Cody Steimle
SID: 862137374
Marco Baez
SID: 862040802

CS166 Databases Final Project

Project Overview

For the final project of CS166 we were tasked with creating an online messenger that contains a functioning friends list system with a friend level rule enforced using java and PostgreSQL. Upon completion of the project our messenger should:

- Allow users to create accounts and login.
- Allow users to change personal info or passwords.
- Allow users to search for another user.
- Make user profiles viewable.
- Allow messages to be sent to anyone on the network.
- Allow messages to be read and deleted.
- Store a friends list and make it accessible.
- Be able to send friend requests with the enforced 3 levels of friendship.
- Be able to view, accept, and reject requests.

Project Setup

Before running the project the files given needed to be cleaned up, so we:

1. Updated create_tables.sql by adding FOREIGN KEY's.
2. Imported Final_data to google drive and exported each page as a csv.
3. Updated load_data.sql by creating COPY sql commands.
4. Updated create_index.sql by created index sql commands.
5. Updated create_tables.sql by allowing UserIDs and Passwords to exceed 10 (10->30).
6. Updated ProfNetwork.java by changing class Messenger -> ProfNetwork and adding void functions at end of file.
7. Renamed create_index.sql to create_indexes (reason: create_db.sh).
8. Modified createdb.sh to work on lab machines.
9. For each csv replaced Âª with ª.
10. Removed contactlist

Commands used to setup and run the program:
Vars: $LABMACHINE (ex: wch132-42)

$UCRID (ex: cstei012)

First get the files on school server:

```
scp -r CS166_Project $LABMACHINE:/extra/$UCRID
```

Then SSH into the lab machines and run:

```
ssh wch132-42
cd /extra/$UCRID/CS166_Project
source ./startPostgreSQL.sh
export DB_NAME="$UCRID_DB"
source ./createPostgreDB.sh
cp ./data/*.csv $PGDATA
source ./sql/scripts/create_db.sh
mkdir $DIR/../classes
source ./java/scripts/compile.sh
```

Files and Data

CS166_Project
- >data
  - Connection.csv
  - Edu_Det.csv
  - Message.csv
  - USR.csv
  - Work_Ex.csv
- >java
  - >lib
    - pg73jdbc3.jar
  - >scripts
    - compile.sh
  - >src
    - ProfNetwork.java
- >sql
  - >scripts
    - create_db.sh
  - >src
    - create_indexes.sql
    - create_tables.sql
    - load_data.sql
- README.txt
- startPostgreSQL.sh
- createPostgreDB.sh
- stopPostgreDB.sh

File purposes:

- README.txt — README guide for running project
- startPostgreSQL.sh — Script for starting PostgreSQL
- createPostgreDB.sh — Script for create the PostgreDB
- stopPostgreDB.sh — Script for stopping the PostgreDB
- pg73jdbc3.jar — Required library file
- compile.sh — Compiles the main class ProfNetwork.java and runs
- ProfNetwork.java — Main class / messenger program
- create_db.sh — Creates the DB used for the messenger
- create_indexes.sql — Creates the indexes for the database
- create_tables.sql — Creates the tables for the project database
- load_data.sql — Loads in the csv data into the database

Design and Implementation

Most of the project just involved using templated java functions and menus and appending to the ProfNetwork.java file. However one design choice outside of this file was the use of a **trigger** on the messages table within the create_tables.sql file. We needed this trigger to be able to insert messages safely into the database without causing a duplicated primary key.

For all of the following functions we used for the project we passed in the variables:

```
function(ProfNetwork esql, String authorisedUser, ...)
```

We will exclude these parameters to save space below but they are essentially global variables that identify the logged in user and the main messenger class.

Functions:

- void CreateUser(...)
  - Inserts a new user into the database, needs userID, password, and email
- String LogIn(...)
  - Queries and checks log in information to move a user to their account main menu. Returns their userID
- void FriendList(...)
  - Queries and displays the logged in users friends list. Sets the current friend level to 0

- void SelectProfile(... , Integer currlvl)
    - Beginning the view process of a profile. Does a friend check and either resets the friend level to 0 if the target user is a friend or increments the friend level.
- boolean FriendCheck(... , String user1, String user2)
    - Takes 2 users and queries / check if they are friends (returns true or false)
- void ViewProfile(... , String viewUser, Integer currlvl)
    - Displays a user's information and friends with the option to send a message, send a request, or view their friend profiles.
- void ViewFriends(... , String thisUser)
    - Queries and prints friendslist for a user
- void UpdateProfile(...)
    - Update query for logged in user information.
- void SendMessage(... , String viewUser)
    - Send a message to the user whose profile their are viewing
- void ViewMessages(...)
    - Allows the logged in user to see their inbox (received messages), can also delete by message ID if desired.
- void ViewSentMessages(...)
    - Allows the logged in user to see their sent messages and can also delete by message ID if desired.
- void DeleteMessage(...)
    - Delete a received message by ID
- void DeleteSentMessage(...)
    - Delete a sent message by ID
- void SendRequest(... , String otherUser, Integer currlvl)
    - Does the level check and sends a request to the user if it passes.
- void ViewIncomingRequests(...)
    - Allows a user to see other users that requested them
- void ViewOutgoingRequests(...)
    - Allows a user to see their pending friend requests
- void ChangeRequest(...)
    - Allows a user to accept or reject a friend request
- void SearchPeople(...)
    - Performs a query to the database on user inputted name
- void ChangePassword(...)
    - Sends a query to the database to update the logged in user’s password

Problems and Challenges

One of the biggest problems we encountered with getting the code running and getting the data to be formatted properly. The steps involved in this process are stated in the Project Setup section, but this was easily the most difficult part of the project unfortunately.

The next problem we faced was dealing with the inserts on the MESSAGE table. Specifically we needed to avoid running into a duplicate primary key error, and also needed to update the index value everytime we inserted. Thankfully due to lab 8, this was easily solved through the use of triggers. We also implemented a trigger for the messageID within the create_tables.sql file. We verified the final index from the csv and made a sequence starting at that value.

For the last major problem, it was implementing the level check. It was a little tricky to comprehend and we ended up coming up with a verified solution. Let's discuss the solution to the level check problem:

1. There is a current friend level (currlvl) variable that is created in both the FriendList function and the SearchPeople Function.

2. These are independent function calls and each initialize currlvl to different values.

3. currlvl=0 for FriendsList, and currlvl=9 for SearchPeople (arbitrary).

4. Both FriendsList and SearchPeople call SelectProfile.

5. SelectProfile does a friend check and either resets the currlvl if friend (currlvl=0) or increments it otherwise (currlvl++)

6. SelectProfile calls ViewProfile then SendRequest or SelectProfile again (all while passing this currlvl value)

7. If SendRequest is called the level check occurs. A request will only work if a user has a currlvl < 3 OR the user has less than 5 friends (queries for this information)

Notable Features

- We also made it so, just like a real command line program, passwords are hidden when being typed.

- We used LIKE in the SearchPeople query to be able essentially use a form of regular expressions. We also perform the search on userID and name, so searching “Kim” will give us all rows where the name or userID contains the word “Kim” anywhere in it (case sensitive).

Example Queries

Search for people:

```
SELECT name, userId
FROM USR
WHERE name LIKE ‘%’ + name + ‘%’ OR userId LIKE ‘%’ + name + ‘%’
```

Change password:

```
UPDATE USR SET password = '%s'
WHERE userId = '%s'
```

Change friend request:

```
UPDATE CONNECTION_USR SET status = '%s'
WHERE userid = '%s' AND connectionid = '%s'
```

ViewIncomingRequests:

```
SELECT userid, status
FROM CONNECTION_USR
WHERE connectionId= '%s' AND status = 'Request'
```

ViewOutgoingRequests:

```
SELECT connectionId, status
FROM CONNECTION_USR
WHERE userId= '%s' AND status = 'Request'
```

FriendCheck:

```
SELECT *
FROM CONNECTION_USR
 WHERE C.status = 'Accept' AND ((C.userId = '%s' AND C.connectionId
='%s') OR (C.connectionId = '%s' AND C.userId ='%s'))
```

ViewProfile:

```
SELECT U.userId, U.email, U.name, U.dateofBirth
FROM USR U
WHERE U.userId = '%s'
```

ViewFriends:

```
SELECT U.userId, U.email, U.name, U.dateofBirth
FROM USR U, (

SELECT C1.connectionId
FROM CONNECTION_USR C1
WHERE C1.status = 'Accept' AND C1.userId = '%s'

UNION

SELECT C2.userId
FROM CONNECTION_USR C2
WHERE C2.status = 'Accept' AND C2.connectionId = '%s'

) AS FRIEND

WHERE U.userId = FRIEND.connectionId
```

UpdateProfile:

```
UPDATE USR SET name = '%s', dateofBirth= '%s'
WHERE userId = '%s'
```

Example Pictures of Working Project

```
*********************************************************
        Welcome To Cody and Marco's Messenger
*********************************************************

Connecting to database...Connection URL: jdbc:postgresql:/

Done
MAIN MENU
---------
1. Create user
2. Log in
9. < EXIT
Please make your choice: 2
        Enter user login: Jorge
        Enter user password:

MAIN MENU
---------
1. Goto Friend List
2. Update Profile
3. Messages
4. Requests
5. Search People
6. Change Password
.........................
9. Log out

Please make your choice:
```

```
MAIN MENU
---------
1. Goto Friend List
2. Update Profile
3. Messages
4. Requests
5. Search People
6. Change Password
.........................
9. Log out

Please make your choice: 5

Enter the name or userId to search for: Misty
name    userid
Elton Ferry                                     Misty.Von
Mrs. Jacinthe Bashirian                         Misty.Sawayn
Ms. Misty Nader                                 Urban.Dooley
Miss Misty Huel                                 Kaylin_Hackett
Laney Moen                                      Misty
Misty Mraz                                      Annamarie
Misty Denesik                                   Abelardo.Stoltenberg
Misty Fisher                                    Jesus.Luettgen
Misty Nicolas                                   Jacinto_Torphy
Ms. Misty Vandervort                            Marcella.Miller
Walter Weimann                                  Misty.Lynch
Lacey Ondricka                                  Misty.Bartoletti
Yasmin Tremblay                                 Misty_Murray
Madison Kozey                                   Misty.Klein
Dana Toy                                        Misty.Glover

1. View Profile
2. Search Again
.........................
9. Back

Please make your choice:
```

```
--------------Francesco's Profile--------------
userid  email   name    dateofbirth
Francesco       Jerel.Daniel@verda.biz  Viviane Rempel Sr.


--------------Francesco's Friends--------------
userid  email   name    dateofbirth
Jorge   Katrina.Gerlach@elfrieda.info   Kaylah Kreiger
Maeve_Legros    Jeanie.Beer@fanny.biz   Jaquan McDermott
Brennon_Kub     Marie@sharon.tv Harvey Kling
Jefferey.Feest  Jaclyn.Kohler@melba.io  Lou Cummings


1. Send Friend Request
2. Send Message
3. View a Friend of Francesco's profile
.........................
9. Back

Please make your choice: 2

Enter Message: Hello
Message Sent!
```

```
MAIN MENU
---------
1. Goto Friend List
2. Update Profile
3. Messages
4. Requests
5. Search People
6. Change Password
.........................
9. Log out

Please make your choice: 3

--------------Messages--------------
1. View Recieved Messages
2. View Sent Messages
.........................
9. Back

Please make your choice: 2

msgid   recieverid      contents        sendtime
55623   Francesco                       Hello



Would you like to delete an messages?
1. Delete Message.........................
9. Back

Please make your choice:
```