Collaborative To-Do list

Session 2023 - 2027

Submitted by:

Muhammad Omer 2023-CS-68

Supervised by:

Muhammad Awais

Course:

CSC-102 Programming Fundamentals

Department of Computer Science

University of Engineering and Technology

Lahore Pakistan

Here you can find the major parts of your Proposal documentation

- ## Short Description of your project

  I wanted to create a collaborative application that boosts the productivity of the people using it.So I created this to do list which is both minimal and collaborative.In it users can not only add tasks to their to do list but also to their peers to do list and request help on tasks they cannot finish alone.

- ## Users of Application

  I wanted this applications to be used by peers so there is no fixed hierarchy. All of the users are on the same level of authority and do not have the capability to change or see any of the other users data. All users have their own files in which data is their data is stored.

- ## Functional Requirements

  The user can do the following functionalities with this application.

  1. Add tasks to himself
  2. View his To-Do list
  3. Remove his tasks
  4. Mark his completed tasks done
  5. View other the list of other peers using this app
  6. Request other peers to help on task
  7. See and mark completed other peoples requested tasks

- ## Wireframes

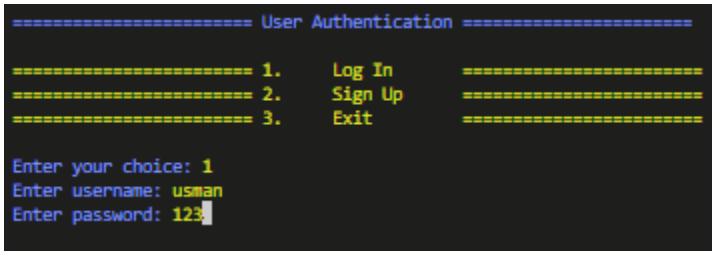

Figure 1: Login Screen

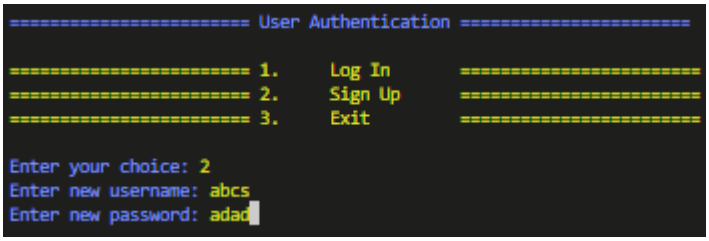

Figure 2: sign up menu

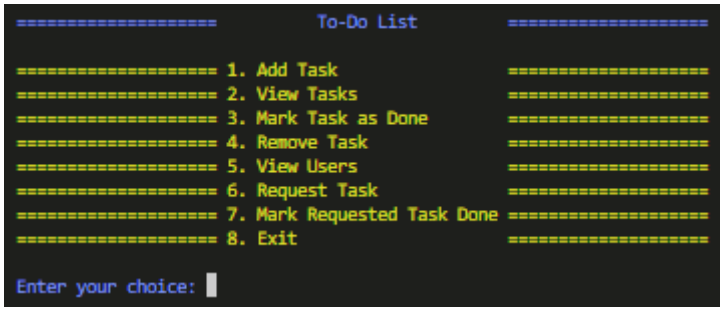

Figure 3: User Main Menu Screen

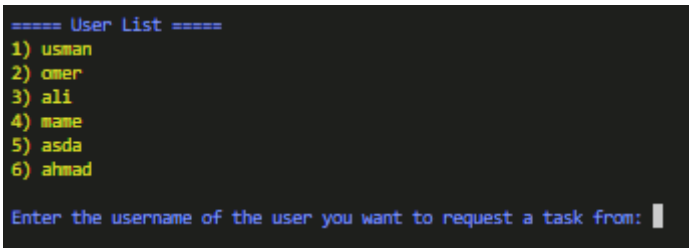

Figure 4: Requesting task menu

- ## Data Structures (Parallel Arrays)

```
string usernames[MAX_USERS];
string passwords[MAX_USERS];
string requestList[MAX_REQUESTS];
int userCount = 0;
int requestCount = 0;
```

- ## Function Prototypes

Void addTask(const string& username, const string& task, bool taskDone[], string taskList[], int& taskCount);

void viewTasks(const string& username, const bool taskDone[], const string taskList[], int taskCount);

void removeTask(const string& username, string taskList[], bool taskDone[], int& taskCount);

void markTaskDone(const string& username, string taskList[], bool taskDone[], int taskCount);

void loadTasksFromFile(const string& username, string taskList[], bool taskDone[], int& taskCount);

void saveTasksToFile(const string& username, const string taskList[], const bool taskDone[], int taskCount);

void requestTask(const string& requester, const string usernames[], string requestList[], int& requestCount);

void viewRequestedTasks(const string& username, const string requestList[], int requestCount);

void markRequestedTaskDone(const string& currentUser, string requestList[], int& requestCount);

void loadRequestsFromFile(const string& username, string requestList[], int& requestCount);

void saveRequestsToFile(const string& username, const string requestList[], int requestCount);

void addUser(string usernames[], string passwords[], int& userCount, const string& username, const string& password);

bool authenticateUser(const string usernames[], const string passwords[], int userCount, string& currentUser);

void saveUsersToFile(const string usernames[], const string passwords[], int userCount);

void loadUsersFromFile(string usernames[], string passwords[], int& userCount);

void viewUsers(const string usernames[], int userCount);

int ascii_to_int(string convert_to_int);

void resizeConsole(int width, int height);

- **Functions Working Flow**