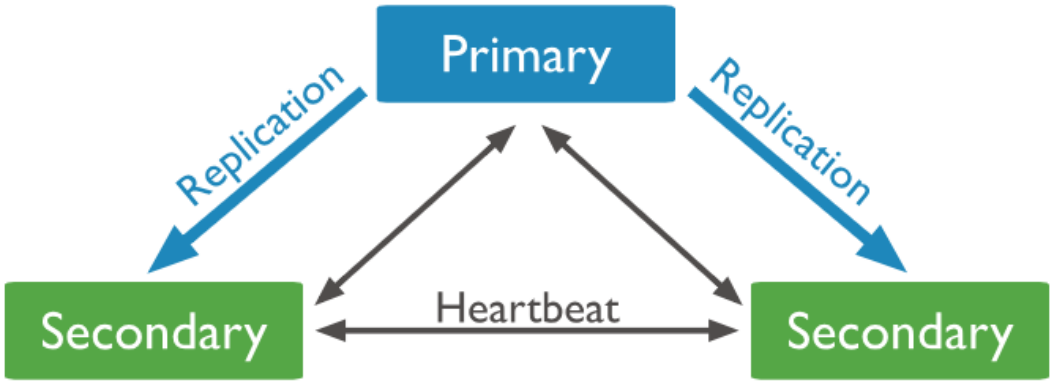
ITboons challenge

To run this repository you will need

ReactJS , MongoDB and nodeJS, npm package installer, git and a browser

The first task is to create a 3 node mongodb cluster

the architecture of of the mongodb cluster would be something like this.



There is one primary machine and two secondary machines

All the three machines communicate with each other through the heartbeat signal.

If the primary machine fails then the other two machines have a poll and the winner is then made a primary machine

And then once the failed primary machine gets booted up it then again joins the replicaset and it becomes the secondary machine

At a time there has to be a minimum of 2 machines if not then the remaining machine will not be a primary machine.

There are various ways to perform reads and write operations on the 3 node cluster but in this case as per the question I have gone with the approach where

The reads are done through the secondary machines and the writes are done to the primary machine

To do that you will need to perform the following steps.

1) Install mongodb in your machine

Now to make sure that the installation is working fine
Type the command - mongod in your terminal
Open a new terminal and type mongo -> you should be able to see a mongo shell.
Once that is done. The second step is to create 3 separate mongodb nodes.

2) Type the command

mongod --replSet itboons --logpath 1.log --dbpath rs1 --port 27018 &

mongod --replSet itboons --logpath 2.log --dbpath rs2 --port 27019 &

mongod --replSet itboons --logpath 3.log --dbpath rs3 --port 27020 &

what these commands does is , it creates 3 mongodb instances on three different ports(27018,27019,27020) respectively

and it creates one replicaSet with the name “itboons” as provided in the command with –replSet followed by the name

It also creates three different log files with the names 1.log , 2.log, 3.log respectively and you can mention the path to those log files as per your own convience

Then the command also creates 3 different folders for the database storage .. rs1,rs2,rs3 respectively.

And the & at the end of the command signifies that I run in the background. So you run these 3 commands 3 different mongodb instances are up and running in your machine.

To check that try this command

Ps -ef | grep mongod

This command will show the 3 processes on the machine.

After you have created 3 instances and after they are up and running . the next step is to choose either one of these three machines as primary for now and open its mongo shell

So in this case I have choosen the instance running on port 27018

Write the command

“mongo -host localhost:27018”

This command will take you to the shell of instance running on port 27018

After getting to the shell .

3) We need to specify the configurations of the replicaSet so in this case I have created a variable config and this is what it consists

   ```
   config = {_id: "itboons", members :[{_id : 0, host : "localhost:27018"}
   ,{_id : 1, host : "localhost:27019"}
   ,{_id : 2, host : "localhost:27020"}]};
   ```

This variable consists of the name of the replicaset and the members of the replica set so for this case we have as shown above

After assigning the value to the variable. we need to initiate the replica set

So for that we do

4) rs.initiate(config)

   This will initiate the replicaset with the specified configurations and members.
   After you have done initiating the replica set
   You can check whether the replicaset is working or not .
   By using the command

   rs.status()

   the output should be something like this

```
itboons:PRIMARY> rs.status()
{
        "set" : "itboons",
        "date" : ISODate("2018-05-21T05:24:41.952Z"),
        "myState" : 1,
        "term" : NumberLong(11),
        "heartbeatIntervalMillis" : NumberLong(2000),
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1526880274, 1),
                        "t" : NumberLong(11)
                },
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1526880274, 1),
                        "t" : NumberLong(11)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1526880274, 1),
                        "t" : NumberLong(11)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1526880274, 1),
                        "t" : NumberLong(11)
                }
        },
        "members" : [
                {
                        "_id" : 1,
                        "name" : "localhost:27018",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 2859,
                        "optime" : {
                                "ts" : Timestamp(1526880274, 1),
                                "t" : NumberLong(11)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1526880274, 1),
                                "t" : NumberLong(11)
                        },
                        "optimeDate" : ISODate("2018-05-21T05:24:34Z"),
                        "optimeDurableDate" : ISODate("2018-05-21T05:24:34Z"),
                        "lastHeartbeat" : ISODate("2018-05-21T05:24:41.878Z"),
                        "lastHeartbeatRecv" : ISODate("2018-05-21T05:24:41.680Z"),
                        "pingMs" : NumberLong(0),
                        "syncingTo" : "localhost:27019",
                        "configVersion" : 3
                },
                {
                        "_id" : 2,
                        "name" : "localhost:27019",
```

```
                        "name" : "localhost:27019",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 2861,
                        "optime" : {
                                "ts" : Timestamp(1526880274, 1),
                                "t" : NumberLong(11)
                        },
                        "optimeDate" : ISODate("2018-05-21T05:24:34Z"),
                        "electionTime" : Timestamp(1526878172, 1),
                        "electionDate" : ISODate("2018-05-21T04:49:32Z"),
                        "configVersion" : 3,
                        "self" : true
                },
                {
                        "_id" : 3,
                        "name" : "localhost:27020",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 2859,
                        "optime" : {
                                "ts" : Timestamp(1526880274, 1),
                                "t" : NumberLong(11)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1526880274, 1),
                                "t" : NumberLong(11)
                        },
                        "optimeDate" : ISODate("2018-05-21T05:24:34Z"),
                        "optimeDurableDate" : ISODate("2018-05-21T05:24:34Z"),
                        "lastHeartbeat" : ISODate("2018-05-21T05:24:41.878Z"),
                        "lastHeartbeatRecv" : ISODate("2018-05-21T05:24:41.909Z"),
                        "pingMs" : NumberLong(0),
                        "syncingTo" : "localhost:27019",
                        "configVersion" : 3
                }
        ],
        "ok" : 1,
        "operationTime" : Timestamp(1526880274, 1),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1526880274, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
```


This is the proof that the replication set is configured and up and running

Now only one step is left before the replicaset is completed

After the status is as shown above you need to mention that the two secondary machines as slaves ok

If we don't do that then the data will be replicated on the secondary machines but we wont be able to see that so to be able to access the data we write

"db.setSlaveOk()"

In both the secondary machines
Once you have done that
You can test the machine by inserting a document in the primary machine and you can go and check in the secondary machine whether the data is replicated or not.

So this is basically how to create a simple replicaSet

Now comes the part where we need to mention that where to read and where to write . So for that there is an option in replicatset which is called

ReadPreference :

So for the kind of arrangement we want ie read from secondary and write to primary we need to mention that option in the url on our backend side where we connect the mongodb

So in this case

mongo.connect('mongodb://localhost:27018,localhost:27019,localhost:27020/test?replicaSet=itboons&readPreference=secondary')

what this does is , this option lets all the post api calls to write to the primary mongodb instance and
all the read would be done from the secondary machine

so if there is no secondary machine and only primary machine (which will never be the case in our scenario because only one machine cannot conduct a poll and even though it would have been a primary machine it cannot stay primary since it need n/2 votes to remain primary and in this case it wouldn't be the case) then we will be able to write to the database but we wont be read since the option that we have chosen is secondary

but if the above scenario occurs then there is another option that can tackle that situation which is

"secondaryPreferred" option this will let the read happen from the secondary and if the secondary is not available then it will read from primary .

After the replicaSet is up and running

Clone the git repository get into the folder

5) get into backend folder and run “npm install” this will install all the node modules from the pagkage.json file

6) No run “npm start” this will start the node application and since the database is on the localhost no need to make any changes in the code .

7) After that you get into the frontend folder . run “npm install”

8) Then run “npm start” this will run the react server on the localhost:3000 from there you can do register and login .

9) Now since everything is running you can turn off the primary node in the mongodb and the application will still be running and one secondary will become primary and the database becomes fault tolerance

Once you start the other turned off mongodb instance it will then become the secondary machine and will join the replicaset.