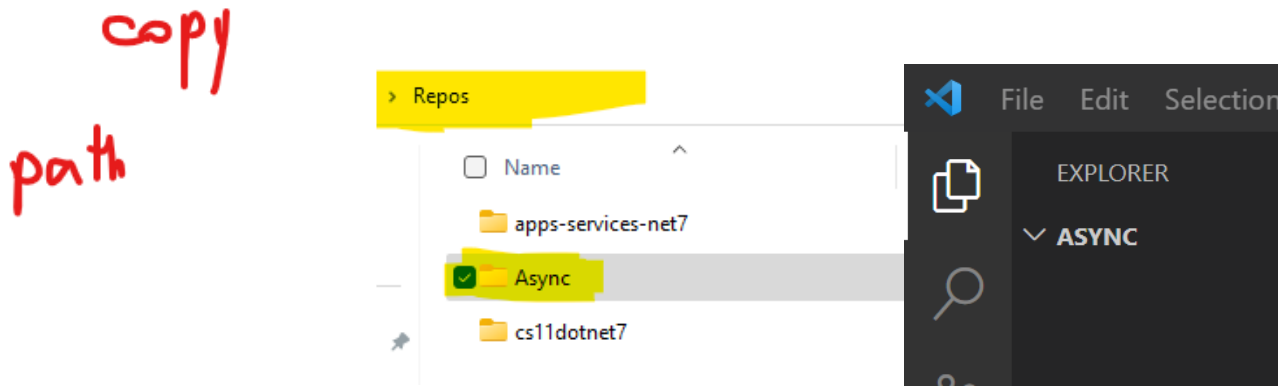


HW #5. Async and Await

Due Date: Listed on Canvas (this is theoretical assignment / essay. Should be submitted all in 1 PDF)

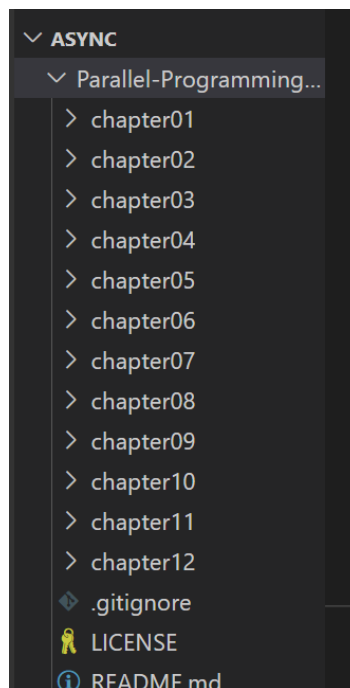
Create a new folder/directory, open it in VSC and clone the code of the book on Parallel Programming as follows:



Choose Terminal -> New Terminal, run the command below

`git clone https://github.com/PacktPublishing/Parallel-Programming-and-Concurrency-with-C-sharp-10-and-.NET-6`


Observe the result:



TASK 1. Provide screenshots proving that part 1 was completed.

Register if not registered, and open ChatGPT website: <https://chat.openai.com/chat> . Ask it to **add comments to all runnable C# code from Chapter 2, one by one.**

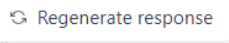
Example below:

 can you add comments to this code? namespace ContactListLocking

```
{
    public class ContactListManager
    {
        private readonly List<Contact> contacts;
        private readonly ReaderWriterLockSlim contactLock = new ReaderWriterLockSlim();


        public ContactListManager(List<Contact> initialContacts)
        {
            contacts = initialContacts;
        }

        public void AddContact(Contact newContact)
        {
            try
            {
                contactLock.EnterWriteLock();
                contacts.Add(newContact);
            }
            catch { }
        }
    }
}
```





Sure, here's the code with comments added to explain what each part does:

csharp  Copy code

```
namespace ContactListLocking
{
    public class ContactListManager
    {
        // The list of contacts that this manager will handle.
        private readonly List<Contact> contacts;

        // The lock used to synchronize access to the contacts list.
        private readonly ReaderWriterLockSlim contactLock = new ReaderWriterLockSlim();

        // Constructor that initializes the list of contacts.
        public ContactListManager(List<Contact> initialContacts)
        {
            contacts = initialContacts;
        }

        // Adds a new contact to the list of contacts.
        public void AddContact(Contact newContact)
        {
            try
            {
                contactLock.EnterWriteLock();
                contacts.Add(newContact);
            }
            catch { }
        }
    }
}
```

TASK 2. Make the ChatGPT generate comments for your code. Read through the commented programs to make sure all logic is transparent and clear. If some explanation is missing /needed, request for more clarification. Run the code. Provide Source code and output fully visible. Your source code should demonstrate the comments generated by chat GPT and run successfully. Explain **in your own words** what programs from Chapter 2 do and what you learn from them.

TASK 3. Do the same as in 2 with all runnable examples from Chapter 3.

TASK 4. Do the same as in 2 with all runnable examples from Chapter 5.

TASK 5. Do the same as in 2 with all runnable examples from Chapter 6.

TASK 6. Do the same as in 2 with all runnable examples from Chapter 8.

TASK 7. Answer the following questions:

- a) Did ChatGPT by adding comments to your programs help you to understand the code? Yes/No
- b) Was your experience with ChatGPT smooth/plesent? Yes/No
- c) Do you think moving forward you might do this for other code that is new and hard to comprehend? Yes/No