

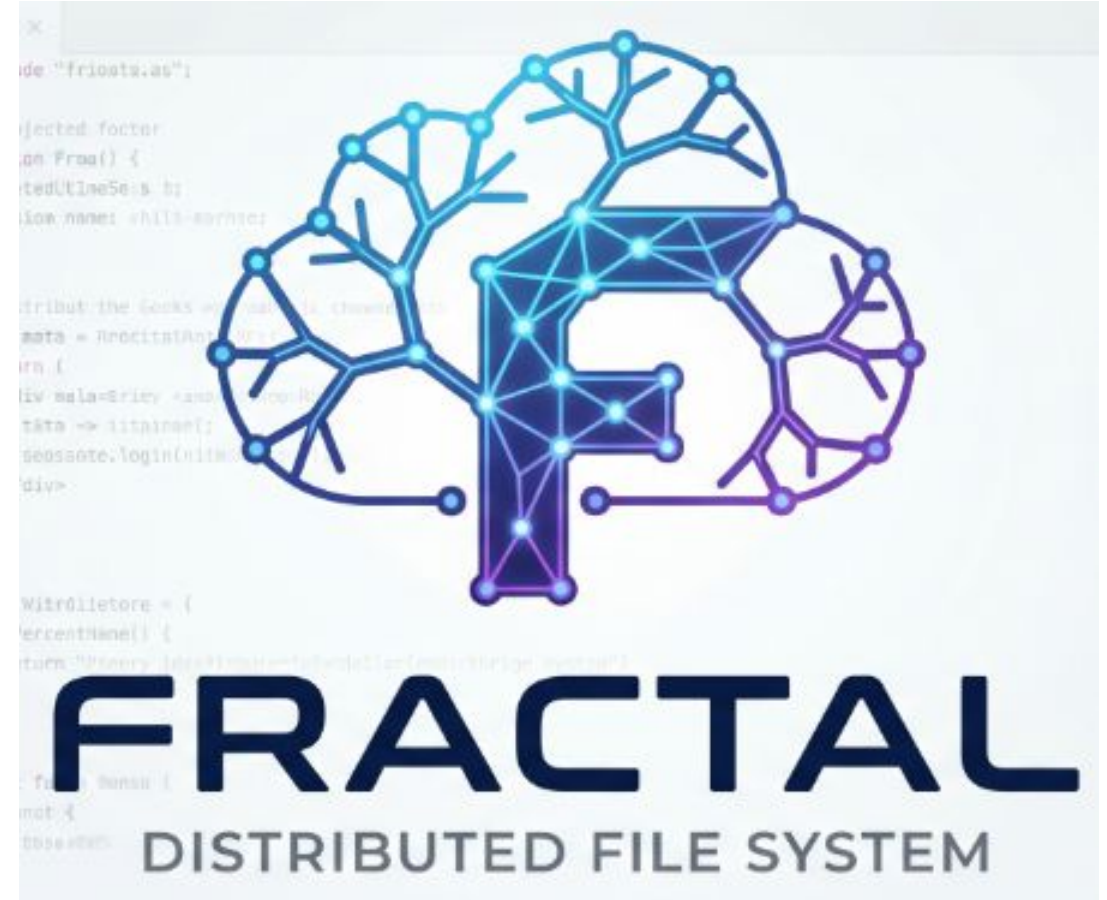
DISTRIBUTED SYSTEMS COURSE 2024/2025

Fractal Distributed File System

Matteo Zaccarini

CORE PRINCIPLES

- Master-Slave Architecture
- Chunk-Based Storage Engine
- CP Consistency over Availability (CAP Theorem)
- CLI tool



Master-Slave Architecture



NameNode

- **Centralized authority** coordinating namespace state, cluster health, and chunk routing blueprints
- Keeps the **namespace ledger** in RAM (**persisted** via **fsimage.json** snapshots), while tracking worker health strictly in ephemeral RAM. .
- Exchanges lightweight unary gRPC messages with the **client**, **never routing payload byte**.



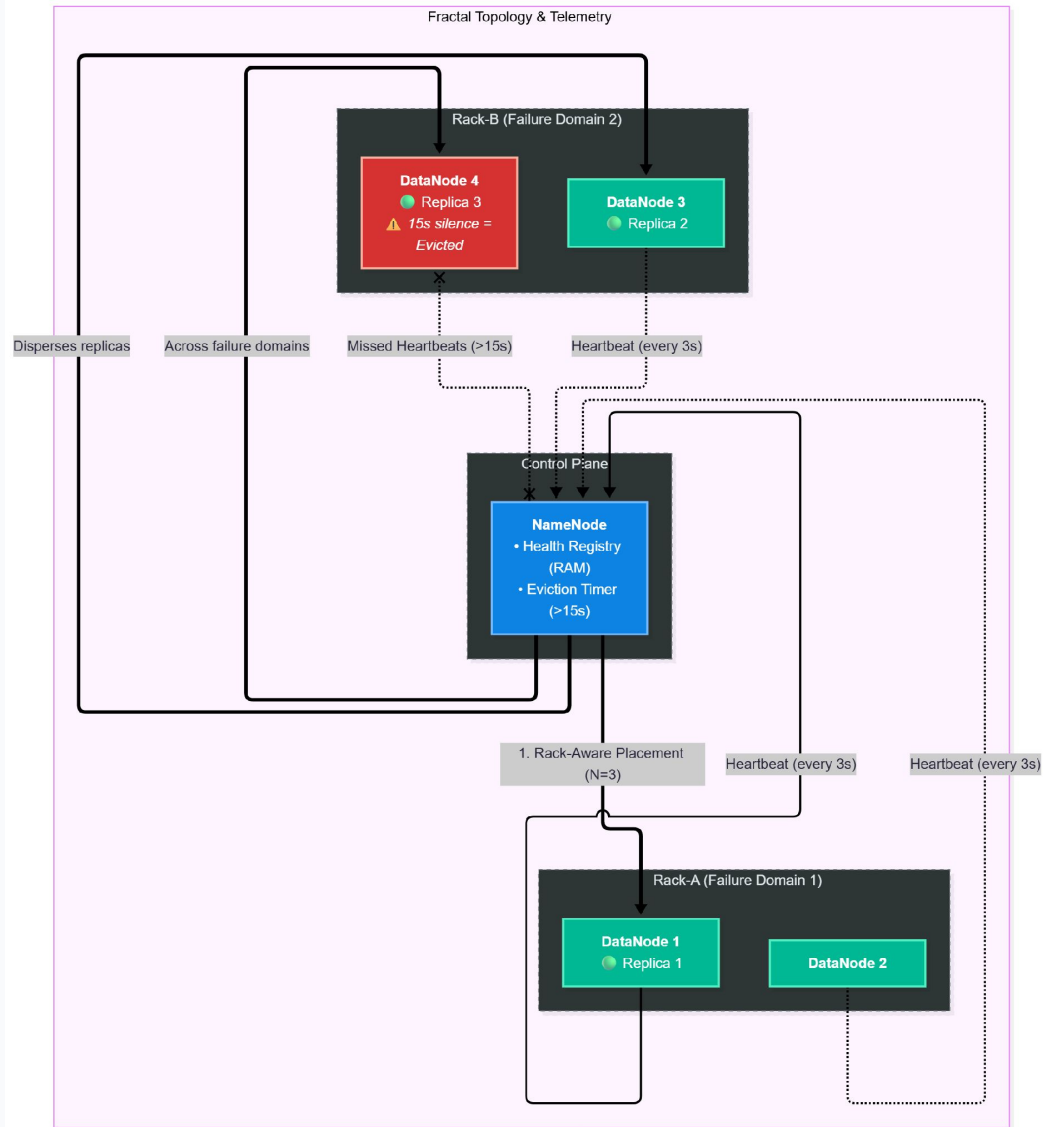
4 DataNodes

- Autonomous storage workers persisting and serving **immutable .dat chunks** on isolated host volumes
- Exchanges raw data directly with the **client** using **64 KB gRPC chunks** over TCP streams.

	namenode ● fractal-base-image 9000:9000 ↗	 		namenode	2026/10/01 19:41:31 INFO: booting NameNode... Attempting to load state from /app/data/file_system_image.json 2026/10/01 19:41:31 SUCCESS: FSImage loaded. Cluster state fully restored. 2026/10/01 19:41:31 INFO: NameNode is ALIVE and listening on port 9000
	datanode-3 ● fractal-base-image 8003:8003 ↗	 		datanode-1	2026/10/01 19:41:32 [INFO] [datanode-1] is ALIVE on port 8001. Local storage mapped to: /app/data/datanode-1
	datanode-2 ● fractal-base-image 8002:8002 ↗	 		datanode-3	2026/10/01 19:41:32 [INFO] [datanode-3] is ALIVE on port 8003. Local storage mapped to: /app/data/datanode-3
	datanode-4 ● fractal-base-image 8004:8004 ↗	 		datanode-2	2026/10/01 19:41:32 [INFO] [datanode-2] is ALIVE on port 8002. Local storage mapped to: /app/data/datanode-2
	datanode-1 ● fractal-base-image 8001:8001 ↗	 		datanode-4	2026/10/01 19:41:32 [INFO] [datanode-4] is ALIVE on port 8004. Local storage mapped to: /app/data/datanode-4
				datanode-2	2026/10/01 19:41:35 [INFO] Heartbeat successfully acknowledged by NameNode.
				datanode-3	2026/10/01 19:41:35 [INFO] Heartbeat successfully acknowledged by NameNode.
				datanode-1	2026/10/01 19:41:35 [INFO] Heartbeat successfully acknowledged by NameNode.
				datanode-4	2026/10/01 19:41:35 [INFO] Heartbeat successfully acknowledged by NameNode.

```
zacca@PerlaNera:/mnt/c/Users/zacca/Desktop/Laurea Magistrale Ing. Informatica Matteo/PRIMO ANNO/PRIMO SEMESTRE/Distributed Systems/fractal$ docker exec -it namenode cat /app/data/file_system_image.json
{
  "files": {
    "150mb.mp4": [
      "150mb.mp4-chunk-2",
      "150mb.mp4-chunk-0",
      "150mb.mp4-chunk-1"
    ]
  },
  "chunk_locations": {
    "150mb.mp4-chunk-0": [
      "datanode-3:8003",
      "datanode-1:8001",
      "datanode-2:8002"
    ],
    "150mb.mp4-chunk-1": [
      "datanode-3:8003",
      "datanode-2:8002",
      "datanode-1:8001"
    ],
    "150mb.mp4-chunk-2": [
      "datanode-2:8002",
      "datanode-4:8004",
      "datanode-3:8003"
    ]
  }
}
```

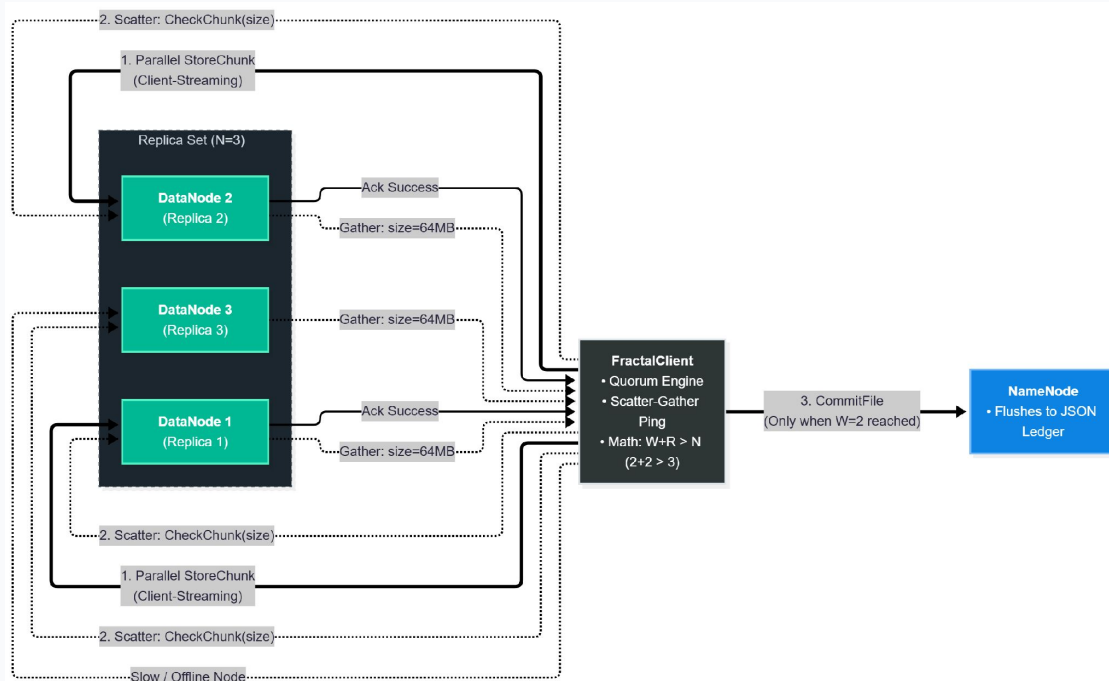
Command	Syntax	Expected Outcome
Help	<code>fractal --help</code>	Prints the global help menu, listing all available commands, usage syntax, and active flags.
Status	<code>fractal status</code>	Queries the NameNode's RAM and prints a live health table of the cluster, showing active DataNodes count, active DataNode IPs, rack assignments, and the timestamp of their last heartbeat.
Create	<code>fractal create [path/to/local/file]</code>	Ingests the local file, fragments it into 64 MB chunks, streams replicas to the DataNodes (enforcing W=2 Quorum), and registers the file in the NameNode. Prints a success confirmation.
Read	<code>fractal read [remote_file_name]</code>	Fetches the routing blueprint, validates the R=2 Read Quorum, and seamlessly streams the binary data back from the DataNodes. The reassembled file is saved to an automatically generated <code>downloads/</code> directory.
Update	<code>fractal update [path/to/local/file]</code>	Uploads the file under a temporary timestamped name (e.g., <code>v1718293_file.txt</code>), atomically swaps the metadata pointer in the NameNode, and asynchronously purges the orphaned chunks from the DataNodes.
Burn	<code>fractal burn [remote_file_name]</code>	Eradicates the file entirely. The NameNode deletes the logical namespace mapping, and the client sends commands to the target DataNodes to permanently delete the physical <code>.dat</code> binaries from their local drives.
List	<code>fractal list</code>	Queries the NameNode and prints a formatted table displaying all archived files and their respective chunk counts. Returns a specific prompt if the cluster is empty.



FAULT TOLERANCE

- **Rack-Awareness** placement engine that disperses $N = 3$ replicas across distinct racks (**Rack-A**, **Rack-B**)
- DataNodes publish lightweight, unidirectional **heartbeats** to the NameNode every **3 seconds**.
- Triggers an automated eviction after **15 seconds** of silence, immediately removing dead workers from allocation pools

QUORUM CONSENSUS



- **Strict Strong Consistency ($W + R > N$):** configures an $N = 3, W = 2, R = 2$ quorum model
- **Decentralized Consensus Engine:** offloads data plane coordination entirely to the client, keeping the NameNode unburdened.

CONSENSUS

Mathematical Parameters

$W=2$

Write Quorum required to
commit

$R=2$

Read Quorum required to
download

$3s$

Worker heartbeat interval

3

Replication Factor

SELF-HEALING

Automated Read-Repair

1

Detect

Size mismatch identified across replicas.

2

Stream

Valid chunk retrieved for user download.

3

Heal

Background thread re-uploads to stale node.

Technological Foundation



Go

Used to compile statically linked binaries that require no external host libraries.
Exploited for its concurrency primitives.



gRPC

Multiplexed long-lived streaming TCP connections.

ClientStreaming and ServerStreaming allows the system to continuously pump 64MB files across the network in 64KB increments.



Protocol Buffers

Language-agnostic binary payload serialization.



Docker

Isolated multi-rack topology simulation.

```
PS C:\Users\zacca\Desktop\Laurea Magistrale Ing. Informatica Matteo\PRIMO ANNO\PRIMO SEMESTRE\Distributed Systems\fractal> fractal create ./mock_files/150mb.mp4
```

VALIDATION

Chaos Engineering Testing



Write Failover

Graceful masking of single node crashes.



Partition Defense

Forced aborts on quorum failure.



Read Repair

Silent background system reconciliation.

```
=== RUN    TestReadQuorum_FailsWhenTwoReplicasDie
    helpers_test.go:36: Booting cluster...
    helpers_test.go:67: Waiting for 4 DataNodes to register heartbeats...
    helpers_test.go:90: SUCCESS: All 4 DataNodes are online and ready!
2026/10/01 23:12:08 Starting upload for '150mb.mp4' (size: 157286400 bytes)
2026/10/01 23:12:08 File split into into 3 chunks
2026/10/01 23:12:08 Broadcasting 150mb.mp4-chunk-0 to 3 nodes...
2026/10/01 23:12:10 Write Quorum reached for 150mb.mp4-chunk-0
2026/10/01 23:12:10 Broadcasting 150mb.mp4-chunk-1 to 3 nodes...
2026/10/01 23:12:13 Write Quorum reached for 150mb.mp4-chunk-1
2026/10/01 23:12:13 Broadcasting 150mb.mp4-chunk-2 to 3 nodes...
2026/10/01 23:12:14 Write Quorum reached for 150mb.mp4-chunk-2
2026/10/01 23:12:14 SUCCESS: file 150mb.mp4 is safely stored.
    chaos_read_quorum_test.go:42: killing containers: [datanode-3 datanode-2]
    chaos_read_quorum_test.go:48: Attempting to read the file...
2026/10/01 23:12:18 Requesting download blueprint for '150mb.mp4'...
2026/10/01 23:12:18 Blueprint received! file is split into 3 chunks. starting assembly...
2026/10/01 23:12:18 Pulling 150mb.mp4-chunk-0 (enforcing R=2 Quorum)...
2026/10/01 23:12:18 Pinging [datanode-3:8003 datanode-2:8002 datanode-1:8001] for 150mb.mp4-chunk-0 metadata...
2026/10/01 23:12:18 NODE UNREACHABLE: ping to datanode-2:8002 failed. Ignoring for Read Repair.
2026/10/01 23:12:18 NODE UNREACHABLE: ping to datanode-3:8003 failed. Ignoring for Read Repair.
    chaos_read_quorum_test.go:56: SUCCESS: system correctly rejected download due to killing [datanode-3:8003 datanode-2:8002]. Error caught: Cluster failed
to serve data: FAILURE: could not reach Read Quorum for 150mb.mp4-chunk-0. Cluster lost data
    helpers_test.go:44: Tearing down cluster...
--- PASS: TestReadQuorum_FailsWhenTwoReplicasDie (38.06s)
```

```
=== RUN    TestReadQuorum_SuccessWhenOneReplicaDies
helpers_test.go:36: Booting cluster...
helpers_test.go:67: Waiting for 4 DataNodes to register heartbeats...
helpers_test.go:90: SUCCESS: All 4 DataNodes are online and ready!
2026/10/01 23:12:37 Starting upload for '150mb.mp4' (size: 157286400 bytes)
2026/10/01 23:12:37 File split into into 3 chunks
2026/10/01 23:12:37 Broadcasting 150mb.mp4-chunk-0 to 3 nodes...
2026/10/01 23:12:39 Write Quorum reached for 150mb.mp4-chunk-0
2026/10/01 23:12:39 Broadcasting 150mb.mp4-chunk-1 to 3 nodes...
2026/10/01 23:12:55 Write Quorum reached for 150mb.mp4-chunk-1
2026/10/01 23:12:55 Broadcasting 150mb.mp4-chunk-2 to 3 nodes...
2026/10/01 23:12:57 Write Quorum reached for 150mb.mp4-chunk-2
2026/10/01 23:12:57 SUCCESS: file 150mb.mp4 is safely stored.
    chaos_read_quorum_test.go:90: killing containers: [datanode-2]
    chaos_read_quorum_test.go:96: Attempting to read the file...
2026/10/01 23:12:59 Requesting download blueprint for '150mb.mp4'...
2026/10/01 23:12:59 Blueprint received! file is split into 3 chunks. starting assembly...
2026/10/01 23:12:59 Pulling 150mb.mp4-chunk-0 (enforcing R=2 Quorum)...
2026/10/01 23:12:59 Pinging [datanode-2:8002 datanode-3:8003 datanode-4:8004] for 150mb.mp4-chunk-0 metadata...
2026/10/01 23:12:59 NODE UNREACHABLE: ping to datanode-2:8002 failed. Ignoring for Read Repair.
2026/10/01 23:12:59 Read Quorum reached for 150mb.mp4-chunk-0!
2026/10/01 23:13:01 Pulling 150mb.mp4-chunk-1 (enforcing R=2 Quorum)...
2026/10/01 23:13:01 Pinging [datanode-4:8004 datanode-2:8002 datanode-1:8001] for 150mb.mp4-chunk-1 metadata...
2026/10/01 23:13:01 NODE UNREACHABLE: ping to datanode-2:8002 failed. Ignoring for Read Repair.
2026/10/01 23:13:01 Read Quorum reached for 150mb.mp4-chunk-1!
2026/10/01 23:13:04 Pulling 150mb.mp4-chunk-2 (enforcing R=2 Quorum)...
2026/10/01 23:13:04 Pinging [datanode-2:8002 datanode-4:8004 datanode-3:8003] for 150mb.mp4-chunk-2 metadata...
2026/10/01 23:13:04 NODE UNREACHABLE: ping to datanode-2:8002 failed. Ignoring for Read Repair.
2026/10/01 23:13:04 Read Quorum reached for 150mb.mp4-chunk-2!
2026/10/01 23:13:05 Success! File fully reassembled and saved as 'downloads\150mb.mp4'
    chaos_read_quorum_test.go:102: SUCCESS: system correctly downloaded 150mb.mp4
helpers_test.go:44: Tearing down cluster...
--- PASS: TestReadQuorum_SuccessWhenOneReplicaDies (47.65s)
```

```
=== RUN    TestReadRepair_RestoresMissingChunk
    helpers_test.go:36: Booting cluster...
    helpers_test.go:67: Waiting for 4 DataNodes to register heartbeats...
    helpers_test.go:90: SUCCESS: All 4 DataNodes are online and ready!
2026/10/01 23:13:45 Starting upload for '150mb.mp4' (size: 157286400 bytes)
2026/10/01 23:13:45 File split into into 3 chunks
2026/10/01 23:13:45 Broadcasting 150mb.mp4-chunk-1 to 3 nodes...
2026/10/01 23:13:50 Write Quorum reached for 150mb.mp4-chunk-1
2026/10/01 23:13:50 Broadcasting 150mb.mp4-chunk-2 to 3 nodes...
2026/10/01 23:13:54 Write Quorum reached for 150mb.mp4-chunk-2
2026/10/01 23:13:54 Broadcasting 150mb.mp4-chunk-0 to 3 nodes...
2026/10/01 23:13:56 Write Quorum reached for 150mb.mp4-chunk-0
2026/10/01 23:13:56 SUCCESS: file 150mb.mp4 is safely stored.
    chaos_read_repair_test.go:38: Targeting node localhost:8001 for sabotage on chunk 150mb.mp4-chunk-1...
    chaos_read_repair_test.go:55: 150mb.mp4-chunk-1 on datanode-1:8001 sabotaged. Triggering download to force Read Repair...
2026/10/01 23:13:56 Requesting download blueprint for '150mb.mp4'...
2026/10/01 23:13:56 Blueprint received! file is split into 3 chunks. starting assembly...
2026/10/01 23:13:56 Pulling 150mb.mp4-chunk-0 (enforcing R=2 Quorum)...
2026/10/01 23:13:56 Pinging [datanode-4:8004 datanode-1:8001 datanode-2:8002] for 150mb.mp4-chunk-0 metadata...
2026/10/01 23:13:56 Read Quorum reached for 150mb.mp4-chunk-0!
2026/10/01 23:13:57 Pulling 150mb.mp4-chunk-1 (enforcing R=2 Quorum)...
2026/10/01 23:13:57 Pinging [datanode-1:8001 datanode-4:8004 datanode-3:8003] for 150mb.mp4-chunk-1 metadata...
2026/10/01 23:13:57 datanode-1:8001 is alive but missing 150mb.mp4-chunk-1
2026/10/01 23:13:57 Read Quorum reached for 150mb.mp4-chunk-1!
2026/10/01 23:13:59 Pulling 150mb.mp4-chunk-2 (enforcing R=2 Quorum)...
2026/10/01 23:13:59 Pinging [datanode-3:8003 datanode-1:8001 datanode-2:8002] for 150mb.mp4-chunk-2 metadata...
2026/10/01 23:13:59 Read Quorum reached for 150mb.mp4-chunk-2!
2026/10/01 23:14:01 Background heal of 150mb.mp4-chunk-1 successful for datanode-1:8001!
2026/10/01 23:14:01 Success! File fully reassembled and saved as 'downloads\150mb.mp4'
    chaos_read_repair_test.go:72: SUCCESS: Read Repair worked! Chunk 150mb.mp4-chunk-1 was successfully restored on datanode-1:8001
    helpers_test.go:44: Tearing down cluster...
--- PASS: TestReadRepair_RestoresMissingChunk (57.68s)
```

FUTURE WORKS

SPOF Mitigation

Warm standby Secondary NameNode.

Namespace Federation

Multiple masters for horizontal scaling.

Protect Data Integrity

SHA-256 silent bit-rot detection.

Garbage Collection and Block Reports

Delete orphaned physical chunks.

Under-Replication Healing

When a DataNode dies, chunks are migrate to others DataNode to keep Replication Factor = 3.

Grazie per l'attenzione
