

Fundamentals of Distributed Systems

1. Vector Clocks and Causal Ordering

This project simulates a real-world scenario where operations across nodes must respect causal relationships. The system includes multiple Python-based nodes. Each node maintains a local key-value store and a vector clock.

On every local write, the node increments its own clock. When sending messages, it includes this clock. When a node receives a message, it checks if the causal dependencies are met using vector clock comparison. If dependencies are satisfied, the message is applied immediately. If not, the message is buffered. This mechanism prevents issues where an update is applied before the data it depends on is available.

For example, if Node A writes a value and Node B reads and modifies it, all nodes must apply A's write before B's update.

I containerized the system using Docker, with a docker-compose.yml to simulate a 3-node environment. A client script simulates realistic scenarios, like a read followed by a write depending on it. The logs show that updates only occur when causal order is preserved, even if network messages arrive out of order. This verifies that vector clocks are functioning as intended and causal consistency is maintained. This project deepened my understanding of logical time, causality, and how buffering helps preserve consistency in distributed systems. All code and tests are organized in the required folder structure.

Folder structure creation in GIT Bash:

```
MINGW64:/c/Users/admin/vector-clock-kv-store

admin@DESKTOP-28C02RB MINGW64 ~
$ winget install --id Git.Git -e --source winget
Found an existing package already installed. Trying to upgrade the installed package...
No available upgrade found.
No newer package versions are available from the configured sources.

admin@DESKTOP-28C02RB MINGW64 ~
$ mkdir vector-clock-kv-store

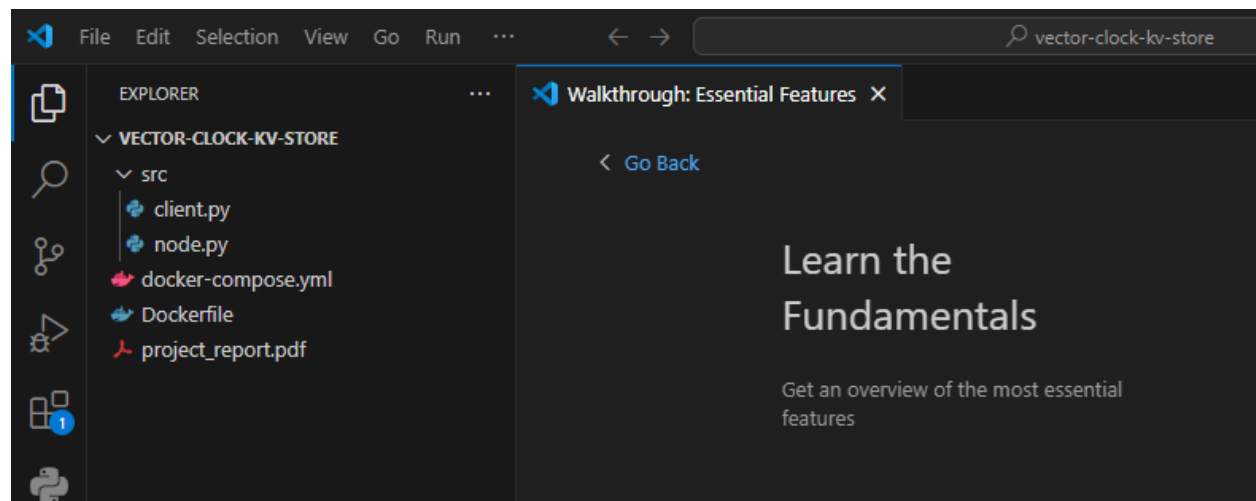
admin@DESKTOP-28C02RB MINGW64 ~
$ cd vector-clock-kv-store

admin@DESKTOP-28C02RB MINGW64 ~/vector-clock-kv-store
$ mkdir src

admin@DESKTOP-28C02RB MINGW64 ~/vector-clock-kv-store
$ touch src/node.py src/client.py Dockerfile docker-compose.yml project_report.pdf
```

```
admin@DESKTOP-28C02RB MINGW64 ~/vector-clock-kv-store
$ ls -R
.:
Dockerfile  docker-compose.yml  project_report.pdf  src/
./src:
client.py  node.py
```

Folder Structure in Visual Studio Code:



1. Node Implementation with Vector Clocks:

```
PS C:\Users\admin\vector-clock-kv-store\src> set NODE_ID=node1
>> set PEERS=node2:5000,node3:5000
>> python node.py
* Serving Flask app 'node'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.1.13:5000
Press CTRL+C to quit
127.0.0.1 - - [25/Jun/2025 14:54:37] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [25/Jun/2025 14:54:37] "GET /favicon.ico HTTP/1.1" 404 -
```

```
pretty-print ☐
{"node_id": "node1", "store": {}, "vector_clock": {"node1": 0}}
```

2. Vector Clock Logic:

```
PS C:\Users\admin\vector-clock-kv-store\src> Invoke-RestMethod -Uri http://localhost:5000/write -Method POST -ContentType "application/json" -Body '{"key":"foo","value":"bar"}'
>>

status  store      vector_clock
-----  -----  -
success @{foo=bar} @{node1=1}
```

Activate Windows
Go to Settings to activate

```
PS C:\Users\admin\vector-clock-kv-store\src>
```

3. Causal Write Propagation:

```
PS C:\Users\admin\vector-clock-kv-store\src> Invoke-RestMethod -Uri http://localhost:5000/write -Method POST -ContentType "application/json" -Body '{"key":"x","value":"1"}'
>>

status      store  vector_clock
-----  -----  -
local_write_success @{x=1} @{node1=1}
```

Activate Windows
Go to Settings to activate

```
PS C:\Users\admin\vector-clock-kv-store\src>
```

4. Containerization and Networking:

```
PS C:\Users\admin\vector-clock-kv-store> docker-compose up
time="2025-06-25T15:33:52+05:30" level=warning msg="C:\Users\admin\vector-clock-kv-store\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 4/4
 ✓ Network vector-clock-kv-store_default    Created                                0.1s
 ✓ Container vector-clock-kv-store-node3-1  Created                                0.3s
 ✓ Container vector-clock-kv-store-node2-1  Created                                0.3s
 ✓ Container vector-clock-kv-store-node1-1  Created                                0.3s
Attaching to node1-1, node2-1, node3-1
node1-1 | * Serving Flask app 'node'
```

Activate Windows
Go to Settings to activate

```
status      store  vector_clock
-----  -----  -
delivered @{k=A} @{node1=1; node2=0; node3=0}
```

5. Verification and Scenario Testing:

```
RawContent      : HTTP/1.1 200 OK
                  Connection: close
                  Content-Length: 99
                  Content-Type: application/json
                  Date: Wed, 25 Jun 2025 10:09:38 GMT
                  Server: Werkzeug/3.1.3 Python/3.10.18

                  {"buffered":[],"node_id":"node1","...
Forms           : {}
Headers         : {[Connection, close], [Content-Length, 99], [Content-Type, application/json], [Date,
```

```
admin@DESKTOP-28C02RB MINGW64 ~/vector-clock-kv-store (master)
$ git commit -m "First Commit"
[master (root-commit) 6ed7cfa] First Commit
4 files changed, 44 insertions(+)
create mode 100644 Dockerfile
create mode 100644 docker-compose.yml
create mode 100644 project_report.pdf
create mode 160000 src
```