Project 4: Async Chat Simulator (C#)

This project demonstrates a real-world C# async application that simulates a chat system with multiple users, asynchronous messaging, and thread-safe logging.

Project Objectives

- Multiple concurrent users
- Async/await based messaging
- Central chat room broadcasting
- Async file logging with thread safety

Architecture Overview

ChatUser → sends messages asynchronously
ChatRoom → broadcasts messages
ChatLogger → logs messages asynchronously to file

Key Code Snippets

Chat Logger (Async File Logging):

```
class ChatLogger
{
    private static readonly string logFile = "chatlog.txt";
    private static readonly SemaphoreSlim semaphore = new SemaphoreSlim(1, 1);

    public static async Task LogAsync(string message)
    {
        await semaphore.WaitAsync();
        try
        {
            await File.AppendAllTextAsync(logFile, message + Environment.NewLine);
        }
        finally
        {
            semaphore.Release();
        }
    }
}
```

Chat User:

```
class ChatUser
{
    public async Task StartChatAsync()
    {
        await Task.Delay(1000);
        await chatRoom.BroadcastAsync(userName, "Hello!");
    }
}
```

Concepts Practiced

- Async/await
- Task.WhenAll
- Concurrency

- Thread-safe logging using SemaphoreSlim

Enhancement Ideas

- CancellationToken support
- Message queues using Channel
- User join/leave notifications