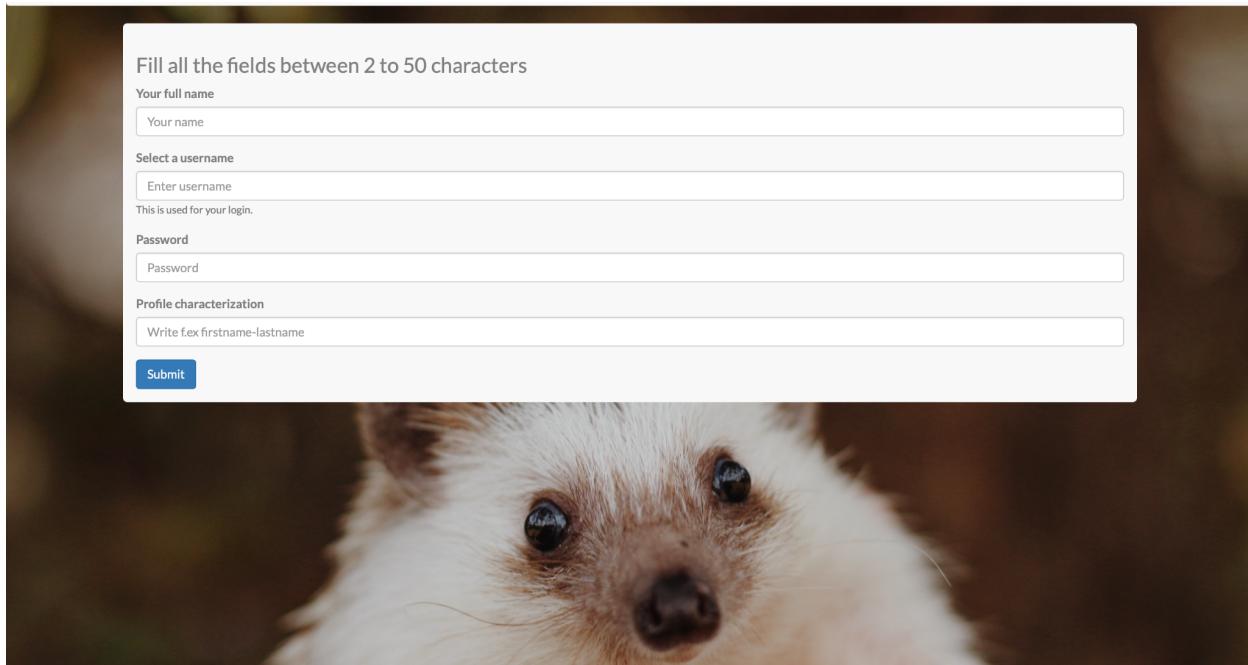
Related responsibilities are in to one manageable unit. High cohesion clearly defines the purpose of the element. It provides benefits of code reuse and low coupling. All operations in class represent a central purpose to implement all operations related to the message board.(read/retrieve message, post message, comment on message, delete message, like message).

### Code (Github link)

[https://github.com/IshitaBharadwaj/OOAD\\_LinkedIn](https://github.com/IshitaBharadwaj/OOAD_LinkedIn)

# OUTPUT

## Signup page



Fill all the fields between 2 to 50 characters

Your full name

Select a username

This is used for your login.

Password

Profile characterization

Write f.ex firstname.lastname

## Login Page

Please sign in

•

## Profile

abc



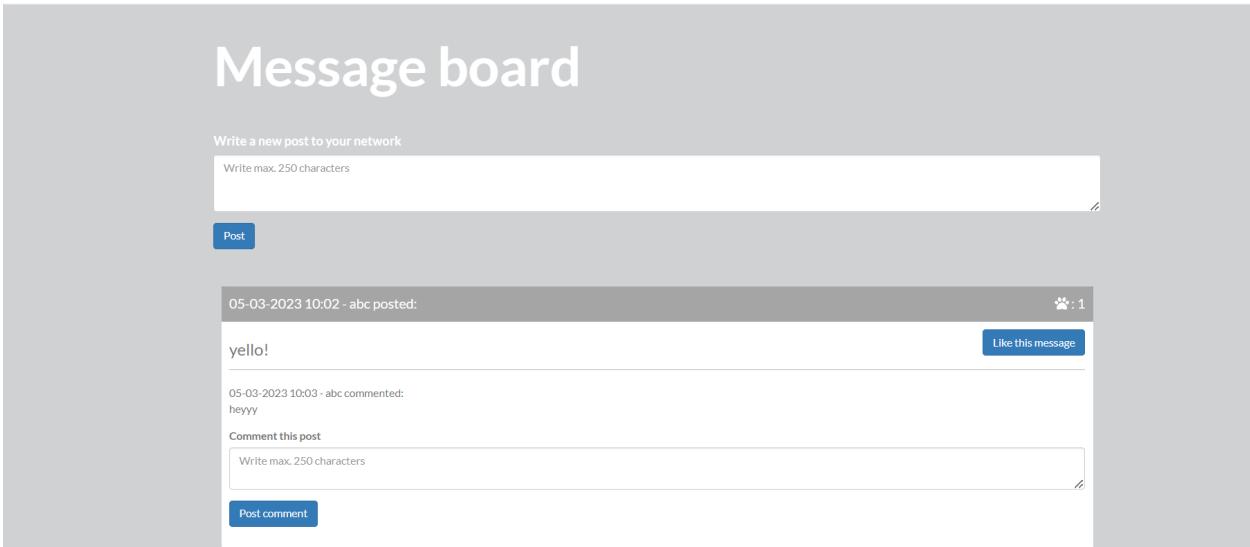
Alter your picture:

Choose File no file selected

Your skills:

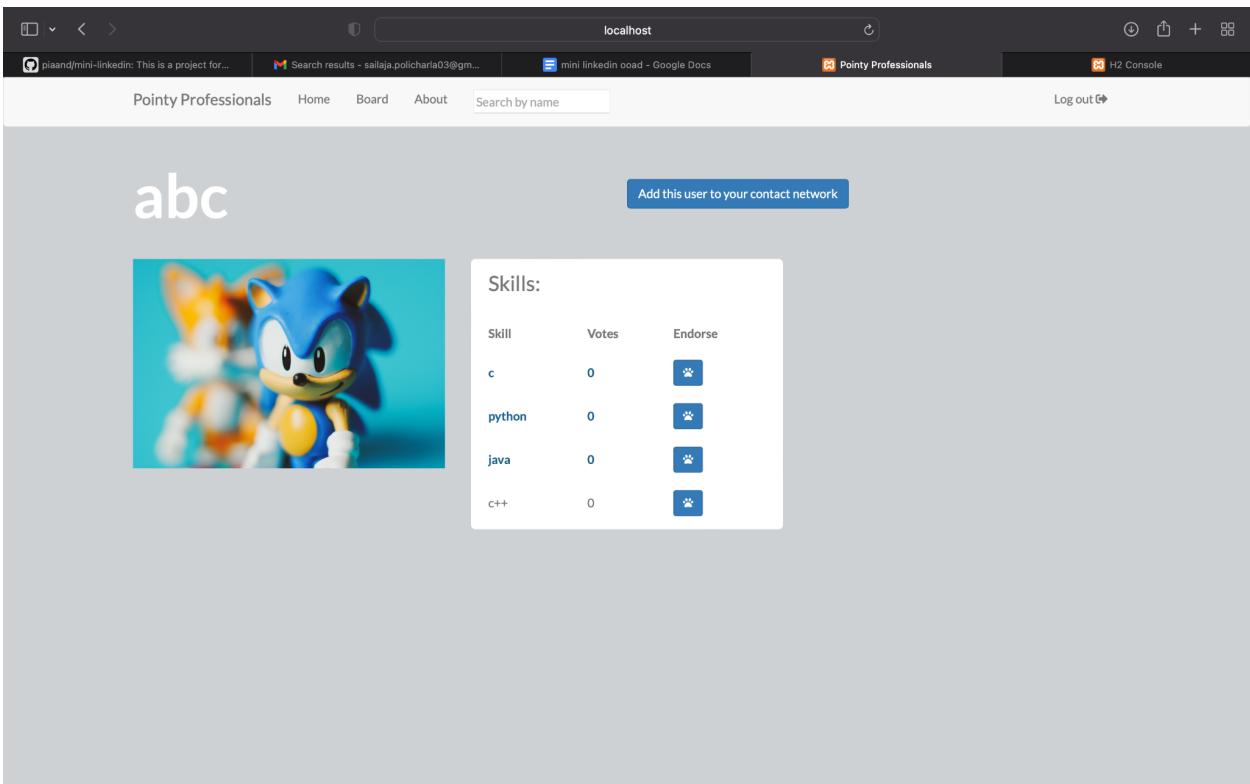
c	<input type="button" value="Delete this skill"/>
python	<input type="button" value="Delete this skill"/>
java	<input type="button" value="Delete this skill"/>
c++	<input type="button" value="Delete this skill"/>

## Message Board



The screenshot shows a message board interface. At the top, there's a text input field with placeholder "Write max. 250 characters" and a "Post" button. Below this, a post from user "abc" is displayed with the timestamp "05-03-2023 10:02". The post content is "yello!". There's a "Like this message" button with a count of "1". A comment from user "def" is shown below, with the timestamp "05-03-2023 10:03" and the content "heyyy". There's also a "Comment this post" section with a text input field and a "Post comment" button.

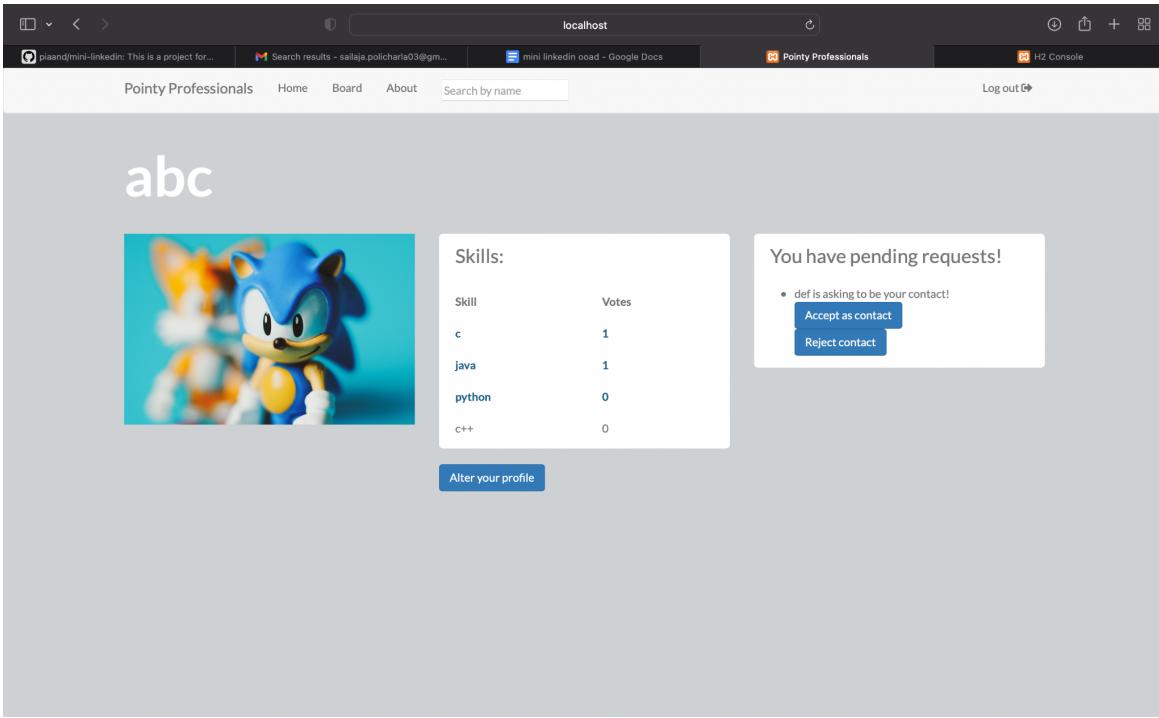
Signed in as user "def" and searched for user "abc"



The screenshot shows a user profile page for "abc". At the top, there's a navigation bar with links for "Home", "Board", "About", and a search bar. A "Log out" link is also present. The main content area features a large photo of Sonic the Hedgehog. To the right, there's a "Add this user to your contact network" button. Below the photo, a section titled "Skills:" lists the following data:

Skill	Votes	Endorse
c	0	
python	0	
java	0	
c++	0	

## Pending Requests showing on user profile



The screenshot shows a web browser window with multiple tabs open. The active tab is titled 'localhost' and displays a user profile for a user named 'abc'. The profile picture is a blue hedgehog. Below the picture is a table showing the user's skills and their respective votes:

Skill	Votes
c	1
java	1
python	0
c++	0

At the bottom of this section is a blue button labeled 'Alter your profile'.

To the right of the skills table is a box containing the message: 'You have pending requests!' followed by a list item: 'def is asking to be your contact!'. Below this list are two buttons: 'Accept as contact' and 'Reject contact'.

After accepting connection request by user "abc" from user "def", "def" is added to the contact list of user abc



Screenshot of a web browser showing a user profile page for 'abc'. The page includes a profile picture of Sonic the Hedgehog, a skills section listing 'c' (1 vote), 'java' (1 vote), 'python' (0 votes), and 'c++' (0 votes), and a contact list section for 'def' with a 'Delete contact' button. A 'Logout' link is visible in the top right.

Message posted by user "def" was commented by user "abc"

Screenshot of a web browser showing a post from 'def' and a comment from 'abc' on a social network platform.

**Post from def:**  
25-04-2023 14:20 - def posted:  
hello everyone

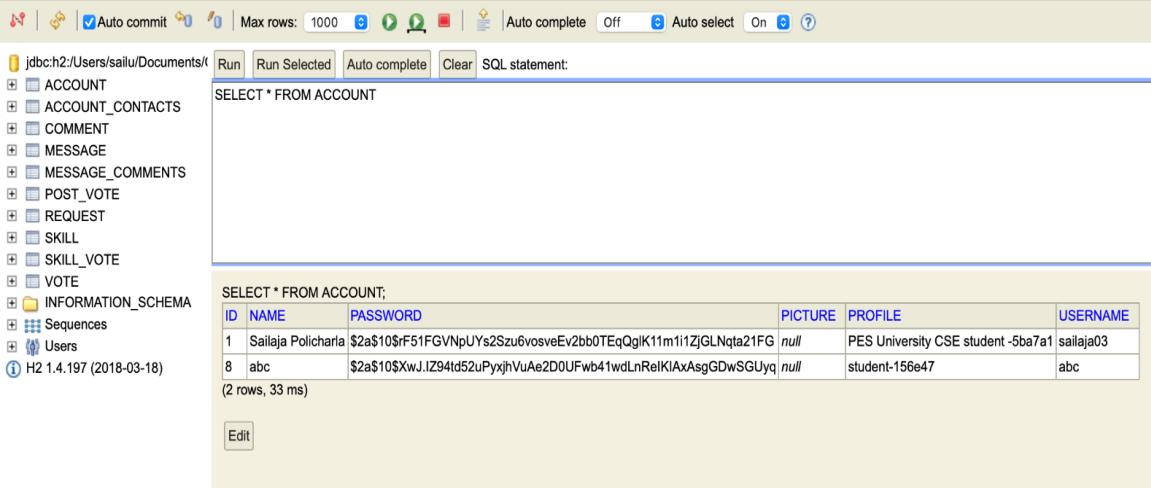
**Comment from abc:**  
25-04-2023 14:20 - abc commented:  
hi

**Post from abc:**  
25-04-2023 13:55 - abc posted:  
hello!

# DATABASE

## ACCOUNT

Account table provides information about the user account like name, their password, picture, profile and their username that they use to login.



The screenshot shows the MySQL Workbench interface with the following details:

- Toolbar:** Includes icons for connection status, auto commit (on), max rows (1000), and various database operations.
- Connection:** jdbc:h2:/Users/sailu/Documents/ (selected)
- Buttons:** Run, Run Selected, Auto complete, Clear, SQL statement.
- SQL Editor:** Contains the query: `SELECT * FROM ACCOUNT;`
- Table:** Results of the query:

ID	NAME	PASSWORD	PICTURE	PROFILE	USERNAME
1	Sailaja Policharla	\$2a\$10\$rF51FGVNpUYs2Szu6vosveEv2bb0TEqQgIK11m1i1ZjGLNqta21FG	null	PES University CSE student -5ba7a1	sailaja03
8	abc	\$2a\$10\$XwJ.IZ94td52uPyxjhVuAe2D0UFwb41wdLnRelKIAxAsgGDwSGUyq	null	student-156e47	abc

(2 rows, 33 ms)
- Buttons:** Edit

## ACCOUNT CONTACTS

This shows the connections that one account has by showing the account ID and the account ID it has as part of its connections

jdbc:h2:/Users/sailu/Documents/ (Auto commit) | Max rows: 1000 | Auto complete Off | Auto select On | SQL statement:

Run Run Selected Auto complete Clear

```
SELECT * FROM ACCOUNT_CONTACTS
```

ACCOUNT ACCOUNT\_CONTACTS COMMENT MESSAGE MESSAGE\_COMMENTS POST\_VOTE REQUEST SKILL SKILL\_VOTE VOTE INFORMATION\_SCHEMA Sequences Users H2 1.4.197 (2018-03-18)

SELECT \* FROM ACCOUNT\_CONTACTS;

ACCOUNT_ID	CONTACTS_ID
17	8
8	17

(2 rows, 14 ms)

## COMMENT

This table shows which account has commented what to a message posted by the owner

jdbc:h2:/Users/sailu/Documents/ (Auto commit) | Max rows: 1000 | Auto complete Off | Auto select On | SQL statement:

Run Run Selected Auto complete Clear

```
SELECT * FROM COMMENT
```

ACCOUNT ACCOUNT\_CONTACTS COMMENT MESSAGE MESSAGE\_COMMENTS POST\_VOTE REQUEST SKILL SKILL\_VOTE VOTE INFORMATION\_SCHEMA Sequences Users H2 1.4.197 (2018-03-18)

SELECT \* FROM COMMENT;

ID	COMMENTOR_NAME	COMMENTOR_PROFILE	CONTENT	CREATED
16	abc	student-156e47	hi	2023-04-25 14:01:49.236

(1 row, 26 ms)

Edit

## MESSAGE

This database shows the message that is posted on the respective user's message board



jdbc:h2:/Users/sailu/Documents/

Run Run Selected Auto complete Clear SQL statement:

SELECT \* FROM MESSAGE

(1 row, 10 ms)

EDIT

ACCOUNT ACCOUNT\_CONTACTS COMMENT MESSAGE MESSAGE\_COMMENTS POST\_VOTE REQUEST SKILL SKILL\_VOTE VOTE INFORMATION\_SCHEMA Sequences Users

H2 1.4.197 (2018-03-18)

SELECT \* FROM MESSAGE;

ID	AUTHOR_NAME	AUTHOR_PROFILE	CONTENT	CREATED	LIKES
14	abc	student-156e47	hello!	2023-04-25 13:55:54.94	0

(1 row, 10 ms)

EDIT

## MESSAGE COMMENTS

jdbc:h2:/Users/sailu/Documents/

Run Run Selected Auto complete Clear SQL statement:

SELECT \* FROM MESSAGE\_COMMENTS

(1 row, 14 ms)

MESSAGE\_ID COMMENTS\_ID

14	16
----	----

EDIT

ACCOUNT ACCOUNT\_CONTACTS COMMENT MESSAGE MESSAGE\_COMMENTS POST\_VOTE REQUEST SKILL SKILL\_VOTE VOTE INFORMATION\_SCHEMA Sequences Users

H2 1.4.197 (2018-03-18)

SELECT \* FROM MESSAGE\_COMMENTS;

## POST\_VOTE

jdbc:h2:/Users/sailu/Documents/

Run Run Selected Auto complete Clear SQL statement:

```
SELECT * FROM POST_VOTE
```

SELECT \* FROM POST\_VOTE;

ID	MESSAGE_ID	LIKER_ID
15	14	8

(1 row, 10 ms)

Edit

## REQUEST

This table shows the details of the account that has sent a connection request to another account and the status of the connection

jdbc:h2:/Users/sailu/Documents/

Run Run Selected Auto complete Clear SQL statement:

```
SELECT * FROM REQUEST
```

SELECT \* FROM REQUEST;

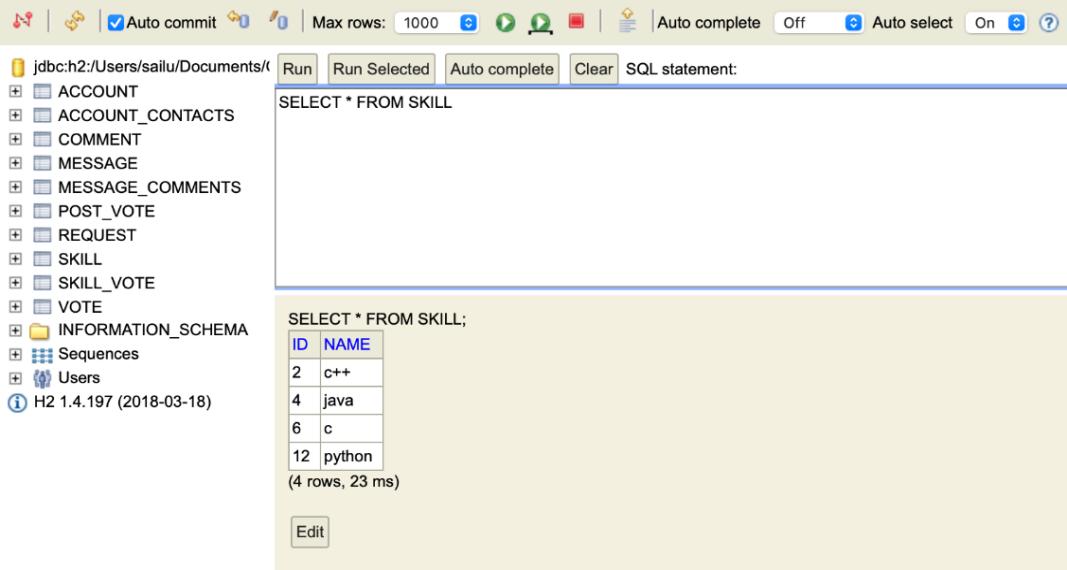
ID	MODIFIED	STATUS	TARGET	SUBMITTER_ID
18	2023-04-25 14:08:36.056	accepted	student-156e47	17

(1 row, 17 ms)

Edit

## SKILL

This table shows the particular skills that are added by the user to their profile



The screenshot shows a JDBC interface with the following details:

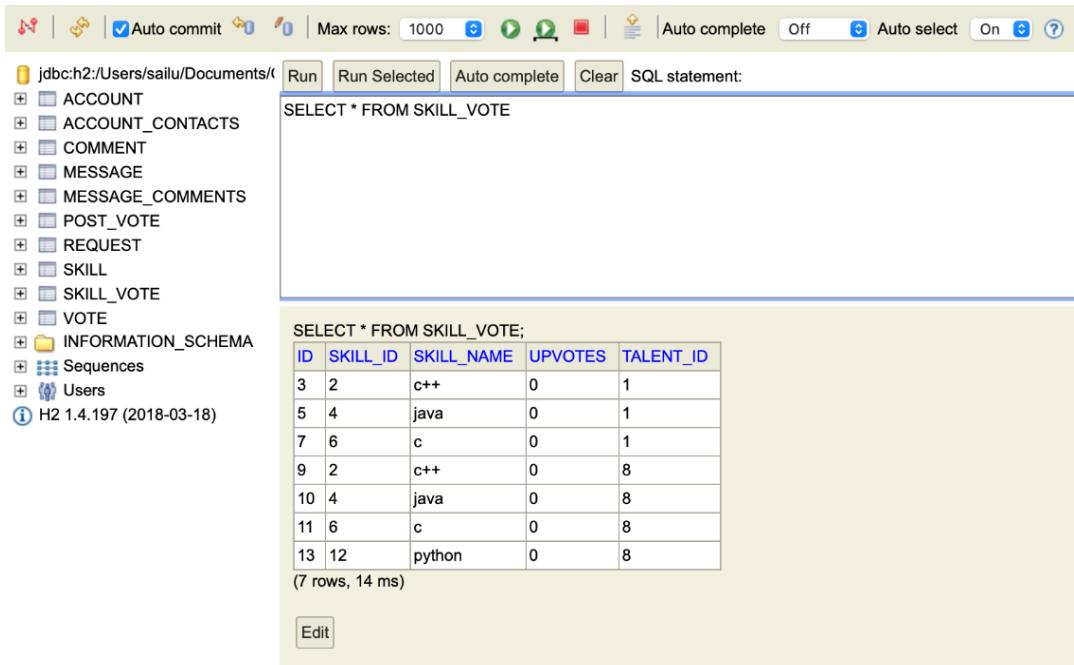
- Connection URL: jdbc:h2:/Users/sailu/Documents/
- Auto commit: On
- Max rows: 1000
- SQL statement: SELECT \* FROM SKILL
- Result set (in a light yellow box):
 

ID	NAME
2	c++
4	java
6	c
12	python

 (4 rows, 23 ms)
- Buttons: Run, Run Selected, Auto complete, Clear, Edit

## SKILL VOTE

This table shows the skills of a particular account and the number of accounts that have endorsed these skills



jdbc:h2:/Users/sailu/Documents/

Run Run Selected Auto complete Clear SQL statement:

```
SELECT * FROM SKILL_VOTE
```

SELECT \* FROM SKILL\_VOTE;

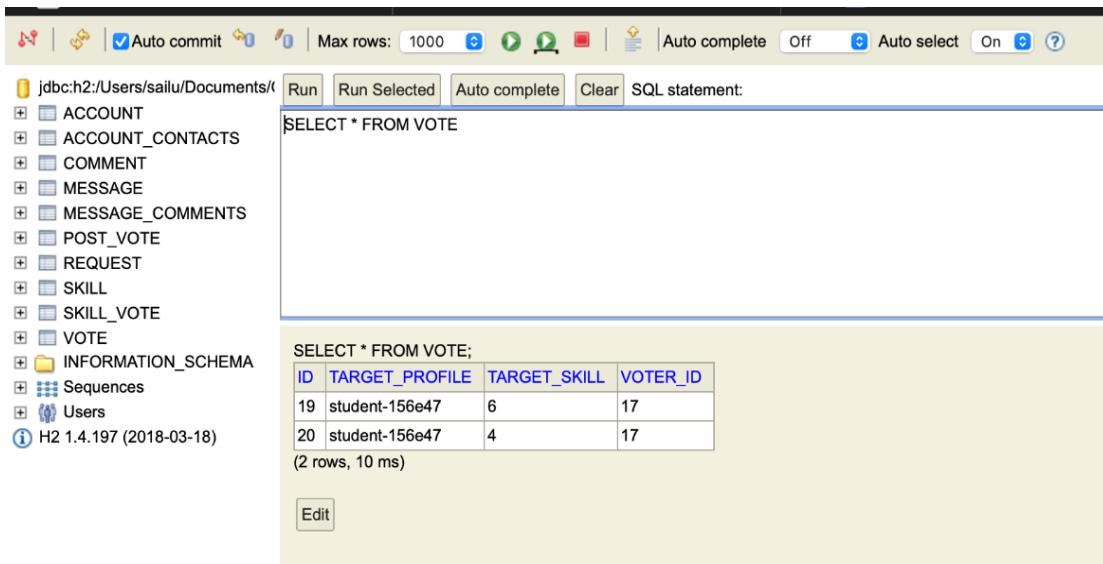
ID	SKILL_ID	SKILL_NAME	UPVOTES	TALENT_ID
3	2	c++	0	1
5	4	java	0	1
7	6	c	0	1
9	2	c++	0	8
10	4	java	0	8
11	6	c	0	8
13	12	python	0	8

(7 rows, 14 ms)

Edit

VOTE

This shows the target profile and the skills that were voted



jdbc:h2:/Users/sailu/Documents/

Run Run Selected Auto complete Clear SQL statement:

```
SELECT * FROM VOTE
```

SELECT \* FROM VOTE;

ID	TARGET_PROFILE	TARGET_SKILL	VOTER_ID
19	student-156e47	6	17
20	student-156e47	4	17

(2 rows, 10 ms)

Edit