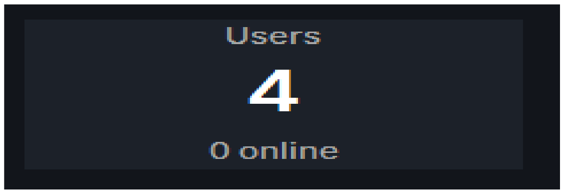
Users

From API we get number of users. We call endpoint `Userlists/count`



```
  UserlListCountDto users = await ServiceHub.Instance.Userlist.GetCountDataAsync();
  int? totalUsers = users.TotalUsers;
  cardUsers.TextCenter = totalUsers.ToString(); //white big number
```

We get online user list from SignalR endpoint `GetClientsSnapshot`

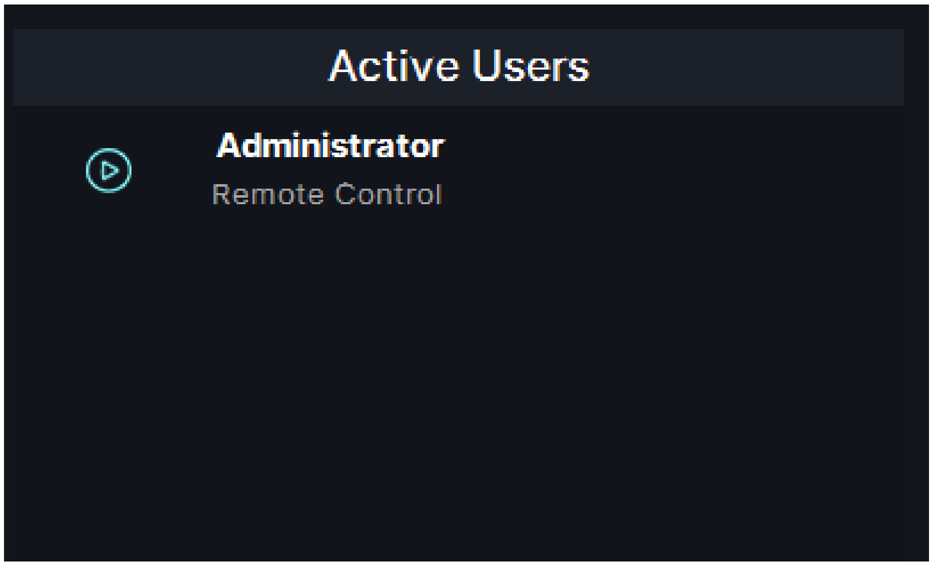
I did like this

```
public async Task<IReadOnlyCollection<ClientSession>> GetClientsOnLine()
     => await
_connection.InvokeAsync<IReadOnlyCollection<ClientSession>>("GetClientsSnapshot");
```

And we get ClientSession. We take only users whose UserId is not null

```
onlineControl1.LoadData(allusers.Where(p => p.UserId != null).ToList());
```

And bind that to grid and we get grid like this in winforms

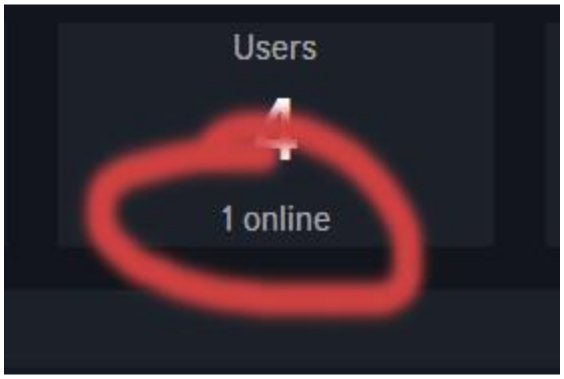


For now image in front depends on application, and it can be like this

Remote Control
Scheduler
CGControl

I have some icons if you want, or put what you think is nice. So we don't use user's image for now, no need. So just Name and Application. Just remember, you get data as RemoteControl, and you must then put in grid Remote Control (with space between)

Once grid is filled, then count unique users, since 1 user can be connected to multiple apps, and show in card, text down



```
var uniqueUsers = Clients
    .Select(x => x.UserId)
    .Where(id => !string.IsNullOrWhiteSpace(id))
    .Distinct()
    .Count();
```

```
And set text cardUsers.TextDown = uniqueUsers.ToString()+ " online";
```

```
In signalR, change status of users like this
```

```
case ServiceMessages.ClientStatusChanged:
     if (comm.group != null && comm.group.Length > 0)
     {
         switch (comm.group.FromSignal<ClientType>())
         {
             case ClientType.PlayoutServer:
                 playoutControl1.PlayoutStatusChanged(comm);
                 GetPlayoutNo();
                 break;
             case ClientType.RemoteControl:
             case ClientType.Scheduler:
             case ClientType.CgControl:
                 onlineControl1.SetStatus(comm);
                 break;
         }
     }
     break;
```

```
public void SetStatus(CommandDto online)
{
    if (online?.data == null)
        return;

    var status = online.data.ToString();
    var existing = Clients.FirstOrDefault(p => p.ClientId == online.clientId);

    switch (status)
    {
        case "0": // offline
            if (existing != null)
                Clients.Remove(existing);
            break;

        case "1": // online / on-air
            if (existing == null)
            {
                Clients.Add(new ClientSession
                {
                    ClientId = online.clientId,
                    GroupId = online.group,
                    UserId = online.userid,
                    UserName = online.playoutId,
                    Name = online.channelId
                });
            }
            else
            {
                //  existing.GroupId = online.group;
            }
            break;
    }

    GetActiveClientsNo();
}
```