A Project Report on UBMarketplace

Phase 4
Distributed Systems
Master's in Computer Science

MAY 6, 2022

UNIVERSITY AT BUFFALO

INDEX

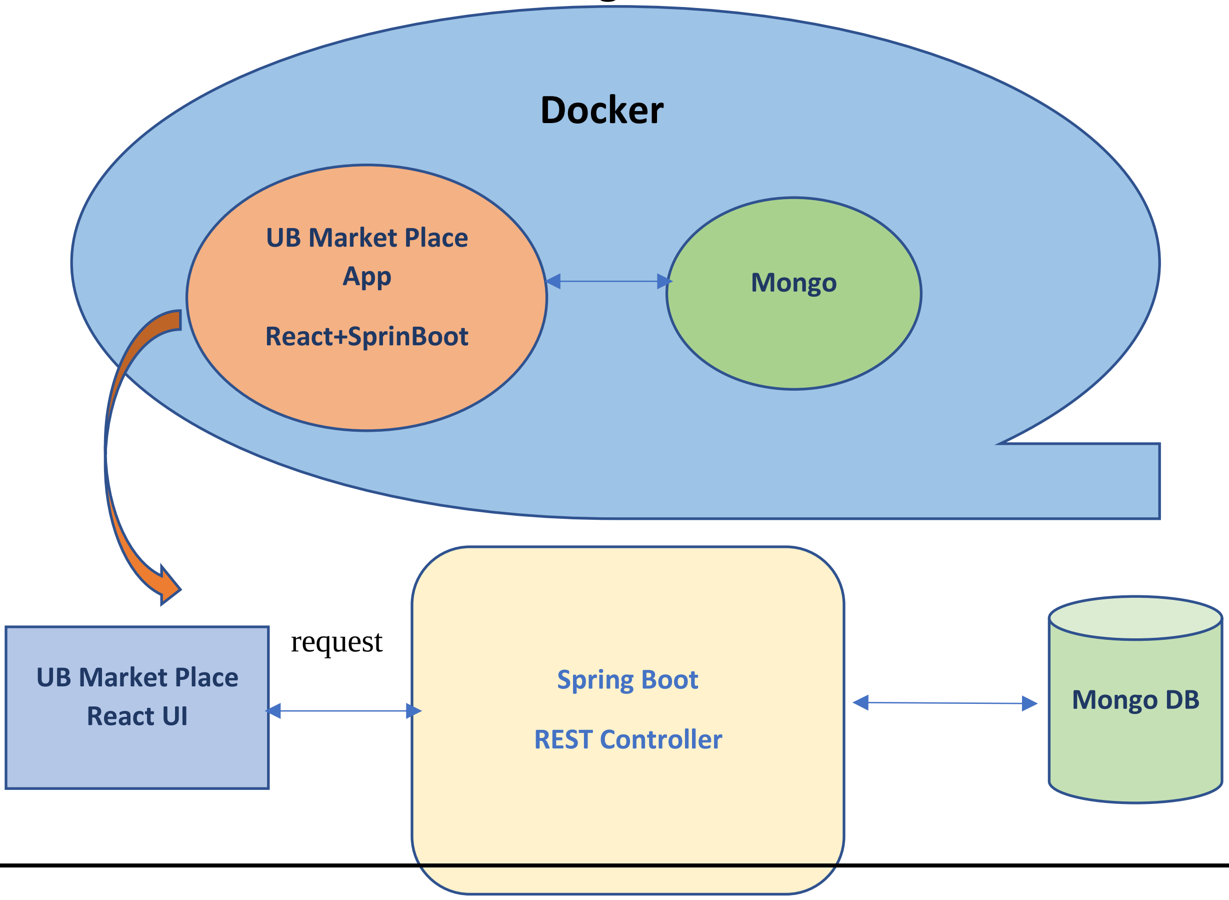
Introduction

The idea of our project is to resolve the issue that the incoming students or current students at the university at buffalo might face. Let's consider Sara who will be joining the college and is interested in staying off-campus. She needs all the necessary goods for a proper stay like a study table, a chair, a desktop, or even a bicycle and so on. Buying all the things on her own might become a huge burden as they cost a lot. What if UB has a marketplace where students trade things at a cheaper price. This is exactly where our idea aroused as we have faced this issue and thought as many students will eventually leave the campus after their graduation and wanted to sell their products which both incoming and outgoing students of UB could benefit from. Our UB Market Place will be their one-stop shop for trading all the student-related pre-owned products at the best prices. This is all about the project that we are trying to build.

Design Overview

Fig. 1.1

A single Node



response

Above Fig 1.1 shows the design of a single node or container and its working design.

In this phase, we enhanced the project by replicating the containers to 5 nodes and making them work in different servers which are interconnected for Leader Election, Heartbeats, log replication, and Timeouts in RAFT to achieve consensus. Each node runs on 8080, 8081, 8082, 8083, 8084 ports respectively with node names as node1, node2, node3, node4, node5.

Below Fig 1.2 represents 5 nodes in the system connected to achieve consensus, and the leader election mechanism, log replication is explained below.

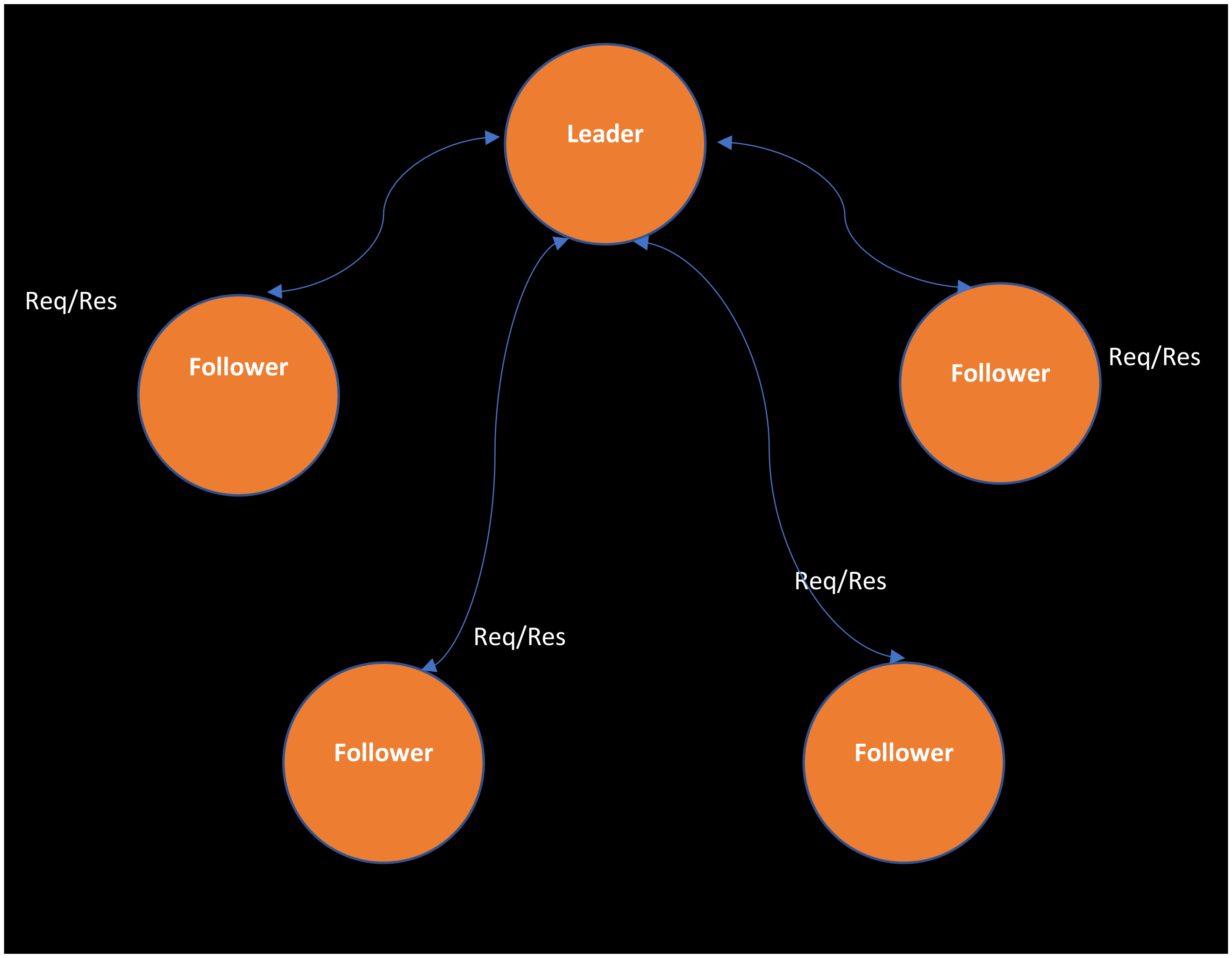


Fig 1.2 Nodes Communication through leader

Each node is connected and communicates over Sockets using UDP (User Datagram Protocol).

Working Principle

As shown in the above figure 1.2, nodes/state machines are connected to implement consensus in RAFT. At any point in time, at most one node acts as the leader which monitors and forwards requests to each node and makes sure the logs and commits of the majority of followers are in sync. Each node in the system consists of a currentTerm, votedFor, timeoutInterval, and heartBeatInterval, log parameters (PrevIndex, LogIndex, prevlogterm, Commitindex, term, success) to store the values at each instance of changes in the node role.

Parameters	Description
currentTerm	Represents the term number and updates the counter for every election.
votedFor	Represents the vote for which the current node voted.
timeoutInterval	Time interval which makes a node timeout for a certain range of time. Here we generated time randomly in a range of 500ms and 1000ms to make the node timeout.
heartBeatInterval	Range of time in which heartbeat is sent in equal intervals.
Log	Entries of commands of every node to be performed are sent by the leader to the follower nodes.
Prevlogindex	Log index of the leader before committing the current request

Prevlogterm	Previous Term of the leader
CommitIndex	Index of committed requests in the current term.
Term	The current term of the consensus
Success	Committed or not?

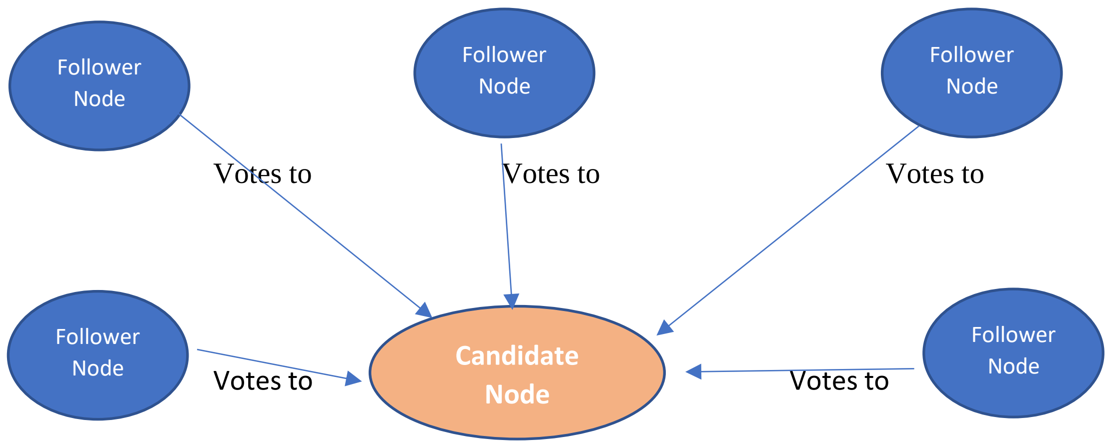
At the start of the servers, currentTerm, and votedFor are null since the leader election does not happen. All the nodes in the system are set as followers at first. The timeout counter starts, once the timeout is set to zero for a node, the leader election starts with updating the currentTerm counter by 1 and starts the election. The timed-out node is set as a CANDIDATE where this node requests a vote from the other nodes by incrementing their terms. Followers send vote acknowledgment to candidates. Follower node may reject vote based on the below condition:

```
If (lastTermV > lastTermC) OR (lastTermV == lastTermC && lastIndexV > lastIndexC)
```

Once the candidate receives the majority number of votes i.e., half the number of nodes +1, then the candidate will act as leader. This leader handles requests from the client and forwards them to its followers in the system. Parallelly, heartbeats are sent to followers by the leader for every heartbeat interval time. If heartbeats are not sent within the set time interval, then it is assumed as the leader is not active, or if a follower is timed out, then a new term leader election starts, and these repeats. There may be a condition of time-outs for multiple nodes, hence those acts as candidates.

For every term, elected leader checks for the logs of their followers by comparing followers' log term and log index. If it is observed that the logs of the follower are not in sync with the leader. Then hence leader decrements its log index by 1 and compares the log. This decremental process continues till the respective follower node matches with the leader's log. Once the log index matches with the leader, then incrementally nowhere updates the follower logs. The same process is applied to every follower for every new term. Once the logs of the majority of followers are updated and are in sync, then the current request or command from the client will be committed.

Leader Election:



Follower votes to the candidate only if the below condition is True:

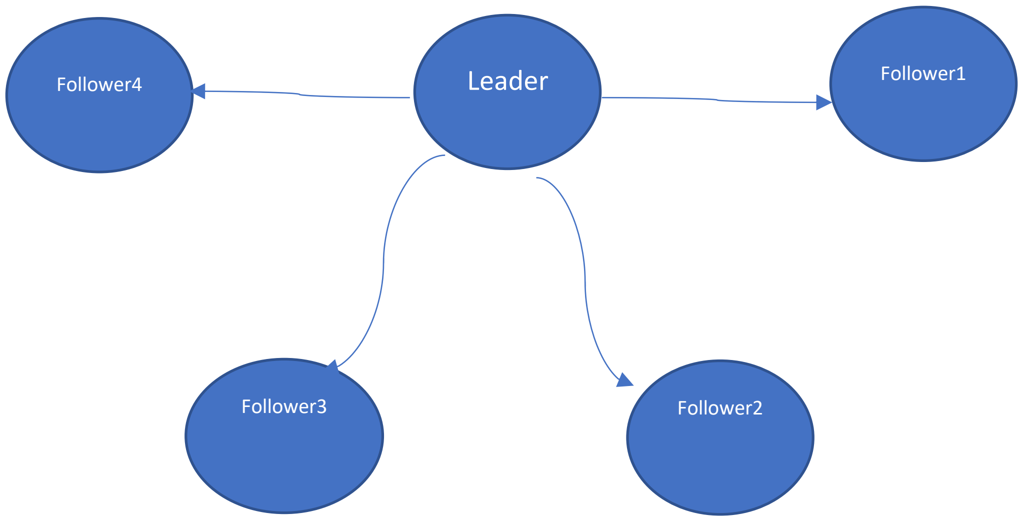
If (lastTermV > lastTermC) OR (lastTermV == lastTermC && lastIndexV > lastIndexC)

For each election, the term field is incremented and the candidate node which gets the highest votes is set as leader and others as follower nodes. Timeout interval, heartbeat sender, the listener at followers run in separate threads

Listeners are the followers who acknowledge the heartbeats sent by the leader of the system. These timeout checkers, heartbeats, and listeners will run on separate threads.

Log replication:

index:	1	2	3	4
Log of leader:	Cmd1	Cmd2	Cmd3	



index:	1	2	3	4
Log of follower2:	Cmd1			

The leader has access to the log index of all followers:

From the above example the next indexes of followers are 3.

nextIndex -> {node1: 3, node2: 2, node3: 3, node4: 3}

From the above picture, the log of leader and follower2 are not in sync, hence leader decrements the index by 1 i.e., to index = 2, and now compares the log. This is repeated until log indices are the same. Since the log is in sync till index 2, hence incrementally nowhere updates the follower2 logs. The same process is applied to every follower every new term. Once the logs of most followers are updated and are in sync, then the current request or command from the client will be committed.

Technical Implementation

As shown in Fig. 1.1, implementation of the above problem statement mentioned in the introduction is a collaboration of several technological components namely User Interface, Database, API (Application Programming Interface), and Container to deploy the project on the local server.

Technologies Used:

01. ReactJS – To develop a basic User Interface that accepts the Name, ID, and age of the student.
02. Spring Boot Framework – Framework used to develop API to interact with the database.

For Each Node:

ReactJS is used to develop a User Interface that contains a form that accepts Name, UB_ID, and age with some basic validations.

These parameters of students are implemented in student class in which the objects of the student class are used by the Spring Boot application to parse and store data in the database. Database fields are created using the student object itself in the application.

Database properties are incorporated in the application properties file of the application.

In this phase, log replication, leader election, heartbeats, timeouts, client requests are handled.

Interconnection:

Five child Nodes are created i.e., created separate containers for each node. These nodes are interconnected such that they communicate with each other.

Docker- compose.YAML file is updated to connect these nodes using networks parameter.

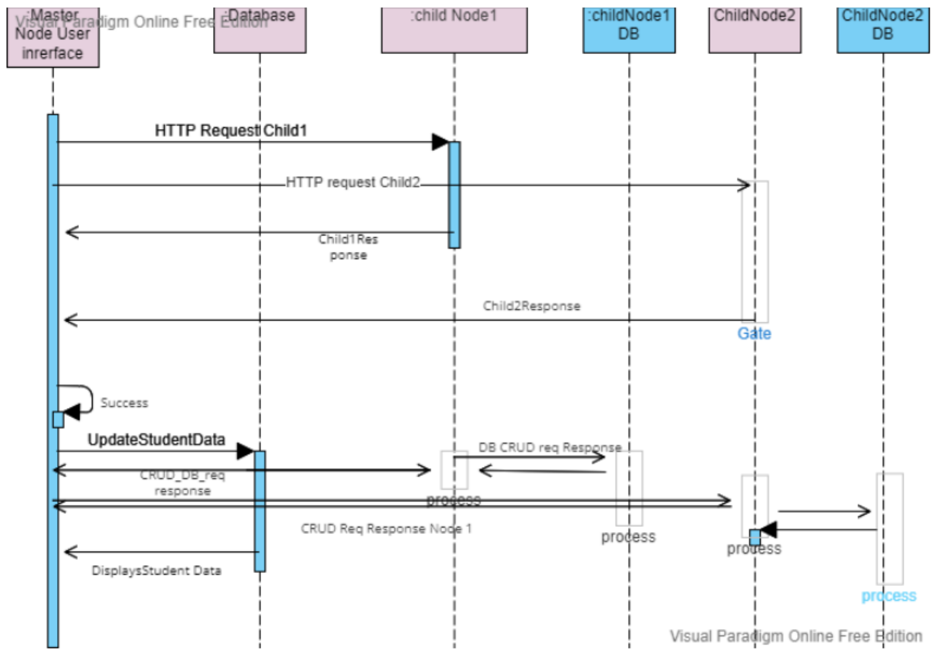
In this phase of implementation, no client requests are made or handled through RAFT. Hence focusing on the internal working of nodes as discussed above.

Finally, Maven is used to cleaning and build the dependencies required for the application.

Phase2 Log VS Phase 4 log replications:

In phase2, the log replication is based on the master node whereas, in the current phase, the log replication is purely based on the consensus i.e., achieved by leader election, heartbeats, timeouts, and log consistency.

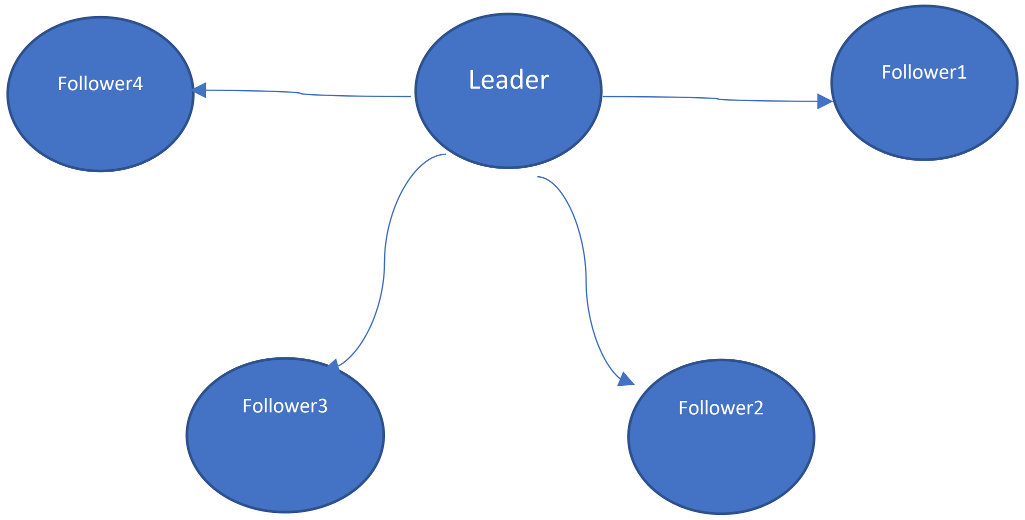
For each term, the leader checks the log of their followers and makes sync with it by checking each log index detrimentally until a log index match is found.



***Designed in external tool**

Phase4: Log replication:

index:	1	2	3	4
Log of leader:	Cmd1	Cmd2	Cmd3	



index:	1	2	3	4
Log of Follower2:	Cmd1	Cmd2		

From the above picture, we now see that there is an increase in the log index in leader so leader updates the logs of followers in next RPC by using the nextIndex as mentioned below. Once the logs of most followers are updated and are in sync, then the current request or command from the client will be committed. This leads to the incrementation of commitIndex.

The leader has access to log index of all followers:

nextIdx -> {node1: 3, node2: 3, node3: 3, node4: 3}

Deployment of Application using docker container

Docker is a platform that enables developing, deploying, and delivering applications in the form of containers that can run irrespective of the infrastructure of the local system, that's the service provided by the docker called OS-level virtualization.

These containers are linked using a docker-compose file which contains the specifications and properties of the Spring Boot application and MongoDB.

Below are the few commands to build and run the container.

01. docker-compose build or maven build.
    To build and link containers mentioned in the docker-compose file.
02. docker-compose up
    To pull and run the containers.

The server can be accessed via http://localhost:8080/

We have tested the conditions using API calls,

We made names for nodes as node1, node2, node3, node4, node5

Controller

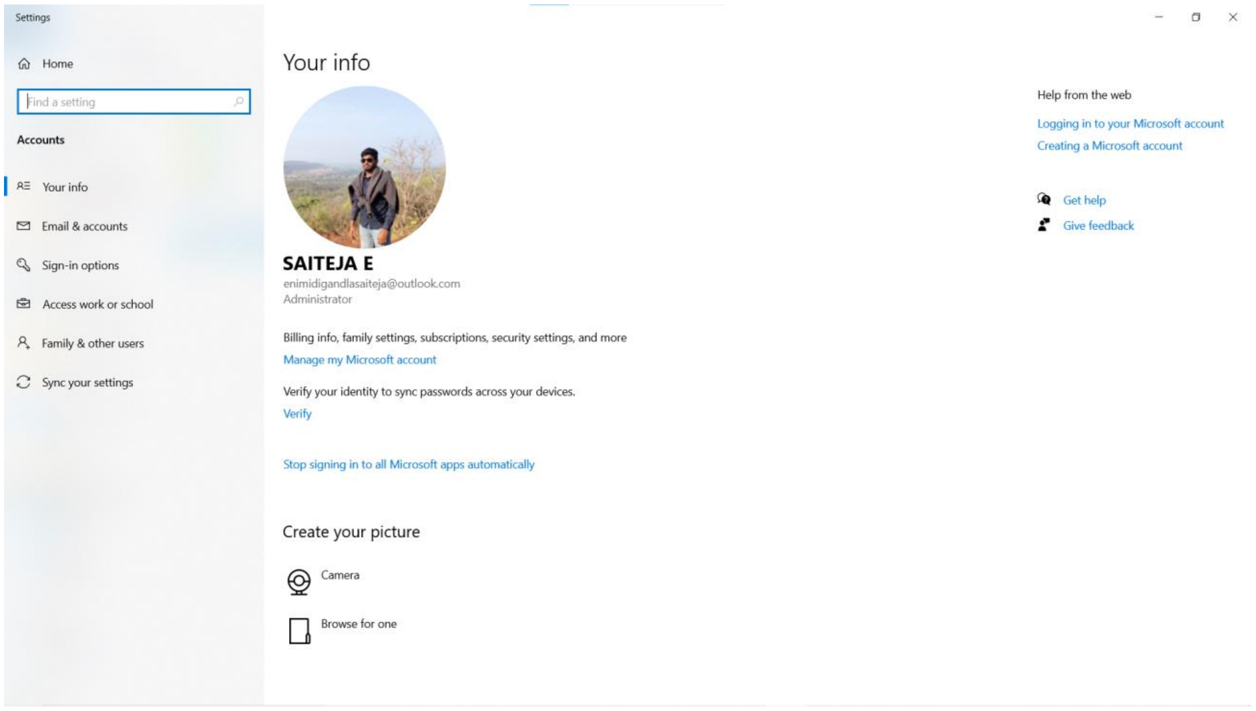
Please check your terminal for all the log information. We will discuss more in our video recording.

- To know which node is the leader use
  http://localhost:8080/getLeaderInfo
- To make a leader as a follower use the following
  http://localhost:8080/convertToFollower?node=node1
- To make a node timeout use the following API
  http://localhost:8080/timeout?node=node2
- To shutdown/ stop a node please use docker application and check the logs in terminal.
- STORE-http://localhost:8080/controller/store?key=4&value=4&node=node2
- RETRIEVE- http://localhost:8080/controller/retrieve?node=node4

*Change node parameter to node1, node2, node3, node4 and key, value params if necessary.

Execution and Validation:

Identity of My Machine



Your info

SAITEJA E
enimidigandlasaiteja@outlook.com
Administrator

Billing info, family settings, subscriptions, security settings, and more
Manage my Microsoft account

Verify your identity to sync passwords across your devices.
Verify

Stop signing in to all Microsoft apps automatically

Create your picture

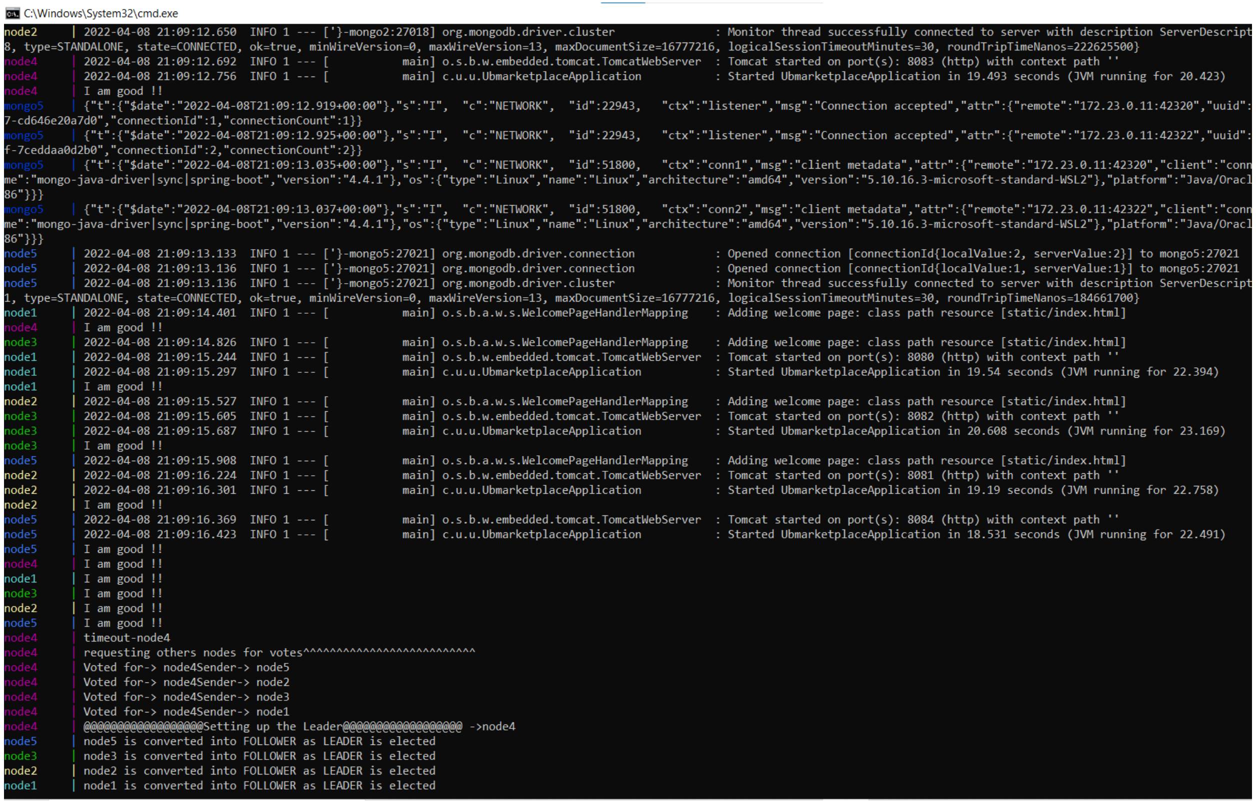
Camera

Browse for one

Help from the web
Logging in to your Microsoft account
Creating a Microsoft account

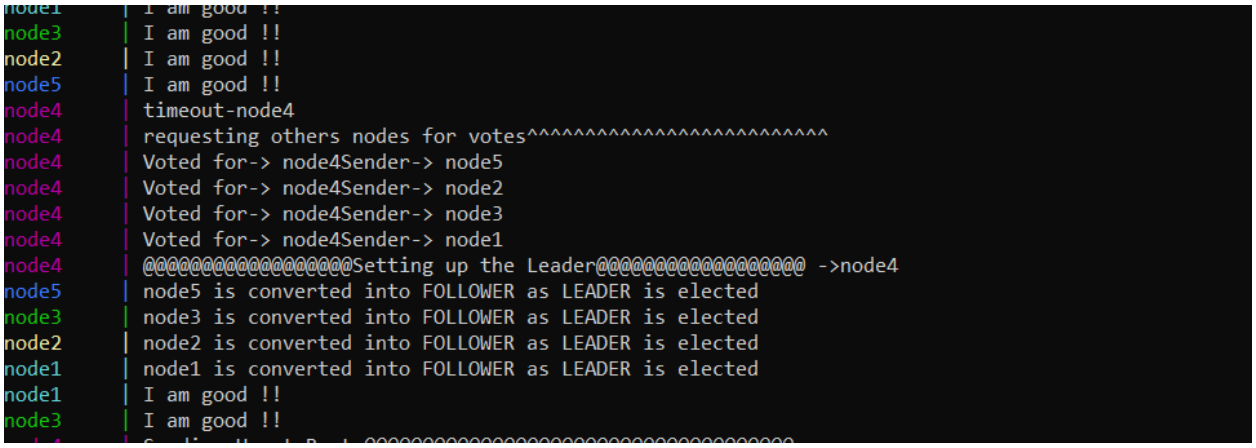
Get help
Give feedback

On start of the nodes:



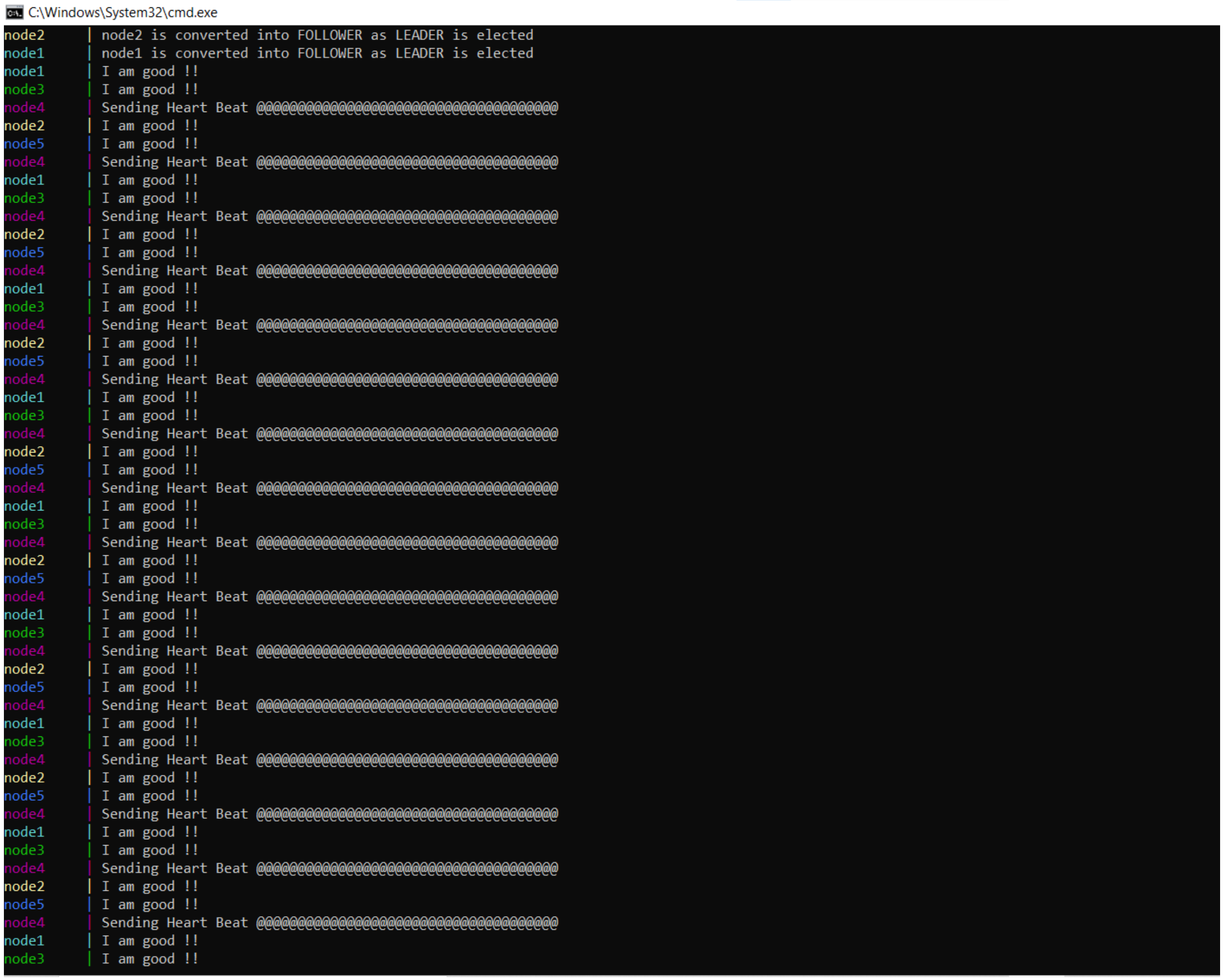
```
C:\Windows\System32\cmd.exe
node2     | 2022-04-08 21:09:12.650  INFO 1 --- ['}-mongo2:27018] org.mongodb.driver.cluster               : Monitor thread successfully connected to server with description ServerDescript
8, type=STANDALONE, state=CONNECTED, ok=true, minWireVersion=0, maxWireVersion=13, maxDocumentSize=16777216, logicalSessionTimeoutMinutes=30, roundTripTimeNanos=222625500}
node4     | 2022-04-08 21:09:12.692  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8083 (http) with context path ''
node4     | 2022-04-08 21:09:12.756  INFO 1 --- [           main] c.u.u.UbmarketplaceApplication           : Started UbmarketplaceApplication in 19.493 seconds (JVM running for 20.423)
node4     | I am good !!
mongo5    | {"t":{"$date":"2022-04-08T21:09:12.919+00:00"},"s":"I",  "c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.23.0.11:42320","uuid":
7-cd646e20a7d0","connectionId":1,"connectionCount":1}}
mongo5    | {"t":{"$date":"2022-04-08T21:09:12.925+00:00"},"s":"I",  "c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.23.0.11:42322","uuid":
f-7ceddaa0d2b0","connectionId":2,"connectionCount":2}}
mongo5    | {"t":{"$date":"2022-04-08T21:09:13.035+00:00"},"s":"I",  "c":"NETWORK",  "id":51800,   "ctx":"conn1","msg":"client metadata","attr":{"remote":"172.23.0.11:42320","client":"conn
me":"mongo-java-driver|sync|spring-boot","version":"4.4.1"},"os":{"type":"Linux","name":"Linux","architecture":"amd64","version":"5.10.16.3-microsoft-standard-WSL2"},"platform":"Java/Oracl
86"}}}
mongo5    | {"t":{"$date":"2022-04-08T21:09:13.037+00:00"},"s":"I",  "c":"NETWORK",  "id":51800,   "ctx":"conn2","msg":"client metadata","attr":{"remote":"172.23.0.11:42322","client":"conn
me":"mongo-java-driver|sync|spring-boot","version":"4.4.1"},"os":{"type":"Linux","name":"Linux","architecture":"amd64","version":"5.10.16.3-microsoft-standard-WSL2"},"platform":"Java/Oracl
86"}}}
node5     | 2022-04-08 21:09:13.133  INFO 1 --- ['}-mongo5:27021] org.mongodb.driver.connection            : Opened connection [connectionId{localValue:2, serverValue:2}] to mongo5:27021
node5     | 2022-04-08 21:09:13.136  INFO 1 --- ['}-mongo5:27021] org.mongodb.driver.connection            : Opened connection [connectionId{localValue:1, serverValue:1}] to mongo5:27021
node5     | 2022-04-08 21:09:13.136  INFO 1 --- ['}-mongo5:27021] org.mongodb.driver.cluster               : Monitor thread successfully connected to server with description ServerDescript
1, type=STANDALONE, state=CONNECTED, ok=true, minWireVersion=0, maxWireVersion=13, maxDocumentSize=16777216, logicalSessionTimeoutMinutes=30, roundTripTimeNanos=184661700}
node1     | 2022-04-08 21:09:14.401  INFO 1 --- [           main] o.s.b.a.w.s.WelcomePageHandlerMapping    : Adding welcome page: class path resource [static/index.html]
node4     | I am good !!
node3     | 2022-04-08 21:09:14.826  INFO 1 --- [           main] o.s.b.a.w.s.WelcomePageHandlerMapping    : Adding welcome page: class path resource [static/index.html]
node1     | 2022-04-08 21:09:15.244  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
node1     | 2022-04-08 21:09:15.297  INFO 1 --- [           main] c.u.u.UbmarketplaceApplication           : Started UbmarketplaceApplication in 19.54 seconds (JVM running for 22.394)
node1     | I am good !!
node2     | 2022-04-08 21:09:15.527  INFO 1 --- [           main] o.s.b.a.w.s.WelcomePageHandlerMapping    : Adding welcome page: class path resource [static/index.html]
node3     | 2022-04-08 21:09:15.605  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8082 (http) with context path ''
node3     | 2022-04-08 21:09:15.687  INFO 1 --- [           main] c.u.u.UbmarketplaceApplication           : Started UbmarketplaceApplication in 20.608 seconds (JVM running for 23.169)
node3     | I am good !!
node5     | 2022-04-08 21:09:15.908  INFO 1 --- [           main] o.s.b.a.w.s.WelcomePageHandlerMapping    : Adding welcome page: class path resource [static/index.html]
node2     | 2022-04-08 21:09:16.224  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8081 (http) with context path ''
node2     | 2022-04-08 21:09:16.301  INFO 1 --- [           main] c.u.u.UbmarketplaceApplication           : Started UbmarketplaceApplication in 19.19 seconds (JVM running for 22.758)
node2     | I am good !!
node5     | 2022-04-08 21:09:16.369  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8084 (http) with context path ''
node5     | 2022-04-08 21:09:16.423  INFO 1 --- [           main] c.u.u.UbmarketplaceApplication           : Started UbmarketplaceApplication in 18.531 seconds (JVM running for 22.491)
node5     | I am good !!
node4     | I am good !!
node1     | I am good !!
node3     | I am good !!
node2     | I am good !!
node5     | I am good !!
node4     | timeout-node4
node4     | requesting others nodes for votes^^^^^^^^^^^^^^^^^^^^^^^^^^
node4     | Voted for-> node4Sender-> node5
node4     | Voted for-> node4Sender-> node2
node4     | Voted for-> node4Sender-> node3
node4     | Voted for-> node4Sender-> node1
node4     | @@@@@@@@@@@@@@@@@@@Setting up the Leader@@@@@@@@@@@@@@@@@@@ ->node4
node5     | node5 is converted into FOLLOWER as LEADER is elected
node3     | node3 is converted into FOLLOWER as LEADER is elected
node2     | node2 is converted into FOLLOWER as LEADER is elected
node1     | node1 is converted into FOLLOWER as LEADER is elected
```

Node1 got timeout4 and requesting modes to other nodes and got the majority votes, hence selected as leader.



```
node1     | I am good !!
node3     | I am good !!
node2     | I am good !!
node5     | I am good !!
node4     | timeout-node4
node4     | requesting others nodes for votes^^^^^^^^^^^^^^^^^^^^^^^^^^
node4     | Voted for-> node4Sender-> node5
node4     | Voted for-> node4Sender-> node2
node4     | Voted for-> node4Sender-> node3
node4     | Voted for-> node4Sender-> node1
node4     | @@@@@@@@@@@@@@@@@@@Setting up the Leader@@@@@@@@@@@@@@@@@@@ ->node4
node5     | node5 is converted into FOLLOWER as LEADER is elected
node3     | node3 is converted into FOLLOWER as LEADER is elected
node2     | node2 is converted into FOLLOWER as LEADER is elected
node1     | node1 is converted into FOLLOWER as LEADER is elected
node1     | I am good !!
node3     | I am good !!
```

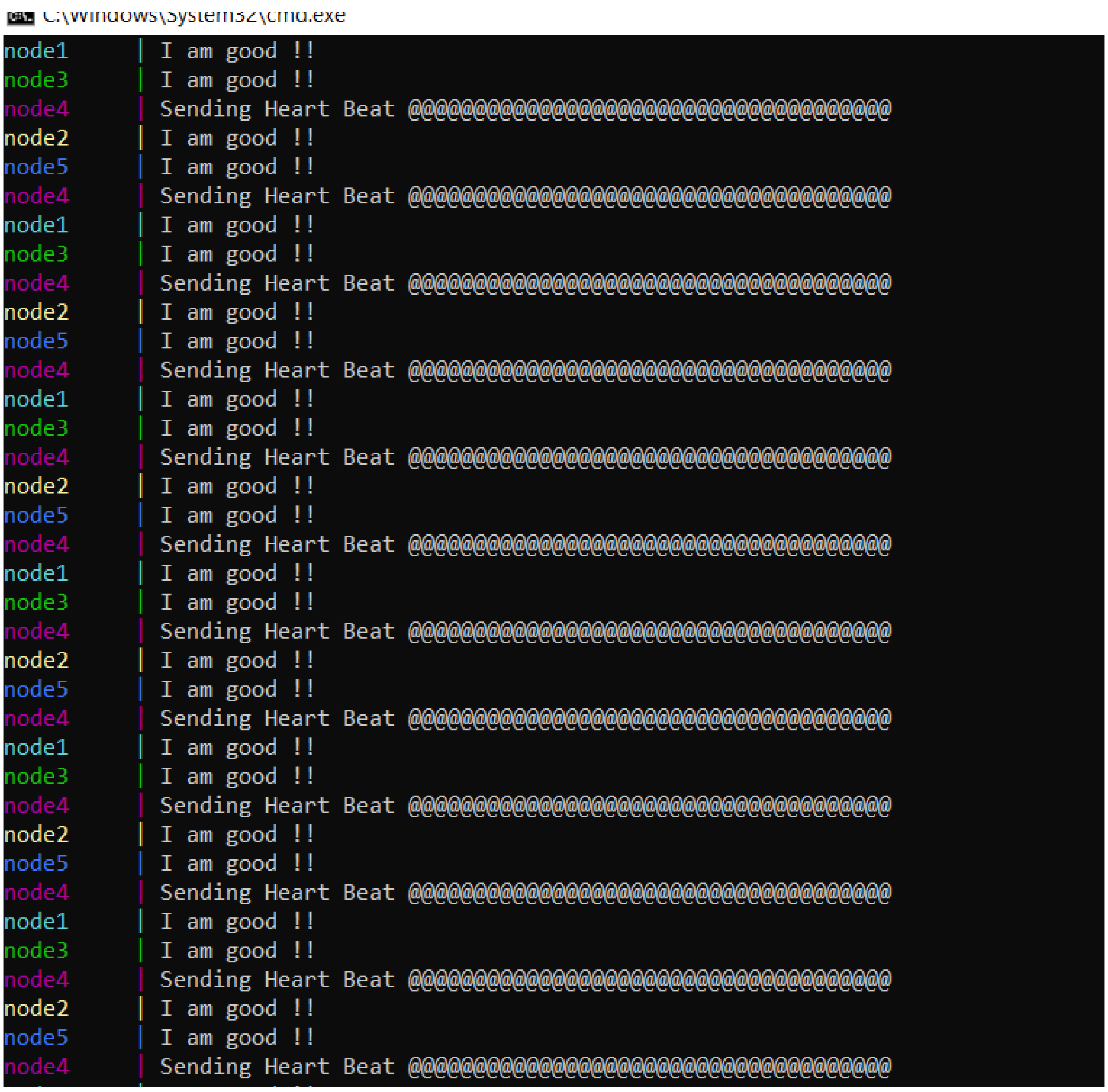
Heartbeats: “Sending Heart Beat”



C:\Windows\System32\cmd.exe

```
node2      | node2 is converted into FOLLOWER as LEADER is elected
node1      | node1 is converted into FOLLOWER as LEADER is elected
node1      | I am good !!
node3      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2      | I am good !!
node5      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node3      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2      | I am good !!
node5      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node3      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2      | I am good !!
node5      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node3      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2      | I am good !!
node5      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node3      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2      | I am good !!
node5      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node3      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2      | I am good !!
node5      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node3      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2      | I am good !!
node5      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node3      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2      | I am good !!
node5      | I am good !!
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node3      | I am good !!
```

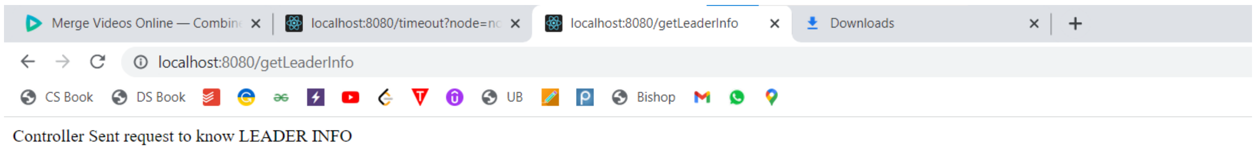
Heartbeats response: “I am good”



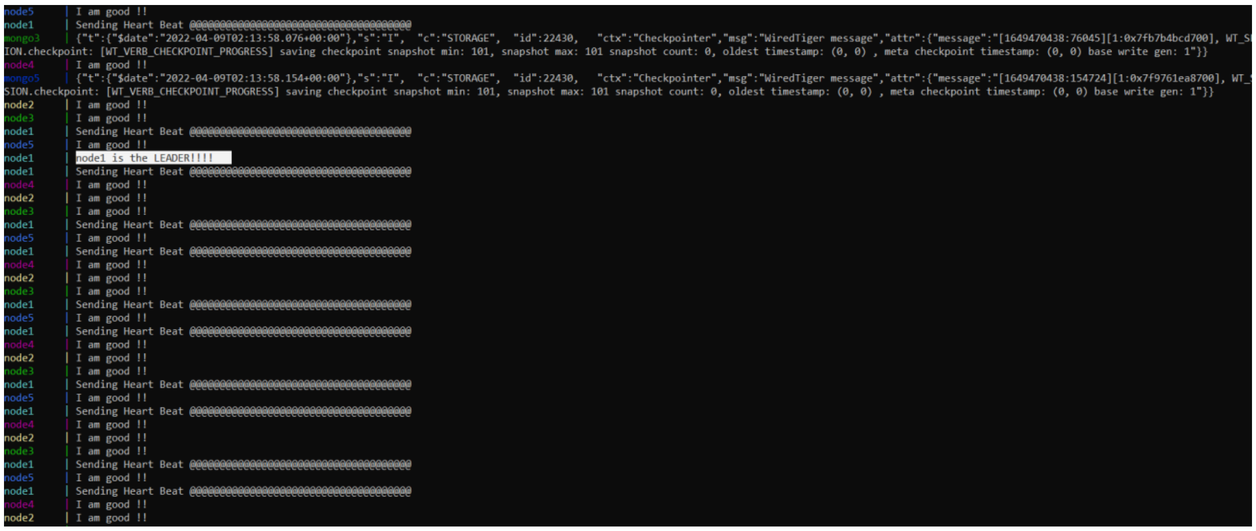
C:\Windows\System32\cmd.exe

```
node1     | I am good !!
node3     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node5     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1     | I am good !!
node3     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node5     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1     | I am good !!
node3     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node5     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1     | I am good !!
node3     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node5     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1     | I am good !!
node3     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node5     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1     | I am good !!
node3     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node5     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

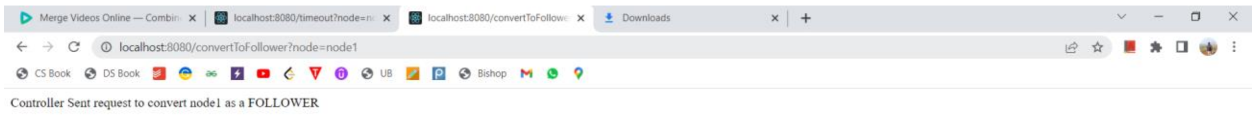
Getting leader info through API call:


localhost:8080/getLeaderInfo

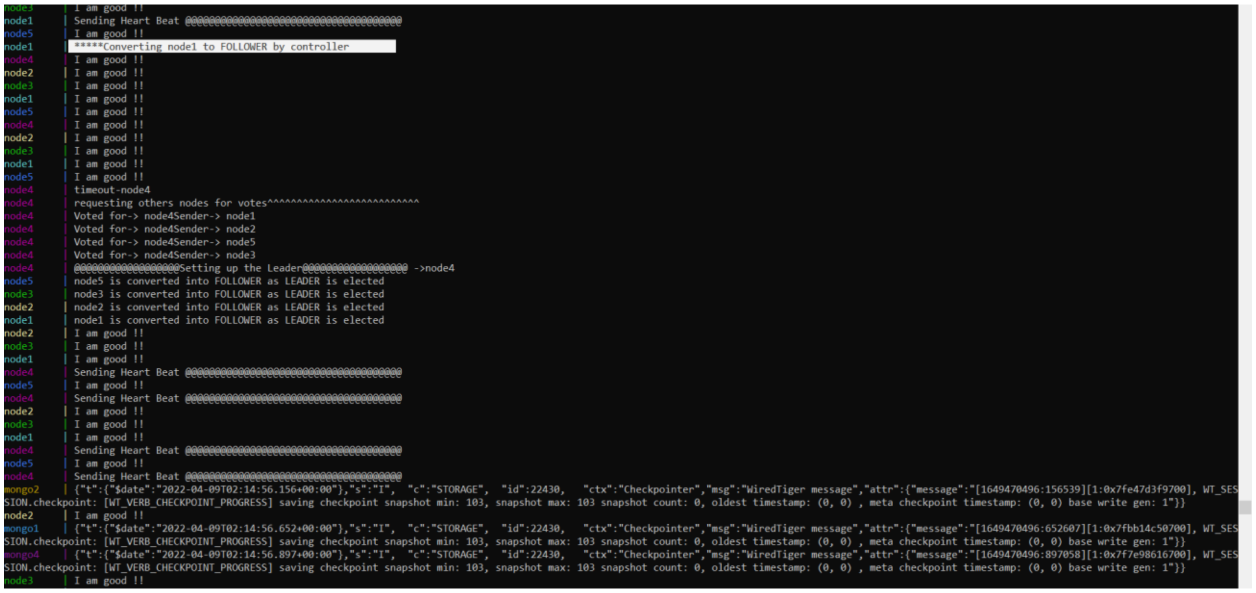
Controller Sent request to know LEADER INFO


```
node5      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
mongo3     | {"t":{"$date":"2022-04-09T02:13:58.076+00:00"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[1649470438:76045][1:0x7fb7b4bcd700], WT_S
ION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 101, snapshot max: 101 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0) base write gen: 1"}}
node4      | I am good !!
mongo5     | {"t":{"$date":"2022-04-09T02:13:58.154+00:00"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[1649470438:154724][1:0x7f9761ea8700], WT_
SION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 101, snapshot max: 101 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0) base write gen: 1"}}
node2      | I am good !!
node3      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5      | I am good !!
node1      | node1 is the LEADER!!!!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4      | I am good !!
node2      | I am good !!
node3      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4      | I am good !!
node2      | I am good !!
node3      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4      | I am good !!
node2      | I am good !!
node3      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4      | I am good !!
node2      | I am good !!
node3      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5      | I am good !!
node1      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4      | I am good !!
node2      | I am good !!
```

Making leader as follower:


localhost:8080/convertToFollower?node=node1

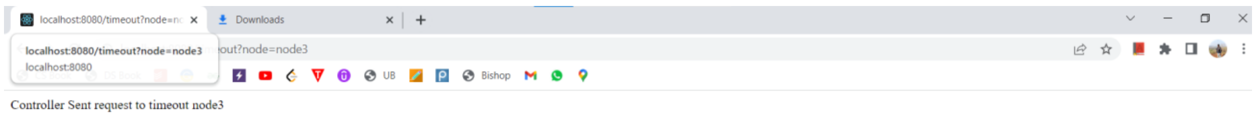
Controller Sent request to convert node1 as a FOLLOWER



```
node3     | I am good !!
node1     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5     | I am good !!
node1     | *****Converting node1 to FOLLOWER by controller
node4     | I am good !!
node2     | I am good !!
node3     | I am good !!
node1     | I am good !!
node5     | I am good !!
node4     | I am good !!
node2     | I am good !!
node3     | I am good !!
node1     | I am good !!
node5     | I am good !!
node4     | timeout-node4
node4     | requesting others nodes for votes^^^^^^^^^^^^^^^^^^^^^^^^^^^
node4     | Voted for-> node4Sender-> node1
node4     | Voted for-> node4Sender-> node2
node4     | Voted for-> node4Sender-> node5
node4     | Voted for-> node4Sender-> node3
node4     | @@@@@@@@@@@@@@@@@@Setting up the Leader@@@@@@@@@@@@@@@@@@ ->node4
node5     | node5 is converted into FOLLOWER as LEADER is elected
node3     | node3 is converted into FOLLOWER as LEADER is elected
node2     | node2 is converted into FOLLOWER as LEADER is elected
node1     | node1 is converted into FOLLOWER as LEADER is elected
node2     | I am good !!
node3     | I am good !!
node1     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node3     | I am good !!
node1     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5     | I am good !!
node4     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
mongo2    | {"t":{"$date":"2022-04-09T02:14:56.156+00:00"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[1649470496:156539][1:0x7fe47d3f9700], WT_SES
SION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 103, snapshot max: 103 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0) base write gen: 1"}}
node2     | I am good !!
mongo1    | {"t":{"$date":"2022-04-09T02:14:56.652+00:00"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[1649470496:652607][1:0x7fbb14c50700], WT_SES
SION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 103, snapshot max: 103 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0) base write gen: 1"}}
mongo4    | {"t":{"$date":"2022-04-09T02:14:56.897+00:00"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[1649470496:897058][1:0x7f7e98616700], WT_SES
SION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 103, snapshot max: 103 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0) base write gen: 1"}}
node3     | I am good !!
```

From the above, we can see that leader node 1 is converted to the follower. Hence a new election is being called and the process is repeated.

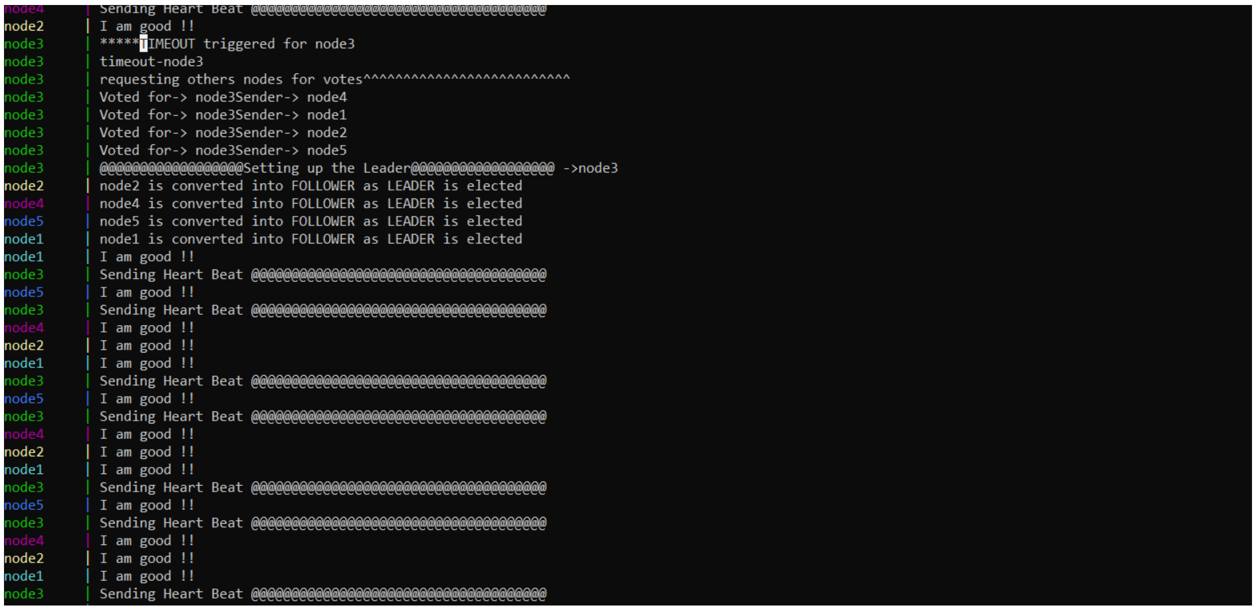
Making node3 timeout.



localhost:8080/timeout?node=nc
Downloads
localhost:8080/timeout?node=node3
localhost:8080
UB
Bishop

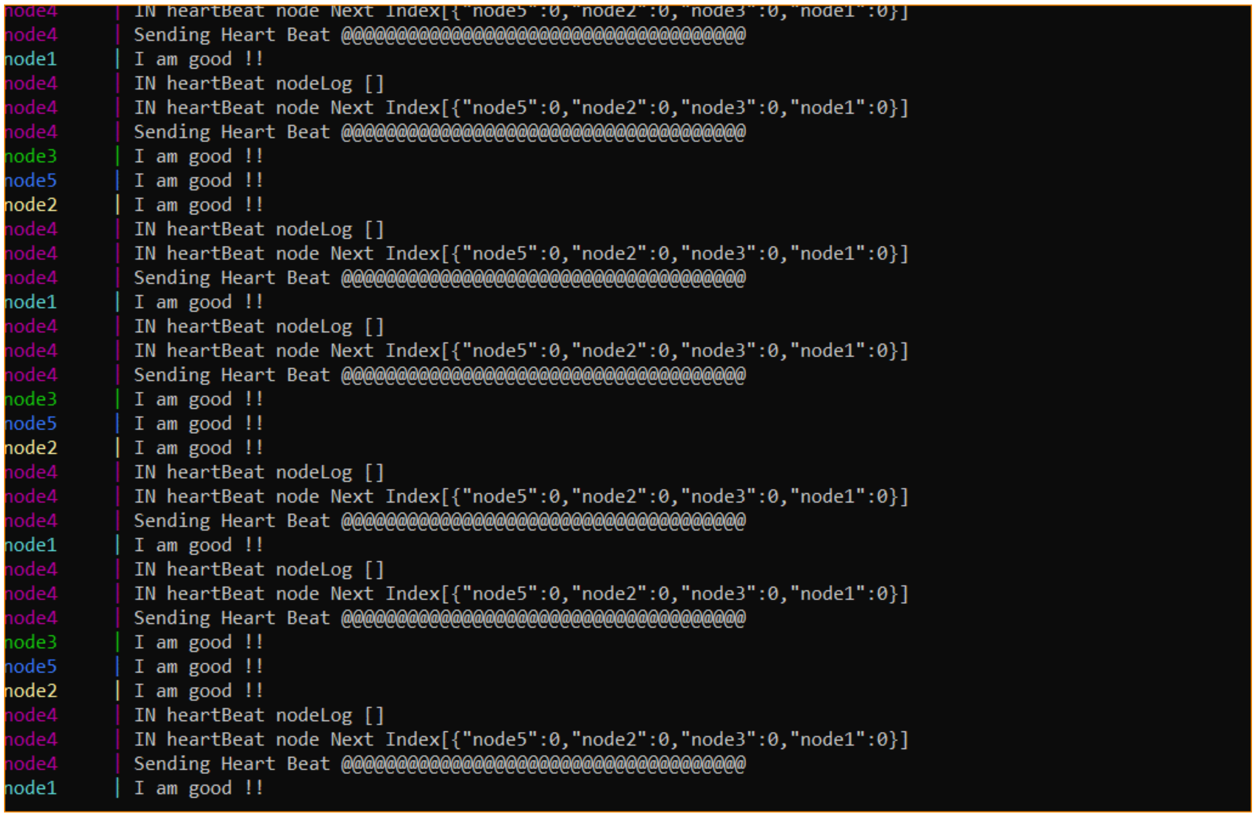
Controller Sent request to timeout node3

Hence a new election was called and the same process is being repeated.



```
node4    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2    | I am good !!
node3    | *****TIMEOUT triggered for node3
node3    | timeout-node3
node3    | requesting others nodes for votes^^^^^^^^^^^^^^^^^^^^^^^^^
node3    | Voted for-> node3Sender-> node4
node3    | Voted for-> node3Sender-> node1
node3    | Voted for-> node3Sender-> node2
node3    | Voted for-> node3Sender-> node5
node3    | @@@@@@@@@@@@@@@@@@Setting up the Leader@@@@@@@@@@@@@@@@@@ ->node3
node2    | node2 is converted into FOLLOWER as LEADER is elected
node4    | node4 is converted into FOLLOWER as LEADER is elected
node5    | node5 is converted into FOLLOWER as LEADER is elected
node1    | node1 is converted into FOLLOWER as LEADER is elected
node1    | I am good !!
node3    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5    | I am good !!
node3    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4    | I am good !!
node2    | I am good !!
node1    | I am good !!
node3    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5    | I am good !!
node3    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4    | I am good !!
node2    | I am good !!
node1    | I am good !!
node3    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5    | I am good !!
node3    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4    | I am good !!
node2    | I am good !!
node1    | I am good !!
node3    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

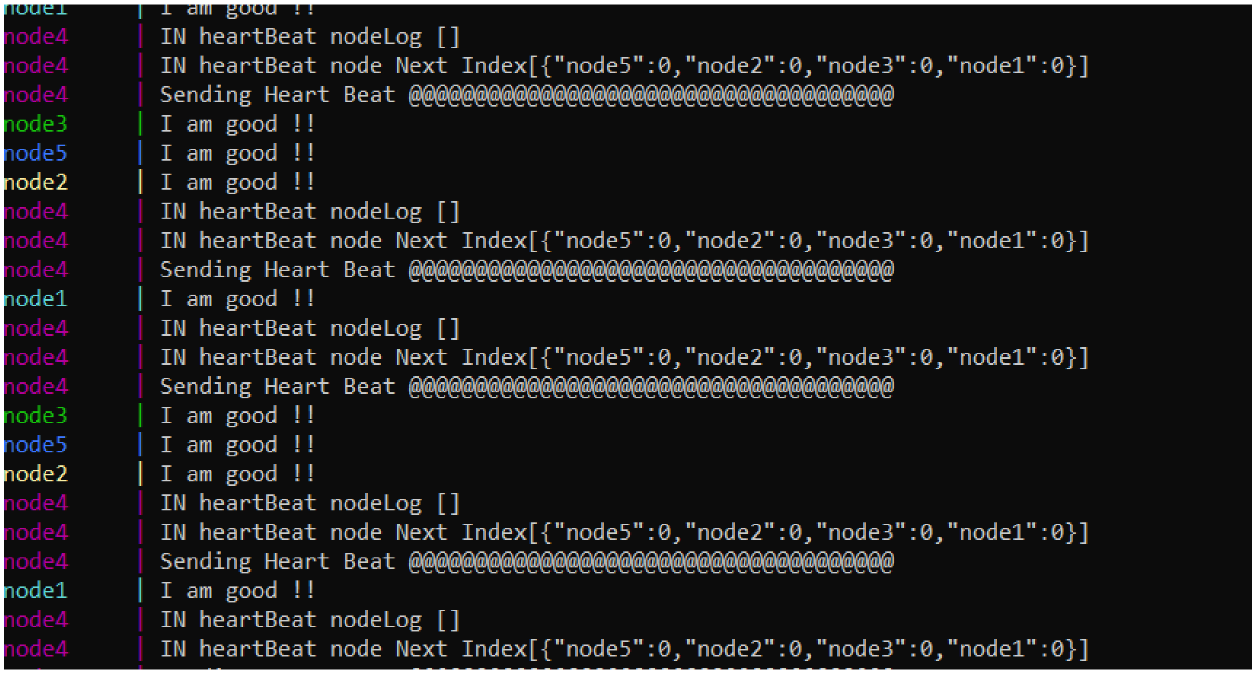
Lets store a key value via leader node, here cuurently leader node is node4:



```
node4    | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1    | I am good !!
node4    | IN heartBeat nodeLog []
node4    | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3    | I am good !!
node5    | I am good !!
node2    | I am good !!
node4    | IN heartBeat nodeLog []
node4    | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1    | I am good !!
node4    | IN heartBeat nodeLog []
node4    | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3    | I am good !!
node5    | I am good !!
node2    | I am good !!
node4    | IN heartBeat nodeLog []
node4    | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1    | I am good !!
node4    | IN heartBeat nodeLog []
node4    | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3    | I am good !!
node5    | I am good !!
node2    | I am good !!
node4    | IN heartBeat nodeLog []
node4    | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4    | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1    | I am good !!
```

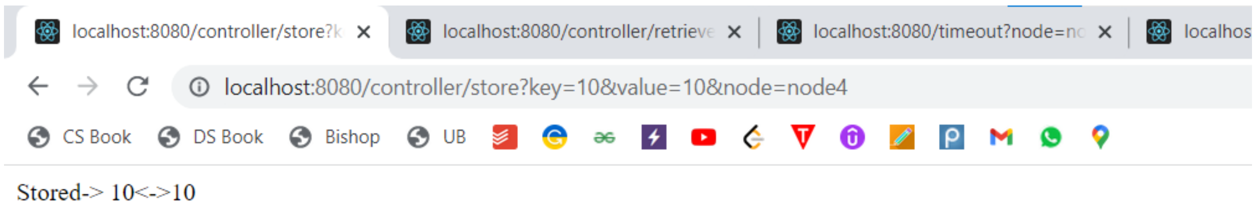
Lets send a store request via node4, since node4 is the leader:

Initially :



```
node1      | I am good !!
node4      | IN heartBeat nodeLog []
node4      | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3      | I am good !!
node5      | I am good !!
node2      | I am good !!
node4      | IN heartBeat nodeLog []
node4      | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node4      | IN heartBeat nodeLog []
node4      | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3      | I am good !!
node5      | I am good !!
node2      | I am good !!
node4      | IN heartBeat nodeLog []
node4      | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node4      | IN heartBeat nodeLog []
node4      | IN heartBeat node Next Index[{"node5":0,"node2":0,"node3":0,"node1":0}]
```

Request submission:

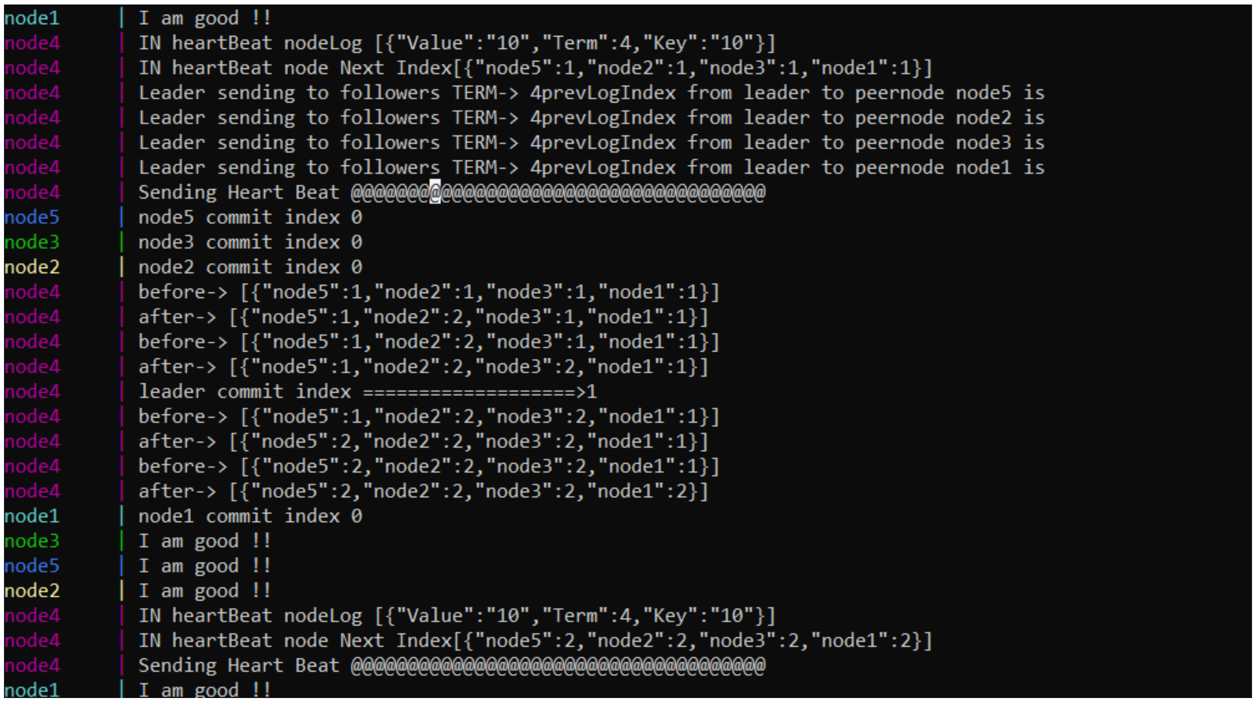


localhost:8080/controller/store?k | localhost:8080/controller/retrieve | localhost:8080/timeout?node=nc | localhos

localhost:8080/controller/store?key=10&value=10&node=node4

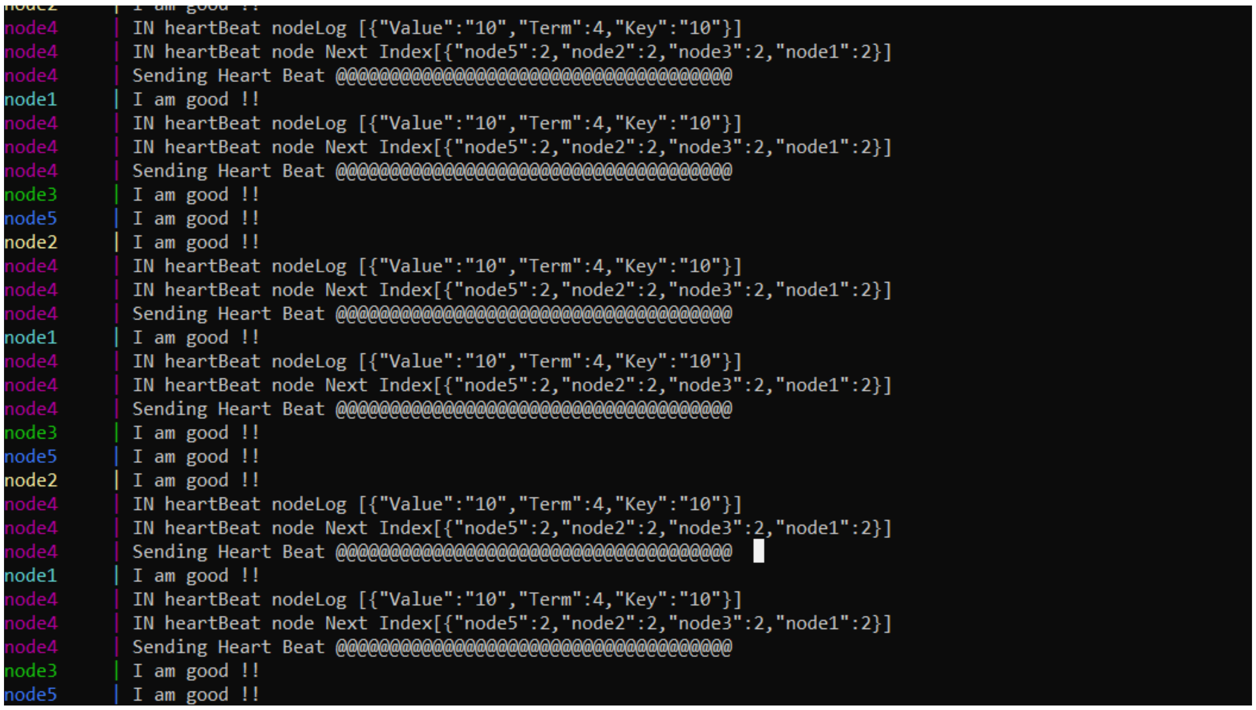
Stored-> 10<->10

Commit Index intially:



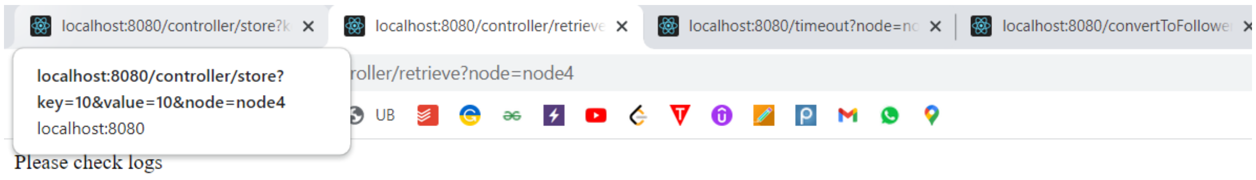
```
node1      | I am good !!
node4      | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4      | IN heartBeat node Next Index[{"node5":1,"node2":1,"node3":1,"node1":1}]
node4      | Leader sending to followers TERM-> 4prevLogIndex from leader to peernode node5 is
node4      | Leader sending to followers TERM-> 4prevLogIndex from leader to peernode node2 is
node4      | Leader sending to followers TERM-> 4prevLogIndex from leader to peernode node3 is
node4      | Leader sending to followers TERM-> 4prevLogIndex from leader to peernode node1 is
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5      | node5 commit index 0
node3      | node3 commit index 0
node2      | node2 commit index 0
node4      | before-> [{"node5":1,"node2":1,"node3":1,"node1":1}]
node4      | after-> [{"node5":1,"node2":2,"node3":1,"node1":1}]
node4      | before-> [{"node5":1,"node2":2,"node3":1,"node1":1}]
node4      | after-> [{"node5":1,"node2":2,"node3":2,"node1":1}]
node4      | leader commit index ===================>1
node4      | before-> [{"node5":1,"node2":2,"node3":2,"node1":1}]
node4      | after-> [{"node5":2,"node2":2,"node3":2,"node1":1}]
node4      | before-> [{"node5":2,"node2":2,"node3":2,"node1":1}]
node4      | after-> [{"node5":2,"node2":2,"node3":2,"node1":2}]
node1      | node1 commit index 0
node3      | I am good !!
node5      | I am good !!
node2      | I am good !!
node4      | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4      | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
```

Log updated:



```
node4      | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4      | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node4      | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4      | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3      | I am good !!
node5      | I am good !!
node2      | I am good !!
node4      | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4      | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node4      | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4      | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3      | I am good !!
node5      | I am good !!
node2      | I am good !!
node4      | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4      | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1      | I am good !!
node4      | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4      | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4      | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3      | I am good !!
node5      | I am good !!
```

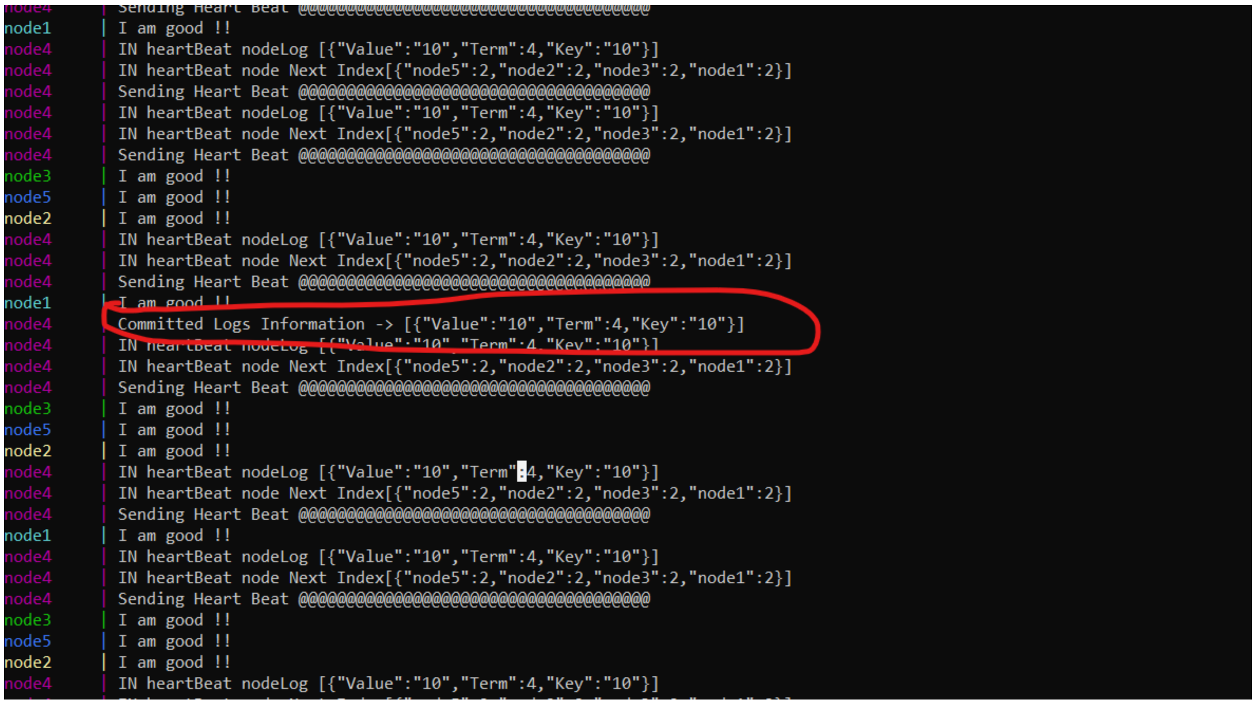
Log retrieval via node4, since node4 is the leader:


localhost:8080/controller/store?k
localhost:8080/controller/retrieve
localhost:8080/timeout?node=nc
localhost:8080/convertToFollowe

localhost:8080/controller/store?key=10&value=10&node=node4
localhost:8080

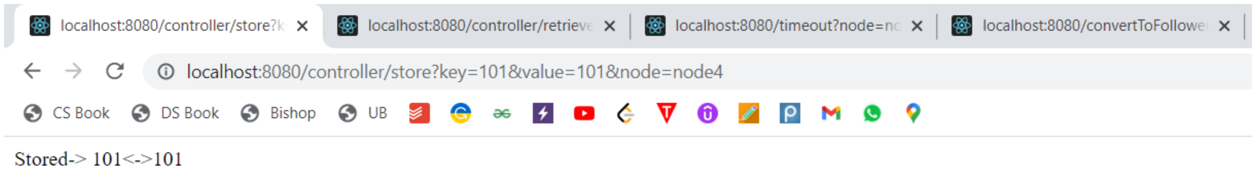
roller/retrieve?node=node4

Please check logs



```
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4       | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4       | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3       | I am good !!
node5       | I am good !!
node2       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4       | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1       | I am good !!
node4       | Committed Logs Information -> [{"Value":"10","Term":4,"Key":"10"}]
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4       | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3       | I am good !!
node5       | I am good !!
node2       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4       | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
node4       | IN heartBeat node Next Index[{"node5":2,"node2":2,"node3":2,"node1":2}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3       | I am good !!
node5       | I am good !!
node2       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}]
```

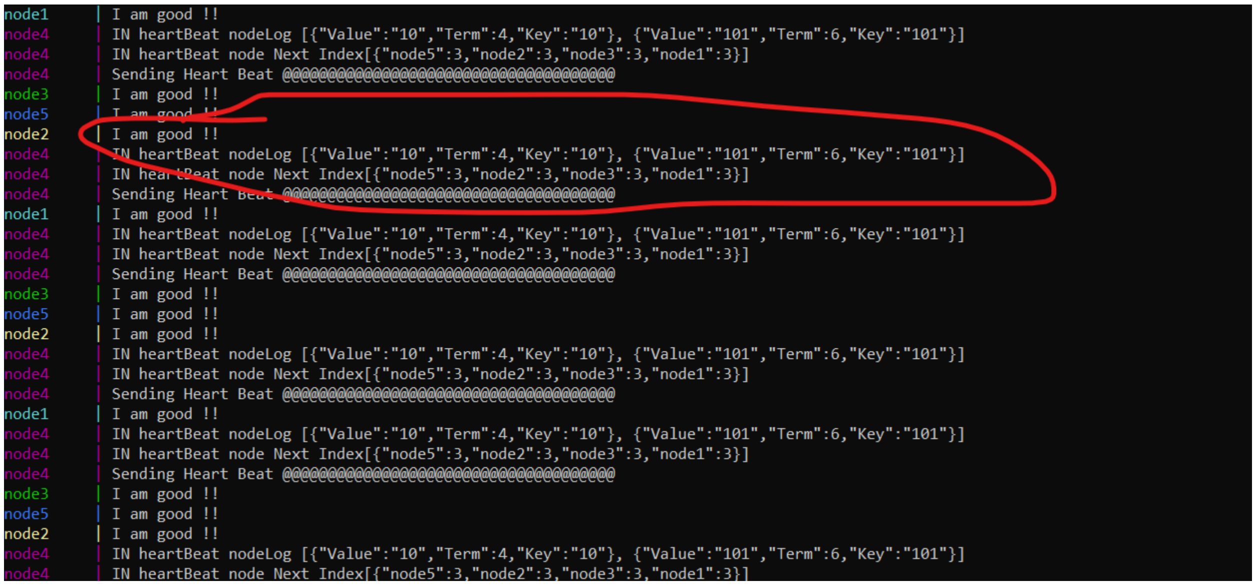
Store once more:



localhost:8080/controller/store?k
localhost:8080/controller/retrieve
localhost:8080/timeout?node=nc
localhost:8080/convertToFollower
CS Book
DS Book
Bishop
UB

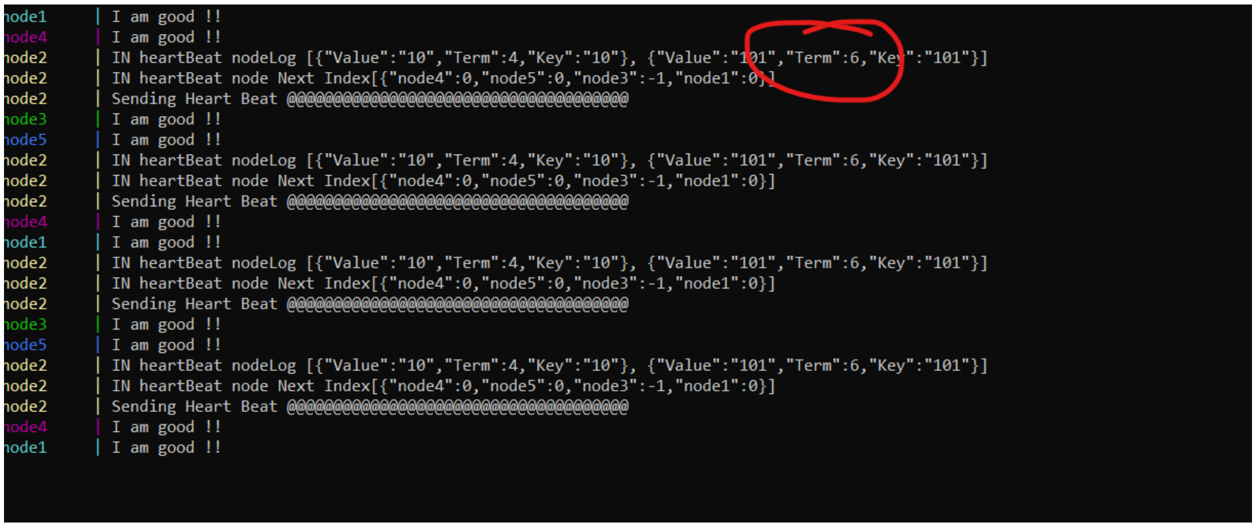
localhost:8080/controller/store?key=101&value=101&node=node4

Stored-> 101<->101



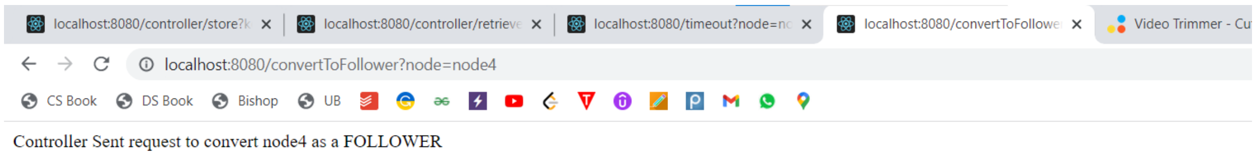
```
node1       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node4       | IN heartBeat node Next Index[{"node5":3,"node2":3,"node3":3,"node1":3}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3       | I am good !!
node5       | I am good !!
node2       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node4       | IN heartBeat node Next Index[{"node5":3,"node2":3,"node3":3,"node1":3}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node4       | IN heartBeat node Next Index[{"node5":3,"node2":3,"node3":3,"node1":3}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3       | I am good !!
node5       | I am good !!
node2       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node4       | IN heartBeat node Next Index[{"node5":3,"node2":3,"node3":3,"node1":3}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node4       | IN heartBeat node Next Index[{"node5":3,"node2":3,"node3":3,"node1":3}]
node4       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3       | I am good !!
node5       | I am good !!
node2       | I am good !!
node4       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node4       | IN heartBeat node Next Index[{"node5":3,"node2":3,"node3":3,"node1":3}]
```

Current term:



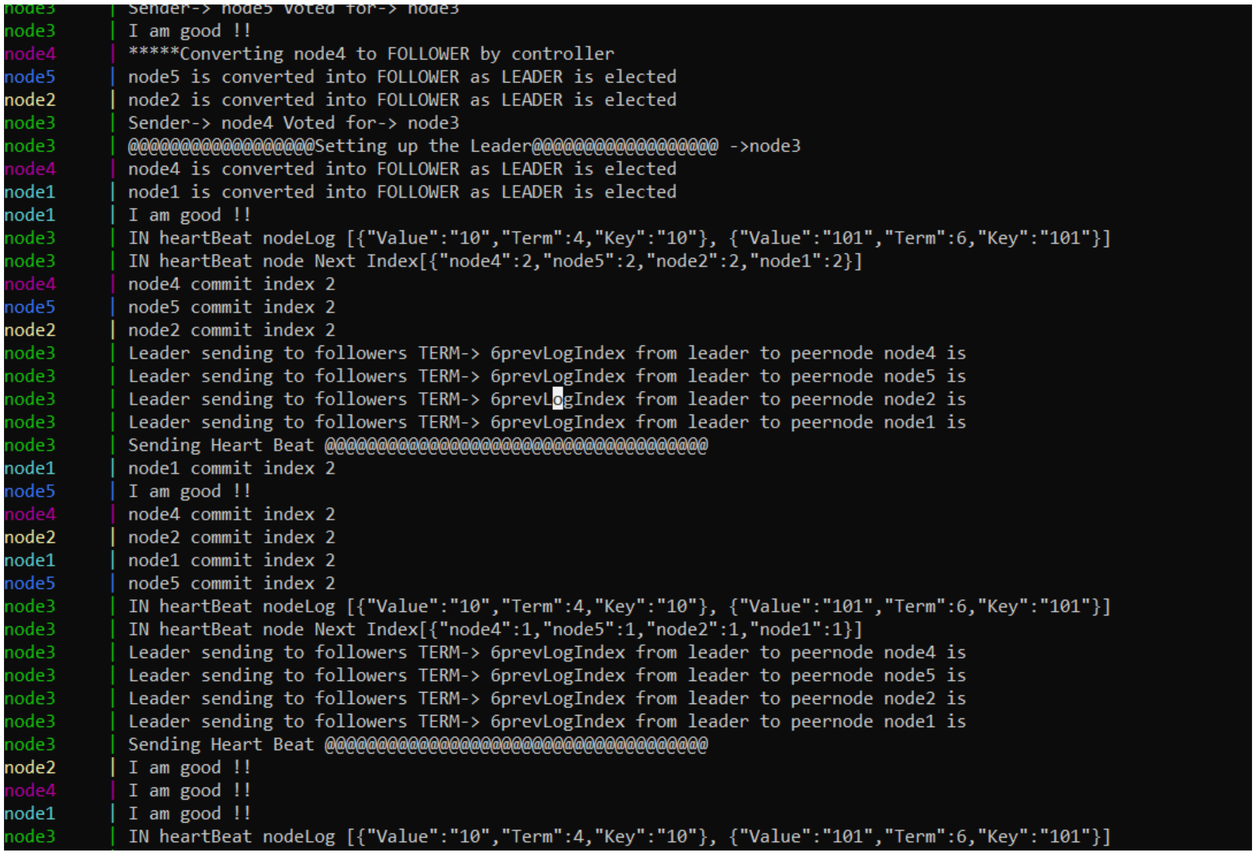
```
node1       | I am good !!
node4       | I am good !!
node2       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node2       | IN heartBeat node Next Index[{"node4":0,"node5":0,"node3":-1,"node1":0}]
node2       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3       | I am good !!
node5       | I am good !!
node2       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node2       | IN heartBeat node Next Index[{"node4":0,"node5":0,"node3":-1,"node1":0}]
node2       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4       | I am good !!
node1       | I am good !!
node2       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node2       | IN heartBeat node Next Index[{"node4":0,"node5":0,"node3":-1,"node1":0}]
node2       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3       | I am good !!
node5       | I am good !!
node2       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node2       | IN heartBeat node Next Index[{"node4":0,"node5":0,"node3":-1,"node1":0}]
node2       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node4       | I am good !!
node1       | I am good !!
```

Converting node4 to follower:

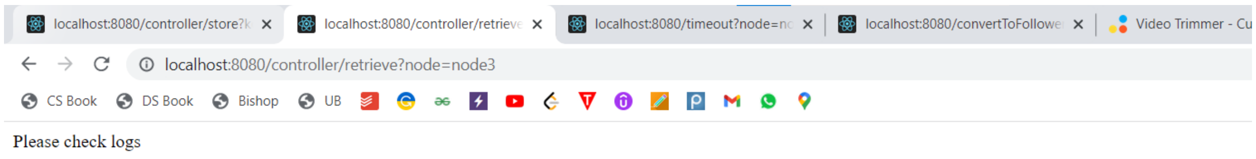

localhost:8080/convertToFollower?node=node4

Controller Sent request to convert node4 as a FOLLOWER

Node3 is elected as leader:

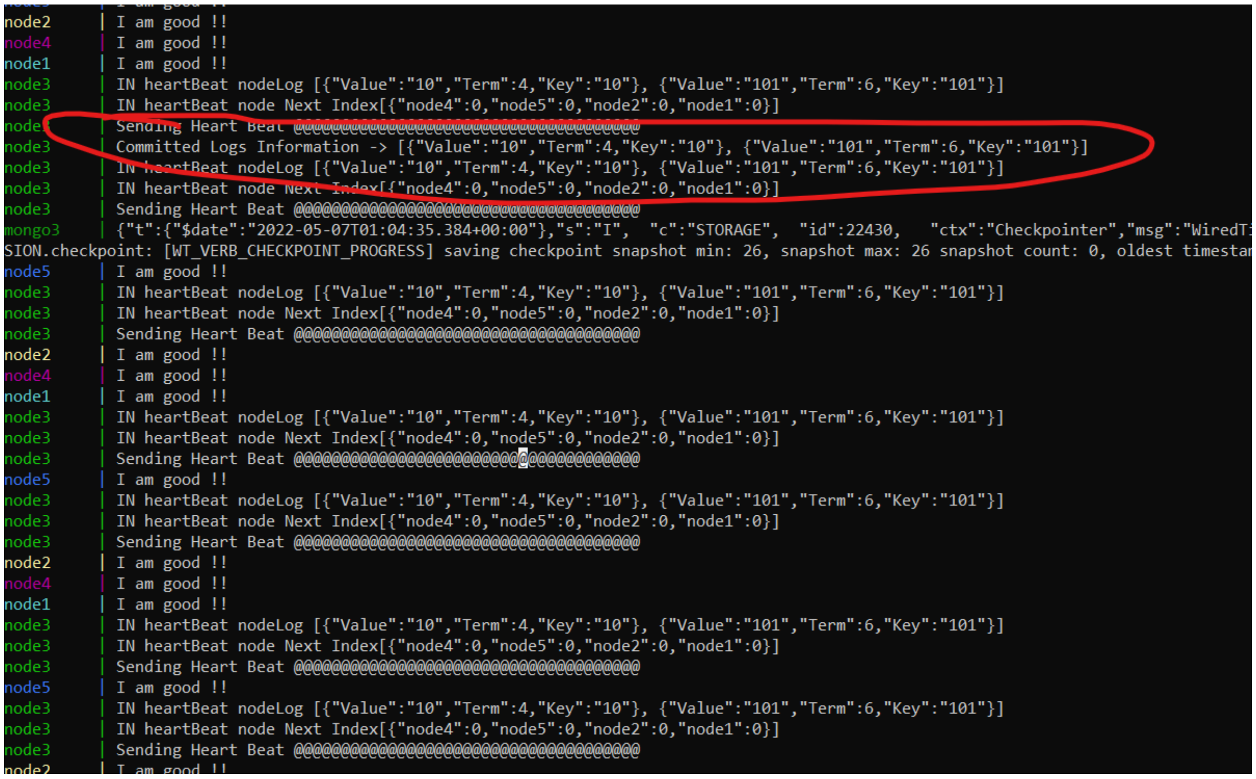

```
node3       | Sender-> node5 Voted for-> node3
node3       | I am good !!
node4       | *****Converting node4 to FOLLOWER by controller
node5       | node5 is converted into FOLLOWER as LEADER is elected
node2       | node2 is converted into FOLLOWER as LEADER is elected
node3       | Sender-> node4 Voted for-> node3
node3       | @@@@@@@@@@@@@@@@@@Setting up the Leader@@@@@@@@@@@@@@@@@@@ ->node3
node4       | node4 is converted into FOLLOWER as LEADER is elected
node1       | node1 is converted into FOLLOWER as LEADER is elected
node1       | I am good !!
node3       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3       | IN heartBeat node Next Index[{"node4":2,"node5":2,"node2":2,"node1":2}]
node4       | node4 commit index 2
node5       | node5 commit index 2
node2       | node2 commit index 2
node3       | Leader sending to followers TERM-> 6prevLogIndex from leader to peernode node4 is
node3       | Leader sending to followers TERM-> 6prevLogIndex from leader to peernode node5 is
node3       | Leader sending to followers TERM-> 6prevLogIndex from leader to peernode node2 is
node3       | Leader sending to followers TERM-> 6prevLogIndex from leader to peernode node1 is
node3       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node1       | node1 commit index 2
node5       | I am good !!
node4       | node4 commit index 2
node2       | node2 commit index 2
node1       | node1 commit index 2
node5       | node5 commit index 2
node3       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3       | IN heartBeat node Next Index[{"node4":1,"node5":1,"node2":1,"node1":1}]
node3       | Leader sending to followers TERM-> 6prevLogIndex from leader to peernode node4 is
node3       | Leader sending to followers TERM-> 6prevLogIndex from leader to peernode node5 is
node3       | Leader sending to followers TERM-> 6prevLogIndex from leader to peernode node2 is
node3       | Leader sending to followers TERM-> 6prevLogIndex from leader to peernode node1 is
node3       | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2       | I am good !!
node4       | I am good !!
node1       | I am good !!
node3       | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
```

Retrieve request via node3:


localhost:8080/controller/store?k
localhost:8080/controller/retrieve
localhost:8080/timeout?node=nc
localhost:8080/convertToFollowe
Video Trimmer - Cu
CS Book
DS Book
Bishop
UB

localhost:8080/controller/retrieve?node=node3

Please check logs



```
node2     | I am good !!
node4     | I am good !!
node1     | I am good !!
node3     | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3     | IN heartBeat node Next Index[{"node4":0,"node5":0,"node2":0,"node1":0}]
node3     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node3     | Committed Logs Information -> [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3     | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3     | IN heartBeat node Next Index[{"node4":0,"node5":0,"node2":0,"node1":0}]
node3     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
mongo3    | {"t":{"$date":"2022-05-07T01:04:35.384+00:00"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"Checkpointer","msg":"WiredTi
SION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 26, snapshot max: 26 snapshot count: 0, oldest timestan
node5     | I am good !!
node3     | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3     | IN heartBeat node Next Index[{"node4":0,"node5":0,"node2":0,"node1":0}]
node3     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node4     | I am good !!
node1     | I am good !!
node3     | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3     | IN heartBeat node Next Index[{"node4":0,"node5":0,"node2":0,"node1":0}]
node3     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5     | I am good !!
node3     | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3     | IN heartBeat node Next Index[{"node4":0,"node5":0,"node2":0,"node1":0}]
node3     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
node4     | I am good !!
node1     | I am good !!
node3     | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3     | IN heartBeat node Next Index[{"node4":0,"node5":0,"node2":0,"node1":0}]
node3     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node5     | I am good !!
node3     | IN heartBeat nodeLog [{"Value":"10","Term":4,"Key":"10"}, {"Value":"101","Term":6,"Key":"101"}]
node3     | IN heartBeat node Next Index[{"node4":0,"node5":0,"node2":0,"node1":0}]
node3     | Sending Heart Beat @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
node2     | I am good !!
```

References:

1. https://docs.docker.com/
2. https://raft.github.io/
3. https://raft.github.io/raft.pdf