Code with Object Oriented Philosophy

Review: SignIn SignUp Application

Make a SignIn and SignUp Application using Class that will check if the user is stored in the file then it will allow it to LogIn. If the user SignUp then the record is stored in the file in comma separated format.

Step 1: Make the class

Make a class.

```
class credential
{
    public string name;
    public string password;
}
```

Step 2: Make the Menu

```
static int menu()
        {
            int option;
            Console.WriteLine("1. SignIn");
            Console.WriteLine("2. SignUp");
            Console.WriteLine("Enter Option");
            option = int.Parse(Console.ReadLine());
            return option;
        }
```

Step 3: Read the data from the file

```
static void readData(string path, List<credential> users)
{
    if (File.Exists(path))
    {
        StreamReader fileVariable = new StreamReader(path);
        string record;
        while ((record = fileVariable.ReadLine()) != null)
        {
            credential info = new credential();
            info.name = parseData(record, 1);
            info.password = parseData(record, 2);
            users.Add(info);
        }
        fileVariable.Close();
    }
    else
    {
        Console.WriteLine("Not Exists");
    }
}
```

```
static string parseData(string record, int field)
{
    int comma = 1;
    string item = "";
    for(int x = 0; x < record.Length; x++)
    {
        if (record[x] == ',')
        {
            comma++;
        }
        else if (comma == field)
        {
            item = item + record[x];
        }
    }
    return item;
}
```

Step 4: Make the SignIn Function

```
static void signIn(string n, string p, List<credential> users)
{
        bool flag = false;
        for (int x = 0; x < users.Count; x++)
        {
            if (n == users[x].name && p == users[x].password)
            {
                 Console.WriteLine("Valid User");
                 flag = true;
                 break;
            }
        }
        if (flag == false)
        {
            Console.WriteLine("Invalid User");
        }
        Console.ReadKey();
}
```

Step 5: Make the SignUp Function

```
static void signUp(string path, string n, string p)
        {
            StreamWriter file = new StreamWriter(path, true);
            file.WriteLine(n + "," + p);
            file.Flush();
            file.Close();
        }
```

Step 6: Main

```
static void Main(string[] args){
     List<credential> users = new List<credential>();
     string path = "G:\\OOP 2022\\BootingCSharp\\textfile.txt";
     int option;
     do{
        readData(path, users);
        Console.Clear();
        option = menu();
        Console.Clear();
        if (option == 1){
            Console.WriteLine("Enter Name: ");
            string n = Console.ReadLine();
            Console.WriteLine("Enter Password: ");
            string p = Console.ReadLine();
            signIn(n, p, users);
        }
        else if (option == 2){
            Console.WriteLine("Enter New Name: ");
            string n = Console.ReadLine();
            Console.WriteLine("Enter New Password: ");
            string p = Console.ReadLine();
            signUp(path, n, p);
         }
     }
     while (option < 3);
     Console.Read();}
```

Review: SignIn SignUp Application

Do you see any problem in this Solution?

Issues: SignIn SignUp Application

1. We are reading the file on every iteration. The time complexity is increased at it takes a lot of time to read the file on every iteration.
2. We are not adding the object in the list during the signUp function, we are directly adding the attributes in the file.
3. There is no separate function to take input from the user.
4. Right now, the functions are not of single responsibility.

Solution: SignIn SignUp Application

Code that resolves the previous issue is as follows.

Class

```
class MUser
    {
        public string userName;
        public string userPassword;
        public string userRole;

        public MUser(string userName, string userPassword, string userRole)
        {
            this.userName = userName;
            this.userPassword = userPassword;
            this.userRole = userRole;
        }

        public MUser(string userName, string userPassword)
        {
            this.userName = userName;
            this.userPassword = userPassword;
            this.userRole = "NA";
        }

        public bool isAdmin()
        {
            if (userRole == "Admin")
            {
                return true;
            }
            return false;
        }
    }
```

Menu Function

```
static int menu()
        {
            int option;
            Console.WriteLine("1. SignIn");
            Console.WriteLine("2. SignUp");
            Console.WriteLine("Enter Option");
            option = int.Parse(Console.ReadLine());
            return option;
        }
```

ReadData

```
static string parseData(string
record, int field)
{
    int comma = 1;
    string item = "";
    for(int x = 0; x < record.Length;
x++)
    {
        if (record[x] == ',')
        {
            comma++;
        }
        else if (comma == field)
        {
            item = item + record[x];
        }
    }
    return item;
}
```

```
static void readDataFromFile(string path)
{
    if (File.Exists(path))
    {
        StreamReader fileVariable = new StreamReader(path);
        string record;
        while ((record = fileVariable.ReadLine()) != null)
        {
            string userName = parseData(record, 1);
            string userPassword = parseData(record, 2);
            string userRole = parseData(record, 3);
            MUser user = new MUser(userName, userPassword, userRole);
            addUserIntoList(user);
        }
        fileVariable.Close();
    }
    else
    {
        Console.WriteLine("Not Exists");
    }
}
```

For SignUp: TakeInputFromConsole

```
static MUser TakeInputFromConsole()
        {
            Console.WriteLine("Enter UserName");
            string userName = Console.ReadLine();
            Console.WriteLine("Enter UserPassword");
            string userPassword = Console.ReadLine();
            Console.WriteLine("Enter UserRole");
            string userRole = Console.ReadLine();

            MUser user = new MUser(userName, userPassword, userRole);
            return user;

        }
```

For SignUp: AddUserIntoList

```
static void addUserIntoList(MUser user)
        {
            usersList.Add(user);
        }
```

For SignUp: StoreUserIntoFile

```
static void storeUserIntoFile(MUser user, string path)
        {
            StreamWriter file = new StreamWriter(path, true);
            file.WriteLine(user.userName + "," + user.userPassword + "," + user.userRole);
            file.Flush();
            file.Close();


        }
```

For SignIn: takeInputwithOutRole

```
static MUser takeInputwithOutRole()
        {
            Console.WriteLine("Enter UserName");
            string userName = Console.ReadLine();
            Console.WriteLine("Enter UserPassword");
            string userPassword = Console.ReadLine();
            MUser user = new MUser(userName, userPassword);
            return user;
        }
```

For SignIn: Validate User

```
static MUser SignIn()
        {
            MUser user = takeInputwithOutRole();
            foreach (MUser storedUser in usersList)
            {
                if (storedUser.userName == user.userName && storedUser.userPassword == user.userPassword)
                {
                    return storedUser;
                }
            }
            return null;
        }
```

Main Function

```
static List<MUser> usersList = new List<MUser>();

        static void Main(string[] args){
            string path = "Data.txt";
            readDataFromFile(path);
            int option = 0;
            while (option != 3){
                Console.Clear();
                option = menu();
                if (option == 1){
                    MUser user = SignIn();
                    if (user != null){
                        if (user.isAdmin()){
                            Console.WriteLine("This is Admin");
                            //Admin Menu
                        }
                        else{
                            Console.WriteLine("This is User");
                            //User Menu
                        }
                    }
                }
                else if (option == 2){
                    MUser user = TakeInputFromConsole();
                    addUserIntoList(user);
                    storeUserIntoFile(user, path);
                }
                Console.ReadKey();
            }
        }
```

Learning Objective

Write class with appropriate behaviour on the data and separate functions with single responsibility.