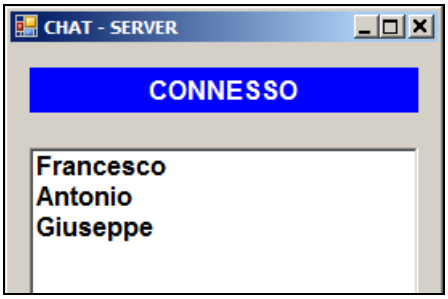
Esempio Pilota: **ChatCollettivaSemplice**

*Realizzare un'**Applicazione di Rete** che consenta a più **Clients** di dialogare fra loro in una Chat Collettiva. I Clients si connettono tutti ad uno stesso **Server** che riceve dai Clients i messaggi da inviare e provvede a ritrasmetterli a tutti gli altri Clients. I Clients possono **Disconnettersi** in qualsiasi momento, seguendo una corretta procedura di disconnessione*

Applicazione Server

Definizione della Form del Server



Codice Form dell'applicazione ChatCollettivaServer

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;
using System.Net.Sockets;
using System.Text;
using System.Text.Json;
using System.Text.Json.Nodes;

namespace ChatCollettivaServer
{
   public partial class frmServer : Form
   {
      public frmServer()
      {
         InitializeComponent();
      }

      /* PROTOCOLLO:
         ---------------------------------------------------------
         Operazione"000" (CLN -> SRV) - invio nominativo al server:
            ["Operazione"] = "000"
            ["Nominativo"] = <stringa>
         ---------------------------------------------------------
         Operazione"001" (CLN -> SRV) - invio messaggio al server:
            ["Operazione"] = "001"
            ["Messaggio"] = <stringa>
         ---------------------------------------------------------
         Operazione"002" (SRV -> CLN) - il server gira il messaggio
                                        a tutti i client:
            ["Operazione"] = "002"
            ["Nominativo"] = <stringa>
            ["Messaggio"] = <stringa>
         ---------------------------------------------------------
         Operazione"003" (CLN -> SRV) - il client comunica al server
                                        che si sta disconnettendo:
            ["Operazione"] = "003"
      */

      TcpListener LST = new TcpListener(57);
      List<Utente> ListaUtenti = new List<Utente>( );

      private void frmServer_Load(object sender, EventArgs e)
      {
         LST.Start();
         tmrAC.Start();
      }

      private void tmrAC_Tick(object sender, EventArgs e)
      {
         if (LST.Pending())
         {
            TcpClient NuovaCN = LST.AcceptTcpClient();

            Utente NuovoUtente = new Utente( NuovaCN );
            ListaUtenti.Add ( NuovoUtente );

            if (tmrAD.Enabled == false)
               tmrAD.Start();

            lblStato.Text = "CONNESSO";
         }
      }

      private void tmrAD_Tick(object sender, EventArgs e)
      {
         Utente UtenteDaDisconnettere = null;

         foreach (Utente UtenteInEsame in ListaUtenti)
         {
            JsonObject JO = Ricevi(UtenteInEsame.CN);

            if (JO != null)
            {
               switch ( (string)JO["Operazione"] )
               {
                  case "000":

                     UtenteInEsame.Nominativo =
                                    (string)JO["Nominativo"];
                     lstElencoUtentiConnessi
                              .Items.Add(UtenteInEsame.Nominativo);
                     break;

                  case "001":

                     string MessaggioRicevuto =
                                    (string)JO["Messaggio"];

                     foreach (Utente UtenteACuiInviare in ListaUtenti)
                     {
                        JsonObject JOmess = new JsonObject()
                        {
                           ["Operazione"] = "002",
                           ["Nominativo"] = UtenteInEsame.Nominativo,
                           ["Messaggio"] = MessaggioRicevuto
                        };
```

```
                Invia (UtenteACuiInviare.CN, JOmess);
            }
            break;

        case "003":
            UtenteDaDisconnettere = UtenteInEsame;
            break;

        default: break;
        }
    }
}

// Chiusura ed Eliminazione della Connessione da Chiudere...

if (UtenteDaDisconnettere != null)
{
    lstElencoUtentiConnessi.Items.Remove
                    (UtenteDaDisconnettere.Nominativo);

    UtenteDaDisconnettere.CN.Close();
    ListaUtenti.Remove ( UtenteDaDisconnettere );

    if (ListaUtenti.Count == 0)
        lblStato.Text = "IN ATTESA...";
}
}
```

```
// Sottoprogrammi INVIA e RICEVI con JSON ...
```

```
private JsonObject Ricevi(TcpClient CN)
{
    JsonObject JO = null;
    if (CN.Connected)
    {
        if (CN.Available > 0)
        {
            NetworkStream NS = CN.GetStream();

            int NumeroDiByteRicevuti = CN.Available;
            byte[] VetDati = new byte[NumeroDiByteRicevuti];

            NS.Read(VetDati, 0, NumeroDiByteRicevuti);
            string StringaJson = UnicodeEncoding.Unicode.GetString(VetDati);

            JO = JsonSerializer.Deserialize<JsonObject>(StringaJson);
        }
    }
    return JO;
}
```

```
private void Invia(TcpClient CN, JsonObject JO)
{
    if (CN.Connected)
    {
        string StringaJSON = JsonSerializer.Serialize(JO);
        NetworkStream NS = CN.GetStream();

        byte[] VetDati = UnicodeEncoding.Unicode.GetBytes(StringaJSON);
        NS.Write(VetDati, 0, VetDati.Length);
    }
}
}
}
```

Codice della Classe "Utente"

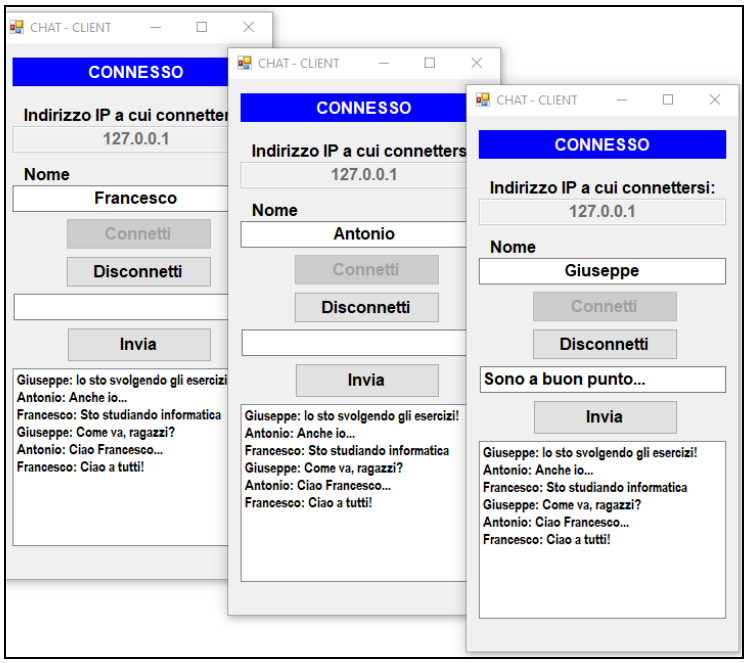
```
using System.Net.Sockets;

namespace ChatCollettivaServer
{
    public class Utente
    {
        public TcpClient CN;
        public string Nominativo;

        public Utente (TcpClient NuovaCN)
        {
            CN = NuovaCN;
            Nominativo = "";
        }
    }
}
```

Applicazione ChatCollettivaClient

Form dell'applicazione ChatCollettivaClient



Codice Form dell'applicazione ChatCollettivaClient

```
using System;
using System.Windows.Forms;

using System.Net.Sockets;
using System.Text;
using System.Text.Json;
using System.Text.Json.Nodes;

namespace ChatCollettivaClient
{
   public partial class frmClient : Form
   {
      public frmClient()
      {
         InitializeComponent();
      }

      TcpClient CN;
      string Nominativo;
```

```
      private void plsConnetti_Click(object sender, EventArgs e)
      {
         CN = new TcpClient();

         try
         { CN.Connect(txtIndirizzoIP.Text, 57); }
         catch
         { MessageBox.Show("Connessione fallita."); }

         if (CN.Connected)
         {
            Nominativo = txtMioNome.Text;

            JsonObject JO = new JsonObject()
            {
               ["Operazione"] = "000",
               ["Nominativo"] = Nominativo
            };

            Invia (CN, JO);

            tmrAD.Start();
            ImpostaFormPerLaChat();
         }
      }
```

```
      private void tmrAD_Tick(object sender, EventArgs e)
      {
         JsonObject JO = Ricevi(CN);

         if (JO != null)
         {
            switch ( (string)JO["Operazione"] )
            {
               case "002":

                  string Nominativo = (string)JO["Nominativo"];
                  string Messaggio = (string)JO["Messaggio"];

                  lstChat.Items.Insert (0, Nominativo + ": " +
                                           Messaggio);
                  break;

               default: break;
            }
         }
      }
```

```
      private void plsInvia_Click(object sender, EventArgs e)
      {
         JsonObject JO = new JsonObject()
         {
            ["Operazione"] = "001",
            ["Nominativo"] = Nominativo,
            ["Messaggio"] = txtMessaggio.Text
         };

         Invia (CN, JO);

         txtMessaggio.Clear();
         txtMessaggio.Focus();
      }
```

```
      private void PlsDisconnetti_Click
                                 (object sender, EventArgs e)
      {
         JsonObject   = new JsonObject()
         { ["Operazione"] = "003" };

         Invia (CN, JO);
```

```
        CN.Close();
        ImpostaFormPerLaConnessione();
    }
```

```
    private void ImpostaFormPerLaChat()
    {
        lblStato.Text = "CONNESSO";
        txtIndirizzoIP.Enabled = false;
        txtMessaggio.Enabled = true;
        plsConnetti.Enabled = false;
        plsDisconnetti.Enabled = true;
        plsInvia.Enabled = true;
    }
```

```
    private void ImpostaFormPerLaConnessione()
    {
        lblStato.Text = "NON CONNESSO";
        txtIndirizzoIP.Enabled = true;
        txtMessaggio.Enabled = false;
        plsConnetti.Enabled = true;
        plsDisconnetti.Enabled = false;
        plsInvia.Enabled = false;
        lstChat.Items.Clear();
    }

    // Sottoprogrammi INVIA e RICEVI con JSON ...

    ... vedi applicazione server ...

  }
}
```