



# VILNIUS UNIVERSITY ŠIAULIAI ACADEMY

BACHELOR PROGRAMME SOFTWARE ENGINEERING

Object-Oriented Programming (OOP)

## **C# Final Project**

**Github report:** <https://github.com/Sifaurrahman/FileSyncNetScout>

Student : MD Sifaur Rahman

Leacturer : Donatas Dervinis, Assist. Prof., Dr.

Šiauliai, 2025

## a) Purpose of Project:

**FileSyncNetScout** = A Distributed File Indexing System Using C#

This C# console-based project demonstrates inter-process communication, multithreading, and CPU core control through a Master-Agent architecture using named pipes.

## b) System Overview

The system consists of three console applications:

**\*\*AgentA\*\*** scans `.txt` files and sends word counts via pipe ``agent1``

**\*\*AgentB\*\*** performs the same via pipe ``agent2``

**\*\*Master\*\*** receives data from both agents, merges and displays word count index

Each component runs on a dedicated CPU core.

## c) Core Features

- ✚ Word Indexing
- ✚ Multithreading
- ✚ Named Pipe Communication
- ✚ CPU Affinity via ``Processor.ProcessorAffinity``
- ✚ Git-based Versioning

## d) Architecture Summary

``AgentA`` and ``AgentB``: Read `.txt` files, count words, send data through named pipes

``Master``: Uses 2 threads to read from pipes concurrently, merges and prints final result.

## e) Technology Stack

Technology	Used For
C# (.NET 7)	All 3 console apps
Named Pipes	Agent-Master communication
Threads	Concurrent file I/O and pipe listening
CPU Affinity	Core isolation of processes
VS Code + Git	Development + version control

## f) CPU Affinity Logic

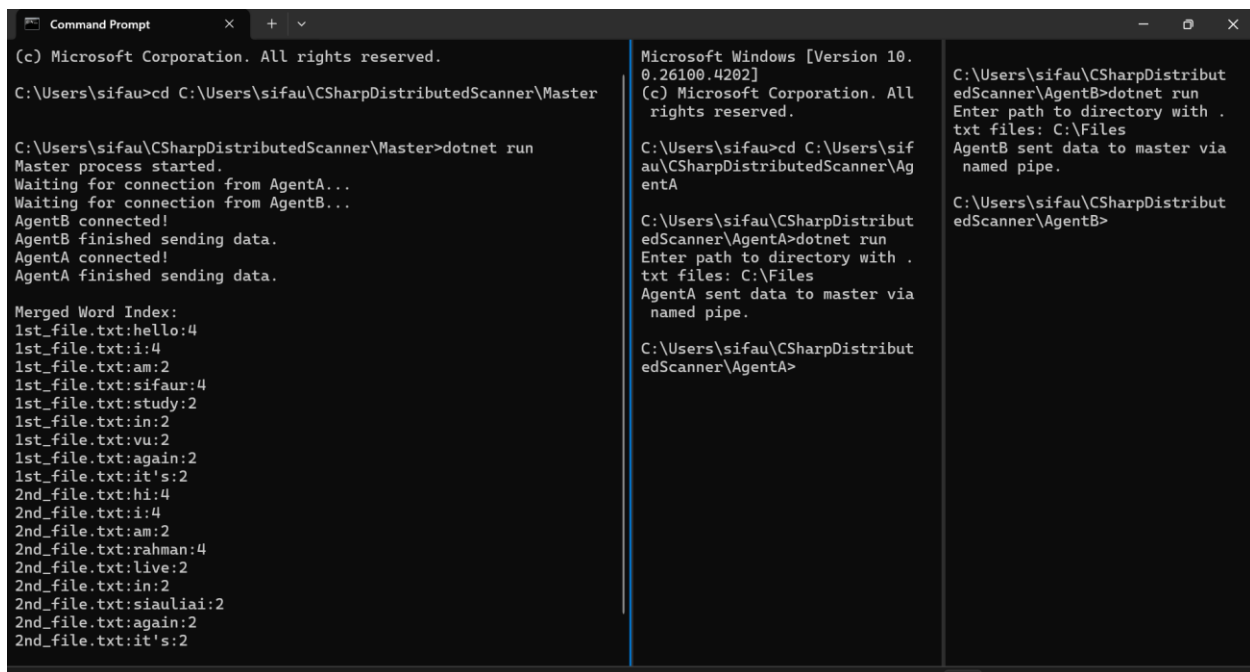
To run each app on its own CPU core:

```
```csharp
```

```
Process current = Process.GetCurrentProcess();
```

```
current.ProcessorAffinity = (IntPtr)(1 << N);
```

## g) Screenshots



```
(c) Microsoft Corporation. All rights reserved.
C:\Users\sifau>cd C:\Users\sifau\CSharpDistributedScanner\Master

C:\Users\sifau\CSharpDistributedScanner\Master>dotnet run
Master process started.
Waiting for connection from AgentA...
Waiting for connection from AgentB...
AgentB connected!
AgentB finished sending data.
AgentA connected!
AgentA finished sending data.

Merged Word Index:
1st_file.txt:hello:4
1st_file.txt:i:4
1st_file.txt:am:2
1st_file.txt:sifaur:4
1st_file.txt:study:2
1st_file.txt:in:2
1st_file.txt:vu:2
1st_file.txt:again:2
1st_file.txt:it's:2
2nd_file.txt:hi:4
2nd_file.txt:i:4
2nd_file.txt:am:2
2nd_file.txt:rahman:4
2nd_file.txt:live:2
2nd_file.txt:in:2
2nd_file.txt:siauliai:2
2nd_file.txt:again:2
2nd_file.txt:it's:2

Microsoft Windows [Version 10.0.26100.4202]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sifau>cd C:\Users\sifau\CSharpDistributedScanner\AgentA

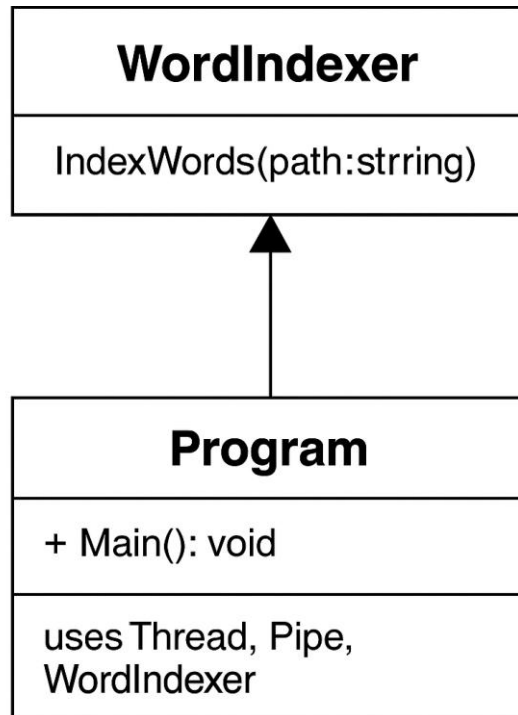
C:\Users\sifau\CSharpDistributedScanner\AgentA>dotnet run
Enter path to directory with .txt files: C:\Files
AgentA sent data to master via named pipe.

C:\Users\sifau\CSharpDistributedScanner\AgentA>

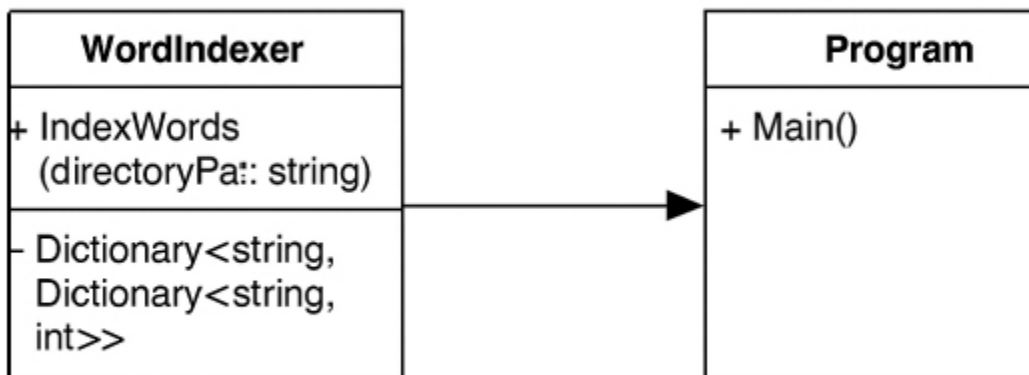
C:\Users\sifau\CSharpDistributedScanner\AgentB>dotnet run
Enter path to directory with .txt files: C:\Files
AgentB sent data to master via named pipe.

C:\Users\sifau\CSharpDistributedScanner\AgentB>
```

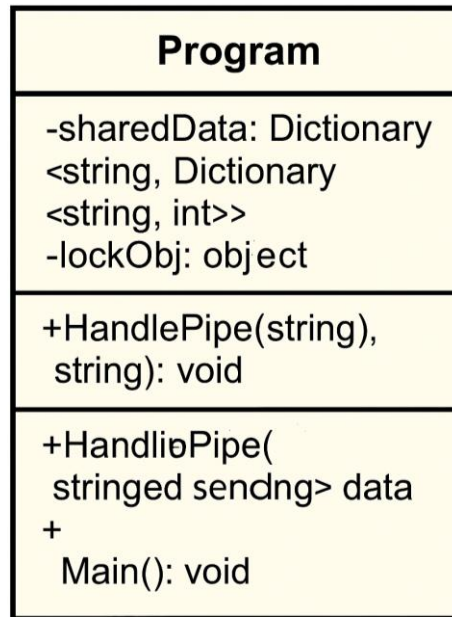
## h) UML Diagram:



**UML Diagram for AgentA**



**UML diagram for AgentB**



**UML diagram for Master**

## i) Challenges and Resolutions

Challenge	How I solve it
Named pipe connection errors	Ensured Master starts before Agents
Null input warning in Console.ReadLine()	Used null-check with ?? ""
CPU Affinity warning	Added correct using System.Diagnostics