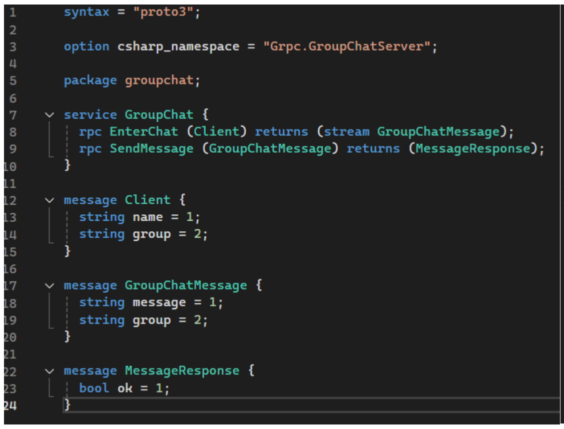
Wagonblast, Gregory
CS2510

Assignment 1: Group RPC Communication

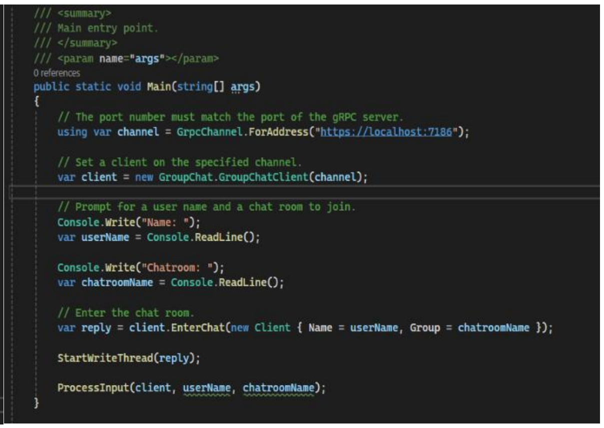
Project Description: This project is based on a .NET Core gRPC client and an ASP.NET Core gRPC Server. Link → Create a .NET Core gRPC client and server in ASP.NET Core | Microsoft Learn. Microsoft complemented the gRPC documentation well with their own C# tutorial. The project was extended to implement a group chat with n-clients and 1-server. This project is a simple console application. Clients are able to input a username and a chatroom name. The chatroom name controls what messages they (the client) receives. The program allows clients to send/receive messages from the server asynchronously. When a client joins a chatroom, they receive all of the unread messages that are stored in a concurrent dictionary, if there are any unread message.

Project Implementation:

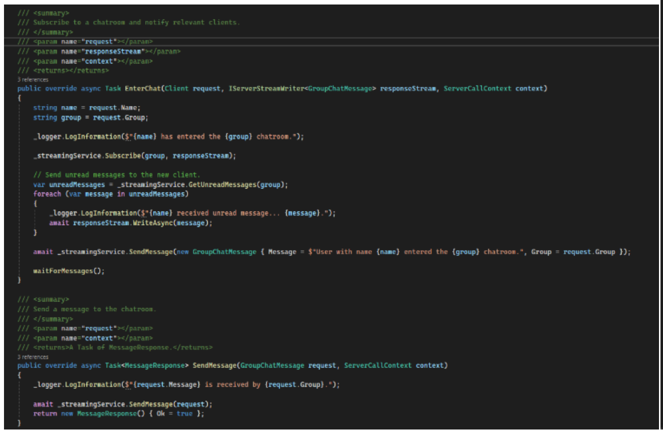
(Top Left) Proto file layout and (Top Right) Client main method, a write thread is started and input s processed from the terminal while it is active. (Bottom Left) Server has the implementation of the protos that were defined. (Bottom Right) This service helps the server implement chatrooms and messaging.



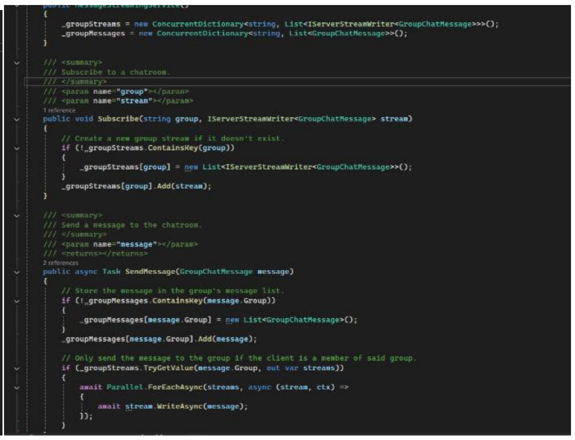
```
syntax = "proto3";

option csharp_namespace = "Grpc.GroupChatServer";

package groupchat;

service GroupChat {
    rpc EnterChat (Client) returns (stream GroupChatMessage);
    rpc SendMessage (GroupChatMessage) returns (MessageResponse);
}

message Client {
    string name = 1;
    string group = 2;
}

message GroupChatMessage {
    string message = 1;
    string group = 2;
}

message MessageResponse {
    bool ok = 1;
}
```



```
/// <summary>
/// Main entry point.
/// </summary>
/// <param name="args"></param>
0 references
public static void Main(string[] args)
{
    // The port number must match the port of the gRPC server.
    using var channel = GrpcChannel.ForAddress("https://localhost:7186");

    // Set a client on the specified channel.
    var client = new GroupChat.GroupChatClient(channel);

    // Prompt for a user name and a chat room to join.
    Console.Write("Name: ");
    var userName = Console.ReadLine();

    Console.Write("Chatroom: ");
    var chatroomName = Console.ReadLine();

    // Enter the chat room.
    var reply = client.EnterChat(new Client { Name = userName, Group = chatroomName });

    StartWriteThread(reply);

    ProcessInput(client, userName, chatroomName);
}
```



```
/// <summary>
/// Subscribe to a chatroom and notify relevant clients.
/// </summary>
/// <param name="request"></param>
/// <param name="responseStream"></param>
/// <param name="context"></param>
/// <returns></returns>
3 references
public override async Task EnterChat(Client request, IServerStreamWriter<GroupChatMessage> responseStream, ServerCallContext context)
{
    string name = request.Name;
    string group = request.Group;

    _logger.LogInformation($"{name} has entered the {group} chatroom.");

    _streamingService.Subscribe(group, responseStream);

    // Send unread messages to the new client.
    var unreadMessages = _streamingService.GetUnreadMessages(group);
    foreach (var message in unreadMessages)
    {
        _logger.LogInformation($"{name} received unread message... {message}.");
        await responseStream.WriteAsync(message);
    }

    await _streamingService.SendMessage(new GroupChatMessage { Message = $"User with name {name} entered the {group} chatroom.", Group = request.Group });

    waitForMessages();
}

/// <summary>
/// Send a message to the chatroom.
/// </summary>
/// <param name="request"></param>
/// <param name="context"></param>
/// <returns>A Task of MessageResponse.</returns>
3 references
public override async Task<MessageResponse> SendMessage(GroupChatMessage request, ServerCallContext context)
{
    _logger.LogInformation($"{request.Message} is received by {request.Group}.");

    await _streamingService.SendMessage(request);
    return new MessageResponse() { Ok = true };
}
```

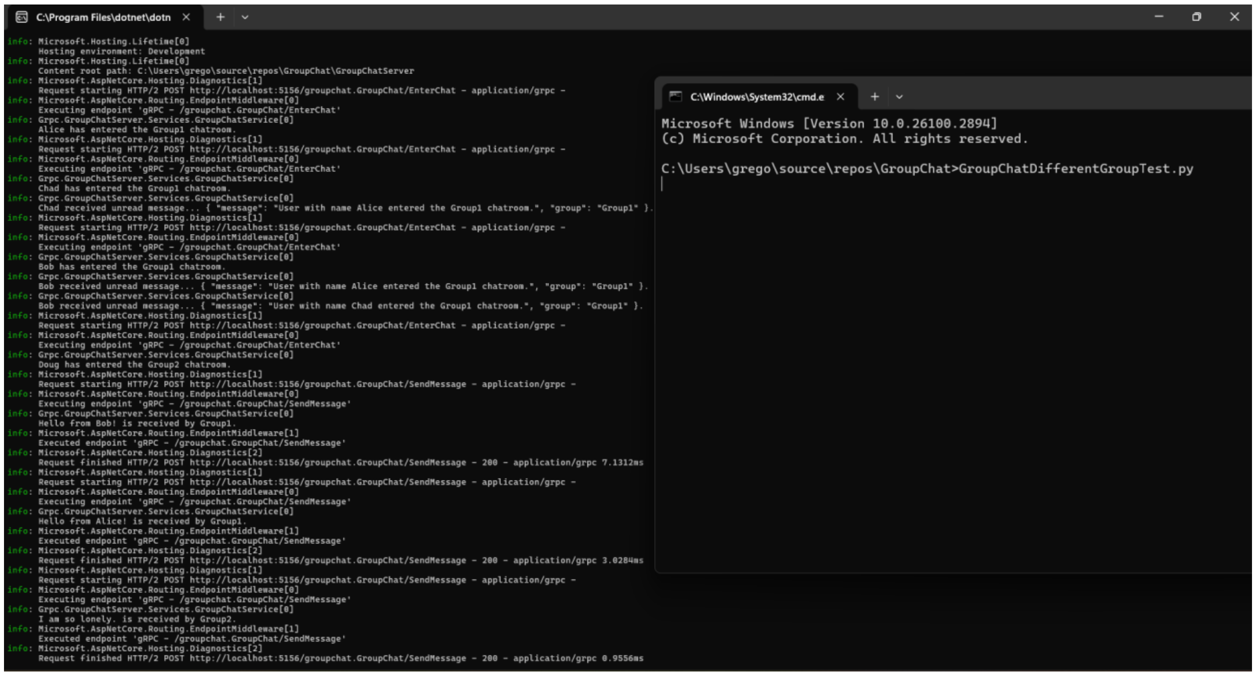


```
{
    _groupStreams = new ConcurrentDictionary<string, List<IServerStreamWriter<GroupChatMessage>>>();
    _groupMessages = new ConcurrentDictionary<string, List<GroupChatMessage>>();
}

/// <summary>
/// Subscribe to a chatroom.
/// </summary>
/// <param name="group"></param>
/// <param name="stream"></param>
1 reference
public void Subscribe(string group, IServerStreamWriter<GroupChatMessage> stream)
{
    // Create a new group stream if it doesn't exist.
    if (!_groupStreams.ContainsKey(group))
    {
        _groupStreams[group] = new List<IServerStreamWriter<GroupChatMessage>>();
    }
    _groupStreams[group].Add(stream);
}

/// <summary>
/// Send a message to the chatroom.
/// </summary>
/// <param name="message"></param>
/// <returns></returns>
2 references
public async Task SendMessage(GroupChatMessage message)
{
    // Store the message in the group's message list.
    if (!_groupMessages.ContainsKey(message.Group))
    {
        _groupMessages[message.Group] = new List<GroupChatMessage>();
    }
    _groupMessages[message.Group].Add(message);

    // Only send the message to the group if the client is a member of said group.
    if (_groupStreams.TryGetValue(message.Group, out var streams))
    {
        await Parallel.ForEachAsync(streams, async (stream, ctx) =>
        {
            await stream.WriteAsync(message);
        });
    }
```

Wagonblast, Gregory
CS2510

Project Testing:

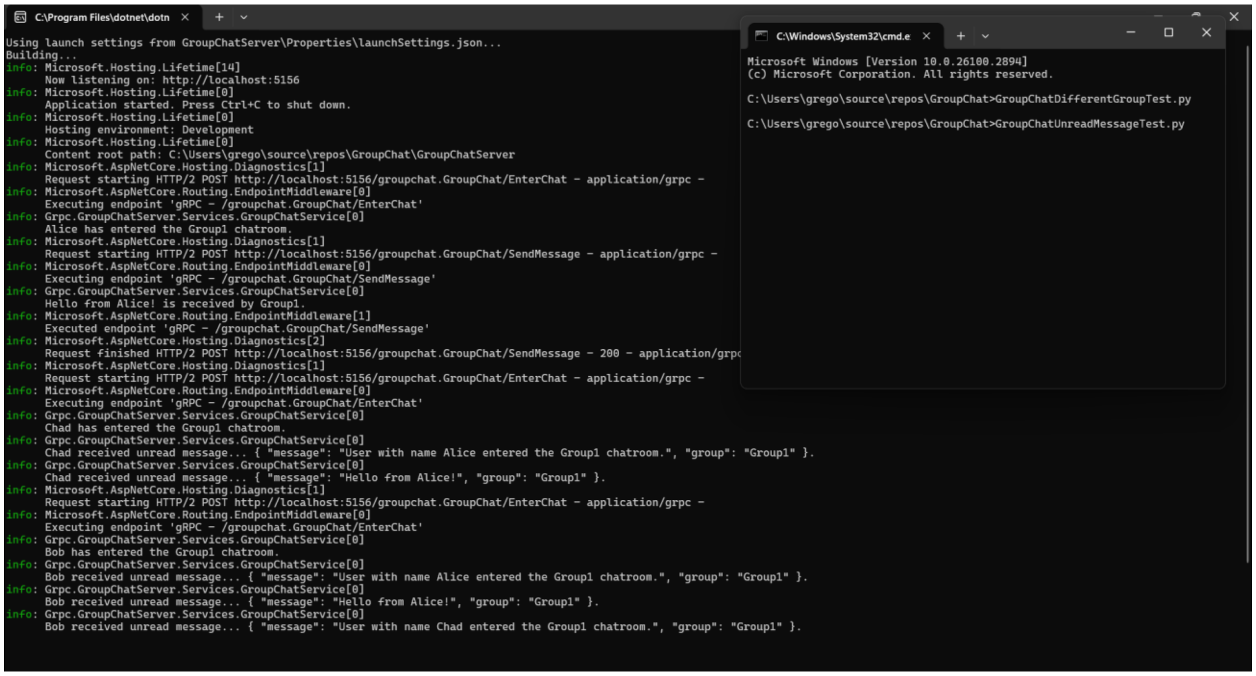
Testing example running the GroupChatDifferentGroupTest.py python script. This shows a user not getting messages when in a separate chatroom. All logs are output to the server.



```
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\grego\source\repos\GroupChat\GroupChatServer
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/EnterChat - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/EnterChat'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Alice has entered the Group1 chatroom.
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/EnterChat - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/EnterChat'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Chad has entered the Group1 chatroom.
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Chad received unread message... { "message": "User with name Alice entered the Group1 chatroom.", "group": "Group1" }.
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/EnterChat - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/EnterChat'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Bob has entered the Group1 chatroom.
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Bob received unread message... { "message": "User with name Alice entered the Group1 chatroom.", "group": "Group1" }.
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Bob received unread message... { "message": "User with name Chad entered the Group1 chatroom.", "group": "Group1" }.
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/EnterChat - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/EnterChat'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Doug has entered the Group2 chatroom.
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/SendMessage - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/SendMessage'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Hello from Bob! is received by Group1.
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[1]
      Executed endpoint 'gRPC - /groupchat.GroupChat/SendMessage'
info: Microsoft.AspNetCore.Hosting.Diagnostics[2]
      Request finished HTTP/2 POST http://localhost:5156/groupchat.GroupChat/SendMessage - 200 - application/grpc 7.1312ms
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/SendMessage - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/SendMessage'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Hello from Alice! is received by Group1.
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[1]
      Executed endpoint 'gRPC - /groupchat.GroupChat/SendMessage'
info: Microsoft.AspNetCore.Hosting.Diagnostics[2]
      Request finished HTTP/2 POST http://localhost:5156/groupchat.GroupChat/SendMessage - 200 - application/grpc 3.0284ms
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/SendMessage - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/SendMessage'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      I am so lonely. is received by Group2.
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[1]
      Executed endpoint 'gRPC - /groupchat.GroupChat/SendMessage'
info: Microsoft.AspNetCore.Hosting.Diagnostics[2]
      Request finished HTTP/2 POST http://localhost:5156/groupchat.GroupChat/SendMessage - 200 - application/grpc 0.9556ms
```

```
Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

C:\Users\grego\source\repos\GroupChat>GroupChatDifferentGroupTest.py
```

Testing example running the GroupChatUnreadMessageTest.py python script. This shows a user joining the chatroom and receiving unread messages. All logs are output to the server.

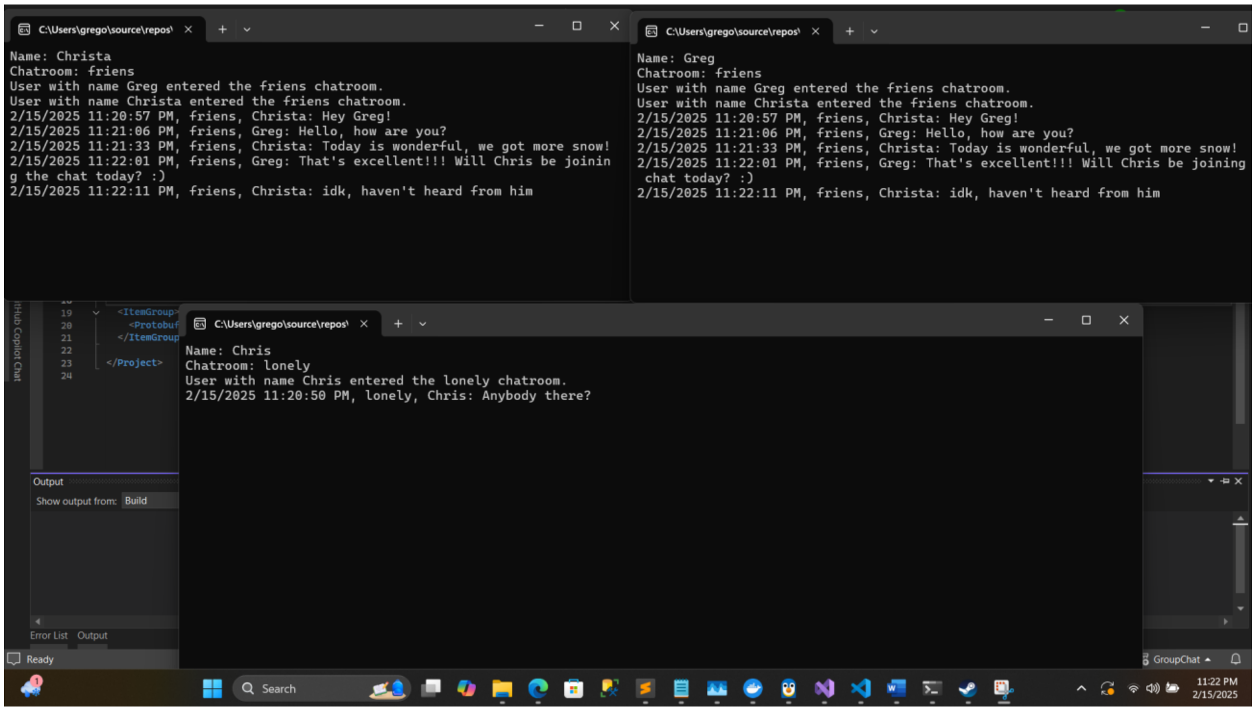


```
Using launch settings from GroupChatServer\Properties\launchSettings.json...
Building...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5156
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\grego\source\repos\GroupChat\GroupChatServer
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/EnterChat - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/EnterChat'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Alice has entered the Group1 chatroom.
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/SendMessage - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/SendMessage'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Hello from Alice! is received by Group1.
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[1]
      Executed endpoint 'gRPC - /groupchat.GroupChat/SendMessage'
info: Microsoft.AspNetCore.Hosting.Diagnostics[2]
      Request finished HTTP/2 POST http://localhost:5156/groupchat.GroupChat/SendMessage - 200 - application/grpc
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/EnterChat - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/EnterChat'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Chad has entered the Group1 chatroom.
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Chad received unread message... { "message": "User with name Alice entered the Group1 chatroom.", "group": "Group1" }.
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Chad received unread message... { "message": "Hello from Alice!", "group": "Group1" }.
info: Microsoft.AspNetCore.Hosting.Diagnostics[1]
      Request starting HTTP/2 POST http://localhost:5156/groupchat.GroupChat/EnterChat - application/grpc -
info: Microsoft.AspNetCore.Routing.EndpointMiddleware[0]
      Executing endpoint 'gRPC - /groupchat.GroupChat/EnterChat'
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Bob has entered the Group1 chatroom.
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Bob received unread message... { "message": "User with name Alice entered the Group1 chatroom.", "group": "Group1" }.
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Bob received unread message... { "message": "Hello from Alice!", "group": "Group1" }.
info: Grpc.GroupChatServer.Services.GroupChatService[0]
      Bob received unread message... { "message": "User with name Chad entered the Group1 chatroom.", "group": "Group1" }.
```

```
Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

C:\Users\grego\source\repos\GroupChat>GroupChatDifferentGroupTest.py

C:\Users\grego\source\repos\GroupChat>GroupChatUnreadMessageTest.py
```

Wagonblast, Gregory
CS2510

Testing example running 3 clients. This shows how the chatrooms control what messages can be seen.

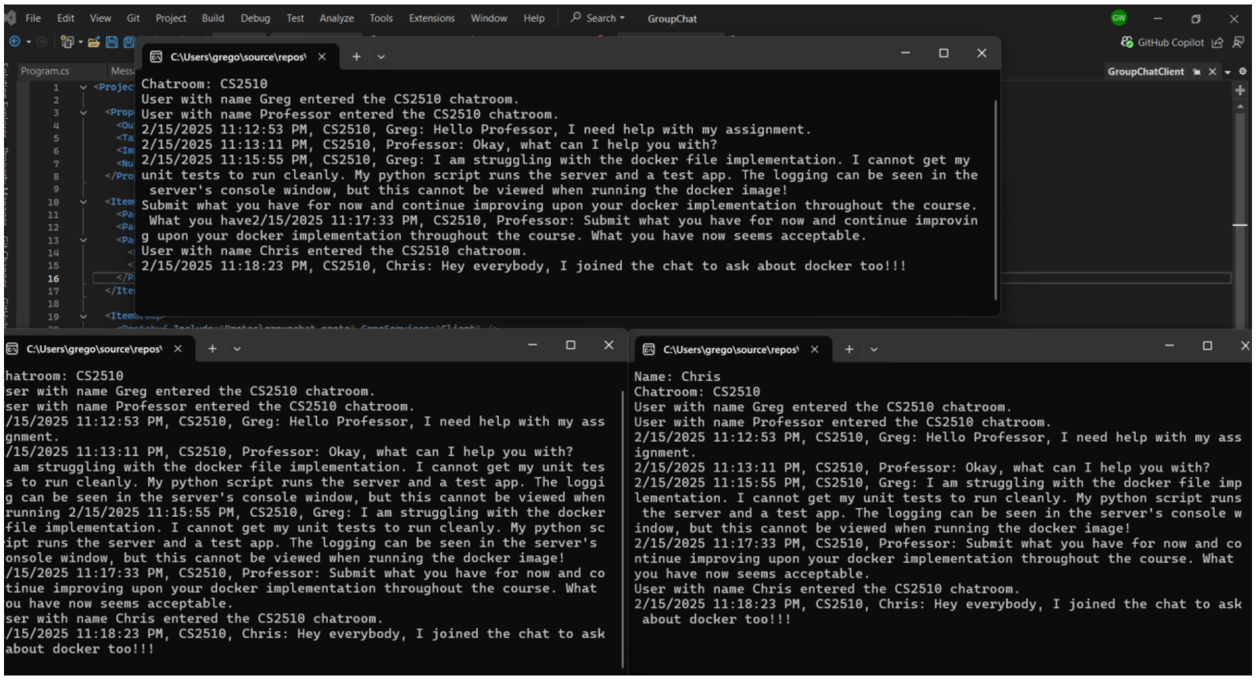


```
Name: Christa
Chatroom: friens
User with name Greg entered the friens chatroom.
User with name Christa entered the friens chatroom.
2/15/2025 11:20:57 PM, friens, Christa: Hey Greg!
2/15/2025 11:21:06 PM, friens, Greg: Hello, how are you?
2/15/2025 11:21:33 PM, friens, Christa: Today is wonderful, we got more snow!
2/15/2025 11:22:01 PM, friens, Greg: That's excellent!!! Will Chris be joinin
g the chat today? :)
2/15/2025 11:22:11 PM, friens, Christa: idk, haven't heard from him
```

```
Name: Greg
Chatroom: friens
User with name Greg entered the friens chatroom.
User with name Christa entered the friens chatroom.
2/15/2025 11:20:57 PM, friens, Christa: Hey Greg!
2/15/2025 11:21:06 PM, friens, Greg: Hello, how are you?
2/15/2025 11:21:33 PM, friens, Christa: Today is wonderful, we got more snow!
2/15/2025 11:22:01 PM, friens, Greg: That's excellent!!! Will Chris be joining
 chat today? :)
2/15/2025 11:22:11 PM, friens, Christa: idk, haven't heard from him
```

```
Name: Chris
Chatroom: lonely
User with name Chris entered the lonely chatroom.
2/15/2025 11:20:50 PM, lonely, Chris: Anybody there?
```

Testing example running 3 clients. This shows how a user receives unread messages when joining a chatroom.



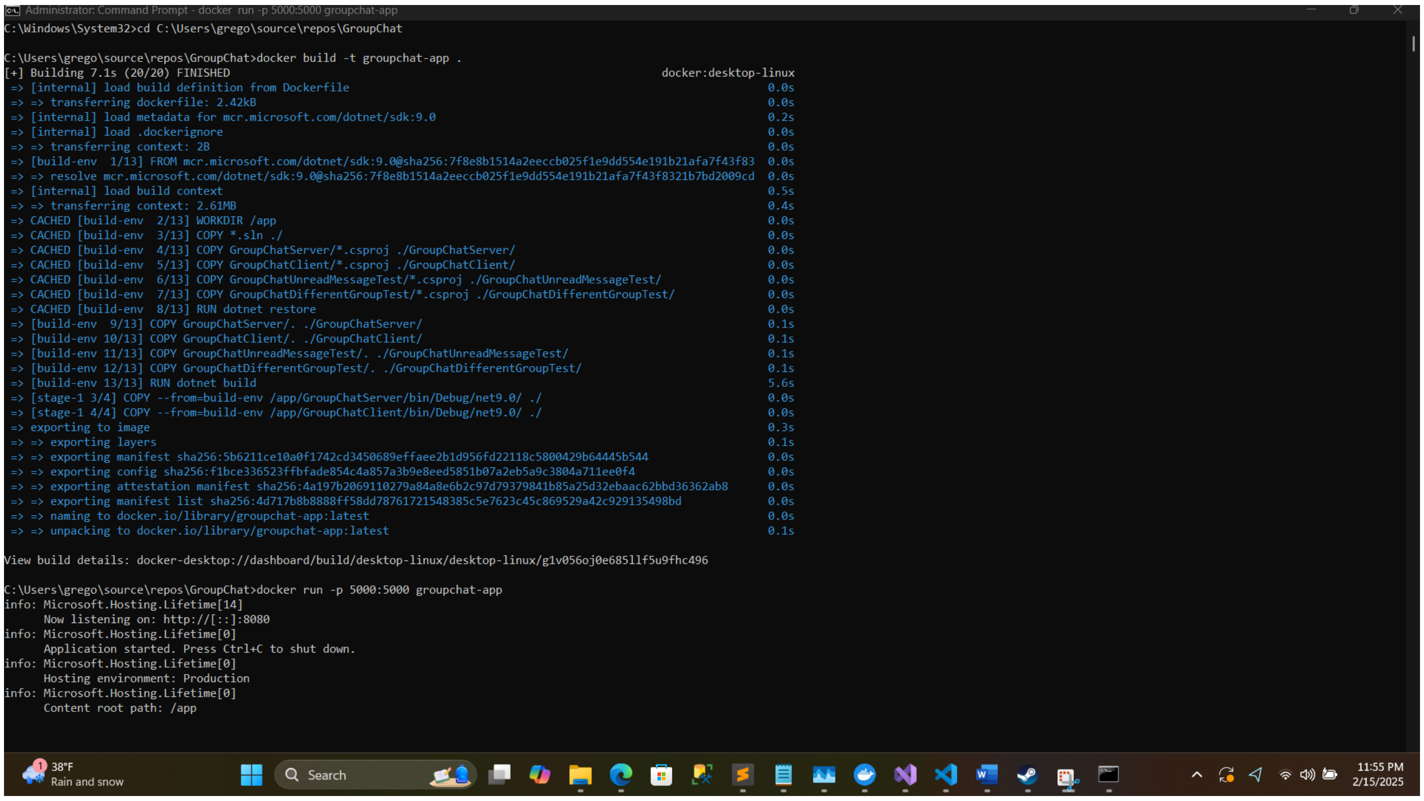
```
Chatroom: CS2510
User with name Greg entered the CS2510 chatroom.
User with name Professor entered the CS2510 chatroom.
2/15/2025 11:12:53 PM, CS2510, Greg: Hello Professor, I need help with my assignment.
2/15/2025 11:13:11 PM, CS2510, Professor: Okay, what can I help you with?
2/15/2025 11:15:55 PM, CS2510, Greg: I am struggling with the docker file implementation. I cannot get my
unit tests to run cleanly. My python script runs the server and a test app. The logging can be seen in the
 server's console window, but this cannot be viewed when running the docker image!
Submit what you have for now and continue improving upon your docker implementation throughout the course.
 What you have2/15/2025 11:17:33 PM, CS2510, Professor: Submit what you have for now and continue improvin
g upon your docker implementation throughout the course. What you have now seems acceptable.
User with name Chris entered the CS2510 chatroom.
2/15/2025 11:18:23 PM, CS2510, Chris: Hey everybody, I joined the chat to ask about docker too!!!
```

```
hatroom: CS2510
ser with name Greg entered the CS2510 chatroom.
ser with name Professor entered the CS2510 chatroom.
/15/2025 11:12:53 PM, CS2510, Greg: Hello Professor, I need help with my ass
gnment.
/15/2025 11:13:11 PM, CS2510, Professor: Okay, what can I help you with?
 am struggling with the docker file implementation. I cannot get my unit tes
s to run cleanly. My python script runs the server and a test app. The loggi
g can be seen in the server's console window, but this cannot be viewed when
running 2/15/2025 11:15:55 PM, CS2510, Greg: I am struggling with the docker
file implementation. I cannot get my unit tests to run cleanly. My python sc
ipt runs the server and a test app. The logging can be seen in the server's
onsole window, but this cannot be viewed when running the docker image!
/15/2025 11:17:33 PM, CS2510, Professor: Submit what you have for now and co
tinue improving upon your docker implementation throughout the course. What
ou have now seems acceptable.
ser with name Chris entered the CS2510 chatroom.
/15/2025 11:18:23 PM, CS2510, Chris: Hey everybody, I joined the chat to ask
about docker too!!!
```

```
Name: Chris
Chatroom: CS2510
User with name Greg entered the CS2510 chatroom.
User with name Professor entered the CS2510 chatroom.
2/15/2025 11:12:53 PM, CS2510, Greg: Hello Professor, I need help with my ass
ignment.
2/15/2025 11:13:11 PM, CS2510, Professor: Okay, what can I help you with?
2/15/2025 11:15:55 PM, CS2510, Greg: I am struggling with the docker file imp
lementation. I cannot get my unit tests to run cleanly. My python script runs
 the server and a test app. The logging can be seen in the server's console w
indow, but this cannot be viewed when running the docker image!
2/15/2025 11:17:33 PM, CS2510, Professor: Submit what you have for now and co
ntinue improving upon your docker implementation throughout the course. What
you have now seems acceptable.
User with name Chris entered the CS2510 chatroom.
2/15/2025 11:18:23 PM, CS2510, Chris: Hey everybody, I joined the chat to ask
 about docker too!!!
```

Wagonblast, Gregory
CS2510

Example of server running in Docker container.



Administrator: Command Prompt - docker run -p 5000:5000 groupchat-app

```
C:\Windows\System32>cd C:\Users\grego\source\repos\GroupChat

C:\Users\grego\source\repos\GroupChat>docker build -t groupchat-app .
[+] Building 7.1s (20/20) FINISHED                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                  0.0s
 => => transferring dockerfile: 2.42kB                                                                                0.0s
 => [internal] load metadata for mcr.microsoft.com/dotnet/sdk:9.0                                                     0.2s
 => [internal] load .dockerignore                                                                                     0.0s
 => => transferring context: 2B                                                                                       0.0s
 => [build-env  1/13] FROM mcr.microsoft.com/dotnet/sdk:9.0@sha256:7f8e8b1514a2eeccb025f1e9dd554e191b21afa7f43f83  0.0s
 => => resolve mcr.microsoft.com/dotnet/sdk:9.0@sha256:7f8e8b1514a2eeccb025f1e9dd554e191b21afa7f43f8321b7bd2009cd  0.0s
 => [internal] load build context                                                                                     0.5s
 => => transferring context: 2.61MB                                                                                   0.4s
 => CACHED [build-env  2/13] WORKDIR /app                                                                             0.0s
 => CACHED [build-env  3/13] COPY *.sln ./                                                                            0.0s
 => CACHED [build-env  4/13] COPY GroupChatServer/*.csproj ./GroupChatServer/                                         0.0s
 => CACHED [build-env  5/13] COPY GroupChatClient/*.csproj ./GroupChatClient/                                         0.0s
 => CACHED [build-env  6/13] COPY GroupChatUnreadMessageTest/*.csproj ./GroupChatUnreadMessageTest/                   0.0s
 => CACHED [build-env  7/13] COPY GroupChatDifferentGroupTest/*.csproj ./GroupChatDifferentGroupTest/                 0.0s
 => CACHED [build-env  8/13] RUN dotnet restore                                                                       0.0s
 => [build-env  9/13] COPY GroupChatServer/. ./GroupChatServer/                                                       0.1s
 => [build-env 10/13] COPY GroupChatClient/. ./GroupChatClient/                                                       0.1s
 => [build-env 11/13] COPY GroupChatUnreadMessageTest/. ./GroupChatUnreadMessageTest/                                 0.1s
 => [build-env 12/13] COPY GroupChatDifferentGroupTest/. ./GroupChatDifferentGroupTest/                               0.1s
 => [build-env 13/13] RUN dotnet build                                                                                5.6s
 => [stage-1 3/4] COPY --from=build-env /app/GroupChatServer/bin/Debug/net9.0/ ./                                     0.0s
 => [stage-1 4/4] COPY --from=build-env /app/GroupChatClient/bin/Debug/net9.0/ ./                                     0.0s
 => exporting to image                                                                                                0.3s
 => => exporting layers                                                                                               0.1s
 => => exporting manifest sha256:5b6211ce10a0f1742cd3450689effaee2b1d956fd22118c5800429b64445b544                     0.0s
 => => exporting config sha256:f1bce336523ffbfade854c4a857a3b9e8eed5851b07a2eb5a9c3804a711ee0f4                       0.0s
 => => exporting attestation manifest sha256:4a197b2069110279a84a8e6b2c97d79379841b85a25d32ebaac62bbd36362ab8         0.0s
 => => exporting manifest list sha256:4d717b8b8888ff58dd78761721548385c5e7623c45c869529a42c929135498bd                0.0s
 => => naming to docker.io/library/groupchat-app:latest                                                               0.0s
 => => unpacking to docker.io/library/groupchat-app:latest                                                            0.1s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/g1v056oj0e6851lf5u9fhc496

C:\Users\grego\source\repos\GroupChat>docker run -p 5000:5000 groupchat-app
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://[::]:8080
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Production
info: Microsoft.Hosting.Lifetime[0]
      Content root path: /app
```