public abstract class GeneralUser

{

    public string Name { get; set; }

    public string Password { get; private set; }

    public GeneralUser(string name = "Null", string password = "Null")

    {

        Name = name;

        Password = password;

    }

    public void LogOut()

    {

        System.Console.WriteLine("Toto se ještě musí dodělat.");

    }

}

public class Manager : GeneralUser

{

    public string Name { get; set; }

    public string Password { get; private set; }

    public Manager(string name = "Null", string password = "Null") : base(name, password)

    {

    }

    public void ListUsers()

    {

        foreach (var row in File.ReadAllLines(Utils.allUsersPathAndFile))

        {

            int index = 1;

            var user = row.Split(';');

            System.Console.WriteLine($"User number: {index++} - {user[0]}");

        }

    }

    public User? GetSelectedUser()

    {

        bool validIndex = false;

        int index;

        do

        {

            Console.Write("Input number of user to be selected: ");

            validIndex = int.TryParse(Console.ReadLine(), out index);

            if (!validIndex) Console.WriteLine("Try again.");

            else index--;

        } while (!validIndex);

        string userLine = File.ReadAllLines(Utils.allUsersPathAndFile)[index];

        User selectedUser = new User

        {

            Name = userLine.Split(';')[0]

        };

        selectedUser.Tasks = selectedUser.GetTasksOfUser();

        return selectedUser;

    }

    public void ManagerMenu()

    {

        bool IsValidInput = false;

        while (!IsValidInput)

        {

            Console.Clear();

            Console.WriteLine("1) List Tasks");

            Console.WriteLine("2) Find Tasks");

            Console.WriteLine("3) Add Task");

            Console.WriteLine("4) Delete Task");

            Console.WriteLine("0) Log out");

            Console.Write("Your choice (0-4): ");

            string choice = Console.ReadLine();

            switch (choice)

            {

                case "1":

                    {

                        ListUsers();

                        User? selectedUser = GetSelectedUser();

                        if (selectedUser != null)

                        {

                            selectedUser.ListTasks();

                            IsValidInput = true;

                            List<Task> selectedUserTasks = selectedUser.GetTasksOfUser();

                        }

                        else System.Console.WriteLine("User not found.");

                        break;

                    }

                case "2":

                    {

                        ListUsers();

                        User? selectedUser = GetSelectedUser();

                        if (selectedUser != null)

                        {

                            IsValidInput = true;

                            selectedUser.FindTasks(); //of specific user

                        }

                        else System.Console.WriteLine("User not found.");

                        break;

                    }

                case "3":

                    {

                        IsValidInput = true;

                        AddTask();

                        break;

                    }

                case "4":

                    {

                        IsValidInput = true;

                        DeleteTask();

                        break;

                    }

                case "0":

                    {

                        IsValidInput = true;

                        LogOut(); //inherited from GeneralUser

                        break;

                    }

                default:

                    {

                        IsValidInput = false;

                        Console.WriteLine("Invalid input. Try again.");

                        break;

                    }

            }

        }

    }

    public void AddTask()

    {

        System.Console.WriteLine("Toto se ještě musí dodělat.");

    }

    public void DeleteTask()

    {

        System.Console.WriteLine("Toto se ještě musí dodělat.");

    }

}

public class User : GeneralUser

{

    public string Name { get; set; }

    public string Password { get; private set; }

    public List<Task> Tasks { get; set; }

    public User(string name = "Null", string password = "Null") : base(name, password)

    {

    }

    public User(string name, List<Task> tasks) : base(name)

    {

        Tasks = tasks;

    }

    public void UserMenu()

    {

        bool IsValidInput = false;

        while (!IsValidInput)

        {

            Console.Clear();

            Console.WriteLine("1) List Tasks");

            Console.WriteLine("2) Find Tasks");

            Console.WriteLine("3) Mark Task as Completed");

            Console.WriteLine("0) Log out");

            Console.Write("Your choice (0-3): ");

            string choice = Console.ReadLine();

            switch (choice)

            {

                case "1":

                    {

                        IsValidInput = true;

                        ListTasks(); //inherited from GeneralUser

                        break;

                    }

                case "2":

                    {

                        IsValidInput = true;

                        FindTasks(); //inherited from GeneralUser

                        break;

                    }

                case "3":

                    {

                        IsValidInput = true;

                        MarkTaskAsCompleted();

                        break;

                    }

                case "0":

                    {

                        IsValidInput = true;

                        LogOut(); //inherited from GeneralUser

                        break;

                    }

                default:

                    {

                        IsValidInput = false;

                        Console.WriteLine("Invalid input. Try again.");

                        break;

                    }

            }

        }

    }

    public void MarkTaskAsCompleted()

    {

        ListTasks();

        bool validIndex = false;

        int index;

        do

        {

            Console.Write("Input number of task to be marked as completed: ");

            validIndex = int.TryParse(Console.ReadLine(), out index);

            if (!validIndex) Console.WriteLine("Try again.");

            else index--;

        } while (!validIndex);

        if (index >= 0 && index < Tasks.Count)

        {

            Tasks[index].Completed = true;

            SaveTasks(Tasks);

            Console.WriteLine("Task marked as completed.");

        }

        else

        {

            Console.WriteLine("Invalid index.");

        }

        Console.ReadKey();

    }

    public void SaveTasks(List<Task> tasks)

    {

        XmlSerializer taskSerializer = new XmlSerializer(typeof(List<Task>));

        //string appDataPath = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData);

        //string pathToDirectory = Path.Combine(appDataPath, "TaskTracker");

        if (!Directory.Exists(Utils.pathToDirectory))

        {

            Directory.CreateDirectory(Utils.pathToDirectory);

        }

        string userXmlFile = $"{Name}.xml";

        string pathToXmlFileInDirectory = Path.Combine(Utils.pathToDirectory, userXmlFile);

        using (StreamWriter writer = new StreamWriter(pathToXmlFileInDirectory))

        {

            taskSerializer.Serialize(writer, tasks);

        }

        /\*var rows = new List<string>();

        foreach (var t in tasks)

        {

            rows.Add($"{t.Description};{t.HighPriority};{t.DueDate.ToString("dd.MM.yyyy")};{t.Completed}");

        }

        File.WriteAllLines($"{Name}.txt", rows);\*/

    }

    public List<Task> GetTasksOfUser()

    {

        List<Task> tasks1 = new List<Task>();

        //string appDataPath = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData);

        //string pathToDirectory = Path.Combine(appDataPath, "TaskTracker");

        if (!Directory.Exists(Utils.pathToDirectory))

        {

            Directory.CreateDirectory(Utils.pathToDirectory);

        }

        string userXmlFile = $"{Name}.xml";

        string pathToXmlFileInDirectory = Path.Combine(Utils.pathToDirectory, userXmlFile);

        if (!File.Exists(pathToXmlFileInDirectory))

        {

            throw new Exception("Tasks not found.");

            //return tasks;

        }

        XmlSerializer taskSerializer = new XmlSerializer(typeof(List<Task>));

        using (StreamReader reader = new StreamReader(pathToXmlFileInDirectory))

        {

            tasks1 = taskSerializer.Deserialize(reader) as List<Task>;

        }

        return tasks1;

    }

    /\*List<Task> tasks = new List<Task>();

    string userFile = $"{Name}.txt";

    string pathToFileInDirectory = Path.Combine(pathToDirectory, userFile);

    if (!File.Exists(pathToFileInDirectory))

    {

        throw new Exception("Tasks not found.");

        //return tasks;

    }

    foreach (var row in File.ReadAllLines(pathToFileInDirectory))

    {

        var data = row.Split(';');

        tasks.Add(new Task

        {

            Description = data[0],

            HighPriority = bool.Parse(data[1]),

            DueDate = DateTime.Parse(data[2]),

            Completed = bool.Parse(data[3])

        });

    }

    return tasks;

    }\*/

    public void ListTasks()

    {

        System.Console.WriteLine($"User: {Name} - List of tasks: ");

        foreach (Task t in Tasks)

        {

            int index = 1;

            System.Console.WriteLine($"Task number: {index++} - {t.Description}; High priority: {t.HighPriority}; Due date: {t.DueDate.ToString("dd.MM.yyyy")}; Completed: {t.Completed}");

        }

    }

    public void FindTasks()

    {

        System.Console.WriteLine("Toto se ještě musí dodělat – hledání tasků podle Description.Contains");

    }

}

public class Task

{

    public string Description { get; set; }

    public bool HighPriority { get; set; }

    public DateTime DueDate { get; set; }

    public bool Completed { get; set; }

}

public class Utils

{

    //for public static methods and var

    public static string appDataPath = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData);

    public static string pathToDirectory = Path.Combine(appDataPath, "TaskTracker");

    public static string allUsersFile = "users.txt";

    public static string allManagersFile = "managers.txt";

    public static string allUsersPathAndFile = Path.Combine(pathToDirectory, allUsersFile);

    public static string allManagersPathAndFile = Path.Combine(pathToDirectory, allManagersFile);

    static string EncodePassword(string password)

    {

        return Convert.ToBase64String(Encoding.UTF8.GetBytes(password));

    }

    static string DecodePassword(string codedPassword)

    {

        return Encoding.UTF8.GetString(Convert.FromBase64String(codedPassword));

    }

    public static User? LogInUser()

    {

        Console.Write("Input user name: ");

        string name = Console.ReadLine();

        Console.Write("Input password: ");

        string password = Console.ReadLine();

        string codedPassword = EncodePassword(password);

        if (!File.Exists(allUsersPathAndFile) || !Directory.Exists(pathToDirectory))

        {

            Console.WriteLine("No registered users found.");

            Console.ReadKey();

            return null;

        }

        foreach (var row in File.ReadAllLines(allUsersPathAndFile))

        {

            var data = row.Split(';');

            if (data[0] == name && data[1] == codedPassword)

            {

                User loggedUser = new User

                {

                    Name = name

                };

                loggedUser.Tasks = loggedUser.GetTasksOfUser();

                Console.WriteLine($"Login successful, welcome {loggedUser.Name}.");

                Console.ReadKey();

                return loggedUser;

            }

        }

        Console.WriteLine("Invalid credentials.");

        Console.ReadKey();

        return null;

    }

    public static Manager? LogInManager()

    {

        Console.Write("Input manager name: ");

        string name = Console.ReadLine();

        Console.Write("Input password: ");

        string password = Console.ReadLine();

        string codedPassword = EncodePassword(password);

        if (!File.Exists(allManagersPathAndFile) || !Directory.Exists(pathToDirectory))

        {

            Console.WriteLine("No registered managers found.");

            Console.ReadKey();

            return null;

        }

        foreach (var row in File.ReadAllLines(allManagersFile))

        {

            var data = row.Split(';');

            if (data[0] == name && data[1] == codedPassword)

            {

                Manager loggedManager = new Manager

                {

                    Name = name

                };

                Console.WriteLine($"Login successful, welcome {loggedManager.Name}.");

                Console.ReadKey();

                return loggedManager;

            }

        }

        Console.WriteLine("Invalid credentials.");

        Console.ReadKey();

        return null;

    }

class Program

{

    //public string souborUzivatele = "uzivatele.txt";

    static void Main(string[] args)

    {

        GeneralUser loggedUser = new User();

        GeneralUser loggedManager = new Manager();

        do

        {

            if (loggedUser == null && loggedManager == null)

            {

                Console.Clear();

                Console.WriteLine("------------------------");

                Console.WriteLine("TASK TRACKER - Main Menu");

                Console.WriteLine("------------------------");

                Console.WriteLine("1) Log in as User");

                Console.WriteLine("2) Log in as Manager");

                Console.WriteLine("3) Sign up as new User");

                Console.WriteLine("0) End program");

                Console.Write("Your choice (0-3): ");

                string choice = Console.ReadLine();

                switch (choice)

                {

                    case "1":

                        {

                            loggedUser = Utils.LogInUser();

                            break;

                        }

                    case "2":

                        {

                            loggedManager = Utils.LogInManager();

                            break;

                        }

                }

            }

            else if (loggedUser != null)

            {

                (loggedUser as User).UserMenu();

            }

            else if (loggedManager != null)

            {

                (loggedManager as Manager).ManagerMenu();

            }

        } while (loggedUser == null && loggedManager == null);

    }

}