

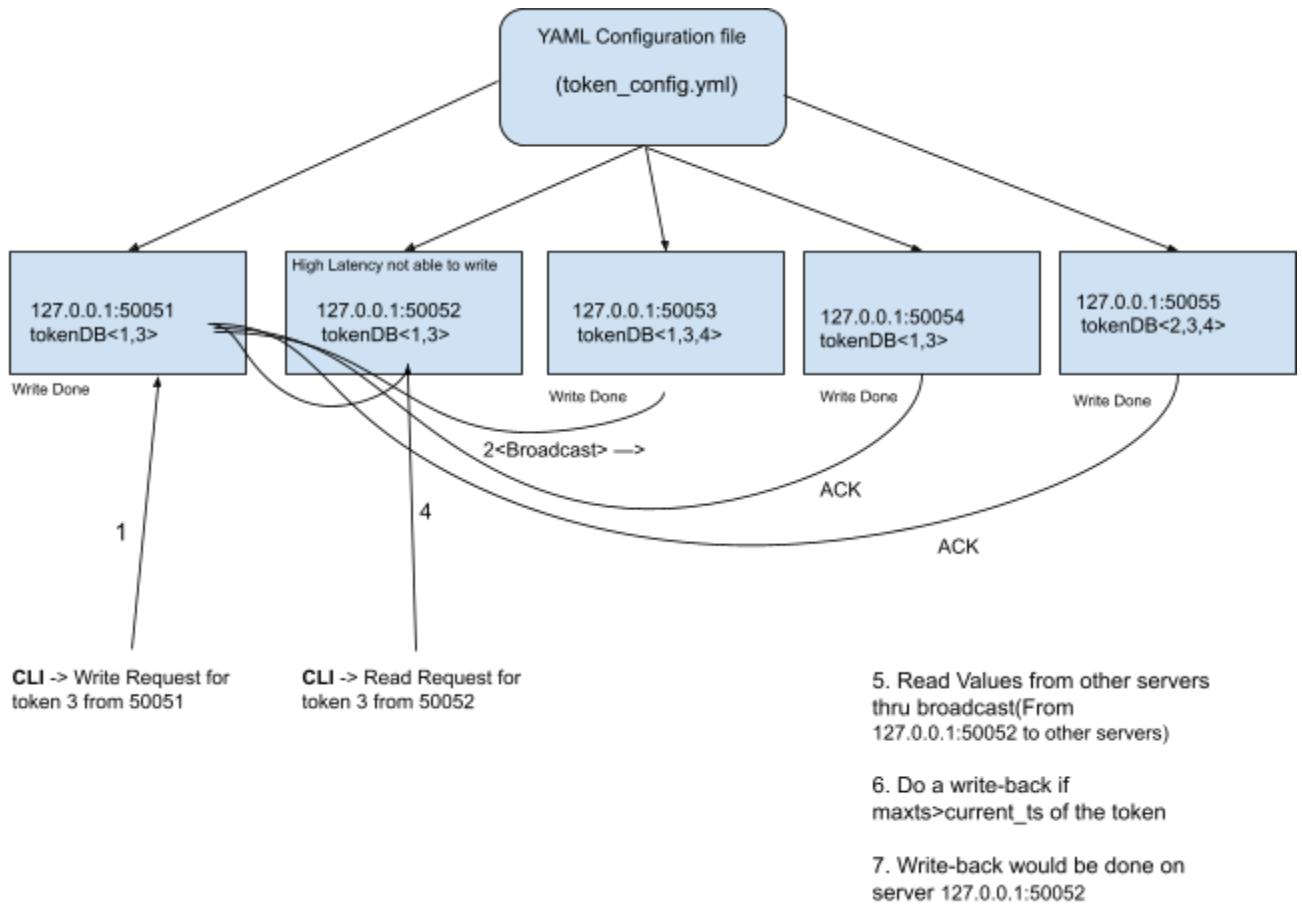
# **Fail-Silent Replicated Token Manager with Atomic Semantics**

## **1. Introduction**

This project is basically the extension of the previous project. Here, we have added functionalities to support replication and atomic semantics. We have simulated a fail-silent model on the tokens which are maintained by the individual servers according to the access given to them. The access points and the token ids which should be maintained are being read from the YAML file. This configuration is then applied to the servers when they are spawned. Additionally, the replication scheme is maintained using the YAML file. Once the configuration has been read and servers are spawned the client would now start requesting write and read requests. The authorization would be checked before serving the requests. For example, if a client gives a write request to the server which doesn't have the token mentioned or is not the writer node then that request would fail to run. In this project, I've used partial\_value, name, low, mid, high which can be written and read from the authorized servers.

To maintain the atomicity and replication scheme I've used the read-impose-write-all(Quorum) protocol. In this protocol, the write request would first broadcast the values written in the request to all the reader nodes. Once it gets the acknowledgement from the majority of the servers (like from 2 servers out of 3) it would return the write value. Now, if the request is made to the reader server it would ask for the values stored in other servers. After getting the values it will check for the value with maximum timestamp. Now, we would have the latest value which was written in the servers. Once we get the latest value we'll do the write-back across other nodes. This implementation has been shown in the demo.

## 2. Architecture



### 3. Implementation/Documentation

Following functionalities has been added to this project.

```
rpc WriteBroadcast (WriteBroadcastRequest) returns (WriteBroadcastResponse) {}  
rpc ReadBroadcast (ReadBroadcastRequest) returns (ReadBroadcastResponse) {}
```

```
message WriteBroadcastRequest {  
    uint64 hash_val=1;  
    bool reading_flag=2;  
    int32 ack=3;  
    string wts=4;  
    string name=5;  
    uint64 low=6;  
    uint64 mid=7;  
    uint64 high=8;  
    int32 token_id=9;  
    string server=10;  
}  
  
message WriteBroadcastResponse {  
    int32 ack=1;  
}
```

```
message ReadBroadcastRequest {  
    int32 token_id=1;  
    // string server=6;  
}  
  
message ReadBroadcastResponse {  
    string wts=1;  
    uint64 finalVal=2;  
    string name=3;  
    uint64 low=4;  
    uint64 mid=5;  
    uint64 high=6;  
}
```

```
func (s *TokenManagerServer) WriteToken(ctx context.Context, in *pb.WriteTokenMsg) (*pb.WriteResponse, error) {
```

This function WriteToken first writes the values for the token that are passed by the client. After writing these values, we need to broadcast these updated values along with the latest timestamp to the other nodes in the system. This function does exactly that.

```
func startBroadcast(partial_val uint64, read_name string, read_high uint64, read_mid uint64, read_low uint64, len_readlist int, wts string, server string, reading_flag bool, cnt *int, cnt_ch chan int, token int32)
```

This startBroadcast function is basically a goroutine used for broadcasting asynchronous rpc calls. These rpc calls are basically write calls to other nodes in the system. The channel returns the number of acks received back to the WriteToken function which keeps a count of the number of acks received.

```
func (s *TokenManagerServer) WriteBroadcast(ctx context.Context, in *pb.WriteBroadcastRequest) (*pb.WriteBroadcastResponse, error)
```

The WriteBroadcast function is the function that is called by the asynchronous write broadcast call. This function writes the latest values of the token and returns an ack.

```
func (s *TokenManagerServer) ReadToken(ctx context.Context, in *pb.Token) (*pb.WriteResponse, error)
```

The ReadToken function issues a broadcast for all the reader nodes based on the read-impose-write-all protocol. This broadcast receives the values of the tokens along with the timestamp as an ack.

```
func startReadBroadcast(token_id int32, wts string, server string, readlist_ch chan readlist) {
```

This is the actual goroutine that is called while broadcasting a read request. The acks received are passed through a channel to the ReadToken function for further evaluation.

```
func (s *TokenManagerServer) ReadBroadcast(ctx context.Context, in *pb.ReadBroadcastRequest) (*pb.ReadBroadcastResponse, error)
```

The ReadBroadcast is the delivery function for the startReadBroadcast function. It returns the value of the appropriate token as an acknowledgement.

#### 4. Demo

- Running the code

Please run the following command if needed -

*go mod init*

To compile the proto buffer file. Run the following command -

```
protoc --go_out=. --go_opt=paths=source_relative --go-grpc_out=.  
--go-grpc_opt=paths=source_relative token_management/token_pb.proto
```

***token\_config.yml*** → This file contains the configuration which would be read by servers.

To run servers individually. Here by default the IP address is 127.0.0.1(i.e localhost). Please change port number as needed.

*For Example -*

```
go build server/tokenservert.go → Terminal 1  
.tokenservert -port 50051 → Terminal 1  
.tokenservert -port 50052 → Terminal 2  
.tokenservert -port 50053 → Terminal 3
```

*Terminal 4 →*

```
go build client/tokenclient.go  
.tokenclient -write -id 1 -name abc -low 5 -mid 25 -high 100 -host 127.0.0.1 -port 50051  
.tokenclient -read -id 3 -host 127.0.0.1 -port 50052
```

***.service-run.sh*** → This bash script will run the above configuration and will kill the servers once the request is served.

- Demonstrating scenario where Read-Write authorizations are checked

```

EXPLORER TOKEN-MANAGER-DIST...
client tokenclient.go
server tokenserv...
token_manage...
token_pb_grpc...
token_pb_pb.go
token_pb.proto
go.mod
go.sum
README.md
readme.pdf
reference-report.pdf
service-run.sh
token_config.yml
token_logs.log
token_manager_cli...
token_manager.sh
tokenclient
tokenserver

PROBLEMS (27) OUTPUT TERMINAL DEBUG CONSOLE

(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -write -id 1 -name abc -low 5 -mid 25 -high 100 -host 127.0.0.1 -port 50054
127.0.0.1:50054
Reading YAML Configuration
[{1: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]}, {2: 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50059 127.0.0.1:50060]}, {3: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}, {4: 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
127.0.0.1:50054
2022/05/17 20:40:39 Unauthorized to do write request
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -read -id 2 -host 127.0.0.1 -port 5 0052
127.0.0.1:50052
Reading YAML Configuration
[{1: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]}, {2: 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50060]}, {3: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}, {4: 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
2022/05/17 20:41:27 Unauthorized to do read request
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed %

```

Below all the necessary servers have already been listening.

```

EXPLORER TOKEN-MANAGER-DIST...
client tokenclient.go
server tokenserv...
token_manage...
token_pb_grpc...
token_pb_pb.go
token_pb.proto
go.mod
go.sum
README.md
readme.pdf
reference-report.pdf
service-run.sh
token_config.yml
token_logs.log
token_manager_cli...
token_manager.sh
tokenclient
tokenserver

PROBLEMS (27) OUTPUT TERMINAL DEBUG CONSOLE

(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -write -id 1 -name abc -low 5 -mid 25 -high 100 -host 127.0.0.1 -port 50051
127.0.0.1:50051
Reading YAML Configuration
[{1: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]}, {2: 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50060]}, {3: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}, {4: 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
127.0.0.1:50051
2022/05/17 20:45:06 Response from the Server: 902613196918738813
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -read -id 2 -host 127.0.0.1 -port 5 0052
127.0.0.1:50052
Reading YAML Configuration
[{1: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]}, {2: 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50060]}, {3: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}, {4: 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
2022/05/17 20:45:14 Unauthorized to do read request
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -read -id 1 -host 127.0.0.1 -port 5 0052
127.0.0.1:50052
Reading YAML Configuration
[{1: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]}, {2: 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50060]}, {3: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}, {4: 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
2022/05/17 20:45:43 Response from the Server: 902613196918738813
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed %

```

- Demonstrating the fail-silent model.

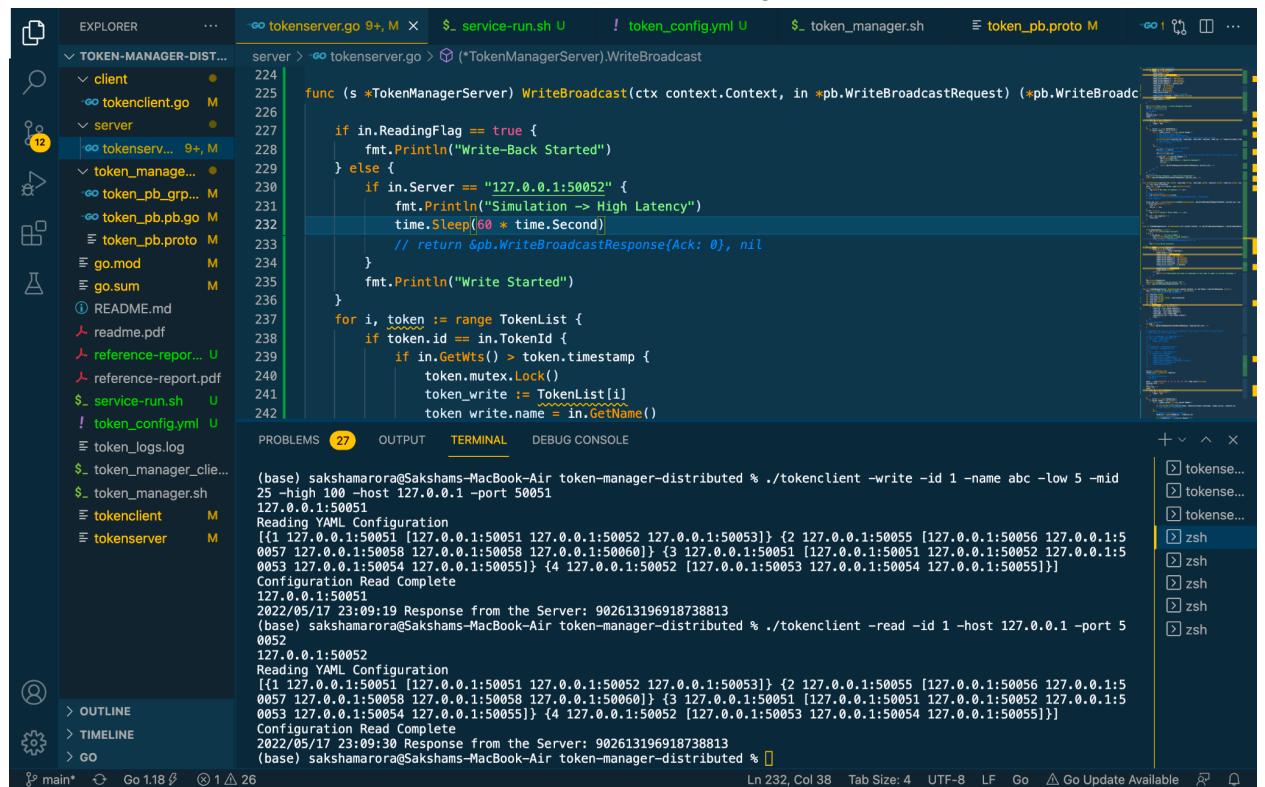
Here I'll be taking the scenario where out of 3 nodes I'll be manually hardcoding 1 server to sleep for a few seconds. Meanwhile, if a read comes on that node it will start the read broadcast functionality. Once the server will know the value of maximum timestamp it will start the write-back and will update its value. This will simulate the behavior when there is high latency and the server is taking time to respond. In the below results we can see that write-back has been done on the server 2 while server 1 and server 3 won't start the write-back since it would already have the updated value.

### Scenario - 1) With sleep

#### Client Request)

Here we can see we have started the read request on server 127.0.0.1:50052.

Additionally, we can see it's returning the correct partial value also other parameters are updated which we can see in server 2(127.0.0.1:50052) logs.



The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER:** Shows the project structure for "TOKEN-MANAGER-DIST..." with files like client, token\_manager..., and tokenserver.go.
- CODE EDITOR:** Displays the tokenserver.go file. The code contains logic for handling a WriteBroadcast request. It includes a check for reading flag, a condition for server 127.0.0.1:50052, and a sleep operation of 60 seconds. It also prints "Write Started" and iterates over tokens to update their timestamps and names.
- TERMINAL:** Shows the command-line output of the application. It starts with configuration reads and then receives a response from the server at 127.0.0.1:50052 containing tokens with IDs 50051, 50052, 50053, 50054, and 50055. The log ends with a final status message.

```

func (s *TokenManagerServer) WriteBroadcast(ctx context.Context, in *pb.WriteBroadcastRequest) (*pb.WriteBroadcastResponse, error) {
    if in.ReadingFlag == true {
        fmt.Println("Write-Back Started")
    } else {
        if in.Server == "127.0.0.1:50052" {
            fmt.Println("Simulation -> High Latency")
            time.Sleep(60 * time.Second)
        }
        fmt.Println("Write Started")
    }
    for i, token := range TokenList {
        if token.id == in.TokenId {
            if in.GetWts() > token.timestamp {
                token.mutex.Lock()
                token_write := TokenList[i]
                token_write.name = in.GetName()
                token_write.timestamp = in.GetWts()
                token_list[i] = token_write
            }
        }
    }
}

```

```

(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -write -id 1 -name abc -low 5 -mid 25 -high 100 -host 127.0.0.1 -port 50051
127.0.0.1:50051
Reading YAML Configuration
[{1 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]} {2 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50059 127.0.0.1:50060]} {3 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]} {4 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
127.0.0.1:50051
2022/05/17 23:09:19 Response from the Server: 902613196918738813
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -read -id 1 -host 127.0.0.1 -port 50052
127.0.0.1:50052
Reading YAML Configuration
[{1 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]} {2 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50059 127.0.0.1:50060]} {3 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]} {4 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
2022/05/17 23:09:30 Response from the Server: 902613196918738813
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed %

```

## Server 127.0.0.1:50051 logs -

The screenshot shows a terminal window with several tabs open. The current tab displays Go code for a `tokenserver.go` file. The code includes logic for handling a `WriteBroadcast` request from a `TokenManagerServer`. It checks if `in.ReadingFlag` is true, prints "Write-Back Started", and then checks if the server's IP is "127.0.0.1:50052". If so, it prints "Simulation -> High Latency", sleeps for 60 seconds, and then returns a response. Otherwise, it prints "Write Started". Below the code, the terminal shows the command being run: `(base) sakshamara@Sakshams-MacBook-Air token-manager-distributed % go build server/tokenserver.go`. The log output shows the server reading configuration files, starting up, and performing a write operation. It also shows a client connecting and receiving an ACK.

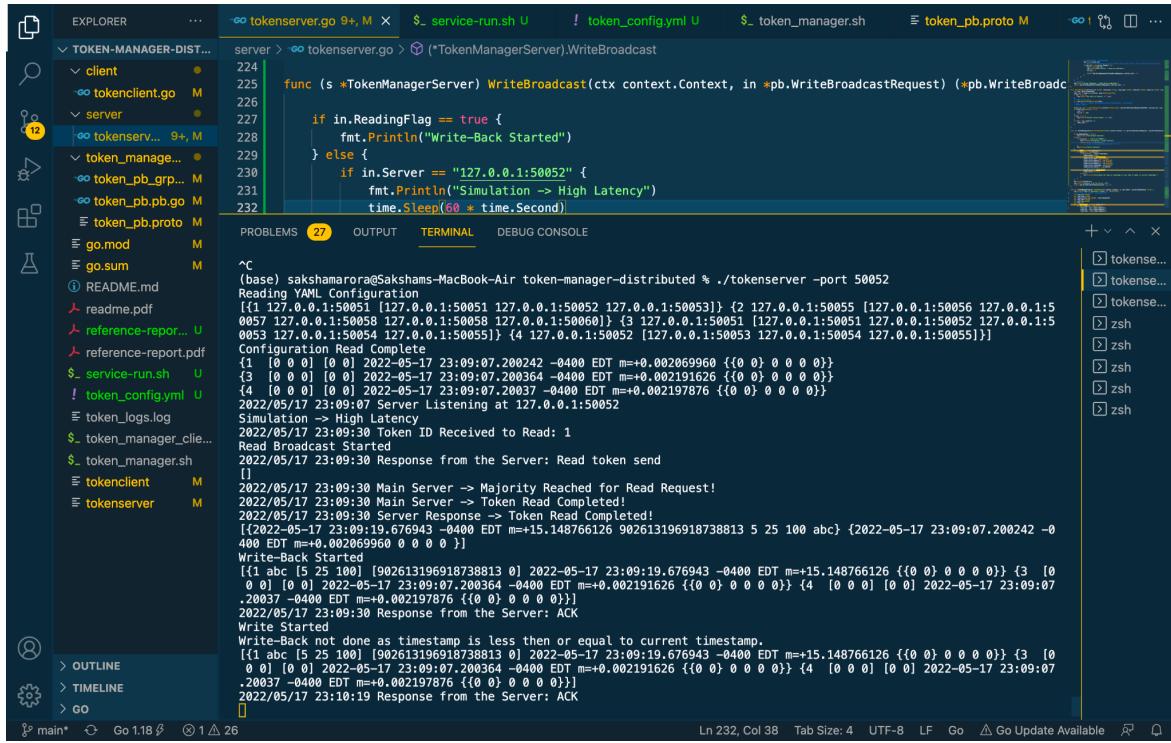
```
func (s *TokenManagerServer) WriteBroadcast(ctx context.Context, in *pb.WriteBroadcastRequest) (*pb.WriteBroadcastResponse, error) {
    if in.ReadingFlag == true {
        fmt.Println("Write-Back Started")
    } else {
        if in.Server == "127.0.0.1:50052" {
            fmt.Println("Simulation -> High Latency")
            time.Sleep(60 * time.Second)
        }
        // return &pb.WriteBroadcastResponse{Ack: 0}, nil
    }
    fmt.Println("Write Started")
}

PROBLEMS 27 OUTPUT TERMINAL DEBUG CONSOLE

(base) sakshamara@Sakshams-MacBook-Air token-manager-distributed % go build server/tokenserver.go
(base) sakshamara@Sakshams-MacBook-Air token-manager-distributed % ./tokenserver -port 50051
Reading YAML Configuration
[{"1": 127.0.0.1:50051, "2": 127.0.0.1:50052, "3": 127.0.0.1:50055}, {"2": 127.0.0.1:50056, "1": 127.0.0.1:50051, "3": 127.0.0.1:50052}, {"3": 127.0.0.1:50051, "1": 127.0.0.1:50055, "2": 127.0.0.1:50054}, {"4": 127.0.0.1:50052, "1": 127.0.0.1:50053, "2": 127.0.0.1:50055}, {"5": 127.0.0.1:50053, "1": 127.0.0.1:50054, "2": 127.0.0.1:50055}], [{"1": 127.0.0.1:50051, "2": 127.0.0.1:50052, "3": 127.0.0.1:50055}], Configuration Read Complete
[1] [0 0 0] [0 0] 2022-05-17 23:09:04.534106 -0400 EDT m=<0.005452459 {{0 0} 0 0 0 0}}
[3] [0 0 0] [0 0] 2022-05-17 23:09:04.534259 -0400 EDT m=<0.005606001 {{0 0} 0 0 0 0}}
2022/05/17 23:09:04 Server Listening at 127.0.0.1:50051
2022/05/17 23:09:19 Token ID Received to Write: 1
Main Server -> Write Broadcast Started
Write Started
Write-Back not done as timestamp is less than or equal to current timestamp.
[1 abc [5 25 100] [902613196918738813 0] 2022-05-17 23:09:19.676943 -0400 EDT m=<+15.148766126 {{0 0} 0 0 0 0}} {{3 [0 0 0] [0 0] 2022-05-17 23:09:04.534259 -0400 EDT m=<0.005606001 {{0 0} 0 0 0 0}}}}
2022/05/17 23:09:19 Response from the Server: ACK
1
2
3
2022/05/17 23:09:19 Main Server -> Majority Reached!
Read Broadcast Started
2022/05/17 23:09:30 Response from the Server: Read token send
Write-Back Started
Write-Back not done as timestamp is less than or equal to current timestamp.
[1 abc [5 25 100] [902613196918738813 0] 2022-05-17 23:09:19.676943 -0400 EDT m=<+15.148766126 {{0 0} 0 0 0 0}} {{3 [0 0 0] [0 0] 2022-05-17 23:09:04.534259 -0400 EDT m=<0.005606001 {{0 0} 0 0 0 0}}}}
2022/05/17 23:09:30 Response from the Server: ACK

> OUTLINE
> TIMELINE
> GO
```

## Server 127.0.0.1:50052 logs -



```
server > ./tokenserver.go <+ M X $ _ service-run.sh U ! token_config.yml U $ _ token_manager.sh E token_pb.proto M -> ...  
server > ./tokenserver.go > (*TokenManagerServer).WriteBroadcast  
224  
225 func (s *TokenManagerServer) WriteBroadcast(ctx context.Context, in *pb.WriteBroadcastRequest) (*pb.WriteBroadcastResponse, error) {  
226     if in.ReadingFlag == true {  
227         fmt.Println("Write-Back Started")  
228     } else {  
229         if in.Server == "127.0.0.1:50052" {  
230             fmt.Println("Simulation -> High Latency")  
231             time.Sleep(60 * time.Second)  
232         }  
233     }  
234     return nil, nil  
235 }  
236  
237 (base) sakhamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenserver -port 50052  
Reading YAML Configuration  
[{"1": 127.0.0.1:50051 [127.0.0.1:50052 127.0.0.1:50053] {2 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50059 127.0.0.1:50060} {3 127.0.0.1:50051 [127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]} {4 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]} }  
Configuration Read Complete!  
[1 [0 0 0] [0 0 0] 2022-05-17 23:09:07.200242 -0400 EDT m+=0.002069960 {{0 0} 0 0 0 0}]  
(3 [0 0 0] [0 0 0] 2022-05-17 23:09:07.200364 -0400 EDT m+=0.002191626 {{0 0} 0 0 0 0})  
(4 [0 0 0] [0 0 0] 2022-05-17 23:09:07.20037 -0400 EDT m+=0.002197876 {{0 0} 0 0 0 0})  
2022/05/17 23:09:07 Server Listening at 127.0.0.1:50052  
Simulation -> High Latency  
2022/05/17 23:09:30 Token ID Received to Read: 1  
Read Broadcast Started  
2022/05/17 23:09:30 Response from the Server: Read token send  
[]  
2022/05/17 23:09:30 Main Server -> Majority Reached for Read Request!  
2022/05/17 23:09:30 Main Server -> Token Read Completed!  
2022/05/17 23:09:30 Server Response -> Token Read Completed!  
[{"2022-05-17 23:09:19.676943 -0400 EDT m+=15.148766126 902613196918738813 5 25 100 abc} {2022-05-17 23:09:07.200242 -0400 EDT m+=0.002069960 0 0 0 0}]  
Write-Back Started  
[{"1 abc [5 25 100] [902613196918738813 0] 2022-05-17 23:09:19.676943 -0400 EDT m+=15.148766126 {{0 0} 0 0 0 0} } {3 [0 0 0] [0 0 0] 2022-05-17 23:09:07.200364 -0400 EDT m+=0.002191626 {{0 0} 0 0 0 0} } {4 [0 0 0] [0 0 0] 2022-05-17 23:09:07.20037 -0400 EDT m+=0.002197876 {{0 0} 0 0 0 0} } ]  
2022/05/17 23:09:30 Response from the Server: ACK  
Write Started  
Write-Back not done as timestamp is less than or equal to current timestamp.  
[{"1 abc [5 25 100] [902613196918738813 0] 2022-05-17 23:09:19.676943 -0400 EDT m+=15.148766126 {{0 0} 0 0 0 0} } {3 [0 0 0] [0 0 0] 2022-05-17 23:09:07.200364 -0400 EDT m+=0.002191626 {{0 0} 0 0 0 0} } {4 [0 0 0] [0 0 0] 2022-05-17 23:09:07.20037 -0400 EDT m+=0.002197876 {{0 0} 0 0 0 0} } ]  
2022/05/17 23:10:19 Response from the Server: ACK
```

In the above image we can see that there was latency in that server so it was not able to write. In the meantime the read request was fired on that server and then we can see that the write-back has started. After some time, we see an older write request was trying to execute but couldn't go further as the timestamp was higher now.

## Server 127.0.0.1:50053 logs -

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER:** Shows the project structure under "TOKEN-MANAGER-DIST...".
- CODE EDITOR:** Displays the `tokenserver.go` file. The code implements a `WriteBroadcast` method for a `TokenManagerServer`. It checks if `in.ReadingFlag` is true and prints "Write-Back Started". If false, it checks if `in.Server` is "127.0.0.1:50052" and if so, prints "Simulation -> High Latency", sleeps for 60 seconds, and then returns a response. Otherwise, it prints "Write Started" and iterates through a `TokenList`, checking if each token's ID matches the current `tokenId`. If it does, it compares the token's timestamp with the current time. If the token's timestamp is greater than the current time, it returns an error.
- TERMINAL:** Shows the command-line output of running the server and sending a broadcast message. The log includes reading configuration from `token_config.yml`, starting the server at port 50053, and receiving responses from the server indicating ACK and simulation mode due to high latency.
- OUTPUT:** Shows the standard output of the application.
- PROBLEMS:** Shows 27 problems in the code.
- SIDEBAR:** Includes sections for OUTLINE, TIMELINE, and GO.
- STATUS BAR:** Shows the current line (Ln 232), column (Col 38), tab size (Tab Size: 4), and encoding (UTF-8). It also indicates LF line endings.

Scenario 2) No sleep. Server responds with negative acknowledgment(assume as crash)

Client Requests ->

Here we can see the consistency has been achieved as server 127.0.0.1:50052 was crashed.

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER**: Shows the project structure for "TOKEN-MANAGER-DIST...". The "tokenserv..." file is currently selected.
- CODE EDITOR**: Displays the "tokenserver.go" file. The code handles token writes and reads, including logic to handle a crashed server (127.0.0.1:50052). A yellow circle highlights line 248.
- TERMINAL**: Shows the command-line output of running the application and performing writes and reads.
- OUTPUT**: Shows log messages from the application.
- PROBLEMS**: Shows 26 problems.
- SIDE BAR**: Shows recent files and a list of terminals.

```
server > tokenserver.go 9+, M X $ service-run.sh U ! token_config.yml U $ token_manager.sh E token_pb.proto M
server > tokenserver.go > (*TokenManagerServer).WriteBroadcast
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
    if in.Server == "127.0.0.1:50052" {
        fmt.Println("Simulation -> High Latency")
        return &pb.WriteBroadcastResponse{Ack: 0}, nil
    }
    fmt.Println("Write Started")

    for i, token := range TokenList {
        if token.id == in.TokenId {
            if in.GetWts() > token.timestamp {
                token.mutex.Lock()
                token_write := TokenList[i]
                token_write.name = in.GetName()
                token_write.domain[0] = in.GetLow()
                token_write.domain[1] = in.GetMid()
                token_write.domain[2] = in.GetHigh()
                token_write.timestamp = in.GetWts()
                token_write.state[0] = in.HashVal
            }
        }
    }
}

(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -write -id 1 -name abc -low 5 -mid
25 -high 100 -host 127.0.0.1 -port 50051
127.0.0.1:50051
Reading YAML Configuration
[{1 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]} {2 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50059 127.0.0.1:50060]} {3 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]} {4 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
127.0.0.1:50051
2022/05/17 21:15:33 Response from the Server: 902613196918738813
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokenclient -read -id 1 -host 127.0.0.1 -port 5
0052
127.0.0.1:50052
Reading YAML Configuration
[{1 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]} {2 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50059 127.0.0.1:50060]} {3 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]} {4 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
2022/05/17 21:15:50 Response from the Server: 902613196918738813
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed %
```

## Server 127.0.0.1:50051 -

```

EXPLORER ... tokensever.go 9+, M service-run.sh token_config.yml token_manager.sh token_pb.proto ...
TOKEN-MANAGER-DIST...
client
tokenclient.go M
server
tokenserv... 9+, M
token_manage...
token_pb_grpc...
token_pb_pb.go M
token_pb.proto M
go.mod M
go.sum M
README.md
readme.pdf
reference-report.pdf
service-run.sh U
token_config.yml U
token_logs.log
token_manager_clie...
token_manager.sh
tokenclient M
tokensever M
PROBLEMS 26 OUTPUT TERMINAL DEBUG CONSOLE
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokensever -port 50051
Reading Configuration
[{1 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]}, {2 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50060]}, {3 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50054 127.0.0.1:50056]}, {4 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
{1 [0 0 0] [0 0] 2022-05-17 21:15:17.210103 -0400 EDT m=+0.005584376 {{0 0} 0 0 0 0}}
{3 [0 0 0] [0 0] 2022-05-17 21:15:17.210128 -0400 EDT m=+0.005682751 {{0 0} 0 0 0 0}}
2022/05/17 21:15:33 Token ID Received to Write: 1
Main Server -> Write Broadcast Started
127.0.0.1:50051
127.0.0.1:50052
127.0.0.1:50053
Write Started
Write-Back not done as timestamp is less then or equal to current timestamp.
[{1 abc [5 25 100] [902613196918738813 0] 2022-05-17 21:15:33.710336 -0400 EDT m=+16.506488459 {{0 0} 0 0 0 0}} {3 [0 0 0] [0 0] 2022-05-17 21:15:17.210128 -0400 EDT m=+0.005682751 {{0 0} 0 0 0 0}}]
2022/05/17 21:15:33 Response from the Server: ACK
1
2
2
2022/05/17 21:15:33 Main Server -> Majority Reached!
Read Broadcast Started
2022/05/17 21:15:50 Response from the Server: Read token send
Write-Back Started
Write-Back not done as timestamp is less then or equal to current timestamp.
[{1 abc [5 25 100] [902613196918738813 0] 2022-05-17 21:15:33.710336 -0400 EDT m=+16.506488459 {{0 0} 0 0 0 0}} {3 [0 0 0] [0 0] 2022-05-17 21:15:17.210128 -0400 EDT m=+0.005682751 {{0 0} 0 0 0 0}}]
2022/05/17 21:15:50 Response from the Server: ACK
Ln 234, Col 1 Tab Size: 4 UTF-8 LF Go △ Go Update Available

```

## Server 127.0.0.1:50052 -

```

EXPLORER ... tokensever.go 9+, M service-run.sh token_config.yml token_manager.sh token_pb.proto ...
TOKEN-MANAGER-DIST...
client
tokenclient.go M
server
tokenserv... 9+, M
token_manage...
token_pb_grpc...
token_pb_pb.go M
token_pb.proto M
go.mod M
go.sum M
README.md
readme.pdf
reference-report.pdf
service-run.sh U
token_config.yml U
token_logs.log
token_manager_clie...
token_manager.sh
tokenclient M
tokensever M
PROBLEMS 26 OUTPUT TERMINAL DEBUG CONSOLE
(base) sakshamarora@Sakshams-MacBook-Air token-manager-distributed % ./tokensever -port 50052
Reading Configuration
[{1 127.0.0.1:50051 [127.0.0.1:50052 127.0.0.1:50053]}, {2 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50060]}, {3 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50054 127.0.0.1:50056]}, {4 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
{1 [0 0 0] [0 0] 2022-05-17 21:15:23.413057 -0400 EDT m=+0.1992668 {{0 0} 0 0 0 0}}
{3 [0 0 0] [0 0] 2022-05-17 21:15:23.413164 -0400 EDT m=+0.002099793 {{0 0} 0 0 0 0}}
{4 [0 0 0] [0 0] 2022-05-17 21:15:23.41317 -0400 EDT m=+0.002106043 {{0 0} 0 0 0 0}}
2022/05/17 21:15:23 Server Listening at 127.0.0.1:50052
Simulation -> High Latency
2022/05/17 21:15:50 Token ID Received to Read: 1
Read Broadcast Started
2022/05/17 21:15:50 Response from the Server: Read token send
[]
2022/05/17 21:15:50 Main Server -> Majority Reached for Read Request!
2022/05/17 21:15:50 Main Server -> Token Read Completed!
2022/05/17 21:15:50 Server Response -> Token Read Completed!
[{2022-05-17 21:15:33.710336 -0400 EDT m=+16.506488459 902613196918738813 5 25 100 abc} {2022-05-17 21:15:33.710336 -0400 EDT m=+16.506488459 902613196918738813 5 25 100 abc}]
Write-Back Started
[{1 abc [5 25 100] [902613196918738813 0] 2022-05-17 21:15:33.710336 -0400 EDT m=+16.506488459 {{0 0} 0 0 0 0}} {3 [0 0 0] [0 0] 2022-05-17 21:15:23.413164 -0400 EDT m=+0.002099793 {{0 0} 0 0 0 0}} {4 [0 0 0] [0 0] 2022-05-17 21:15:23.41317 -0400 EDT m=+0.002106043 {{0 0} 0 0 0 0}}]
2022/05/17 21:15:50 Response from the Server: ACK
Ln 234, Col 1 Tab Size: 4 UTF-8 LF Go △ Go Update Available

```

Above we can see the write-back started for server 2 as it was crashed. And after the write-back it maintains the value with the greater timestamp.

Server 127.0.0.1:50053 -

The screenshot shows a terminal window with several tabs open. The tabs include `service-run.sh`, `token_config.yml`, `token_manager.sh`, and `token_pb.proto`. The `tokenserver.go` file is the active tab, displaying Go code for a token manager server. The code handles reading from a token list and performing write-back operations. The terminal output shows logs from the server, including responses to client requests and internal state changes like "Write Started" and "Write-Back Started". The log output is timestamped and includes details about token values and timestamps.

```
server > ./tokenserver.go > (*TokenManagerServer).WriteBroadcast
225
226
227
228
229
230
231
232
233
234
235
236
237
238
2022/05/17 21:14:29 Response from the Server: ACK
^C
(base) sakshamarrora@Sakshams-MacBook-Air token-manager-distributed % ./tokenserver -port 50053
Reading YAML Configuration
[{1: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053]}, {2: 127.0.0.1:50055 [127.0.0.1:50056 127.0.0.1:50057 127.0.0.1:50058 127.0.0.1:50059 127.0.0.1:50060]}, {3: 127.0.0.1:50051 [127.0.0.1:50051 127.0.0.1:50052 127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}, {4: 127.0.0.1:50052 [127.0.0.1:50053 127.0.0.1:50054 127.0.0.1:50055]}]
Configuration Read Complete
[{1: [0 0 0] [0 0] 2022-05-17 21:15:27.645211 -0400 EDT m=>0.001598959 {[0 0] 0 0 0 0}}, {3: [0 0 0] [0 0] 2022-05-17 21:15:27.645293 -0400 EDT m=>0.001680876 {[0 0] 0 0 0 0}}, {4: [0 0 0] [0 0] 2022-05-17 21:15:27.645296 -0400 EDT m=>0.001684209 {[0 0] 0 0 0 0}}]
2022/05/17 21:15:27 Server Listening at 127.0.0.1:50053
Write Started
[{1: abc [5 25 100] [902613196918738813 0] 2022-05-17 21:15:33.710336 -0400 EDT m=>16.506488459 {[0 0] 0 0 0 0}}, {3: [0 0 0] [0 0] 2022-05-17 21:15:27.645293 -0400 EDT m=>0.001680876 {[0 0] 0 0 0 0}}, {4: [0 0 0] [0 0] 2022-05-17 21:15:27.645296 -0400 EDT m=>0.001684209 {[0 0] 0 0 0 0}}]
2022/05/17 21:15:33 Response from the Server: ACK
Read Broadcast Started
2022/05/17 21:15:50 Response from the Server: Read token send
Write-Back Started
Write-Back not done as timestamp is less then or equal to current timestamp.
[{1: abc [5 25 100] [902613196918738813 0] 2022-05-17 21:15:33.710336 -0400 EDT m=>16.506488459 {[0 0] 0 0 0 0}}, {3: [0 0 0] [0 0] 2022-05-17 21:15:27.645293 -0400 EDT m=>0.001680876 {[0 0] 0 0 0 0}}, {4: [0 0 0] [0 0] 2022-05-17 21:15:27.645296 -0400 EDT m=>0.001684209 {[0 0] 0 0 0 0}}]
2022/05/17 21:15:50 Response from the Server: ACK
```