CSE5306 PA3 — Fault-Tolerant Extensions Using 2PC and Raft

Faisal Ahmad
Student ID: 1002239354

Project Repository

Link

1 Introduction

This report documents the design and implementation of Two-Phase Commit (2PC) and Raft consensus algorithms added to the base distributed polling system from CSE5306 Project 2 (Made by Group-3). The work extends a REST-based polling backend with two independent fault-tolerance mechanisms.
Link to Original Base Project Repository

2 Original Base Project

Summary of base project:

- REST API implemented using Hono (Node.js)
- Stateless backend containers (api1, api2)
- PostgreSQL single database instance
- Nginx load balancer in front of APIs
- Standard CRUD for polls and votes

The base system had **no fault-tolerant write mechanism**. Both 2PC and Raft implementations extend the **CreatePoll** path only.

3 Two-Phase Commit (Q2)

3.1 System Architecture

The 2PC version adds:

- One Coordinator (inside `api/`)
- Four Participant Nodes (`two_pc_participant/`)
- gRPC communication via `two_pc.proto`

Data flow for `/polls`:

1. API receives poll creation request
2. API starts a 2PC transaction:
    - Sends `RequestVote` to all participants
    - If all vote YES: send `SendDecision(COMMIT)`
    - Else: `SendDecision(ABORT)`
3. API writes poll to DB only after COMMIT

This ensures atomicity.

4 Raft Leader Election (Q3)

4.1 Architecture

Implemented in:

`base_rest_raft/raft_node/src/server.ts`

Each of 5 Raft nodes:

- Starts as follower
- Waits randomized 1.5–3.0s election timeout
- If no heartbeat: becomes candidate
- Increments term
- Votes for itself
- Broadcasts RequestVote to peers
- If receives majority: becomes leader
- Sends heartbeat every 1s

The API communicates with Raft using the entry node.

5 Raft Log Replication (Q4)

5.1 Operation Flow

When a client creates a poll:

1. API sends `HandleClient` RPC to Raft
2. If caller is not leader: leader redirect
3. Leader appends log entry (`CREATE_POLL`)
4. Leader sends AppendEntries (log + commitIndex)
5. Followers overwrite log with leader's log and apply committed entries
6. Leader updates commit index immediately (assignment simplification)
7. API writes poll to DB after commit

This satisfies Q4 requirements.

6 Test Cases (Q5)

All logs are stored in:

`base_rest_raft/logs/`

Test Case 1 — Normal Operations

- Start cluster normally
- Submit poll
- Leader receives operation, replicates, commits

```
[TC1] Normal Raft operation: leader election + create poll
[TC1] Log file: logs\test1_normal_20251123_173639.log

[TC1] Creating poll via load balancer on :3005...
curl -X POST http://localhost:3005/polls -H "Content-Type: application/json" -d "{\"question\":\"TC1: normal?\",\"options\":[\"yes\",\"no\"]}"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   180  100   130  100    50   2436    937 --:--:-- --:--:-- --:--:--  3396
{"id":26,"question":"TC1: normal?","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:36:39.616Z","raftLogIndex":2}
[TC1] Done. Check logs for leader election and log application.
[TC1] Now you can run:
  docker compose logs api1 --tail=40
  docker compose logs raft3 --tail=80
```

Test Case 2 — Leader Crash

- Identify leader using logs
- `docker compose stop raftX`
- Observe election timeout
- New leader chosen
- Submit poll → success

```
[TC2] Leader crash and re-election test
[TC2] Using assumed leader service: raft3
[TC2] Log file: logs\test2_leader_crash_20251123_173655.log

[TC2] Stopping leader container: raft3 ...
 Container raft_node3  Stopping
 Container raft_node3  Stopped

[TC2] Immediately trying to create a poll (may fail or be rejected)...
curl -X POST http://localhost:3005/polls ...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   204  100   142  100    62   5061   2210 --:--:-- --:--:-- --:--:--  7285
{"id":27,"question":"TC2: after leader crash?","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:36:56.278Z","raftLogIndex":3}
[TC2] Waiting 5 seconds for re-election...

[TC2] Trying again after re-election...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   202  100   141  100    61   6026   2607 --:--:-- --:--:-- --:--:--  8782
{"id":28,"question":"TC2: after re-election?","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:37:01.311Z","raftLogIndex":4}
[TC2] Now check raft node logs for a new leader being elected.
```

Test Case 3 — Follower Recovery

- Stop follower
- Leader continues operating
- Restart follower
- Leader sends AppendEntries
- Log catches up

```
[TC3] Follower crash and recovery (log catch-up)
[TC3] Using follower service: raft4
[TC3] Log file: logs\test3_follower_recovery_20251123_173814.log

[TC3] Stopping follower: raft4 ...
 Container raft_node4  Stopping
 Container raft_node4  Stopped

[TC3] Creating 2 polls while follower is DOWN...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   202  100   141  100    61   3615   1564 --:--:-- --:--:-- --:--:--  5315
{"id":29,"question":"TC3: while raft4 down 1","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:15.977Z","raftLogIndex":5}  % Total    % Received % Xferd  Average Speed
Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   202  100   141  100    61   7856   3399 --:--:-- --:--:-- --:--:-- 11882
{"id":30,"question":"TC3: while raft4 down 2","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:16.032Z","raftLogIndex":6}
[TC3] Starting follower again: raft4 ...
 Container raft_node4  Starting
 Container raft_node4  Started

[TC3] Waiting 5 seconds for it to receive AppendEntries...

[TC3] Now manually run: docker compose logs raft4 --tail=80
[TC3] Look for: Applying log index=... op=CREATE_POLL
```

Test Case 4 — Concurrent Poll Creation

- Rapid `curl` requests
- Log shows proper ordering and replication

```
[TC4] Multiple back-to-back client polls (log ordering test)
[TC4] Log file: logs\test4_concurrent_20251123_173835.log

[TC4] Creating poll 1 ...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   180  100   130  100    50   6048   2326 --:--:-- --:--:-- --:--:--  8571
{"id":31,"question":"TC4: poll 1?","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:35.807Z","raftLogIndex":7}
[TC4] Creating poll 2 ...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   180  100   130  100    50   7429   2857 --:--:-- --:--:-- --:--:-- 10588
{"id":32,"question":"TC4: poll 2?","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:35.892Z","raftLogIndex":8}
[TC4] Creating poll 3 ...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   180  100   130  100    50   6293   2420 --:--:-- --:--:-- --:--:--  9000
{"id":33,"question":"TC4: poll 3?","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:35.988Z","raftLogIndex":9}
[TC4] Creating poll 4 ...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   181  100   131  100    50   7915   3021 --:--:-- --:--:-- --:--:-- 11312
{"id":34,"question":"TC4: poll 4?","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:36.079Z","raftLogIndex":10}
[TC4] Creating poll 5 ...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   181  100   131  100    50   8434   3219 --:--:-- --:--:-- --:--:-- 12066
{"id":35,"question":"TC4: poll 5?","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:36.162Z","raftLogIndex":11}
[TC4] Now check logs on leader and follower for sequential log indexes.
[TC4] Example:
  docker compose logs raft3 --tail=80
  docker compose logs raft4 --tail=80
```

Test Case 5 — New Node Joining

- `docker compose up -d raft5` after removal
- Node boots as follower
- Receives full log on heartbeat
- Catches up

```
[TC5] New node entering the system (late-joining Raft node)
[TC5] Using node: raft5
[TC5] Log file: logs\test5_new_node_join_20251123_173847.log

[TC5] Stopping raft5 so it is OUT of the cluster...
 Container raft_node5  Stopping
 Container raft_node5  Stopped

[TC5] Creating 3 polls while raft5 is down...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100    62    0     0  100    62      0    303 --:--:-- --:--:-- --:--:--   302
100    62    0     0  100    62      0     51  0:00:01  0:00:01 --:--:--    51
100    62    0     0  100    62      0     27  0:00:02  0:00:02 --:--:--    27
100    62    0     0  100    62      0     19  0:00:03  0:00:03 --:--:--    19
100    62    0     0  100    62      0     14  0:00:04  0:00:04 --:--:--    14
100   188  100   126  100    62     25     12  0:00:05  0:00:05 --:--:--    26
{"error":"Raft cluster did not accept operation","details":"14 UNAVAILABLE: Name resolution failed for target dns:raft5:6000"}  % Total    % Received % Xferd  Average Speed   Time    Time
Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   205  100   143  100    62   5142   2229 --:--:-- --:--:-- --:--:--  7321
{"id":36,"question":"TC5: before raft5 join 2","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:54.191Z","raftLogIndex":12}  % Total    % Received % Xferd  Average Speed
Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   205  100   143  100    62   5649   2449 --:--:-- --:--:-- --:--:--  8200
{"id":37,"question":"TC5: before raft5 join 3","options":["yes","no"],"isActive":true,"createdAt":"2025-11-23T23:38:54.259Z","raftLogIndex":13}
[TC5] Starting raft5 back (late join)...
 Container raft_node5  Starting
 Container raft_node5  Started

[TC5] Waiting 5 seconds for it to sync logs via AppendEntries...

[TC5] Now manually run: docker compose logs raft5 --tail=100
[TC5] Look for: Applying log index=1..3 op=CREATE_POLL
```

7 Directory Structure Summary

Full structure included exactly as submitted (see appendix in repo).

8 Implementation Notes and Challenges

- Election storms occurred early due to short initial timeouts
- Fixed via correct timeout randomization
- Database commits triggered only on successful Raft commit
- 2PC path isolated and independent from Raft path

9 Conclusion

This project successfully implemented:

- Fully working 2PC protocol
- Fully working Raft cluster with leader election
- Log replication system
- Integration with consensus mechanisms into REST stack
- Five documented test cases verifying correctness

All Q2–Q5 requirements were satisfied.

Appendix: Log Files

Located in:

`base_rest_raft/logs/`