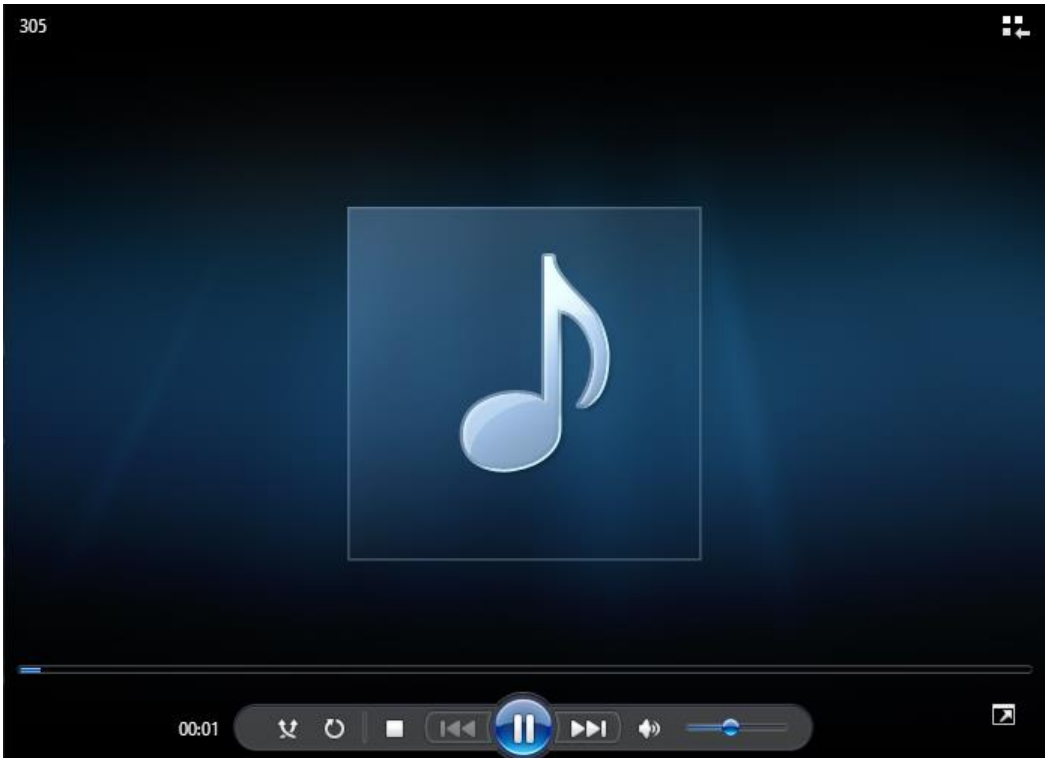
Media Player Application



Description

The project is about a Media Player that requires the user to have an account and login into this account to be able to access the features in the program. The application can store multiple user accounts. After Logging in, user can add a folder of songs, play a song in this folder, add a song, delete a song, search for a song, and skip songs in the playlist.

Functions Used in the Program

- **Registration function**

```
void registration(string* user, string* pass)
{
    static int count = 0;
    profile pro;
    cout << "Enter the username\n";
    cin >> pro.username;
    cout << "Enter the password\n";
    cin >> pro.password;
    user[count] = pro.username;
    pass[count] = pro.password;
    count++;
}
```

It is used to register from the user new data so he can login into the app

- **Login function**

```
int login(string* user, string* pass)//function for logging in
{
    string check1, check2;
    int ind;
    int p = 0, c = 0;
    cout << "Enter the username\n";
    cin >> check1;

    cout << "Enter the password\n";
    cin >> check2;
    for (int i = 0; i < 100; i++)
    {
        if (check1 == user[i])
        {
            p = 1;
            ind = i;
        }
    }
    for (int j = 0; j < 100; j++)
    {
        if (check2 == pass[j])
            c = 1;
    }
    if (p != 0 && c != 0)
    {
        cout << "Login successful, welcome " << check1 << endl;
        return ind;
    }
    else
    {
        cout << "Invalid username or password\n";
        return -1;
    }

}
```

This function used to check if the user have an account or no, by comparing to the registered data, in order to continue to use the app.

- **Folder's menu function**

```
void fold(song** folder_arabic, string folder[][10], int index)//the folder choice
{
    int fold_choice;
    static int vol = 50;
    cout << "-------------------------------------\n";
    cout << "Choose what do you want to do: \n";
    cout << "1. Add folder\n";
    cout << "2. View current folders\n";
    cout << "3. Return to main menu\n";
    cout << "-------------------------------------\n";
    cin >> fold_choice;
    switch (fold_choice)
    {
    case 1:
    {
        int count = 0;
        for (int i = 0; i < 10; i++)
            if (folder[index][i] != "")
                count++;
        cout << "Enter the name of your folder\n";
        cin >> folder[index][count];
        fold(folder_arabic, folder, index);
        break;
    }
```

```
case 2:
{
    int folder_choice;
    int count = 0;
    for (int i = 0; i < 10; i++)
        if (folder[index][i] != "")
            count++;

    for (int i = 0; i < count; i++)
    {
        cout << i + 1 << ". " << folder[index][i] << endl;
    }
    if (count == 0)
        cout << "No folders available\n";
    else
        cout << "Now select the folder you want\n";
    cout << "If you want to return to menu, press 0\n";
    cin >> folder_choice;
    if (folder_choice != 0)
    {
        for (int i = 0; i < count; i++)
        {
            if (folder_choice == i + 1 && (folder_choice <= count))
            {
                cout << "Here's your folder: " << folder[index][i] << endl;
                song_options(folder_arabic, folder, index, folder_choice, vol);
            }


        }
        fold(folder_arabic, folder, index);

    }

    else if (folder_choice == 0)
    {
        fold(folder_arabic, folder, index);
    }

    break;
}
case 3:
{
    break;
}
default:
{
    cout << "Invalid choice\n";
    fold(folder_arabic, folder, index);
    break;
}
break;

}
}
```

It is a function to display the menu options for the folder and the user can choose from the options, and accordingly, one of the cases will happen.

- **Song’s menu function**

```
void song_options(song** folder_arabic, string folder[][10], int index, int folder_choice, int vol)
{
    int process, songnum;
    string songname;
    cout << "----------------------------------------\n" << "1.display the songs and choose one to play\n2.search for a song to play\n3.add a song\n4.delete a song\n";
    cout << "select the number of process you want \n" << "----------------------------------------\n";
    cin >> process;
    switch (process)
    {
    case 1:
    {
        display(folder_arabic, folder_choice);
        cout << "------------------------------------------------------------\n";
        cout << "enter the number of the song that you want to play it\n";
        cout << "------------------------------------------------------------\n";
        cin >> songnum;
        play_sound(folder_arabic, songnum, folder_choice, vol);
        break;
    }
    case 2:
    {
        cout << "------------------------------------------------------------\n";
        cout << "enter the of the name of the song that you want to search it\n";
        cout << "------------------------------------------------------------\n";
        cin >> songname;
        int num = search(folder_arabic, songname, folder_choice);
        if (num == -1)
        {
            cout << "The song you searched for was not found\n";
        }
        else
        {
            cout << "the number of the song that you searched is " << num << endl;
            play_sound(folder_arabic, num, folder_choice, vol);
        }

        break;
    }
}
case 3:
{
    display(folder_arabic, folder_choice);
    adding(folder_arabic, folder_choice);
    display(folder_arabic, folder_choice);
    break;
}
case 4:
{
    display(folder_arabic, folder_choice);
    deleting(folder_arabic, folder_choice);
    display(folder_arabic, folder_choice);
    break;

}
default:
{
    cout << "Invalid choice\n";
    break;
}
break;

}
int choice;
cout << "----------------------------------------\n";
cout << "Now choose what do you want: \n";
cout << "1. Return to the song's menu\n";
cout << "2. Return to the folder's menu\n";
cout << "----------------------------------------\n";
cin >> choice;
switch (choice)
{
case 1:
{
    song_options(folder_arabic, folder, index, folder_choice, vol);
    break;
}
case 2:
{
    break;
}
default:
{
    cout << "Invalid choice\n";
    break;
}
break;
}
```

This function is used to display the menu of options for the songs in a single folder, and at the end, a two-options menu will appear in which if the user wants to continue in this function or return to the folder function.

- **Display songs menu function**

```
void display(song** folder, int folder_choice)
{
    cout << "=========================================\n";
    cout << "the soudtracks are:\n";
    int count = 0;
    for (int i = 0; i < 200; i++)
        if (folder[folder_choice][i].name != "")
            count++;

    for (int i = 0; i < count; i++)
    {
        cout << i + 1 << ". " << folder[folder_choice][i].name << "        " << "its time is " << folder[folder_choice][i].time << endl;
    }
    cout << "=========================================\n";

}
```

This function is the first option in the song's menu, in which it displays the list of songs available in the folder.

- **Single song menu function**

```
void play_sound(song** folder, int index, int folder_choice, int vol)
{
    string name = folder[folder_choice][index - 1].name;
    float time1 = folder[folder_choice][index - 1].time;
    int choice, quan;
    cout << "=========================================\n";
    cout << "playing :  " << name << "  " << "its duration is " << time1 << endl;
    cout << "=========================================\n";
    cout << "select what do you want to do\n1.stop the sound\n2.play the next song\n3.increase the volume\n4.decrease the volume\n";
    cout << "=========================================\n";

    for (int i = 3; i > 0; i++)
    {
        cin >> choice;
        if (choice == 1)
            break;
        else if (choice == 2)
            break;
        else if (choice == 3)
            break;
        else if (choice == 4)
            break;
        cout << "The number you entered is not in the list, choose again\n";
    }

    switch (choice)
    {
    case 1:
    {
        cout << "song is stopped\n";
        break;
    }
    case 2:
    {
        if (folder[folder_choice][index].name == "")
            cout << "There are no songs left\n";
        else
        {
            name = folder[folder_choice][index].name;
            time1 = folder[folder_choice][index].time;
            song_skip(folder, index, folder_choice, name, time1, vol);
        }
        break;
    }
```

```
    }
    case 3:
    {

        cout << "The current volume = " << vol << endl;
        cout << "How much do you want to increase the volume?\n";
        cin >> quan;
        vol = vol + quan;
        if (vol > 100)
        {
            vol = 100;
        }
        cout << "the volume is " << vol << endl;
        break;
    }
    case 4:
    {
        cout << "The current volume = " << vol << endl;
        cout << "How much do you want to decrease the volume?\n";
        cin >> quan;
        vol = vol - quan;
        if (vol < 0)
        {
            vol = 0;
        }
        cout << "the volume is " << vol << endl;
        break;

    }
    default:
    {
        cout << "Invalid choice\n";
        break;
    }

    }
}
```

This function is displayed when the user chooses a specific song to play, it shows a menu for the running song, and the user chooses which option he wants.

- **Song skipping function**

```
void song_skip(song** folder, int index, int folder_choice, string name, float time1, int vol)
{
    int choice, quan;
    cout << "========================================\n";
    cout << "playing :  " << name << "  " << "its duration is " << time1 << endl;
    cout << "========================================\n";
    cout << "select what do you want to do\n1.stop the sound\n2.play the next song\n3.increase the volume\n4.decrease the volume\n";
    cout << "========================================\n";

    for (int i = 3; i > 0; i++)
    {
        cin >> choice;
        if (choice == 1)
            break;
        else if (choice == 2)
            break;
        else if (choice == 3)
            break;
        else if (choice == 4)
            break;
        cout << "The number you entered is not in the list, choose again\n";
    }

    switch (choice)
    {
    case 1:
    {
        cout << "song is stopped\n";
        break;
    }
    case 2:
    {
        index++;
        if (folder[folder_choice][index].name =="")
            cout << "There are no songs left\n";
        else
        {
            name = folder[folder_choice][index].name;
            time1 = folder[folder_choice][index].time;
            song_skip(folder, index, folder_choice, name, time1, vol);
        }
        break;
    }
```

```
    case 3:
    {
        vol = 50;
        cout << "How much do you want to increase the volume?\n";
        cin >> quan;
        vol = vol + quan;
        if (vol > 100)
        {
            vol = 100;
        }
        cout << "the volume is " << vol << endl;
        break;
    }
    case 4:
    {
        vol = 50;
        cout << "How much do you want to decrease the volume?\n";
        cin >> quan;
        vol = vol - quan;
        if (vol < 0)
        {
            vol = 0;
        }
        cout << "the volume is " << vol << endl;
        break;

    }
    default:
    {
        cout << "Invalid choice\n";
        break;
    }

    }
```

This function plays the next song in the folder. The option is chosen by the user from the single song's options.

- **Search for a song function**

```
int search(song** array, string track, int folder_choice)
{
    for (int i = 0; i < 50; i++)

        if (array[folder_choice][i].name == track)
            return i + 1;

    return -1;

}
```

It is an option chosen from the song's menu, it searches for a song in a specific folder, it then returns the position of the song in the folder, if the searched song is found, else it returns -1.

- **Deleting a song function**

```
void deleting(song** array, int folder_choice)
{
    int tot = 200, i, j, found = 0;



    cout << "enter song number to be deleted\n";
    int elem;
    cin >> elem;
    for (i = 0; i < tot; i++)
    {
        if (i == elem - 1)
        {
            for (j = i; j < (tot - 1); j++)

                array[folder_choice][j] = array[folder_choice][j + 1];
            found++;



        }
    }
    if (found == 0)
        cout << "element is not found in the folder";

}
```

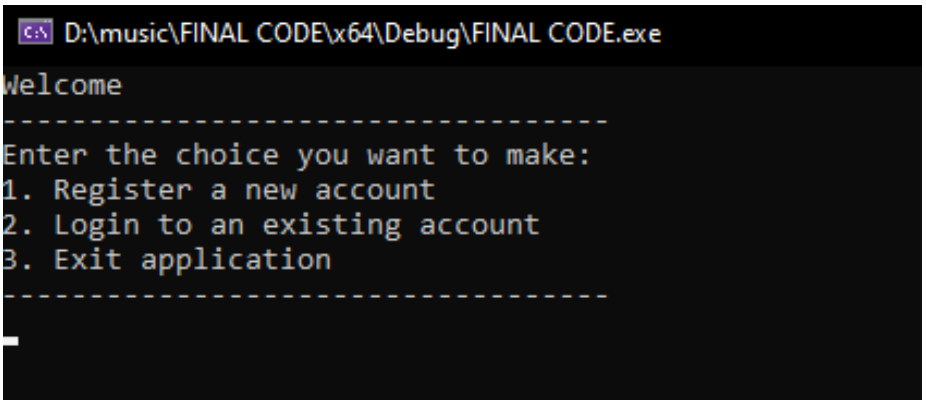
It is an option found in the song's menu, in which you can delete a song from a specific folder.

- **Adding a new song function**

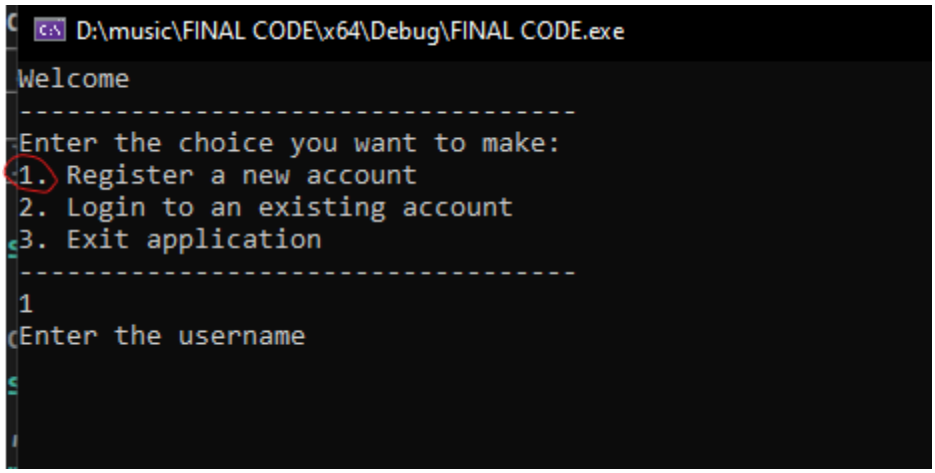
```
void adding(song** array, int folder_choice)
{
    string songname2;
    int n = 0;
    float time;
    cout << "enter the name of the song you want to add\n";
    cin >> songname2;
    cout << "enter its time ";
    cin >> time;

    for (int i = 0; i < 200; i++)
    {
        if (array[folder_choice][i].name != "")
            n++;
    }
    array[folder_choice][n].name = songname2;
    array[folder_choice][n].time = time;


}
```

It is a function that occurs when the user chooses it in the song's menu, in which it adds a new song to a specific folder, by adding its name and its duration.

How To Use the Program features

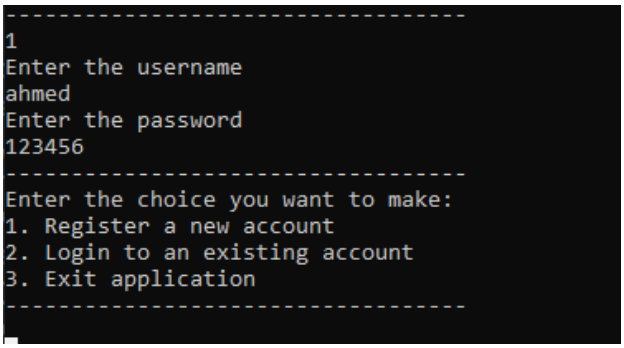


When the program opens it allows the user to choose whether to register a new account or if the user already has an account to login with this account, in order to for the user to access the media player.

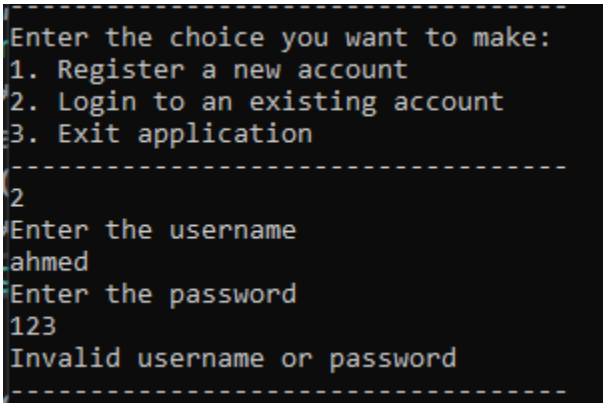


If the user is using the application for the first time, the user needs to register first. So the user need to press(1)

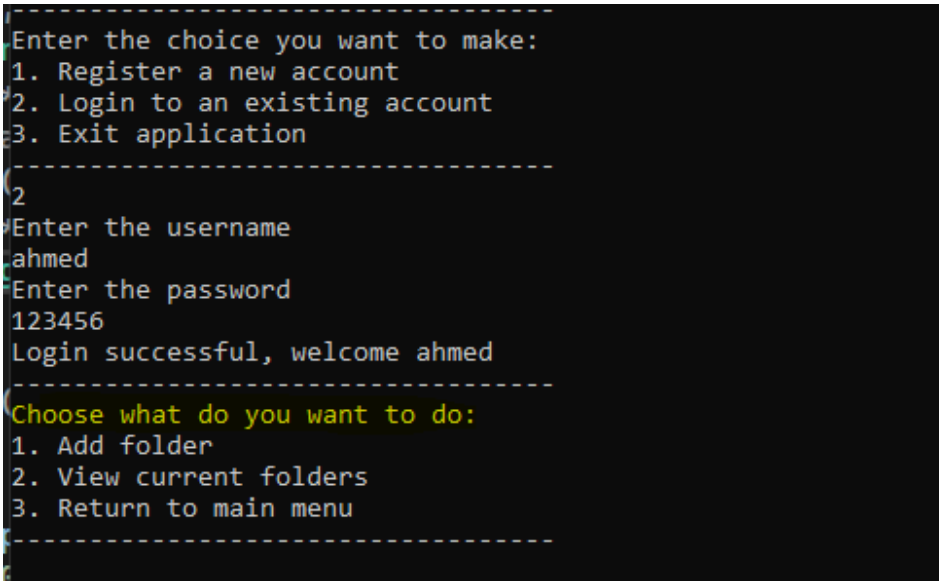
Note: The program can accept as many user accounts as assigned to it.



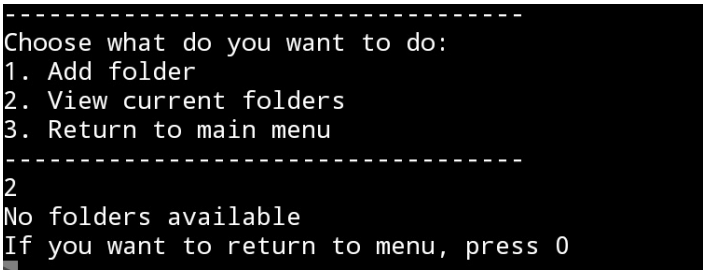
After the user register his name, password, The program saves his data then return once again to the main menu so that the user can login.



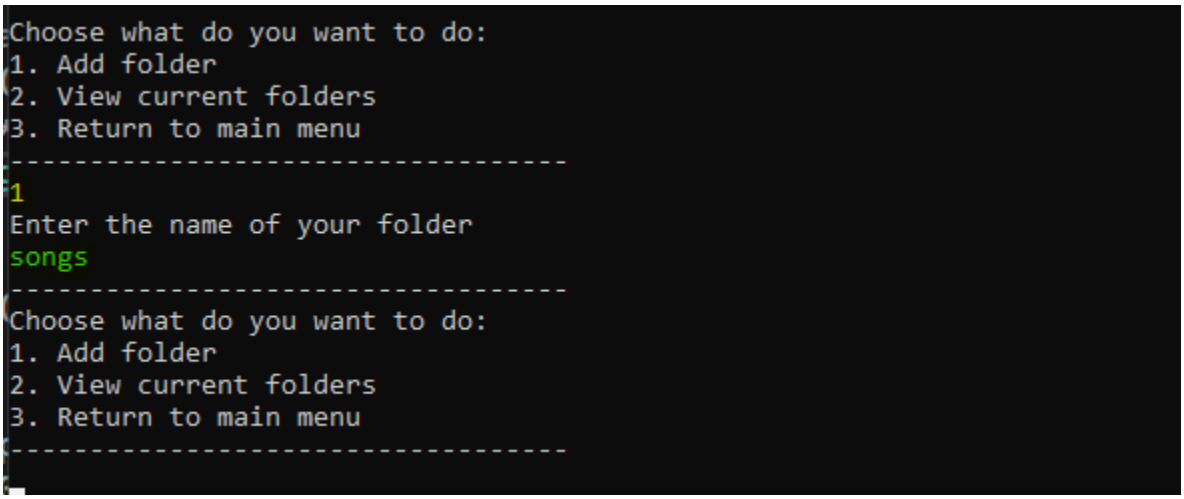
If the user input his username or password are wrong through the process of the login, the program will automatically appear a message that the inputs given are wrong, and will not resume the program until the user enters the correct data.



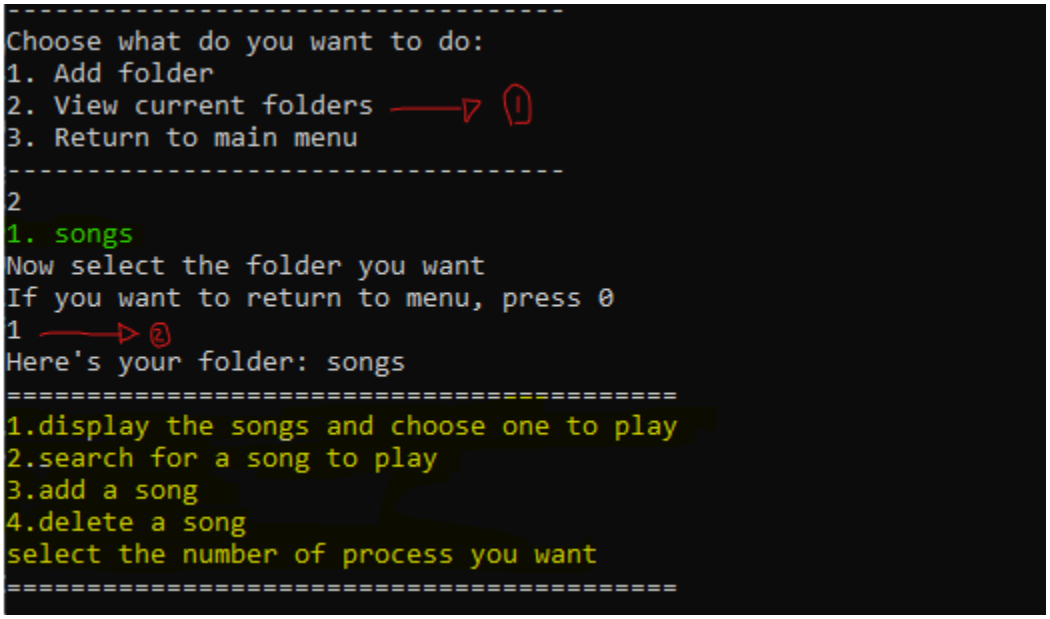
After successfully the user logs in , The Folders menu appears ,contains what actions to be done to the folders in the media player ,and user chooses what action he wants to perform.



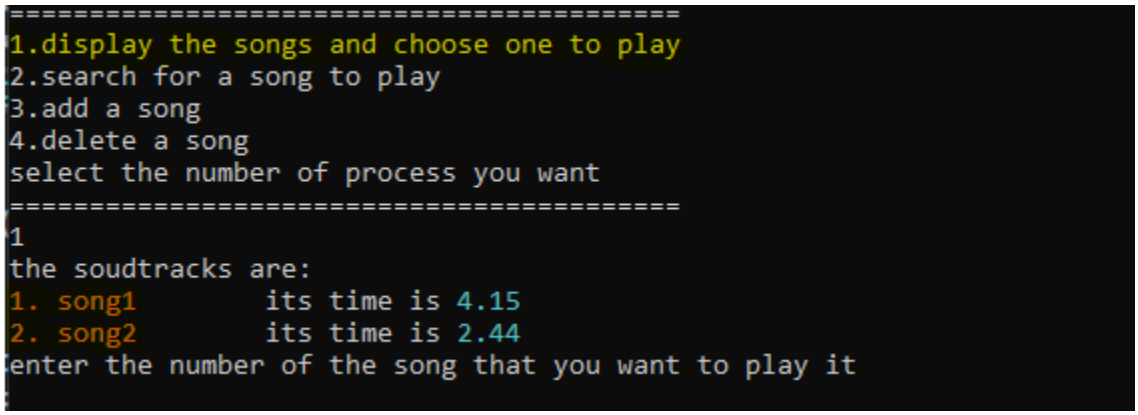
At first, the program will not contain any folders so the user needs to add the folder that contains the songs he needs to play.



After adding for example folder name (songs), user can now by pressing (2) view the menu for this folder.

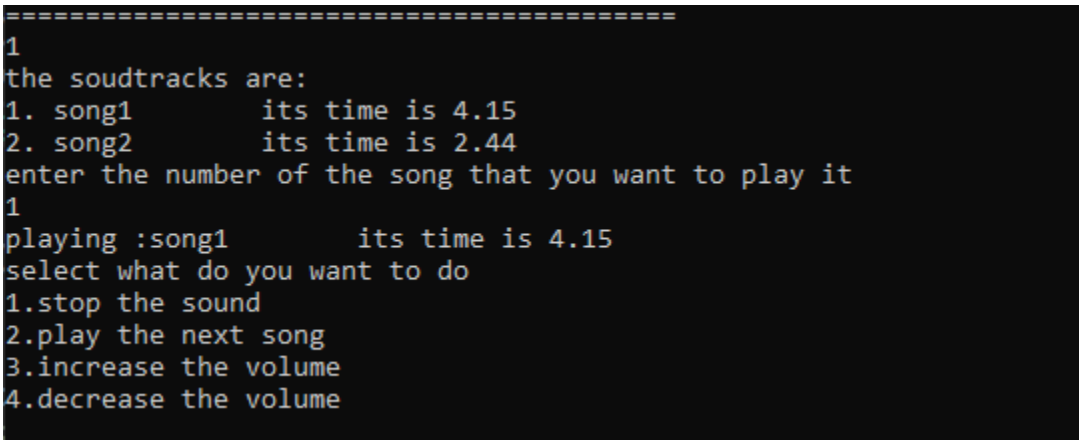


A list of folders appear if he chose so, and the user choose which folder he needs to view. Then, the song's menu appear that contains 1.display 2.search 3.add 4.delete



Pressing (1) views The available tracks and their duration.

Note: Two samples of songs are automatically added in the folder, you can delete them later.



In this phase, the user can choose a song to play and if he does, a menu of editing tools also appear when the song is playing.

```
playing :song1        its time is 4.15
select what do you want to do
1.stop the sound
2.play the next song
3.increase the volume
4.decrease the volume
1
song is stopped
Now choose what do you want:
1. Return to the song's menu
2. Return to the folder's menu
```

The first tool is stop, when pressed the program stops playing the song.

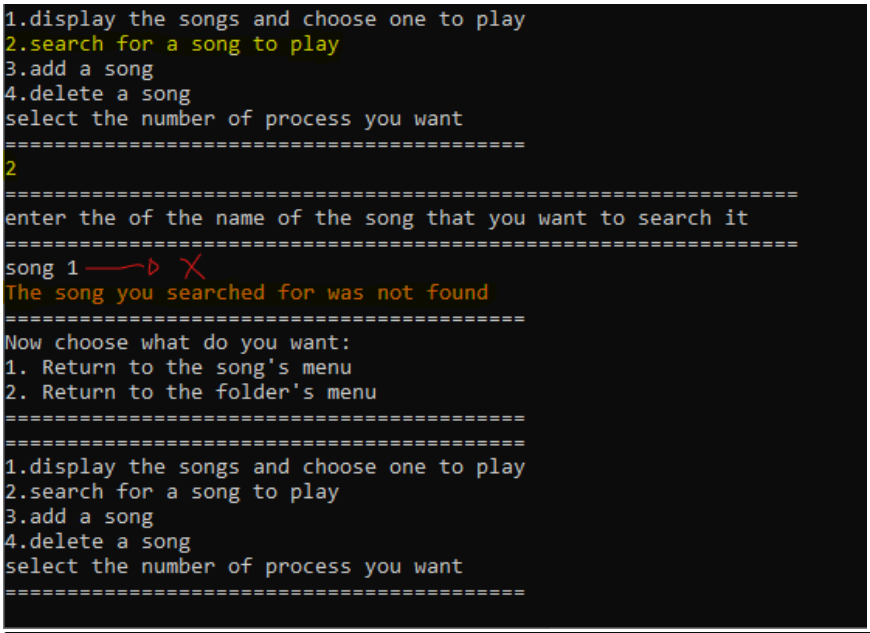
```
playing :song1        its time is 4.15
select what do you want to do
1.stop the sound
2.play the next song
3.increase the volume
4.decrease the volume
2
playing :song2        its time is 2.44
select what do you want to do
1.stop the sound
2.play another song
3.increase the volume
4.decrease the volume
```

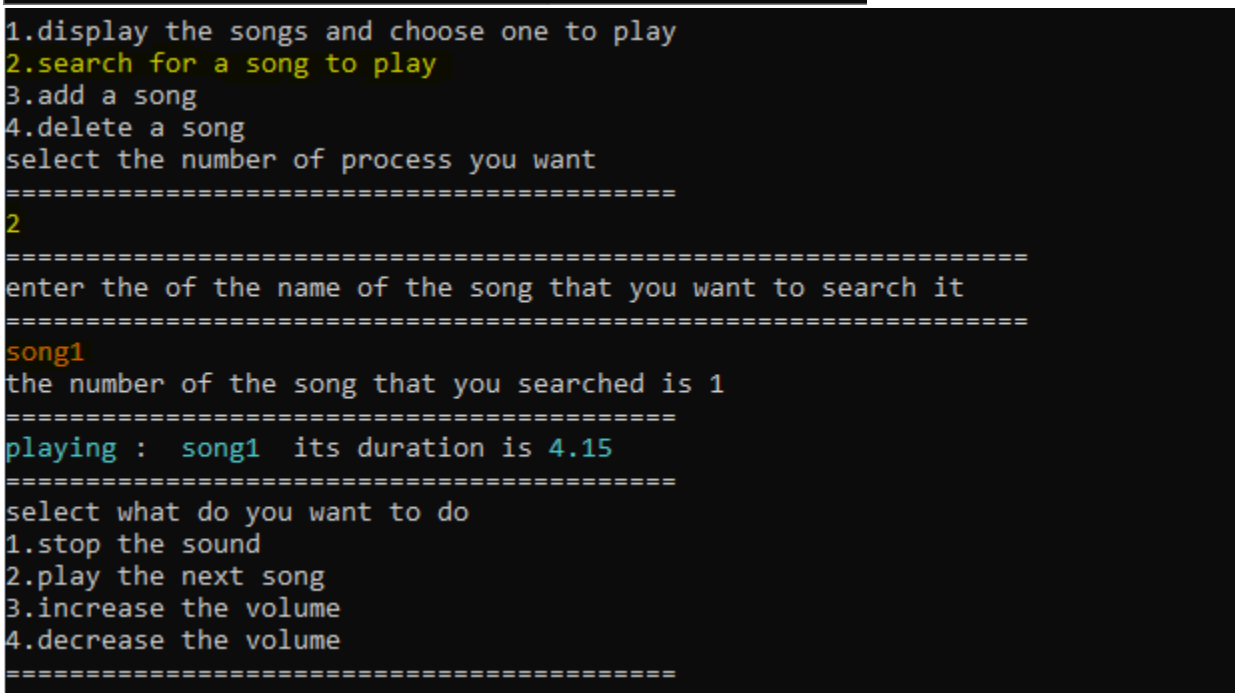
The second tool is play next song, It plays the next song the folder assigned.

```
select what do you want to do
1.stop the sound
2.play another song
3.increase the volume
4.decrease the volume
3
How much do you want to increase the volume?
10
the volume is 60
Now choose what do you want:
1. Return to the song's menu
2. Return to the folder's menu
```

```
select what do you want to do
1.stop the sound
2.play the next song
3.increase the volume
4.decrease the volume
4
How much do you want to decrease the volume?
10
the volume is 40
Now choose what do you want:
1. Return to the song's menu
2. Return to the folder's menu
```

Third and Fourth tool are similar. Both are dealing with the sound of the track playing. When the song initially starts to play , the sound assigned is 50. To increase sound the program asks the user by how much do you want to increase the volume. Similarly, to decrease the sound the program asks the user by how much do he need to decrease the sound.





Second feature in the songs menu is searching for a song, this happens by asking the user for the name of the track he wants in the folder .If the song doesn't exist the search will return not found and will return the user to the song menu. In case the song searched for is found the program automatically plays it .

```
========================================
1.display the songs and choose one to play
2.search for a song to play
3.add a song
4.delete a song
select the number of process you want
========================================
3
the soudtracks are:
1. song1        its time is 4.15
2. song2        its time is 2.44
enter the name of the song you want to add
song3
```

Third feature in the songs menu is adding songs, The program asks the user the name of the song he needs to add and its time.

```
enter the name of the song you want to add
song3
enter its time 3.12
the soudtracks are:
1. song1        its time is 4.15
2. song2        its time is 2.44
3. song3        its time is 3.12
Now choose what do you want:
1. Return to the song's menu
2. Return to the folder's menu
```

The program then will add it to the folder and display its name and time of play.

```
==========================================
1.display the songs and choose one to play
2.search for a song to play
3.add a song
4.delete a song
select the number of process you want
==========================================
4
the soudtracks are:
1. song1        its time is 4.15
2. song2        its time is 2.44
3. song3        its time is 3.12
enter song number to be deleted
1
the soudtracks are:
1. song2        its time is 2.44
2. song3        its time is 3.12
Now choose what do you want:
1. Return to the song's menu
2. Return to the folder's menu
```

The fourth and last feature in the songs menu is deleting a song ,in which the program will ask the user what song he needs to delete from the folder that contains the tracks. For example, the user chose (song1) to be deleted.

The program deletes the song and view to the user the playlist of songs without the song he choose to delete.

```
Now choose what do you want:
1. Return to the song's menu
2. Return to the folder's menu
2
---------------------------------
Choose what do you want to do:
1. Add folder
2. View current folders
3. Return to main menu
---------------------------------
3
---------------------------------
Enter the choice you want to make:
1. Register a new account
2. Login to an existing account
3. Exit application
---------------------------------
3
Thank you for using our application
```

At the end if user wants to close the program first he needs to get out of the songs menu by pressing(2) .Now, the user is in the folders menu, to exit he needs to press (3) .And at last, if he wants to exit the program he needs to press (3) and the program closes and displays a Thank you message .

Summary

Designing a media player using all the topics explained in the course. A media player that asks for login information, can add folders of songs, can play , stop , skip, and control the volume of the songs in the folder . Program that runs until the user determines when to exit from it.