**A PROJECT REPORT ON**

# Realtime Chat Application

**By**

**Nevil Parmar (CE093) (18CEUBG023)**

**B. Tech CE Semester - VI**
**Subject: Service Oriented Computing**

**Guided By:**

| | |
|---|---|
| **Prof. Apurva A. Mehta** | **Prof. Prashant M. Jadav** |
| **Assistant Professor** | **Associate Professor** |
| **Dept. Of Computer Engineering** | **Dept. Of Computer Engineering** |

**Faculty of Technology**
**Department of Computer Engineering**
**Dharmsinh Desai University**

**Faculty of Technology**
**Department of Computer Engineering**
**Dharmsinh Desai University**

# CERTIFICATE

This is to certify that the practical / term work carried out in the subject of **Service Oriented Computing** and recorded in this journal is the bonafide work of

**Nevil Parmar (CE093) (18CEUBG023)**

of B. Tech semester **VI** in the branch of **Computer Engineering** during the academic year **2020-2021**.

Prof. Apurva A. Mehta
Assistant Professor,
Dept. of Computer Eng.,
Faculty of Technology,
Dharmsinh Desai
University, Nadiad

Prof. Prashant M. Jadav
Associate Professor,
Dept. of Computer Eng.,
Faculty of Technology,
Dharmsinh Desai
University, Nadiad

Dr. C. K. Bhensdadia,
Head,
Dept. of Computer Eng.,
Faculty of Technology,
Dharmsinh Desai
University, Nadiad

# Table of Contents

## Abstract

The system is a multiuser chat application with the support of 2 platforms, web and windows. The system mainly works in 2 modes. One to one chat mode & Broadcast mode. Users can create multiple channels, and can share the name of channel with multiple users, and by joining the channel they can chat with multiple users at a time. The system also comes with rich features like using emoji in chat, media sharing support, update profile, forgot password.

## Purpose

The purpose of this document is to collect and present ideas, requirements & analysis done in order to develop this system. This document provides a detailed overview of the system, details of the functionalities provided to users, target audience of the system and their user interfaces.

## Scope

The scope of the system is providing a handy web and desktop application to users where they can spend One – On – One time with friends or in a group. Any user with a registered account on the system can use the functionalities & get benefits of the provided features.

# Introduction

## - Brief Introduction

The system is a multiuser chat application with the support of 2 platforms, web and windows. The system mainly works in 2 modes. One to one chat mode & Broadcast mode. Users can create multiple channels, and can share the name of channel with multiple users, and by joining the channel they can chat with multiple users at a time. The system also comes with rich features like using emoji in chat, media sharing support, manage profile and forgot password.

## - Technology/Platform/Tools used

**Technology:**

- WCF framework
- Dot net core WEB API framework
- ASP.NET framework
- Windows Forms
- WPF (Windows presentation Foundation)
- Signal - R
- SQL Server database.
- JavaScript
- Bootstrap

**Platform:**

- Windows

**Tools:**

- Visual Studio

## Software Requirements Specification - SRS

The system works with general end users only.

## Functional Requirements

### R.1 User module

#### R.1.1 Login

Description: User can login to the system

Exception Flow: If credentials are incorrect or insufficient data is provided

Input: User data

Output: Redirects to Dashboard

#### R.1.2 Register

Description: User can login to the system

Exception flow: If username is already taken or insufficient data is provided, user will be asked to fill the details again.

Description: On successful registration of a new users, the system will send a greeting email to the registered email

Input: User data

Output: Redirects to Login page with success message

#### R.1.3 Update profile

Exception flow: If username is already taken or insufficient data is provided

Input: User profile data

Output: Success message

### R.1.4 Forgot Password

Input: username, old password, new password

Output: Response message

### R.1.5 Logout

Input: User selection

Output: Redirects to Login page with success message

### R.1.6 Delete my account

Input: User selection

Output: Redirects to Login page with success message

### R.1.7 Search user

Input: Username to be searched

Output: List of users matching search criteria

## R.2 One to one chat module

### R.2.1 Send a message to particular user

Input: Message to be sent

Output: Upon successful delivery of a message, the message will be listed onto the chat screen

Description: Users selects the name of the user he want to chat with, and enters the message

### R.3 Broadcast chat module

#### R.3.1 Create channel / Room

Exception flow: If channel exists, error message will be displayed

Input: Channel name

Output: Success message

#### R.3.2 Join channel / Room

Exception flow: If channel does not exist, error message will be displayed

Input: Channel name

Output: Success message

#### R.3.3 Send a message in channel

Input: Message to be sent

Output: Upon successful delivery of a message, message will be listed onto the chat screen

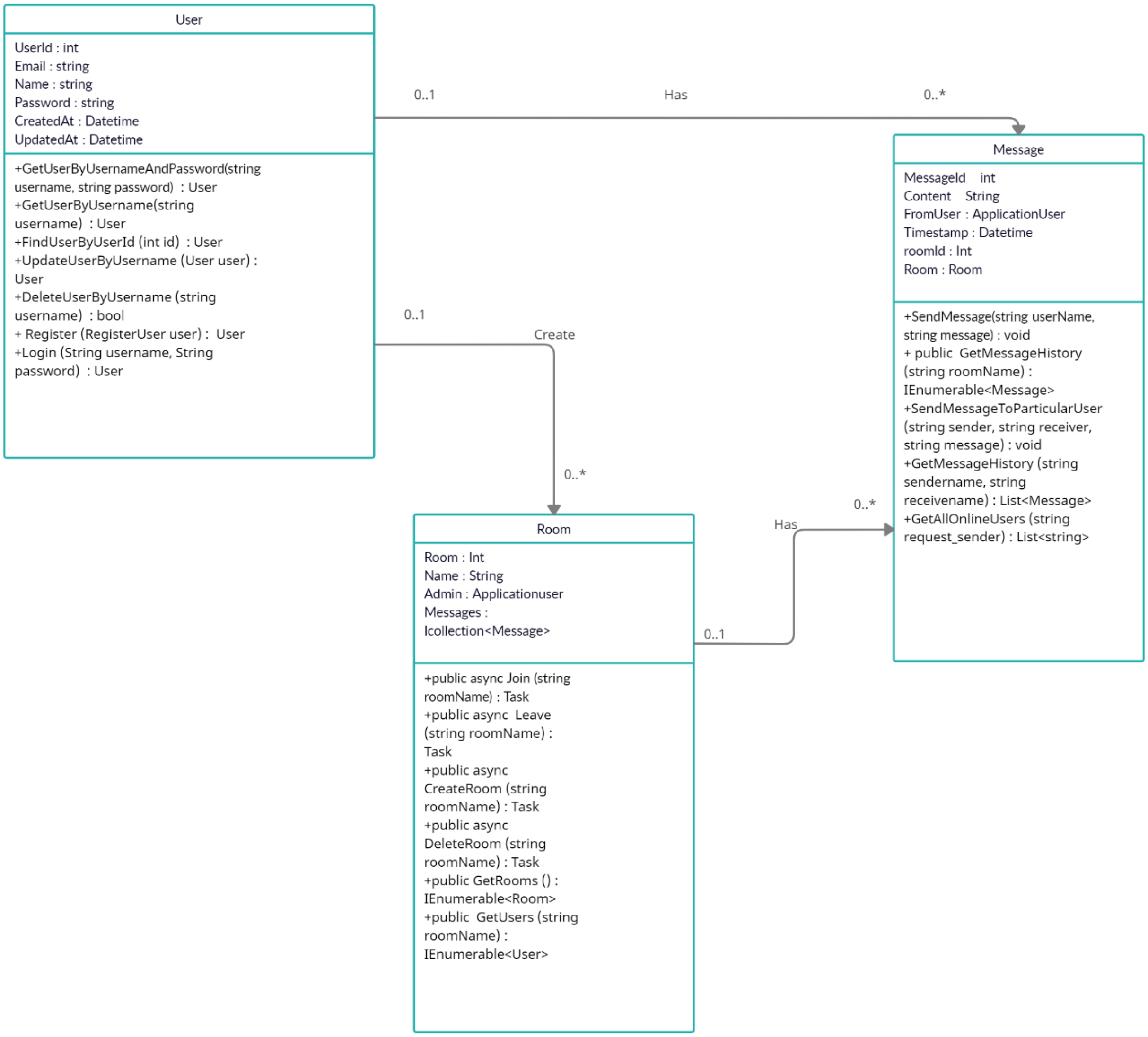
### R.3.4 Delete channel / Room

Input: User Selection

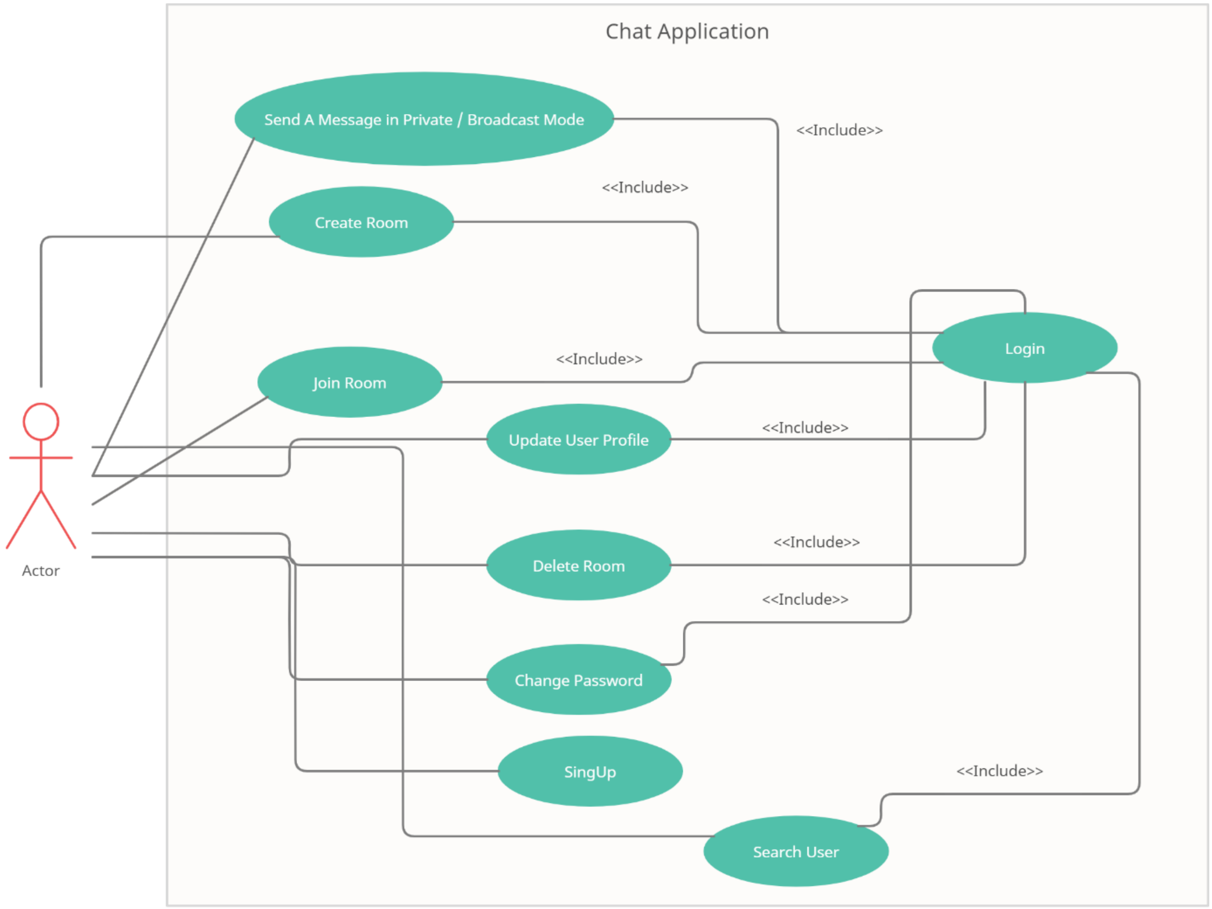
Output: Success message

## Design

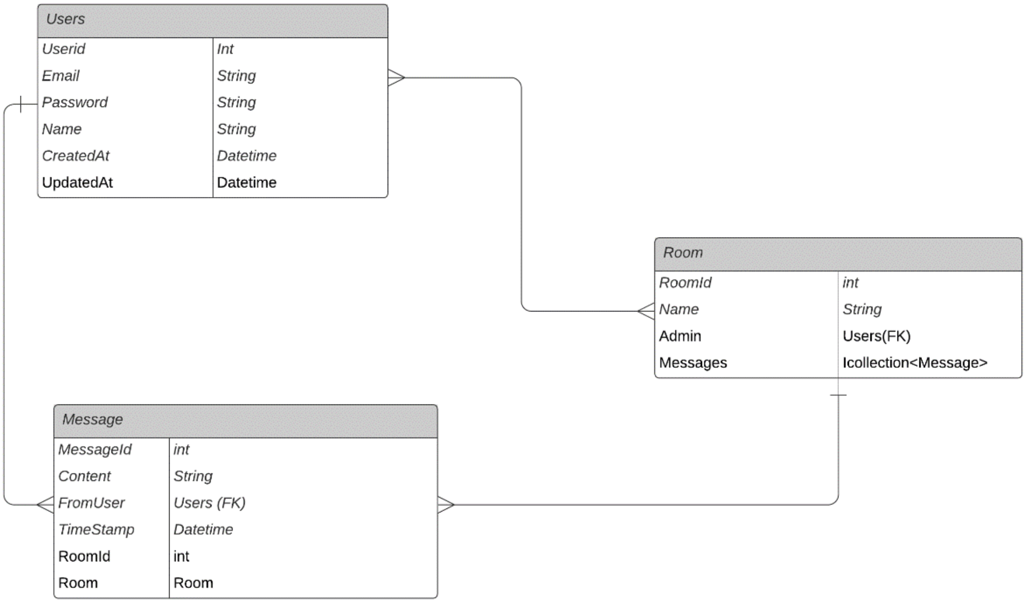
## Class Diagram


User
UserId : int
Email : string
Name : string
Password : string
CreatedAt : Datetime
UpdatedAt : Datetime
+GetUserByUsernameAndPassword(string username, string password) : User
+GetUserByUsername(string username) : User
+FindUserByUserId (int id) : User
+UpdateUserByUsername (User user) : User
+DeleteUserByUsername (string username) : bool
+ Register (RegisterUser user) : User
+Login (String username, String password) : User
0..1
Has
0..*
Message
MessageId int
Content String
FromUser : ApplicationUser
Timestamp : Datetime
roomId : Int
Room : Room
+SendMessage(string userName, string message) : void
+ public GetMessageHistory (string roomName) : IEnumerable<Message>
+SendMessageToParticularUser (string sender, string receiver, string message) : void
+GetMessageHistory (string sendername, string receivename) : List<Message>
+GetAllOnlineUsers (string request_sender) : List<string>
0..1
Create
0..*
Room
Room : Int
Name : String
Admin : Applicationuser
Messages : Icollection<Message>
+public async Join (string roomName) : Task
+public async Leave (string roomName) : Task
+public async CreateRoom (string roomName) : Task
+public async DeleteRoom (string roomName) : Task
+public GetRooms () : IEnumerable<Room>
+public GetUsers (string roomName) : IEnumerable<User>
Has
0..*
0..1

## Use case Diagram



## ER Diagram



## Data Dictionary

## Application User

| No | Field name | Data type | Required | Unique | PK / FK | Ref. Table |
|---|---|---|---|---|---|---|
| 1 | UserId | int | true | true | PK | - |
| 2 | Email | string | true | true | - | - |
| 3 | Name | string | true | false | - | - |
| 4 | Password | string | true | false | - | - |
| 5 | CreatedAt | Datetime | true | false | - | - |

| | | | | | | |
|---|---|---|---|---|---|---|
| 6 | UpdatedAt | Datetime | True | False | - | - |

### Message

| No | Field name | Data type | Required | Unique | PK / FK | Ref. Table |
|---|---|---|---|---|---|---|
| 1 | MessageId | int | true | true | PK | - |
| 2 | Content | String | True | False | - | - |
| 3 | FromUser | ApplicationUser | True | False | FK | Applicationuser |
| 4 | Timestamp | Datetime | True | False | - | - |
| 5 | roomId | Int | True | False | - | - |
| 6 | Room | Room | True | False | FK | Room |

### Room

| No | Field name | Data type | Required | Unique | PK / FK | Ref. Table |
|---|---|---|---|---|---|---|
| 1 | Room | int | true | true | PK | - |
| 2 | Name | String | true | True | - | - |
| 3 | Admin | Application user | True | False | FK | Application User |
| 4 | Messages | Icollection<Message> | False | False | - | - |

## Implementation Details

## Modules

Total modules: 3

1. **Users Module:**

   Deals with user related operations. Using functionalities provided by this module, user can register themselves to the system, log in / log out to the system and manage their user profiles.

2. **ONE-TO-ONE Chat Module:**

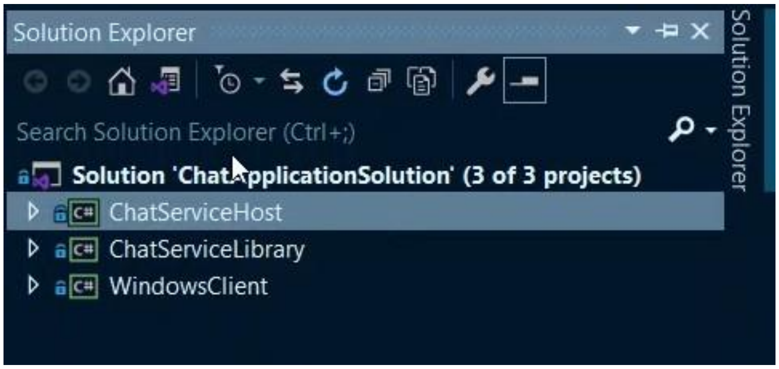
   Deals with user related operations. Using functionalities provided by this module, user can chat with particular user at a time, it also has emoji support and media support in the chat.

3. **Broadcast Chat Module:**

   Deals with user related operations. Using functionalities provided by this module, users can create multiple channels, and joining them will enabled them to chat in a group. It also has emoji support and media support in chat.

## EXTRA

## WCF Project:



Solution Explorer

Search Solution Explorer (Ctrl+;)

**Solution 'ChatApplicationSolution' (3 of 3 projects)**

- ChatServiceHost
- ChatServiceLibrary
- WindowsClient

- Uses TCP Dual binding protocol.
- Implemented using WCF Callback mechanism
- Implemented using WCF duplex service, where users send their msg to the server, and server, in turn, forwards their messages to all the users in a network using Client Callback

**The project solution contains 3 projects.**

**1. ChatServiceLibrary**

- WCF Service Library
- The library contains 3 major services
  1. User Service
  - Deals with user CRUD service operations
  2. Broadcast Service
  - Deals with Broadcast message service operations
  3. One-One Chat Service
  - Deals with a one-one chat message service operations

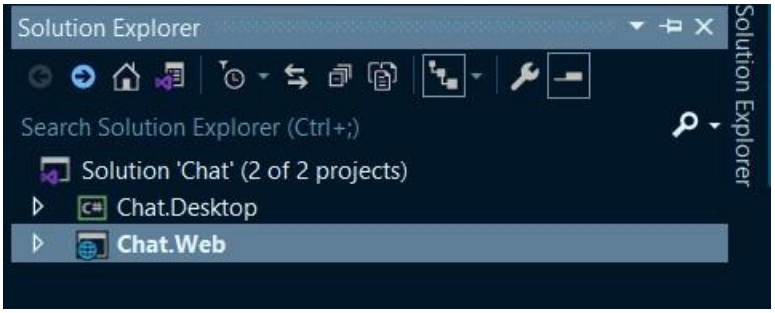
**2. ChatServiceHost**

- Console based host application

**3. WindowsClient**

- Multi-Screen Windows forms type Client application

## WEB API Project:


Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Chat' (2 of 2 projects)
Chat.Desktop
Chat.Web
Solution Explorer

- Implemented using signal-R for real time updates
- Implemented WPF client and Web client

**The project solution contains 2 projects.**

**1. Chat.Web**
- WEB API Project
- The project contains services and Hubs implementation related to chat operations and users management
- It uses Signal-R for real time update in chat

**2. Chat.Desktop**
- WPF desktop client to demonstrate use of single service in different platform

## Function prototypes which implement major functionality

**- Function prototypes of Users Module**

- User GetUserByUsernameAndPassword (string username, string password);
- User GetUserByUsername (string username);
- User FindUserByUserId (int id);
- User UpdateUserByUsername (User user);
- User DeleteUserByUsername (string username);
- User Register (RegisterUser user);
- User Login (String username, String password)

**- Function prototypes of One-One Chat Module**

- bool AddMeToServer (string userName);
- bool LeaveServer (string userName);
- bool LogInState (string userName);
- void SendMessageToParticularUser (string sender, string receiver, string message);
- List<SingleChatMessage> GetMessageHistory (string sendername, string receivename);
- List<string> GetAllOnlineUsers (string request_sender);

**- Function prototypes of Broadcast Module**

- bool LogInState(string userName);
- void SendMessage(string userName, string message);
- public IEnumerable<MessageViewModel> GetMessageHistory (string roomName)
- public async Task Join (string roomName)
- public async Task Leave (string roomName)
- public async Task CreateRoom (string roomName)
- public async Task DeleteRoom (string roomName)
- public IEnumerable<RoomViewModel> GetRooms ()
- public IEnumerable<UserViewModel> GetUsers (string roomName)

# Testing

Unit testing of each module was done after successfully completing the module. Each module was tested individually before integrating them with the whole system.
After integrating each module with the system, integration testing was done in order to check if modules are working properly together.
After completing all integrations, black-box testing of the whole system was carried out to ensure the system works in a correct manner.

**Black box testing of Major functions of the system**

1. **Log in to the system.**

   **Case 1:** Invalid Username or password entered by the user.
   **Output:** Error message on the screen saying "Invalid credentials"

   **Case 2:** Valid credentials.
   **Output:** The user is redirected to the Dashboard page.

2. **Update Profile**

   **Case 1:** username already exists.
   **Output:** Error message on the screen saying "Username already exists"

   **Case 2:** Some of required fields missing in input.
   **Output:** Model validation errors will be displayed to the user.

   **Case 3:** All input data are valid.
   **Output:** Profile updated successfully.

3. **View Chat.**

   **Case 1:** User is not logged in.
   **Output:** Redirected to the login page with error message "Please login!".

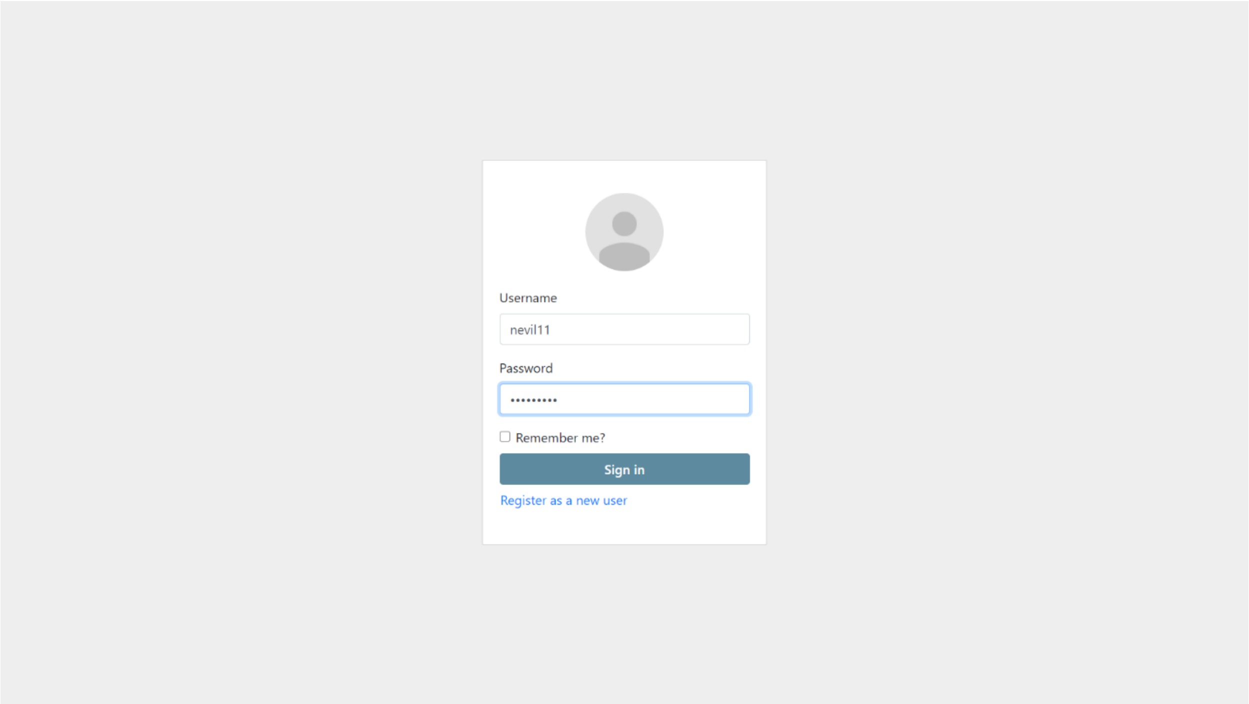
   **Case 2:** If a user exists and has the chat records.
   **Output:** All the chat history will be displayed

**Case 4:** Provided username does not exists in the system.
**Output:** 404 Error.

## Screenshots

### WEBAPI PROJECT:

### Login Page



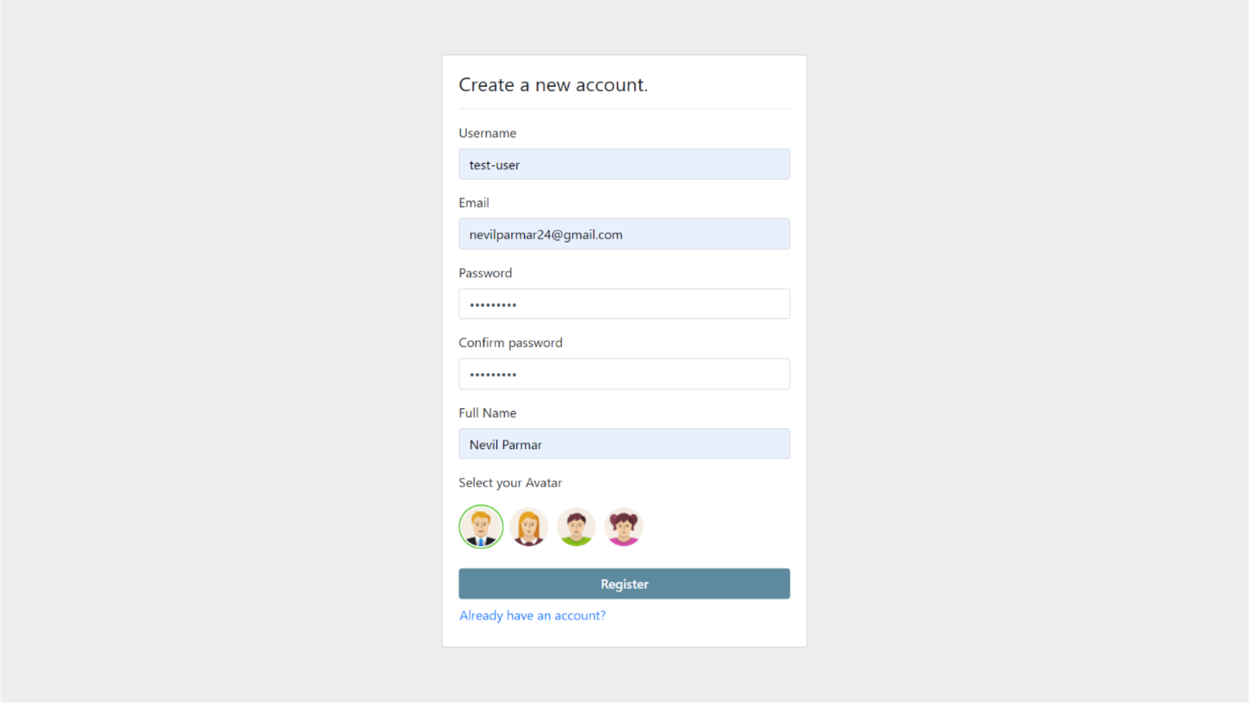
Username

nevil11

Password

•••••••••

## Signup Page



Create a new account.

Username

test-user

Email

nevilparmar24@gmail.com

Password

•••••••••

Confirm password

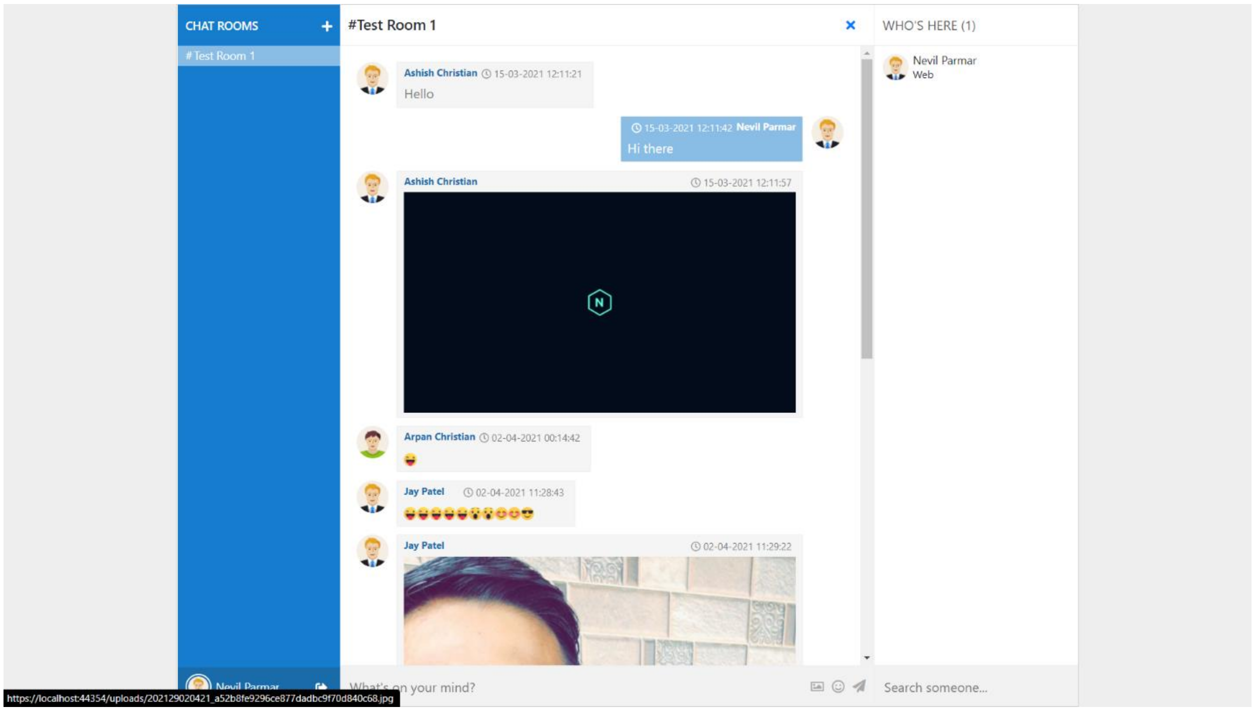
•••••••••

Full Name

Nevil Parmar

Select your Avatar

## Dashboard



Nevil Parmar
Web

Ashish Christian 15-03-2021 12:11:21
Hello

15-03-2021 12:11:42 Nevil Parmar
Hi there

Ashish Christian 15-03-2021 12:11:57

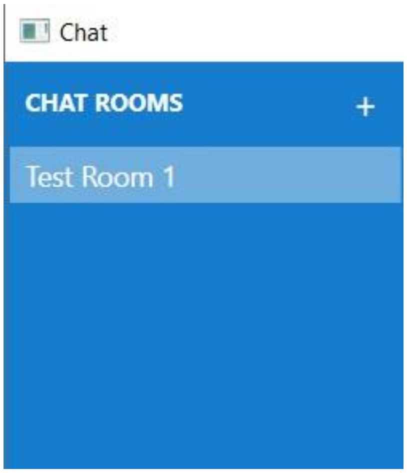
Arpan Christian 02-04-2021 00:14:42
😛

Jay Patel 02-04-2021 11:28:43
😛😛😛😛😛😮😮😊😊😎

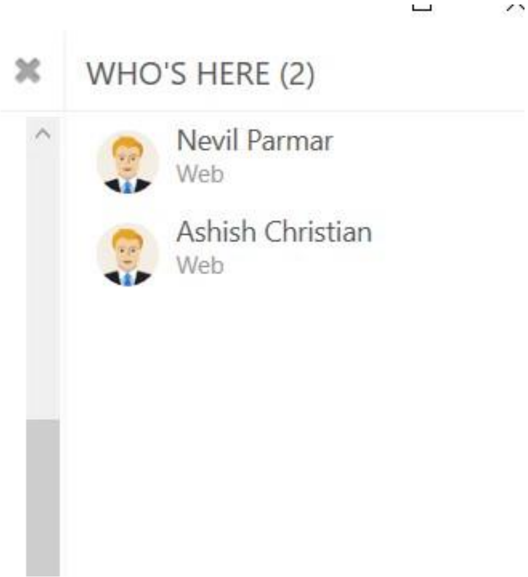
Jay Patel 02-04-2021 11:29:22

Nevil Parmar
What's on your mind?
Search someone...
https://localhost:44354/uploads/202129020421_a52b8fe9296ce877dadbc9f70d840c68.jpg

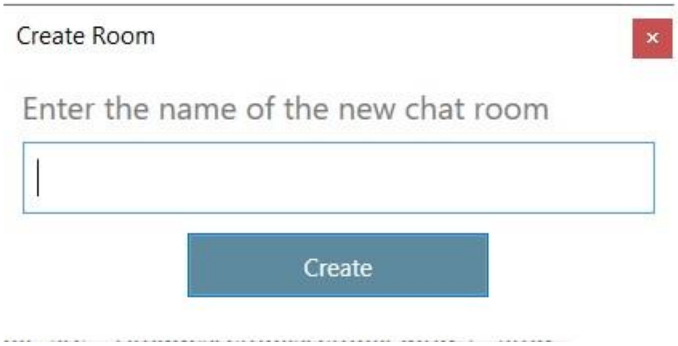
## Chat Rooms



## Online Users

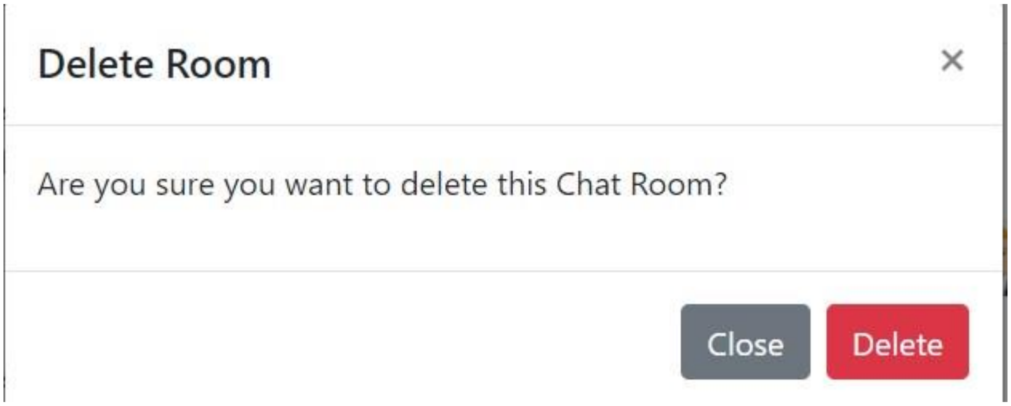


## Create Room



Enter the name of the new chat room

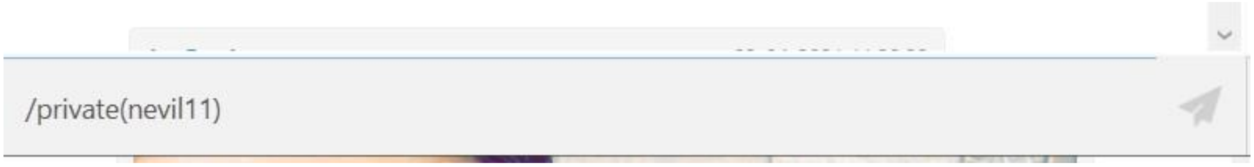
## Delete Room



Delete Room

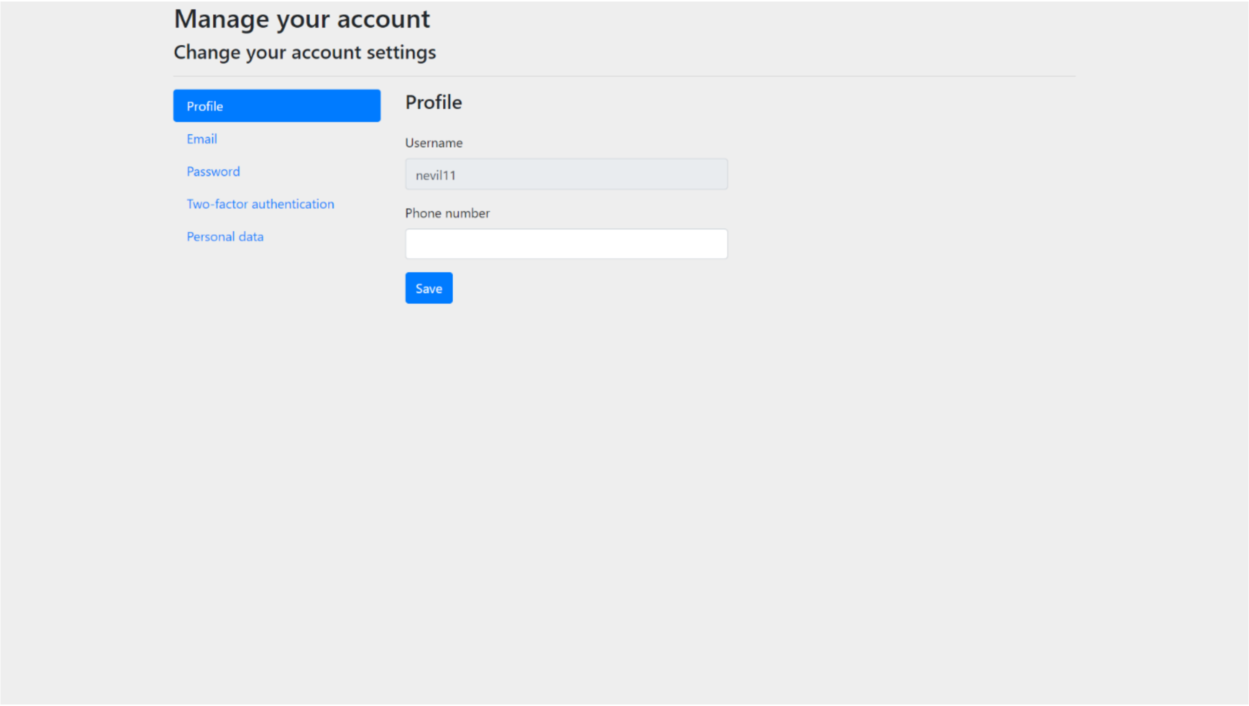
Are you sure you want to delete this Chat Room?

## Chat Private



/private(nevil11)

## Manage Profile



Manage your account

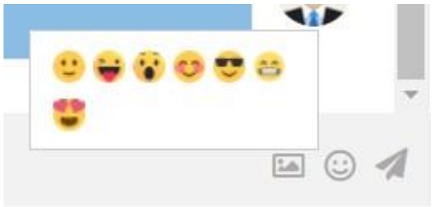
Change your account settings

Profile

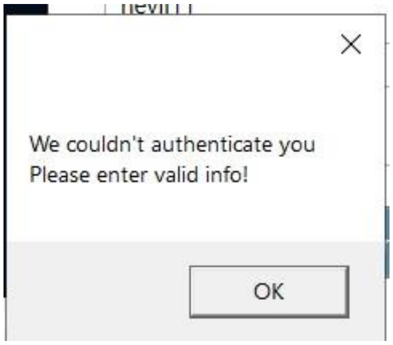
Username: nevil11

Phone number:

## Emoji Support



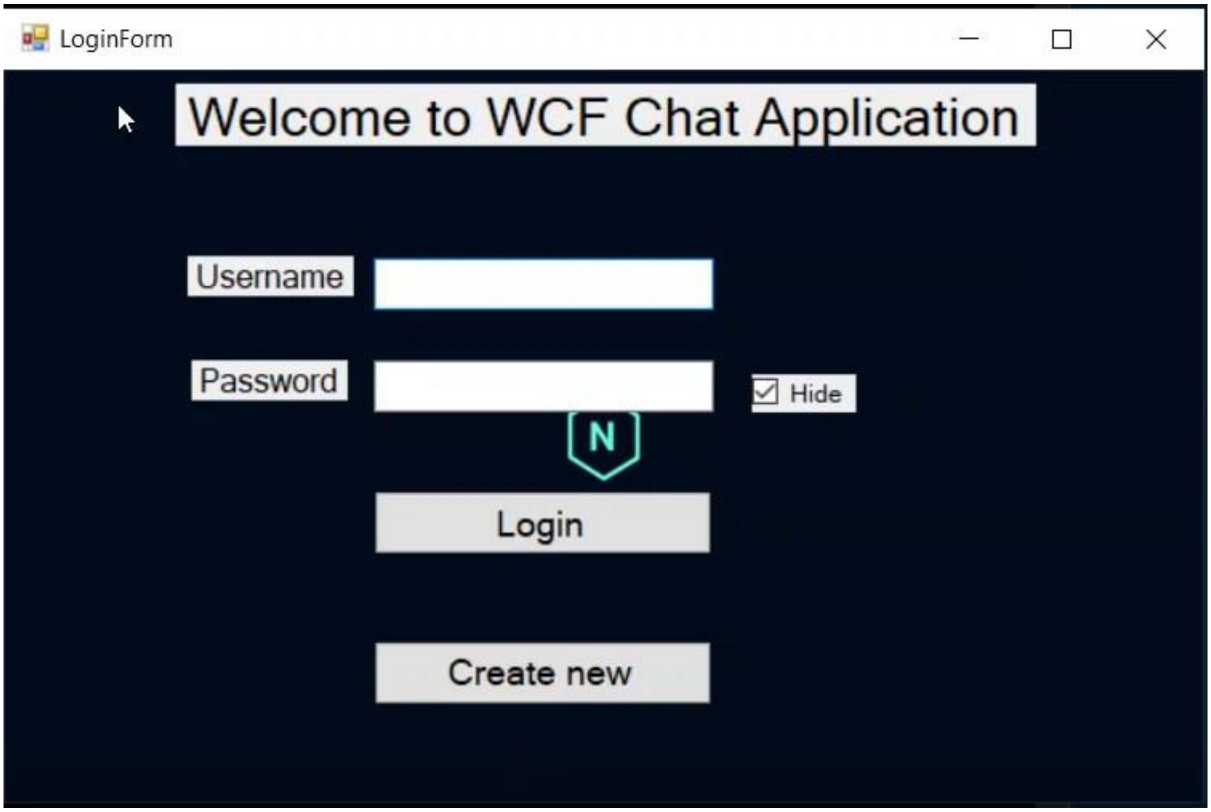
### Error Handling



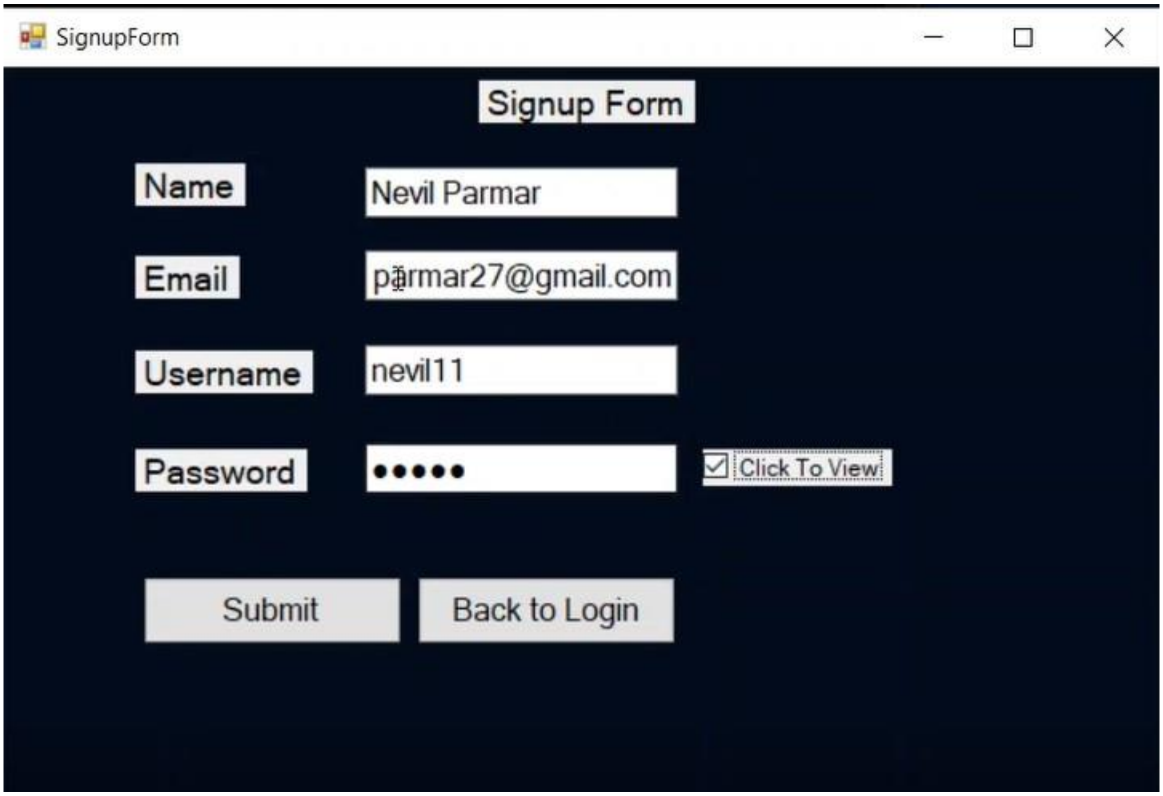
We couldn't authenticate you
Please enter valid info!

### WCF PROJECT:

### Login Page



**Signup Form**



Signup Form

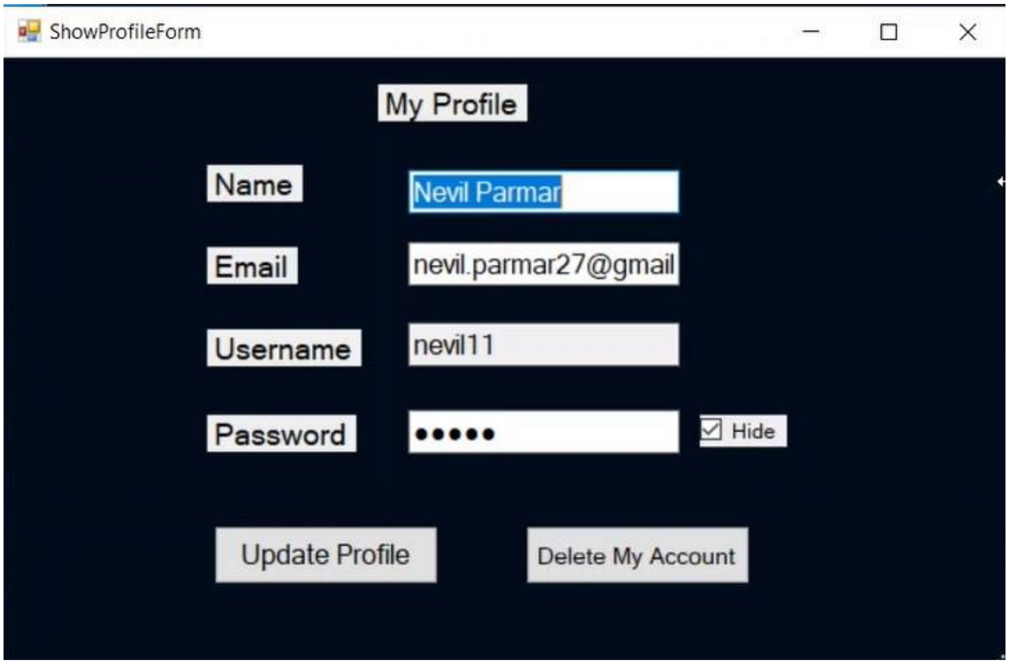
Name: Nevil Parmar

Email: parmar27@gmail.com

Username: nevil11

Password: ●●●●● ☑ Click To View

**Profile Page**



My Profile

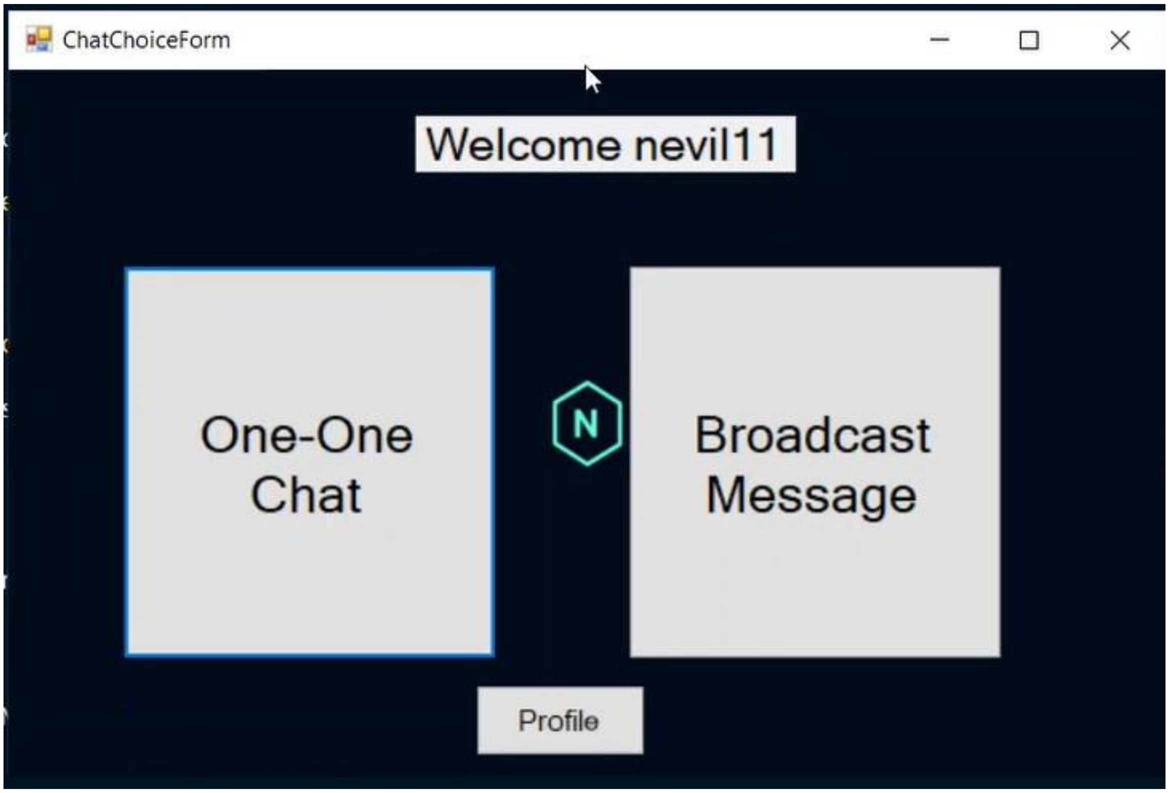
Name: Nevil Parmar

Email: nevil.parmar27@gmail

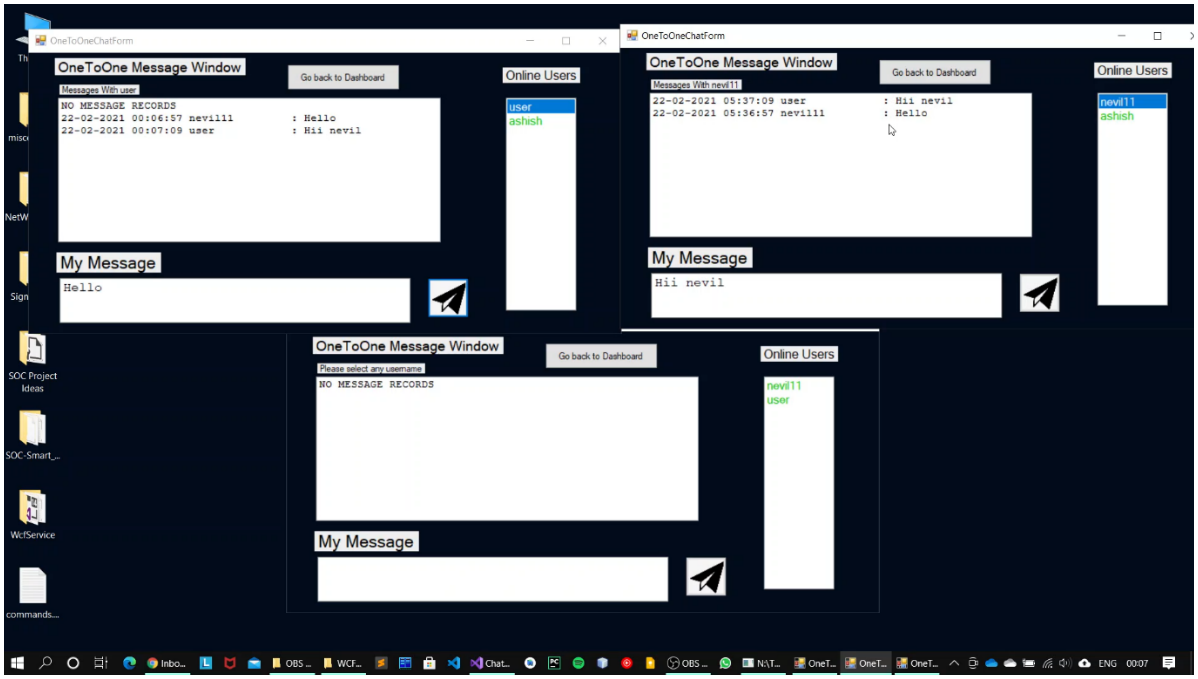
Username: nevil11

Password: ●●●●● ☑ Hide

## Dashboard



## One-One Chat Mode



```
NO MESSAGE RECORDS
22-02-2021 00:06:57 nevil11          : Hello
22-02-2021 00:07:09 user             : Hii nevil
```

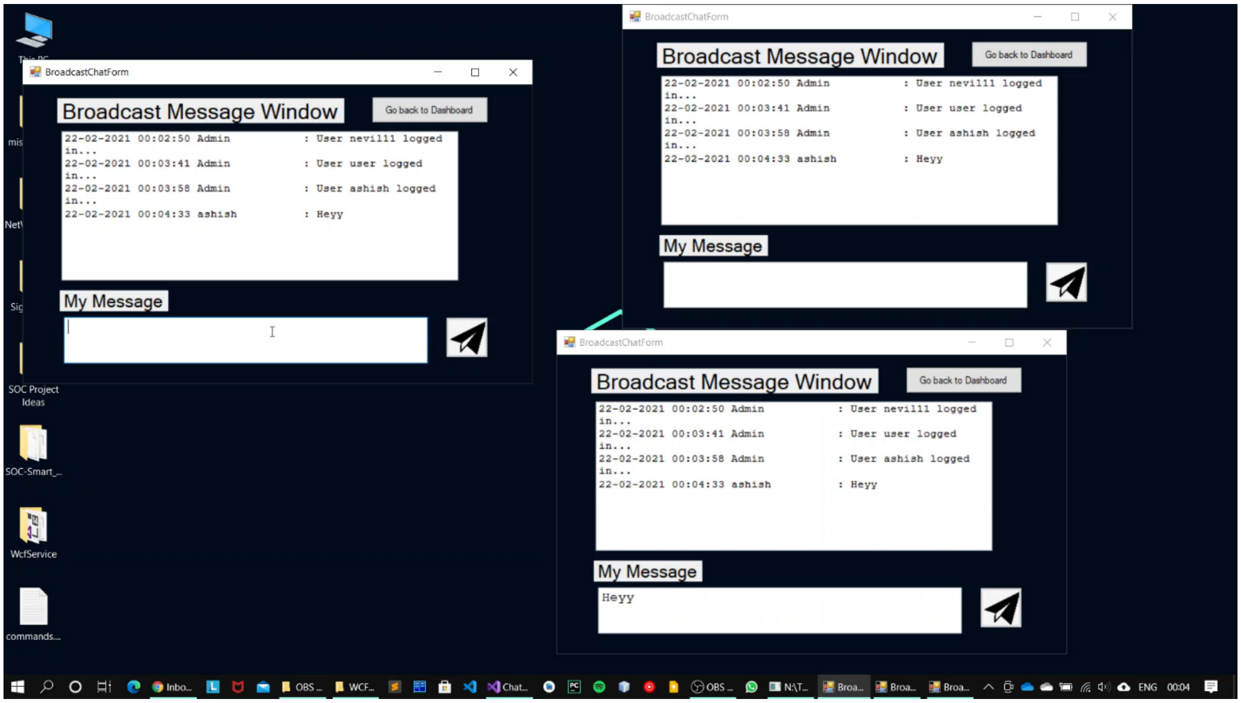
```
Hello
```

```
22-02-2021 05:37:09 user             : Hii nevil
22-02-2021 05:36:57 nevil11          : Hello
```

```
Hii nevil
```

```
NO MESSAGE RECORDS
```

## Broadcast Mode



```
22-02-2021 00:02:50 Admin          : User nevil11 logged
in...
22-02-2021 00:03:41 Admin          : User user logged
in...
22-02-2021 00:03:58 Admin          : User ashish logged
in...
22-02-2021 00:04:33 ashish         : Heyy
```

```
22-02-2021 00:02:50 Admin          : User nevil11 logged
in...
22-02-2021 00:03:41 Admin          : User user logged
in...
22-02-2021 00:03:58 Admin          : User ashish logged
in...
22-02-2021 00:04:33 ashish         : Heyy
```

```
22-02-2021 00:02:50 Admin          : User nevil11 logged
in...
22-02-2021 00:03:41 Admin          : User user logged
in...
22-02-2021 00:03:58 Admin          : User ashish logged
in...
22-02-2021 00:04:33 ashish         : Heyy
```

```
Heyy
```

## Conclusion

### Functionalities implemented successfully:

- Registration
- Login / Logout
- Manage User Profile
- Update user profile
- Forgot password
- Email notifications
- One – One Chat Mode
- Broadcast Chat mode
- Emoji support in chat
- Media Support in chat
- Web + Desktop Clients
- Search User
- Create Channel / Room
- Leave Channel / Room
- Join Channel / Room

## Limitation and Future Extensions

- **Limitations**

- Linux users may only get benefit of web client, as desktop client has only support for windows platform

- **Functionalities not implemented**

- Upload profile picture
- Two Factor Authentication

- **Possible future extensions**

- Provide user profile picture option
- Extend support for Linux users

## Bibliography

- Frameworks:
  **https://docs.microsoft.com/en-us/dotnet/framework/wcf/**
  **https://docs.microsoft.com/en-us/aspnet/core/web-api/?view=aspnetcore-5.0**
  **https://docs.microsoft.com/en-us/dotnet/framework/wcf/**
  **https://docs.microsoft.com/en-us/aspnet/core/tutorials/signalr?view=aspnetcore-5.0&tabs=visual-studio**

- CSS & Bootstrap:
  **https://getbootstrap.com/**

- For debugging:
  **https://stackoverflow.com/**

THANK YOU