Report for Project 3

Group 7: Weiran Zhang, Hongzhuo Chen

GitHub link to this project: https://github.com/Richardchen714/distributed-pv

Two-Phase Commit (2PC) with gRPC (Voting + Decision Phases)

This project extends the distributed-pv system by implementing a fully working Two-Phase Commit (2PC) protocol using gRPC.

It supports:

- Voting Phase (Q1)
- Decision Phase (Q2)
- Cross-language compatibility by using a shared .proto file
- 5+ containerized nodes that communicate over gRPC
- A coordinator that runs the full 2PC protocol and logs all RPC calls

Each node runs two internal services:

1. Participant-Vote Phase Service
2. Participant-Decision Phase Service

Both services are gRPC servers running inside the same container, communicating only through gRPC.

Overview of the Two-Phase Commit Protocol

A 2PC transaction has two phases:

Phase 1 — Voting Phase

The Coordinator sends a Vote RPC request to all participant nodes.
Each Participant responds with:

1. VOTE_COMMIT — ready to commit
2. VOTE_ABORT — cannot commit

This is implemented with the RPC method:

```
rpc Vote(TxRequest) returns (VoteResponse)
```

Phase 2 — Decision Phase

1. Coordinator gathers all votes:
    - If all commit, send GLOBAL_COMMIT
    - If any abort, send GLOBAL_ABORT
2. Each participant receives the decision and:

- Commits (if global commit)
- Aborts (if global abort)

This phase uses:

```
rpc Decide(Ack) returns (Ack)
```

Logging Requirements (Implemented)

Every RPC prints logs in exactly this format:

Client side

```
Phase <phase_name> of Node <node_id> sends RPC <rpc_name> to Phase <phase_name> of Node
<node_id>
```

Server side

```
Phase <phase_name> of Node <node_id> sends RPC <rpc_name> to Phase <phase_name> of Node
<node_id>
```

All vote and decision RPCs produce these logs automatically.

Container Architecture

Each participant node runs two internal gRPC servers:

Service	Port	Purpose
vote_server.py	50051	Handles Vote phase
decision_server.py	60051	Handles final decision

Coordinator uses the same .proto and connects to each node at these ports.

Configuration

Each node uses two environment variables:

Variable	Purpose
`NODE_ID`	Identifies the node
`VOTE`	Either COMMIT or ABORT
`VOTE_PORT`	Port of voting server
`DECISION_PORT`	Port of decision server

Example in `docker-compose.yml`:

```
environment:
  - NODE_ID=node4
  - VOTE=ABORT
```

You can force an abort by setting `VOTE=ABORT`.

How to Run

1. Build images

From the root folder (2pc-grpc/):

```
docker-compose build
```

2. Start the system

```
docker-compose up
```

This launches:

- 5 participant nodes
- 1 coordinator

The coordinator waits a few seconds and automatically starts 2PC for transaction:

```
tx-001
```

3. Read the output logs

You will see messages such as:

- Voting Phase:

```
Phase Coordinator of Node coordinator sends RPC Vote to Phase Participant-Vote of Node node1
Phase Participant-Vote of Node node1 sends RPC Vote to Phase Coordinator of Node coordinator
Phase Coordinator of Node coordinator received VoteResponse from node1: VOTE_COMMIT
```

- Decision Phase:

```
Coordinator decision for tx tx-001: GLOBAL_ABORT
Phase Coordinator of Node coordinator sends RPC Decide to Phase Participant-Decision of Node
node4
Phase Participant-Decision of Node node4 sends RPC Decide to Phase Coordinator of Node
coordinator
Participant-Decision Node node4: aborting transaction (tx ack from coordinator).
```

Changing Votes to Test Behavior

To simulate different outcomes, edit docker-compose.yml:
Example: make node4 abort

```
node4:
  environment:
    - VOTE=ABORT
```

Example: all commit

```
node4:
  environment:
    - VOTE=COMMIT
```

Then rebuild & restart:

```
docker-compose down -v
docker-compose up --build
```

Expected Scenarios

Node Votes	Coordinator Decision	Behavior
All COMMIT	GLOBAL_COMMIT	All participants commit
One or more ABORT	GLOBAL_ABORT	All participants abort

Raft Consensus Protocol (Q3–Q5)

This project integrates **Raft consensus algorithm** into the distributed password vault, ensuring strong consistency across nodes during secret operations.

It supports:

- **Leader election (Q3)**
- **Log replication using gRPC heartbeat (Q4)**
- **Client request forwarding (when request hits a follower)**
- **Leader crash recovery**
- **Integration with Password Vault microservice (Q5)**

Raft nodes are fully containerized using Docker and communicate via gRPC.

Raft Core Components

File	Description
`raft/raft_node.py`	Core Raft implementation (election + replication)
`raft/raft_server.py`	Entrypoint to start each Raft node
`raft/raft_integration.py`	Interface between Raft and application layer
`raft/Q3_test.py`	Demonstrates Raft leader election only (no log replication)
`raft/Q4_test.py`	Local in-process test (leader election and log replication)
`test_raft_addsecret.py`	Final demo: Add Secret via Raft replication

🔌 Raft Cluster Architecture (5 Nodes)

Node ID	Container Name	Address
node1	`raft-node1`	`raft-node1:7001`
node2	`raft-node2`	`raft-node2:7002`
node3	`raft-node3`	`raft-node3:7003`
node4	`raft-node4`	`raft-node4:7004`
node5	`raft-node5`	`raft-node5:7005`

Each node is aware of all other nodes through `docker-compose.yml`.

Implementation Details

Raft Message Formats

```
RequestVote RPC
{
  candidate_id: <Node requesting vote>,
  term: <Current election term>
}

AppendEntries RPC
{
  leader_id: <Current leader node>,
  entries: [
    {
      k: <Index of log entry>,
      t: <Term when entry was added>,
      o: <Client operation stored>
```

```
    }
  ],
  commit_index: <Most recently committed index>
}
```

Leader Election Workflow

```
heartbeat timeout:1s
Election timeout = random between 5.0 ~ 9.0 seconds(Election timeout was extended to 5–8 s
(instead of the standard 1.5–3 s) to avoid split-vote instability during local Docker
testing)

node1: didn't received node3 appendentry(heartbeat1), reach it election timeout, become
candidate -> send requestvote RPC to node2/3/4/5(heartbeat6.5) -> received OK from
node2/3/4/5(heartbeat7.5), become leader

node2: didn't received node3 appendentry(heartbeat1), didn't reach it election timeout-
>received node1 requestvote RPC(heartbeat6.5) -> send OK to node1(heartbeat7.5)

node3: crash(heartbeat1) -> --- -> ----

node4: didn't received node3 appendentry(heartbeat1), didn't reach it election timeout-
>received node1 requestvote RPC(heartbeat6.5) -> send OK to node1(heartbeat7.5)

node5: didn't received node3 appendentry(heartbeat1), didn't reach it election timeout-
>received node1 requestvote RPC(heartbeat6.5) -> send OK to node1(heartbeat7.5)
```

Log Replication Workflow

```
node1/node2/node3(leader)/node4/node5

Workflow: new client request2 -> node1 -> forward request to node3(next heartbeat) ->
node3(received forwarded request->update current entry(request2 pending)), multicast
appendentry RPC to other nodes(1/2/4/5)(next heartbeat) -> other nodes commit request1 and
update their log entry, send ACK to node3(next heartbeat) -> node3 commit request2 and
update entry(c: The index of the most recently committed operation)

node1: received node1 appendentry/received client request(heartbeat0) -> forward request to
node3(heartbeat1) -> received node1 appendentry RPC(heartbeat2) -> commit request1, send
ACK(heartbeat3)

node2: received node1 appendentry(heartbeat0)->received appendentry(heartbeat1)->received
node3 appendentry RPC(heartbeat2) -> commit request1, send ACK(heartbeat3)

node3: sent appendentry to node1/2/4/5(heartbeat0)->received forwarded request2(latest
committed request1)(heatbeat1), update current entry(request2 pending)->multicast
appendentry RPC to other nodes(1/2/4/5)(heartbeat2)->node3 received ACK, commit request2 and
update entry(c: The index of the most recently committed operation)(heartbeat3)

node4: received node1 appendentry(heartbeat0) -> received appendentry(heartbeat1) ->
received node3 appendentry RPC(heartbeat2) -> commit request1, send ACK(heartbeat3)

node5: received node1 appendentry(heartbeat0) -> received appendentry(heartbeat1) ->
received node3 appendentry RPC(heartbeat2) -> commit request1, send ACK(heartbeat3)
```

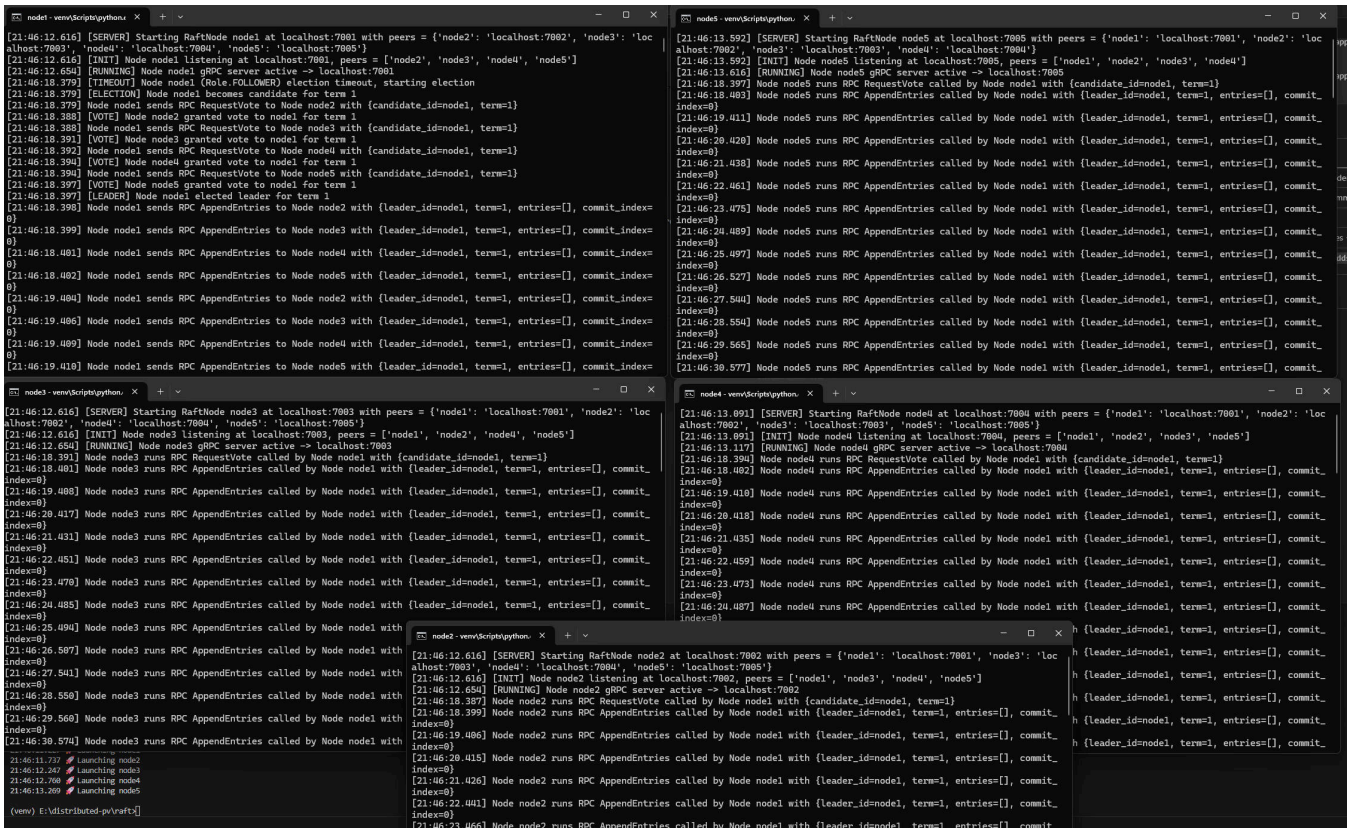
Test Cases

Test Case	Description
TC1 – Q3 Leader Election	Start Raft cluster and observe automatic leader selection
TC2 – Q4 Log Replication	Send operations to leader and verify log commitment across all nodes
TC3 – Leader Crash Recovery	Stop leader container, verify re-election
TC4 – System Initialization (5 Raft nodes + Microservices of Password App)	Start complete system including 5 Raft nodes + Password microservices
TC5 – Add Secret via Raft	Perform password add using `test_raft_addsecret.py`, replicated via Raft

Test Cases details

TC1 – Leader Election

- Screenshot



TC2 – Log Replication

- Screenshot

```
(venv) E:\distributed-pv>py -m raft.Q4_test
[21:51:46.364] [INIT] Node node1 listening at localhost:7001, peers = ['node2', 'node3', 'node4', 'node5']
[21:51:46.382] [RUNNING] Node node1 gRPC server active -> localhost:7001
[21:51:46.382] [INIT] Node node2 listening at localhost:7002, peers = ['node1', 'node3', 'node4', 'node5']
[21:51:46.398] [RUNNING] Node node2 gRPC server active -> localhost:7002
[21:51:46.398] [INIT] Node node3 listening at localhost:7003, peers = ['node1', 'node2', 'node4', 'node5']
[21:51:46.414] [RUNNING] Node node3 gRPC server active -> localhost:7003
[21:51:46.414] [INIT] Node node4 listening at localhost:7004, peers = ['node1', 'node2', 'node3', 'node5']
[21:51:46.430] [RUNNING] Node node4 gRPC server active -> localhost:7004
[21:51:46.430] [INIT] Node node5 listening at localhost:7005, peers = ['node1', 'node2', 'node3', 'node4']
[21:51:46.444] [RUNNING] Node node5 gRPC server active -> localhost:7005
🚀 All nodes are up. Waiting for leader election...
[21:51:51.536] [TIMEOUT] Node node3 (Role.FOLLOWER) election timeout, starting election
[21:51:51.537] [ELECTION] Node node3 becomes candidate for term 1
[21:51:51.537] Node node3 sends RPC RequestVote to Node node1 with {candidate_id=node3, term=1}
[21:51:51.544] Node node1 runs RPC RequestVote called by Node node3 with {candidate_id=node3, term=1}
[21:51:51.544] [VOTE] Node node1 granted vote to node3 for term 1
[21:51:51.544] Node node3 sends RPC RequestVote to Node node2 with {candidate_id=node3, term=1}
[21:51:51.545] Node node2 runs RPC RequestVote called by Node node3 with {candidate_id=node3, term=1}
[21:51:51.546] [VOTE] Node node2 granted vote to node3 for term 1
[21:51:51.546] Node node3 sends RPC RequestVote to Node node4 with {candidate_id=node3, term=1}
[21:51:51.547] Node node4 runs RPC RequestVote called by Node node3 with {candidate_id=node3, term=1}
[21:51:51.547] [VOTE] Node node4 granted vote to node3 for term 1
[21:51:51.547] Node node3 sends RPC RequestVote to Node node5 with {candidate_id=node3, term=1}
[21:51:51.548] Node node5 runs RPC RequestVote called by Node node3 with {candidate_id=node3, term=1}
[21:51:51.548] [VOTE] Node node5 granted vote to node3 for term 1
[21:51:51.549] [LEADER] Node node3 elected leader for term 1
[21:51:51.549] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:51.550] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:51.550] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:51.551] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:51.551] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:51.552] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:51.553] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:52.554] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:52.558] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:52.559] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:52.562] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:52.563] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:52.565] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:52.565] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:52.567] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:53.567] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:53.574] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:53.575] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:53.578] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:53.579] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:53.582] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:53.583] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:53.585] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:54.586] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:54.587] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:54.588] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:54.589] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:54.589] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:54.591] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:54.591] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:54.592] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:55.592] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:55.594] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:55.594] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:55.596] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:55.596] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:55.597] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:55.597] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[], commit_index=0}
[21:51:55.598] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[], commit_index=0}
📢 Leader elected: node3
```

```
🚀 Sending 3 client operations to test log replication...


📩 Proposing operation: test_op_1
[CLIENT] Sending operation 'test_op_1' to localhost:7001 as client 3189da0d-b3d1-430d-974f-252be2dc0350
[21:51:56.450] Node node1 runs RPC ClientRequest called by Node client with operation='test_op_1'
[21:51:56.450] Node node1 forwards client request to leader node3
[21:51:56.450] Node node1 sends RPC ClientRequest to Node node3 (forwarded client op='test_op_1')
[21:51:56.455] Node node3 runs RPC ClientRequest called by Node node1 with operation='test_op_1'
[21:51:56.455] [LOG] Node node3 (leader) appended entry (k=1, t=1, o='test_op_1') PENDING; will be replicated on next heartbeat
  → Response: success=True, result=Operation 'test_op_1' accepted by leader node3, pending commit
[21:51:56.599] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=0}
[21:51:56.602] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=0}
[21:51:56.603] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=0}
[21:51:56.605] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=0}
[21:51:56.606] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=0}
[21:51:56.609] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=0}
[21:51:56.609] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=0}
[21:51:56.611] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=0}
[APPLY] Node node3 executes raw: test_op_1
[21:51:56.612] [COMMIT] Node node3 commits log entries up to index 1
[21:51:57.613] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=1}
[21:51:57.615] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=1}
[APPLY] Node node1 executes raw: test_op_1
[21:51:57.615] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=1}
[21:51:57.616] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=1}
[APPLY] Node node2 executes raw: test_op_1
[21:51:57.616] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=1}
[21:51:57.617] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=1}
[APPLY] Node node4 executes raw: test_op_1
[21:51:57.618] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=1}
[21:51:57.618] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1')], commit_index=1}
[APPLY] Node node5 executes raw: test_op_1

📩 Proposing operation: test_op_2
[CLIENT] Sending operation 'test_op_2' to localhost:7001 as client 1d9b99a2-0e46-4b7a-9427-7783dbcba06c
[21:51:58.459] Node node1 runs RPC ClientRequest called by Node client with operation='test_op_2'
[21:51:58.460] Node node1 forwards client request to leader node3
[21:51:58.460] Node node1 sends RPC ClientRequest to Node node3 (forwarded client op='test_op_2')
[21:51:58.461] Node node3 runs RPC ClientRequest called by Node node1 with operation='test_op_2'
[21:51:58.461] [LOG] Node node3 (leader) appended entry (k=2, t=1, o='test_op_2') PENDING; will be replicated on next heartbeat
  → Response: success=True, result=Operation 'test_op_2' accepted by leader node3, pending commit
[21:51:58.620] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=1}
[21:51:58.624] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=1}
[21:51:58.625] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=1}
[21:51:58.628] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=1}
[21:51:58.629] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=1}
[21:51:58.631] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=1}
[21:51:58.632] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=1}
[21:51:58.633] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=1}
[APPLY] Node node3 executes raw: test_op_2
[21:51:58.634] [COMMIT] Node node3 commits log entries up to index 2
[21:51:59.634] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=2}
[21:51:59.640] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=2}
[APPLY] Node node1 executes raw: test_op_2
[21:51:59.641] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=2}
[21:51:59.644] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=2}
[APPLY] Node node2 executes raw: test_op_2
[21:51:59.645] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=2}
[21:51:59.648] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=2}
[APPLY] Node node4 executes raw: test_op_2
[21:51:59.649] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=2}
[21:51:59.651] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2')], commit_index=2}
[APPLY] Node node5 executes raw: test_op_2

📩 Proposing operation: test_op_3
[CLIENT] Sending operation 'test_op_3' to localhost:7001 as client 70cea9a6-f17c-405c-bcdb-443135c3307e
[21:52:00.465] Node node1 runs RPC ClientRequest called by Node client with operation='test_op_3'
[21:52:00.465] Node node1 forwards client request to leader node3
[21:52:00.465] Node node1 sends RPC ClientRequest to Node node3 (forwarded client op='test_op_3')
[21:52:00.467] Node node3 runs RPC ClientRequest called by Node node1 with operation='test_op_3'
```

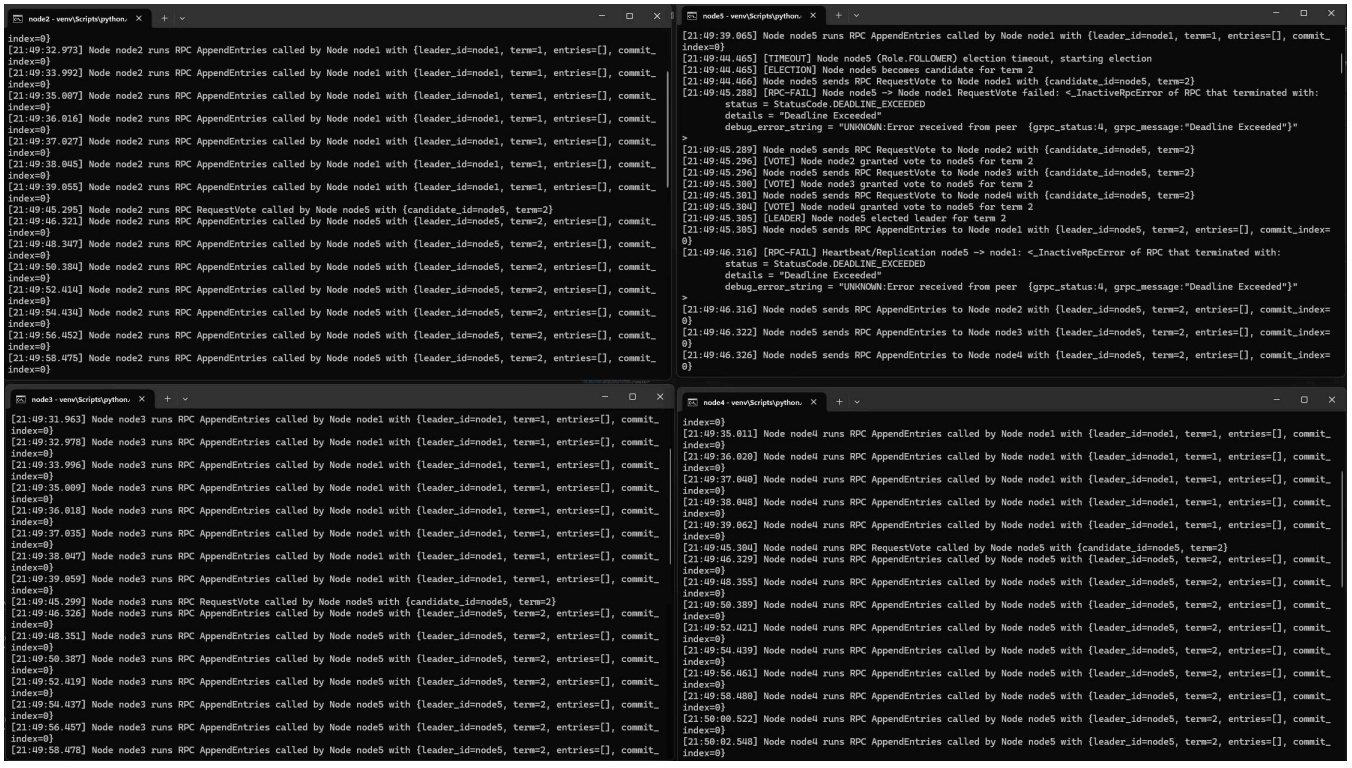
```
📩 Proposing operation: test_op_3
[CLIENT] Sending operation 'test_op_3' to localhost:7001 as client 70cea9a6-f17c-405c-bcdb-443135c3307e
[21:52:00.465] Node node1 runs RPC ClientRequest called by Node client with operation='test_op_3'
[21:52:00.465] Node node1 forwards client request to leader node3
[21:52:00.465] Node node1 sends RPC ClientRequest to Node node3 (forwarded client op='test_op_3')
[21:52:00.467] Node node3 runs RPC ClientRequest called by Node node1 with operation='test_op_3'
[21:52:00.467] [LOG] Node node3 (leader) appended entry (k=3, t=1, o='test_op_3') PENDING; will be replicated on next heartbeat
  → Response: success=True, result=Operation 'test_op_3' accepted by leader node3, pending commit
[21:52:00.651] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=2}
[21:52:00.657] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=2}
[21:52:00.659] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=2}
[21:52:00.662] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=2}
[21:52:00.663] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=2}
[21:52:00.665] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=2}
[21:52:00.666] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=2}
[21:52:00.668] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=2}
[APPLY] Node node3 executes raw: test_op_3
[21:52:00.668] [COMMIT] Node node3 commits log entries up to index 3
[21:52:01.669] Node node3 sends RPC AppendEntries to Node node1 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=3}
[21:52:01.674] Node node1 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=3}
[APPLY] Node node1 executes raw: test_op_3
[21:52:01.676] Node node3 sends RPC AppendEntries to Node node2 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=3}
[21:52:01.679] Node node2 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=3}
[APPLY] Node node2 executes raw: test_op_3
[21:52:01.680] Node node3 sends RPC AppendEntries to Node node4 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=3}
[21:52:01.682] Node node4 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=3}
[APPLY] Node node4 executes raw: test_op_3
[21:52:01.683] Node node3 sends RPC AppendEntries to Node node5 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=3}
[21:52:01.685] Node node5 runs RPC AppendEntries called by Node node3 with {leader_id=node3, term=1, entries=[(k=1,t=1,o='test_op_1'), (k=2,t=1,o='test_op_2'), (k=3,t=1,o='test_op_3')], commit_index=3}
[APPLY] Node node5 executes raw: test_op_3

📌 Final log state of each node:
🔹 node1 log = [{'index': 1, 'term': 1, 'operation': 'test_op_1'}, {'index': 2, 'term': 1, 'operation': 'test_op_2'}, {'index': 3, 'term': 1, 'operation': 'test_op_3'}]
   executed_operations = ['test_op_1', 'test_op_2', 'test_op_3']
🔹 node2 log = [{'index': 1, 'term': 1, 'operation': 'test_op_1'}, {'index': 2, 'term': 1, 'operation': 'test_op_2'}, {'index': 3, 'term': 1, 'operation': 'test_op_3'}]
   executed_operations = ['test_op_1', 'test_op_2', 'test_op_3']
🔹 node3 log = [{'index': 1, 'term': 1, 'operation': 'test_op_1'}, {'index': 2, 'term': 1, 'operation': 'test_op_2'}, {'index': 3, 'term': 1, 'operation': 'test_op_3'}]
   executed_operations = ['test_op_1', 'test_op_2', 'test_op_3']
🔹 node4 log = [{'index': 1, 'term': 1, 'operation': 'test_op_1'}, {'index': 2, 'term': 1, 'operation': 'test_op_2'}, {'index': 3, 'term': 1, 'operation': 'test_op_3'}]
   executed_operations = ['test_op_1', 'test_op_2', 'test_op_3']
🔹 node5 log = [{'index': 1, 'term': 1, 'operation': 'test_op_1'}, {'index': 2, 'term': 1, 'operation': 'test_op_2'}, {'index': 3, 'term': 1, 'operation': 'test_op_3'}]
   executed_operations = ['test_op_1', 'test_op_2', 'test_op_3']

(venv) E:\distributed-pv>
```

TC3 – Leader Crash Recovery

- Screenshot



TC4 – System Initialization (5 Raft nodes + Microservices of Password App)

- Screenshot

```
raft-node4-1  | [03:55:17.136] [SERVER] Starting RaftNode node4 at 0.0.0.0:7004 with peers = {'node1': 'raft-node1:7001', 'node2': 'raft-node2:7002', 'node3': 'raft-node3:7003', 'node5': 'raft-node5:7005'}
raft-node4-1  | [03:55:17.136] [INIT] Node node4 listening at 0.0.0.0:7004, peers = ['node1', 'node2', 'node3', 'node5']
raft-node4-1  | [03:55:17.148] [RUNNING] Node node4 gRPC server active -> 0.0.0.0:7004
raft-node4-1  | [03:55:23.858] [TIMEOUT] Node node4 (Role.FOLLOWER) election timeout, starting election
raft-node4-1  | [03:55:23.858] [ELECTION] Node node4 becomes candidate for term 1
raft-node4-1  | [03:55:23.858] Node node4 sends RPC RequestVote to Node node1 with {candidate_id=node4, term=1}
raft-node4-1  | [03:55:23.874] [VOTE] Node node1 granted vote to node4 for term 1
raft-node4-1  | [03:55:23.874] Node node4 sends RPC RequestVote to Node node2 with {candidate_id=node4, term=1}
raft-node4-1  | [03:55:23.876] [VOTE] Node node2 granted vote to node4 for term 1
raft-node4-1  | [03:55:23.876] Node node4 sends RPC RequestVote to Node node3 with {candidate_id=node4, term=1}
raft-node4-1  | [03:55:23.879] [VOTE] Node node3 granted vote to node4 for term 1
raft-node4-1  | [03:55:23.879] Node node4 sends RPC RequestVote to Node node5 with {candidate_id=node4, term=1}
raft-node4-1  | [03:55:23.881] [VOTE] Node node5 granted vote to node4 for term 1
raft-node4-1  | [03:55:23.881] [LEADER] Node node4 elected leader for term 1
raft-node4-1  | [03:55:23.881] Node node4 sends RPC AppendEntries to Node node1 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:23.882] Node node4 sends RPC AppendEntries to Node node2 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:23.884] Node node4 sends RPC AppendEntries to Node node3 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:23.886] Node node4 sends RPC AppendEntries to Node node5 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:24.887] Node node4 sends RPC AppendEntries to Node node1 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:24.893] Node node4 sends RPC AppendEntries to Node node2 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:24.896] Node node4 sends RPC AppendEntries to Node node3 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:24.898] Node node4 sends RPC AppendEntries to Node node5 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:25.900] Node node4 sends RPC AppendEntries to Node node1 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:25.905] Node node4 sends RPC AppendEntries to Node node2 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:25.907] Node node4 sends RPC AppendEntries to Node node3 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:25.909] Node node4 sends RPC AppendEntries to Node node5 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:26.911] Node node4 sends RPC AppendEntries to Node node1 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:26.916] Node node4 sends RPC AppendEntries to Node node2 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:26.918] Node node4 sends RPC AppendEntries to Node node3 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:26.920] Node node4 sends RPC AppendEntries to Node node5 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:27.922] Node node4 sends RPC AppendEntries to Node node1 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:27.926] Node node4 sends RPC AppendEntries to Node node2 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:27.928] Node node4 sends RPC AppendEntries to Node node3 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:27.930] Node node4 sends RPC AppendEntries to Node node5 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:28.932] Node node4 sends RPC AppendEntries to Node node1 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:28.934] Node node4 sends RPC AppendEntries to Node node2 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:28.936] Node node4 sends RPC AppendEntries to Node node3 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:28.938] Node node4 sends RPC AppendEntries to Node node5 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:29.940] Node node4 sends RPC AppendEntries to Node node1 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:29.943] Node node4 sends RPC AppendEntries to Node node2 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:29.944] Node node4 sends RPC AppendEntries to Node node3 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:29.946] Node node4 sends RPC AppendEntries to Node node5 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node4-1  | [03:55:30.948] Node node4 sends RPC AppendEntries to Node node1 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node2-1  | [03:56:06.292] Node node2 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node3-1  | [03:56:02.253] Node node3 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node5-1  | [03:56:07.310] Node node5 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node1-1  | [03:56:06.289] Node node1 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node2-1  | [03:56:07.305] Node node2 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node3-1  | [03:56:03.265] Node node3 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node1-1  | [03:56:07.301] Node node1 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node2-1  | [03:56:08.315] Node node2 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node1-1  | [03:56:08.313] Node node1 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node1-1  | [03:56:09.323] Node node1 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node5-1  | [03:56:09.333] Node node5 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node3-1  | [03:56:05.285] Node node3 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node2-1  | [03:56:09.327] Node node2 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node1-1  | [03:56:10.339] Node node1 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node5-1  | [03:56:10.349] Node node5 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node3-1  | [03:56:06.294] Node node3 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node2-1  | [03:56:10.344] Node node2 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node5-1  | [03:56:11.357] Node node5 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node1-1  | [03:56:11.349] Node node1 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node3-1  | [03:56:07.308] Node node3 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node2-1  | [03:56:11.352] Node node2 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node1-1  | [03:56:12.362] Node node1 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}

raft-node5-1  | [03:56:12.371] Node node5 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node2-1  | [03:56:12.365] Node node2 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node3-1  | [03:56:08.317] Node node3 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
raft-node1-1  | [03:56:13.374] Node node1 runs RPC AppendEntries called by Node node4 with {leader_id=node4, term=1, entries=[], commit_index=0}
```

TC5 – Add Secret via Raft

- Screenshot

Lessons Learned & Implementation Challenges

During implementation, we discovered that using the standard Raft election timeout configuration of 1.5–3.0 seconds (as recommended in theoretical Raft papers and assignment specification) led to frequent split-vote occurrences and unstable leader selection behavior.
This is likely due to simulation on a single local PC, where multiple Raft nodes compete for computational resources, introducing timing inconsistencies.

To address this problem, we extended the election timeout to 5.0–9.0 seconds, which:

- Eliminated unstable election cycles
- Allowed clean observation of leader re-election after failures
- Prevented frequent split votes during startup