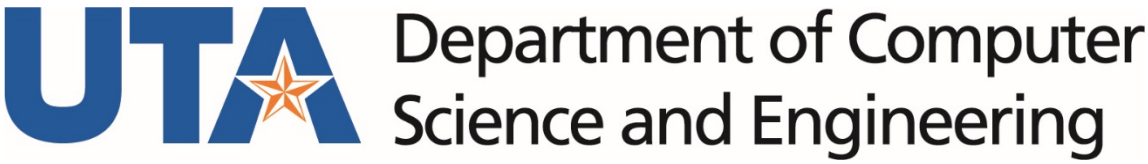


CSE 1320 Project Documentation

PROJECT NAME

Students names, surnames, IDs:
1. Asim Zafaruddin
2. Stephen Mills
3. Daniel Fielder
4.

Mentor: Marika Apostolova

-

SKILLSEARCH

Intermediate programming CSE 1320

Student declaration:

We declare that:

- *We understand what is meant by plagiarism*
- *The implication of plagiarism has been explained to me by our professor*
- *This assignment is all team own work and we have acknowledged any use of the published and unpublished works of other people.*

1 Student name, surname, ID and **signature: ……Asim Zafaruddin; 1002235712……** Stephen Mills 1002264517
2 Student name, surname, ID and **signature: ……………………......** Daniel Fielder 1002107178
3 Student name, surname, ID and **signature: ……………………......**
4 Student name, surname, ID and **signature: ……………………......**

Date:……...4/20/2025....................................

	Total number of pages including this cover page	##
Class Code / Group	CSE 1320	
Lecturer's Name	MARIKA APOSTOLOVA	

Table of Contents

1. **Project Introduction**

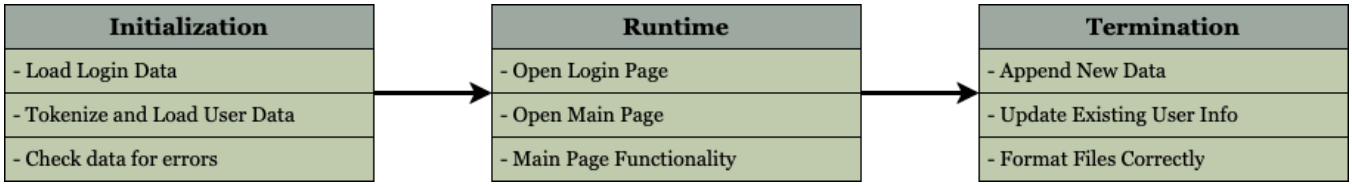
   With this app, we wanted to create a new and reliable way to help users connect and utilize their own skills. This skill-exchange platform allows users to find each other based on the skills they need and what they can provide. In the app, you can update your skills and requests, discover users with matching interests, and rate each other's performance. For example, if a web developer needed knowledge on graphic design, they could search for someone with graphic designing skills and a request for web development. If their skills and requests align, users can collaborate to help each other and rate the experience based on performance.

2. **Project Description**

   This project aims to provide a simple way for people to search for the skills they need and to share their own skills with their local community. It features a user registration system so that users can create and manage their accounts, skill listings, and skill requests. There is also a basic trust/reputation system where other user scores will be displayed.

3. **Project Architecture**

   The program execution itself is split into three parts: 1) Initialization – This is where user info is loaded into the program, 2) Runtime – Once all the user information is loaded in, the user can execute all of the functionality that the program provides (i.e. search for skills, manage requests, view other users), and 3) Termination – Upon exiting the program, all changes that were made during runtime are written into the user data files.



4. **Programming Concepts Used**

File handling- The program utilizes multiple external files to handle the large amount of data needed to operate the program. This includes a file holding the login information for the users, such as their usernames and passwords, along with a file holding user account information, including their skills, skill requests, and trust rating. Lastly, there are the library files to handle the layout of the user information,

Structs- There are multiple structs (in typedef format) used to keep the information for each user properly organized, such as the Fields, User, and ErrorID structs, used for the fields' layout in profile_info.txt, the storage/layout user data in profile_info.txt, and the repository for error messages, respectively.

Functions – A multitude functions integral to the program's operation, including void skillSearch, int newUser, and extern void tokenize, responsible for searching for users with specific skills, the storage of new user information upon registration, and the layout of fields in the file profile_info.txt, respectively.

5. Code Description

One key feature that helped with keeping the user's information intact was the User structs:

```
User users[MAX_USERS] ={
        {"hedge78!4", "Photography", "Photography", 5.80},
        {"quilt@08@", "FitnessWeb_Development", "Photography", 7.10},
        {"igloo5**1", "Graphic_Design", "Podcasting", 5.40}, etc.
```

This struct contains the information of all users in the database, containing their username, skill, requests, and trust level. Once you login, your information is also appended to this struct, so that other people can also find it when they search for skills or requests. With this, we are able to efficiently search for specific people based on characteristics in their profile, such as skills or requests.

Another important feature in the source code is the ability to search for a skill and have a list of users with that skill listed along with all their information:

```
        printf("Enter Skill to Search: \n");
                scanf("%s", skill_search);
                for (int i=0; i<user_count; i++){
                    if (strcmp(users[i].skill, skill_search)==0){
                        printf("\nUser: %s\n Skill: %s\n Request: %s\n Trust:
%.2f\n\n", users[i].name, users[i].skill, users[i].request, users[i].trust);
                    }
                }
                break;
```

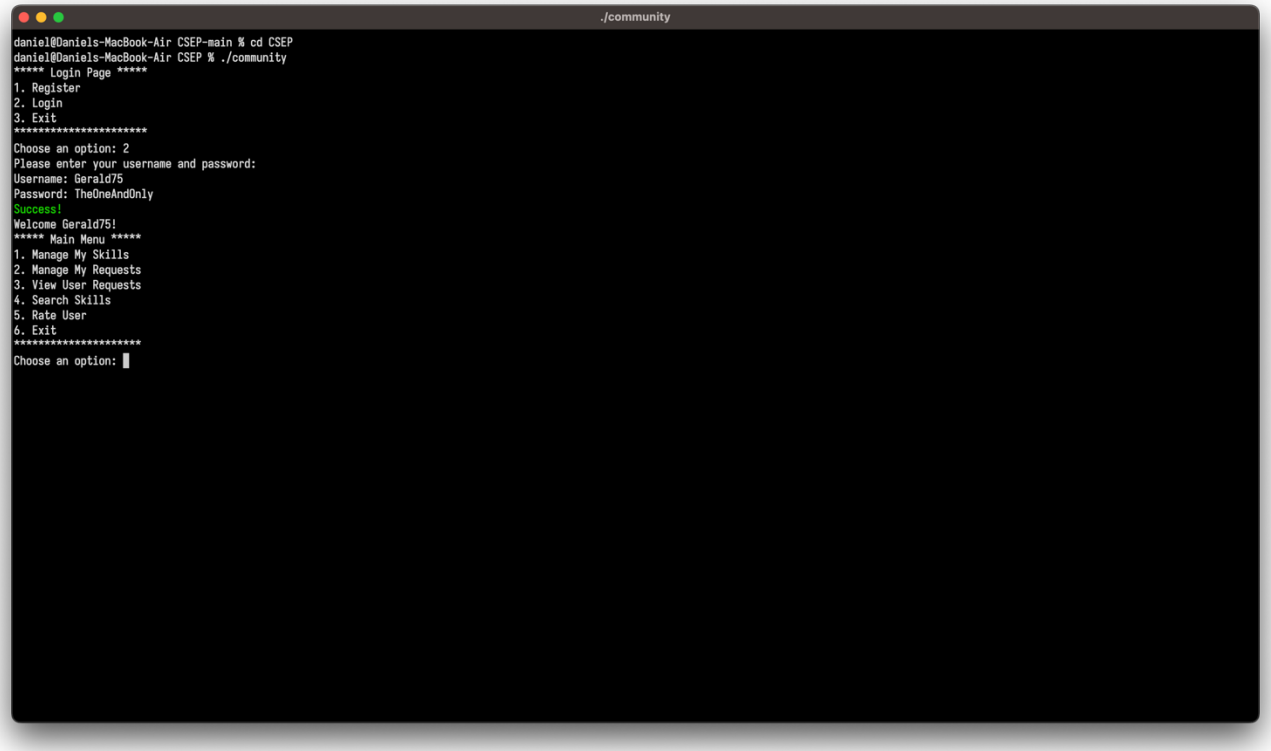
This code snippet uses user input to search for a specific skill and saves that as a variable. Then, it uses a for-loop to check all of the users and compare the strings to see if they are equal. If they are equal, it will print that user's information. This allows users to quickly find people that have a certain skill they want. They can also see what their request and trust level is.

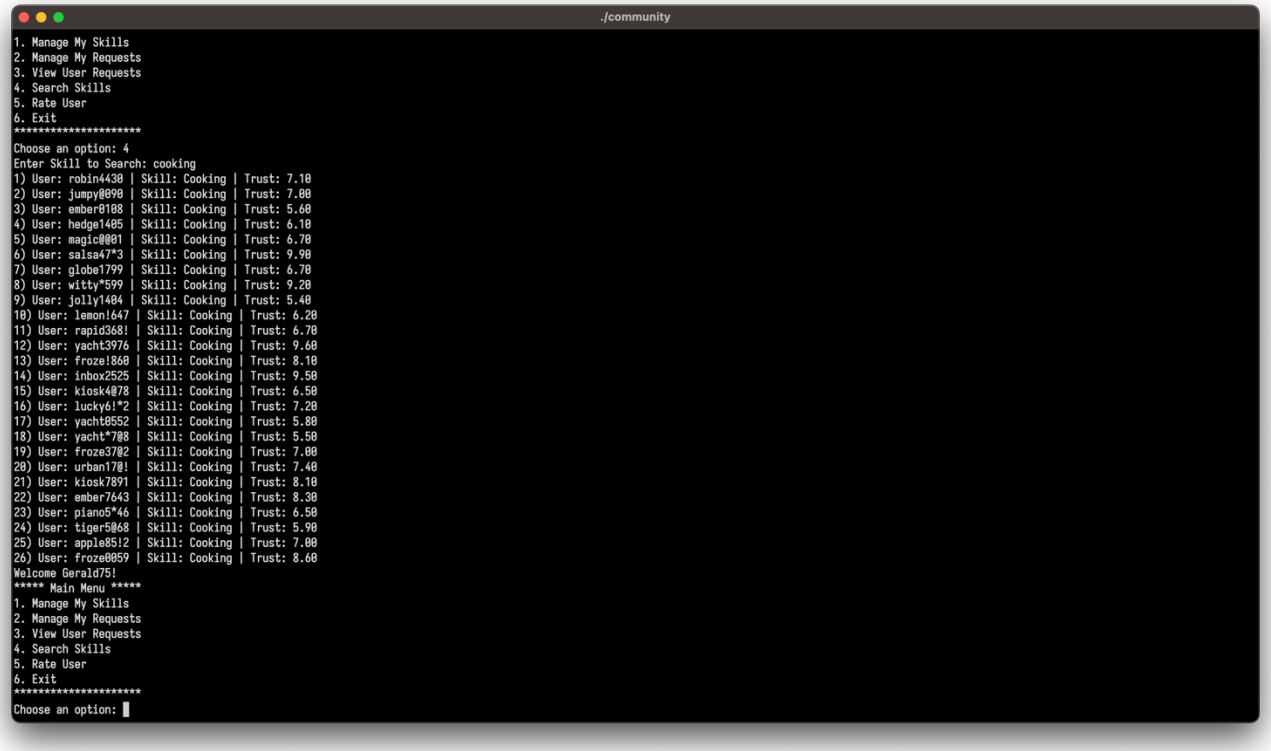
Lastly, a highlight feature in our app is the trust system. This system gives people the opportunity to review their experiences with other users, making it easier for future users to know if they are a good client to choose from. The implemented code is below:

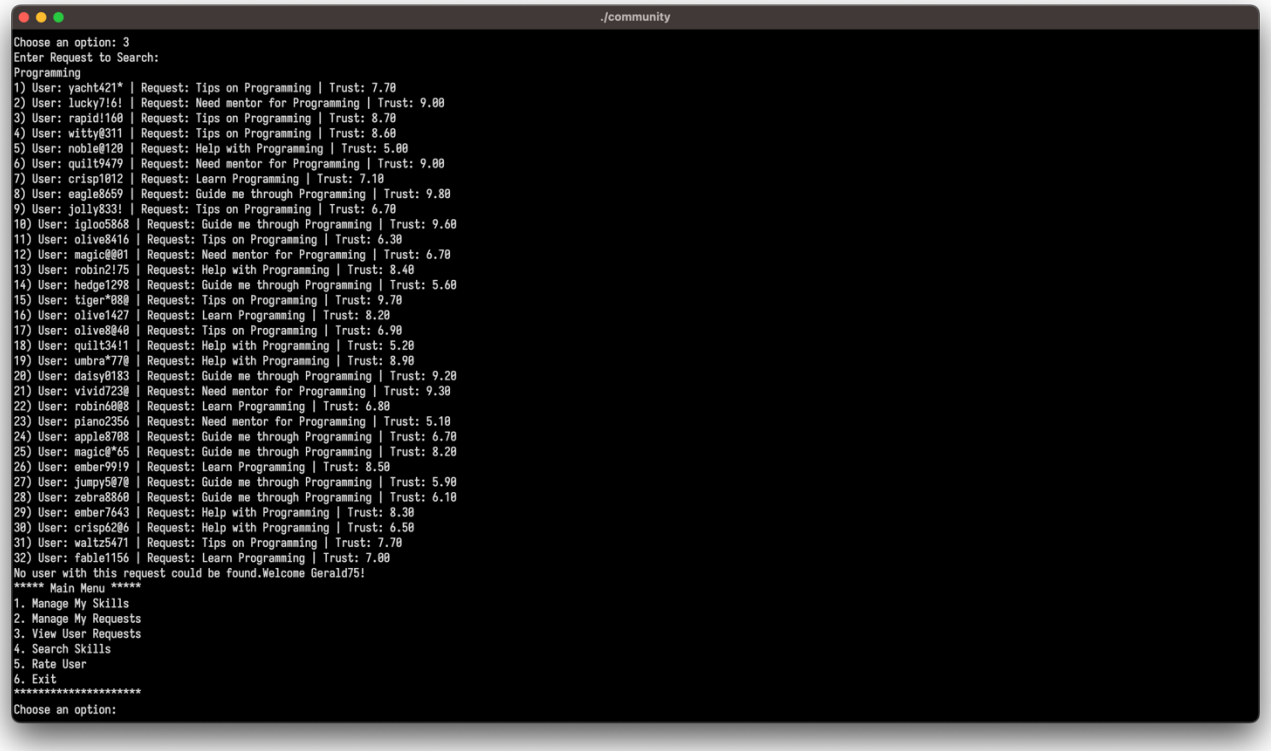
```
// Search users with that name
    int i = 0;
    for (; i < numOfUsers; i++) {
        if (strcmp(allUsers[i].name, userToFind) == 0) {
            temp = allUsers[i].trust;
            allUsers[i].trust = (allUsers[i].trust + rating) / 2;
            break;
        }
    }
    if (allUsers[i].trust < 0 || allUsers[i].trust > 10) {
        LOG(ERR_CORRUPT_DATA, "Invalid trust score was created. Resetting to
previous...");
        allUsers[i].trust = temp;
    }

    return;
}
```

This code snippet shows us the way trust is implemented into the program. First, the user inputs a name to search, and then a for-loop goes through all users and compares strings to see if both strings are equal. Once it finds the user, it will take the initial rating and add the user's new rating and divide it by 2, giving the average of the two. Also, if the user inputs a number that is not between 1 and 10, it will ask the user to input another number. With this, we are able to give the community a way to tell others how well certain people did in their opinion. This makes it easier for future users to tell whether or not they should connect with certain clients.

6. System Testing







7. Group members

Throughout the project, each group member played an important role in successfully completing the assignment. For the main source code, Daniel was focused on developing the main foundation of the program, such as the login information and trust system. For the main menu, Stephen contributed by developing the options to manage and view user requests. Asim also contributed by completing the ability to manage and search skills in the main menu. Communication was consistent, and everyone was allowed to fairly contribute to the project ‘s success.

8. Conclusion and future works

In conclusion, we were able to build a foundation for a skill-sharing platform that allows users to connect and exchange skills. Through the main menu, they can search for skills, manage requests, rate others on their performance, and more. This project allowed us to experiment with different concepts in C, make a cohesive project with multiple people’s code, and communicate with others to achieve success. In the future, we can improve its look and availability, since it is only the source code right now. Also, we can add on to the code to make it more usable. For example, the trust system could be tweaked to where it only lets you rate someone if they have completed a request from you. Also, the “Manage Skills” option could be expanded to where they can add multiple skills instead of just one.