Department of Software Engineering

C#Final Project

Course Name: Object-Oriented Programming

Submitted By:

Rokaiya Akter

Sub-Group: 1

Submitted To:

Donatas Dervinis, Assist. Prof., Dr.

Github Link: https://github.com/Rokaiya-Akter/C--Final-Task

Objective

This project implements a distributed system using C# where two agent programs (ScannerA and ScannerB) scan text files and send word data to a master process using named pipes. The master collects, aggregates, and displays the results.

□ System Components

1. ScannerA (Agent A)

- Reads `.txt` files from a user-specified folder (textsA)
- Processes each file to extract words (either by frequency or sequence)
- Sends data to the master via pipe agent1

2. ScannerB (Agent B)

- Similar to ScannerA but reads from textsB
- Communicates via pipe agent2

3. Master Process

- Waits for connections from both agents
- Receives and processes word data
- Displays either aggregated counts or the sequence of words (based on configuration)

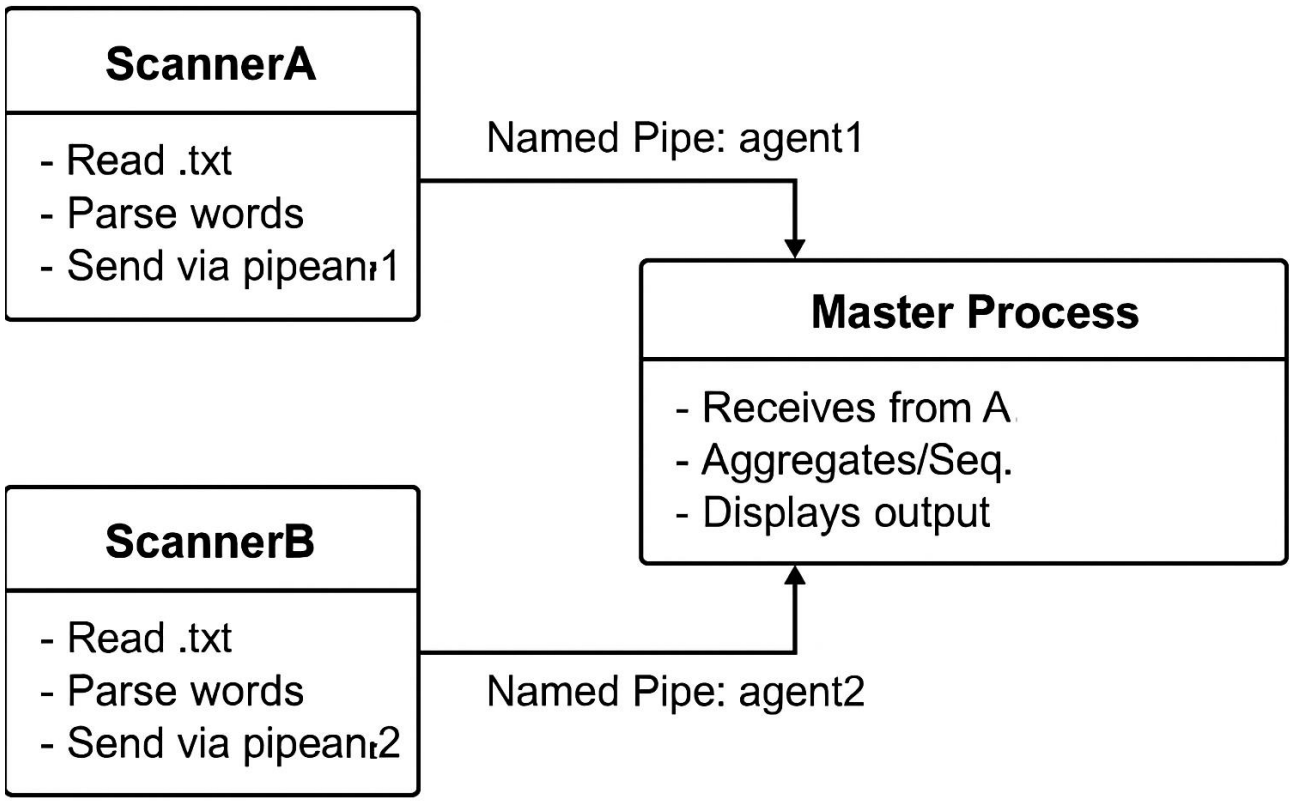
□ Test Environment

- **Operating System:** Windows 10 (x64)
- **.NET Version:** .NET 8.0 SDK + Runtime
- **Editor:** Notepad
- **Language:** C#
- **Communication:** Named Pipes
- **Multithreading:** Each process uses at least one background thread
- **CPU Affinity:** Assigned using ProcessorAffinity

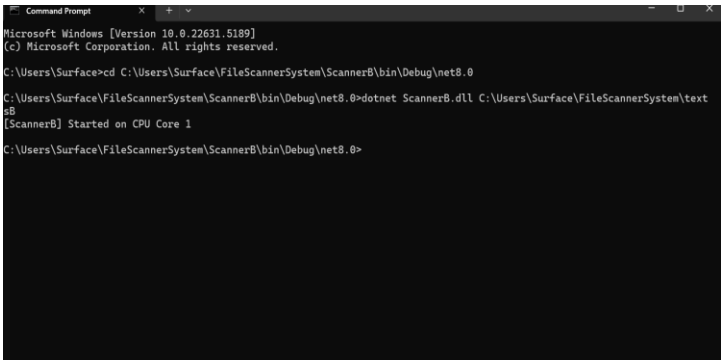
Challenges Encountered

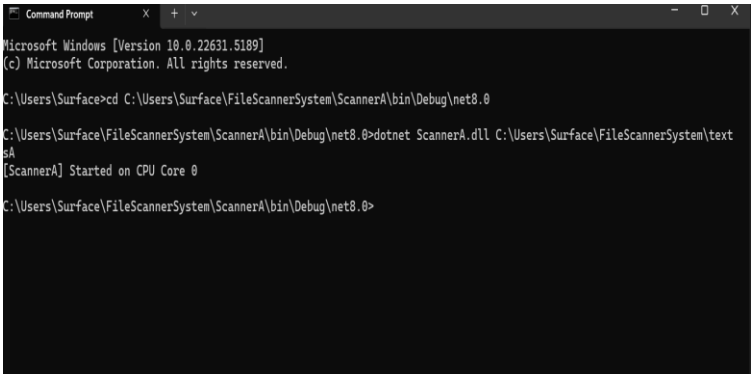
Problem	Solution
.NET 6 runtime missing	Updated project to .NET 8
DirectoryNotFoundException	Created missing textsA and textsB folders
Output was unordered	Modified logic to send words in sequence

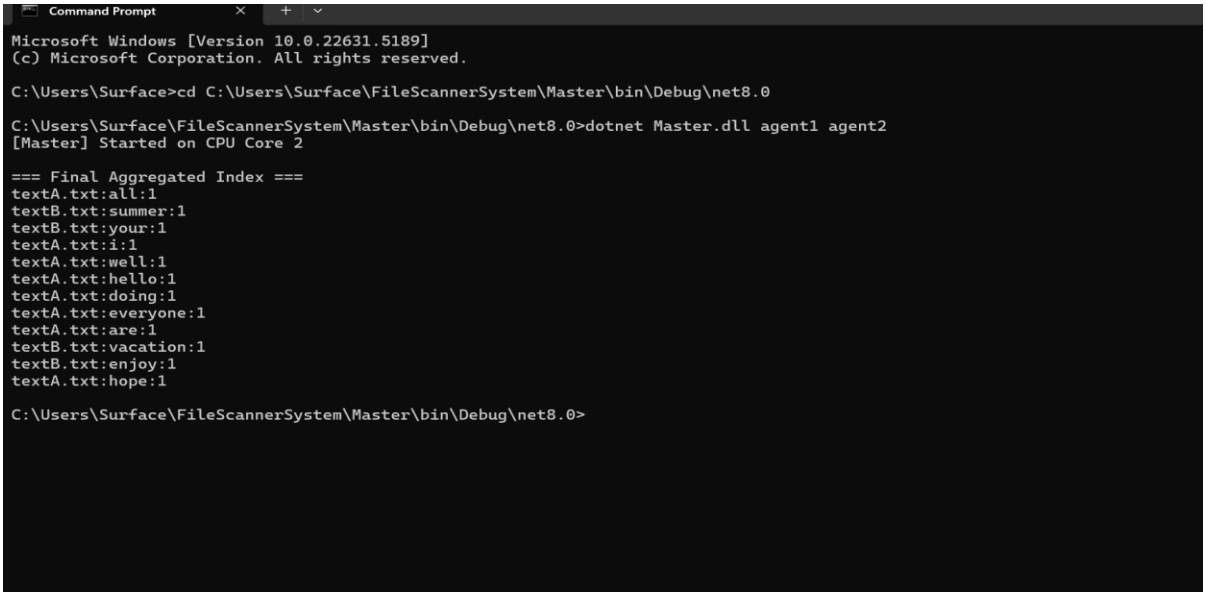
UML DIAGRAM


ScannerA
- Read .txt
- Parse words
- Send via pipeant1
Named Pipe: agent1
Master Process
- Receives from A
- Aggregates/Seq.
- Displays output
ScannerB
- Read .txt
- Parse words
- Send via pipeant2
Named Pipe: agent2

Screenshorts of the output:







Conclusion

The system works as expected:

- Agents can scan .txt files in parallel.
- Master successfully aggregates or lists data from both agents.
- Inter-process communication using named pipes is reliable and fast.
- Learned practical use of multithreading, pipes, and CPU affinity.