



CONESTOGA COLLEGE

NTWK8111 – PROJECT II



ROCKET.CHAT: *BUILD BOOK*

GROUP 1 MEMBERS:

MESTANZA OSCAR	8945212
PATEL SHUBHAM	8984389
SHUKLA HARSHUL	8986048
SINGH HARVEER	8983964

PROFESSOR:

HALLMAN **DREW**

DECEMBER, 2024

TABLE OF CONTENTS

TABLE OF CONTENTS.....	2
INTRODUCTION	3
ACTIVE DIRECTORY	3
DHCP.....	3
APPLICATION: ROCKETCHAT.....	3
FEATURES OF ROCKET CHAT	4
DATABASE: MONGODB.....	4
LOAD BALANCER: NGINX.....	4
APACHE WEB SERVER	4
ARCHITECTURE	5
NETWORK SETTINGS.....	6
ACTIVE DIRECTORY DOMAIN SERVICE.....	8
ACTIVE DIRECTORY REPLICATION	8
DHCP.....	9
UBUNTU DEPLOYMENT FOR MONGODB.....	11
MONGODB DEPLOYMENTS	11
DATABASE CONFIGURATION	13
MONGODB CONFIGURATION FILE	14
DATABASE REPLICATION	15
DEPLOYMENT OF UBUNTU HEADLESS FOR ROCKETCHAT	18
DEPLOYMENT OF ROCKETCHAT	19
ROCKET CHAT	20
NGINX	21
WEB.....	22
CONCLUSION.....	25
REFERENCES	26

INTRODUCTION

Rocket.Chat is an open-source communication platform that aims to make how teams communicate more human with real-time messaging, video calls, and file-sharing. Considered highly adaptable, this self-hosted alternative to other proprietary systems includes group chat, private messaging, and integrations with third-party services. It is suitable for any business, development community, or educational organization.

In this project, the deployment and configuration of Rocket.Chat is focused on a Linux environment. The implementation makes sure that the communication infrastructure is secure, scalable, and efficient by leveraging the stability and flexibility of Linux. This project covers the installation process, system configuration, and key features of Rocket.Chat may be used to transform team collaboration in modern IT environments.

In this setup, the project also covers integrating Linux services, user management, and customization options to make Rocket.Chat suitable for specific organizational needs.

ACTIVE DIRECTORY

Active Directory integration extends the functionality of Rocket.Chat by providing centralized user management and secure authentication. By integrating AD, users will be able to log in to Rocket.Chat using their existing credentials via LDAP integration, thus providing single sign-on and smooth access control. This setup simplifies user provisioning, enforces organizational security policies, and ensures consistent access permissions. Integrating AD with Rocket.Chat is important, especially in an enterprise environment, because it allows for role-based access control, scalability, and smooth user management across hybrid platforms.

DHCP

Dynamic Host Configuration Protocol (DHCP) is a network management protocol that automates assigning IP addresses and other network configuration parameters to devices on a network.

APPLICATION: ROCKETCHAT

Rocket.Chat is a freely downloadable, freeware application for teamwork and immediate communication. It is a great option for businesses searching for a safe and adaptable communication solution because it has features equivalent to those of Slack or Microsoft Teams. The key reasons for choosing this application in our team were that it is one of the most secure on private networks, has an open-source license, compliance, interoperability and RBAC featured as compared to other applications on the primes.

FEATURES OF ROCKET CHAT

- ❖ **Real-Time Communication:** This application organizes seamless real-time communication through different sources of connections like voice calls, video calls and text messaging.
- ❖ **Theming & modification:** Changing colour, font, and other personal choices are also available.
- ❖ **Multiple user support:** This has all the features available for creating user support and even for their roles so that in the department, everyone has their own departments and access according to their positions.
- ❖ **Room Creation Support:** This is the feature all communication applications have these days for making a group to communicate between users.
- ❖ **Security and Privacy:** It has the best privacy and security rating compared to other applications like Meter, Next Cloud and other tools like Zimbra.
- ❖ **Omnichannel Expansions:** This means that it can be integrated with many other tools and applications so that it supports other applications.

DATABASE: MONGODB

MongoDB provides a database built without SQL, focusing on versatility, scalability, and enhanced efficiency. In contrast to conventional relational data tables, MongoDB organizes information using a document-oriented format, making it a great option for current applications such as Rocket.Chat wherever changing and constantly updated processing is essential.

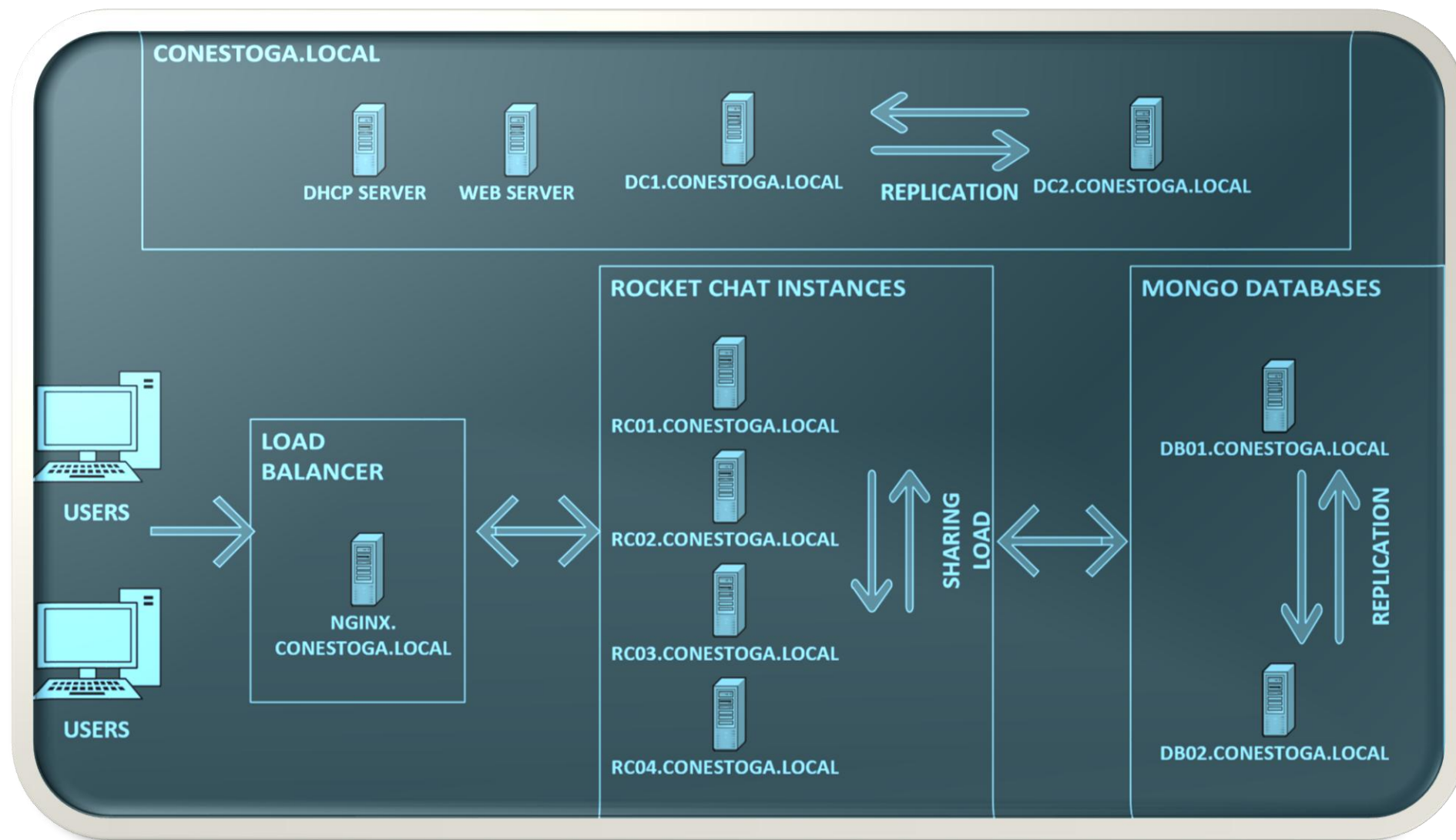
LOAD BALANCER: NGINX

Nginx serves as a proxy service and website host with excellent performance. nginx is the one that produces it. It is frequently used to distribute the load, provide static information, and function as an internet application's reverse router. It can manage applications for constantly changing information using uWSGI and FastCGI, along with additional protocols besides the browser. Nginx is renowned for quickly, scalable, and effectively handling multiple simultaneous interactions.

APACHE WEB SERVER

The largest and most popular open-source internet servers worldwide reside in the Apache HTTP Server, also known as Apache. This is renowned for its solid efficiency, adaptability, and the ability to configure. Apache is frequently used for maintaining web pages, applications, and other services and can deliver dynamic and static information.

ARCHITECTURE



NETWORK SETTINGS

Domain Name: Conestoga.ca

SHUBHAM						
OS	COMPUTER NAME	IP ADDRESS	NETMASK	GATEWAY	DNS1	DNS2
WIN SERVER 2019	CONESTOGA-DC1	172.23.38.231	255.255.255.0	172.22.38.1	142.156.150.112	142.156.150.113
WIN SERVER 2019	CONESTOGA-DC2	172.23.38.232	255.255.255.0	172.22.38.1	142.156.150.112	142.156.150.113
LINUX SERVER	CONESTOGA-RC1	172.23.38.233	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX CLIENT	CONESTOGA-CL1	DYNAMIC IP				
WIN CLIENT	CONESTOGA-CL2	DYNAMIC IP				

HARSHUL						
OS	COMPUTER NAME	IP ADDRESS	NETMASK	GATEWAY	DNS1	DNS2
LINUX SERVER	CONESTOGA-RC2	172.23.38.235	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX SERVER	CONESTOGA-DB1	172.23.38.238	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX CLIENT	CONESTOGA-CL3	DYNAMIC IP				
WIN CLIENT	CONESTOGA-CL4	DYNAMIC IP				

HARVEER						
OS	COMPUTER NAME	IP ADDRESS	NETMASK	GATEWAY	DNS1	DNS2
LINUX SERVER	CONESTOGA-RC3	172.23.38.236	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX SERVER	CONESTOGA-DB2	172.23.38.239	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX CLIENT	CONESTOGA-CL5	DYNAMIC IP				
WIN CLIENT	CONESTOGA-CL6	DYNAMIC IP				

OSCAR						
OS	COMPUTER NAME	IP ADDRESS	NETMASK	GATEWAY	DNS1	DNS2
LINUX SERVER	CONESTOGA-DHCP	172.23.38.234	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX SERVER	CONESTOGA-RC4	172.23.38.237	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX SERVER	CONESTOGA-LB	172.23.38.240	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX SERVER	CONESTOGA-WEB	172.23.38.251	255.255.255.0	172.22.38.1	172.23.38.231	172.23.38.232
LINUX CLIENT	CONESTOGA-CL7	DYNAMIC IP				
WIN CLIENT	CONESTOGA-CL8	DYNAMIC IP				

ACTIVE DIRECTORY DOMAIN SERVICE

Active Directory integration gives Rocket.Chat several capabilities: central management of user accounts and secure authentication. The AD allows the user to log in to Rocket.Chat using their existing credentials through LDAP integration enables SSO and smooths access control.

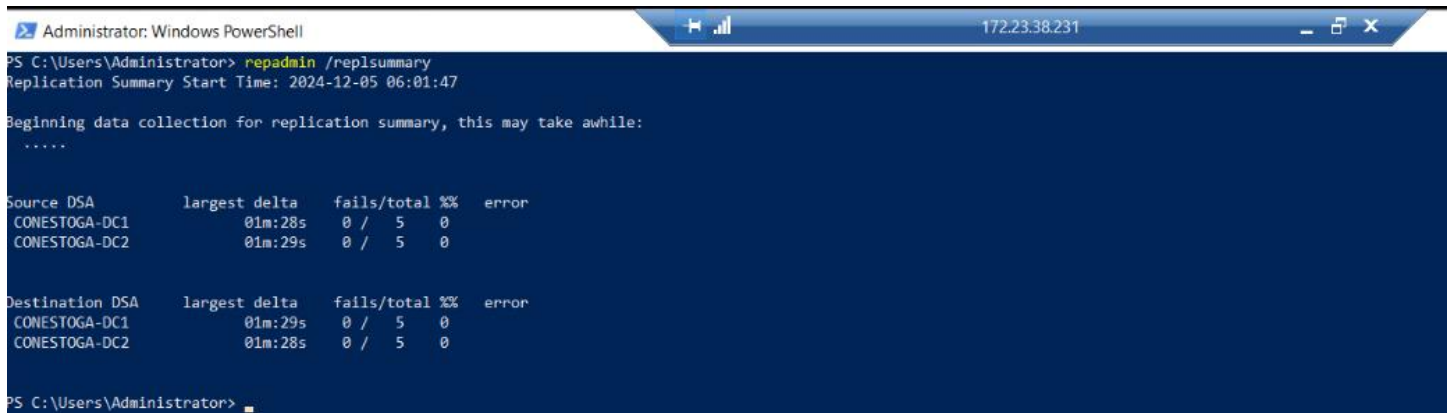
This setup simplifies the provisioning of users, ensuring that accounts and permissions are managed centrally within AD. It enforces the security policies of an organization, such as password complexity and account lockout, thus enhancing the overall security of the Rocket.Chat deployment. Besides, AD integration supports RBAC, which allows user roles and permissions to be aligned with predefined organizational structures.

It is convenient in enterprise environments and has the scalability to handle large volumes of users. It maintains consistent permission and provides hybrid functionality across Windows and Linux by connecting Rocket.Chat to AD, the admins reduce the time and effort in managing users and offer the users a seamless experience with single sign-on authentication and secure control in one place.

ACTIVE DIRECTORY REPLICATION

The primary objectives of Active Directory replication are the following:

- ❖ Ensuring Data Consistency
- ❖ Maintaining Availability
- ❖ Load Balancing
- ❖ Optimizing Network Traffic
- ❖ Managing Replication Topology
- ❖ Reducing Conflicts and ensuring integrity
- ❖ Monitoring



```
Administrator: Windows PowerShell
PS C:\Users\Administrator> repadmin /replsummary
Replication Summary Start Time: 2024-12-05 06:01:47

Beginning data collection for replication summary, this may take awhile:
.....

Source DSA          largest delta    fails/total %%  error
CONESTOGA-DC1       01m:28s        0 / 5 0
CONESTOGA-DC2       01m:29s        0 / 5 0

Destination DSA     largest delta    fails/total %%  error
CONESTOGA-DC1       01m:29s        0 / 5 0
CONESTOGA-DC2       01m:28s        0 / 5 0

PS C:\Users\Administrator>
```

```
Administrator: Windows PowerShell 172.23.38.231
PS C:\Users\Administrator> Get-ADReplicationPartnerMetadata -Target "conestoga.local" -Scope Domain

CompressChanges : False
ConsecutiveReplicationFailures : 0
DisableScheduledSync : False
IgnoreChangeNotifications : False
IntersiteTransport :
IntersiteTransportGuid :
IntersiteTransportType : IP
LastChangeUsn : 127512
LastReplicationAttempt : 12/5/2024 6:00:18 AM
LastReplicationResult : 0
LastReplicationSuccess : 12/5/2024 6:00:18 AM
Partition : DC=conestoga,DC=local
PartitionGuid : ce011677-0294-4d94-a33f-562654b86c40
Partner : CN=NTDS Settings,CN=CONESTOGA-DC2,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=conestoga,DC=local
PartnerAddress : 619abb0a-cf19-49fe-b007-bc34954a00d6._msdcs.conestoga.local
PartnerGuid : 619abb0a-cf19-49fe-b007-bc34954a00d6
PartnerInvocationId : 8b6556ac-98b0-4af9-ae59-a407ee753e82
PartnerType : Inbound
ScheduledSync : True
Server : CONESTOGA-DC1.conestoga.local
SyncOnStartup : True
TwoWaySync : False
UsnFilter : 127512
Writable : True

CompressChanges : False
ConsecutiveReplicationFailures : 0
DisableScheduledSync : False
IgnoreChangeNotifications : False
IntersiteTransport :
IntersiteTransportGuid :
IntersiteTransportType : IP
LastChangeUsn : 127398
LastReplicationAttempt : 12/5/2024 6:00:19 AM
LastReplicationResult : 0
LastReplicationSuccess : 12/5/2024 6:00:19 AM
Partition : DC=conestoga,DC=local
PartitionGuid : ce011677-0294-4d94-a33f-562654b86c40
Partner : CN=NTDS Settings,CN=CONESTOGA-DC1,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=conestoga,DC=local
PartnerAddress : 2baec1c9-a188-4e06-9ab4-fa948fc0eb42._msdcs.conestoga.local
PartnerGuid : 2baec1c9-a188-4e06-9ab4-fa948fc0eb42
PartnerInvocationId : 2baec1c9-a188-4e06-9ab4-fa948fc0eb42
PartnerType : Inbound
ScheduledSync : True
Server : CONESTOGA-DC2.conestoga.local
SyncOnStartup : True
TwoWaySync : False
UsnFilter : 127398
Writable : True
```

DHCP

To deploy a DHCP server based on Linux:

1. Download and install the package using *apt-get install isc-dhcp-server*
2. Configure the file called *dhcpd.conf* in */etc/dhcp/*
3. Enable the service when it boots using *systemctl enable dhcpd.service*
4. Start the service typing *systemctl start dhcpd.service*

```
VMware Fusion  File  Edit  View  Virtual Machine  Window  Help
NTWK8111-DHCP

# set.
#host fantasia {
#  hardware ethernet 08:00:07:26:c0:a5;
#  fixed-address fantasia.example.com;
#}

# You can declare a class of clients and then do address allocation
# based on that.  The example below shows a case where all clients
# in a certain class get addresses on the 10.17.224/24 subnet, and all
# other clients get addresses on the 10.0.29/24 subnet.

#class "foo" {
#  match if substring (option vendor-class-identifier, 0, 4) = "SUNW";
#}

#shared-network 224-29 {
#  subnet 10.17.224.0 netmask 255.255.255.0 {
#    option routers rtr-224.example.org;
#  }
#  subnet 10.0.29.0 netmask 255.255.255.0 {
#    option routers rtr-29.example.org;
#  }
#  pool {
#    allow members of "foo";
#    range 10.17.224.10 10.17.224.250;
#  }
#  pool {
#    deny members of "foo";
#    range 10.0.29.10 10.0.29.230;
#  }
#}

#Configuration for conestoga.local
option domain-name-servers 172.23.38.231, 172.23.38.232;
option domain-name "conestoga.local";

subnet 172.23.38.0 netmask 255.255.255.0 {
option subnet-mask      255.255.255.0;
option broadcast-address 172.23.38.255;
option routers          172.23.38.1;

range                    172.23.38.201 172.23.38.220;
}

drew@conestoga-dhcp:~$
```

option domain-name-servers *"DC1's IP address", "DC2's IP address"*

option domain-name *"domain name"*

subnet *"network"* netmask *"mask"* {

option subnet-mask *"mask";*

option broadcast-address *"broadcast IP";*

option routers *"gateway";*

range *"start IP address end IP address";*

}

UBUNTU DEPLOYMENT FOR MONGODB

These requirements for MongoDB to be deployed must be on our Ubuntu system so that the perfect version of Ubuntu 22.04 with 4 GB RAM and 4 CPU & 50 GB storage space is compatible. After Ubuntu is configured for the login screen of the user credentials, As we have logged in, the system starts the process by the following steps:

- ❖ Start updating and upgrading the system package list.

```
$ sudo apt update , $ sudo apt upgrade
```

- ❖ After the upgrades, install some of the dependencies.

```
$ sudo apt install build-essential
```

- ❖ Make sure that it is pointing to the correct nameserver and Domain.

```
$ sudo nano /etc/resolv.conf
```

- ❖ Connecting the system to the domain and installing dependencies

```
$ sudo apt install -y realmd libnss-sss libpam-sss sssd sssd-tools oddjob oddjob-mkhomedir adcli  
samba-common-bin krb5-user
```

- ❖ Join the system as a Kerberos member and the domain member

```
$ sudo apt kinit Username@DOMAIN-NAME.TLD-EXTENSIONS
```

```
$sudo apt realm join -U Username DOMAIN-NAME-TLD-EXTENSION
```

- ❖ Testing connectivity

```
$ping (website, domain name)
```

MONGODB DEPLOYMENTS

To deploy the MongoDB database, we have to have the Ubuntu version 22.04 jammy edition to fully support all the functionality. After the upper part of the Ubuntu server is deployed and pointed toward the domain, we have to install some of the dependencies so that MongoDB can be installed. Mongo DB is the primary data center for the rocket chat application to store and the functionality to take proper action in the scenario. Below, the instructions will show how our team have deployed the MongoDB on the Ubuntu headless:

❖ Deployment Stage:

- Pre-Deployment Setup: Update the system

```
$sudo apt update && sudo apt upgrade -y
```

- Configuration of the firewall: To allow the default port for the mongodb

```
$sudo ufw allow 27017
```

```
$sudo ufw enable
```

❖ MONGODB Installation: Import the Mongodb Public Key

```
$curl -fsSL https://pgp.mongodb.com/server-6.0.asc
```

```
$sudo gpg -o /usr/share/keyrings/mongodb-server-6.0.gpg --dearmor
```

❖ ADD MongoDB to the repository

```
$echo "deb [signed-by=/usr/share/keyrings/mongodb-server-6.0.gpg] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0 multiverse"
```

```
$ sudo tee /etc/apt/sources.list.d/mongodb-org-6.0.list
```

❖ Installation of MONGODB:

```
$sudo apt update
```

```
$sudo apt install -y mongodb-org
```

❖ Starting the container and the service for the MongoDB

```
$sudo systemctl start mongod
```

```
$sudo systemctl enable mongod
```

❖ Restart the MongoDB Service to take action

```
$sudo systemctl restart mongod
```

DATABASE CONFIGURATION

As we have talked about the primary role of the database in the rocket chat, we would like to show some of the points below for the beneficial and key reasons why this is important to make a database:

- ❖ Data Storage and Management
- ❖ High Performance for real-time applications
- ❖ Flexibility & Scalability
- ❖ Replication & Availability
- ❖ Efficient

Database Deployment and details:

❖ Step 1: Open the MongoDB shell

```
mongosh
```

❖ Step 2: Create Database

```
use rocketchat
```

```
rs.initiate()
```

❖ Step 3: Create a User for the database

```
db.createUser ({  
  user: "root"  
  pwd: "Secret55!"  
  roles: [ { role: "readWrite", db: "rocketchat" } ]  
});
```

❖ Step 4: Verification for user and databases

```
db.getusers()  
  
show databases;  
  
exit;
```

MONGODB CONFIGURATION FILE

To deploy Rocketchat in this project, Mongodb acts as a primary database to store and manage real-time connectivity, user personal details and the application's configuration. However, rocketchat itself is only useful with MongoDB, which is crucial to make it collaborative for a powerful platform. To organize rocketchat, MongoDB is configured in the location of **/etc/mongod.conf**. This is where different configurations like replication, authentication, storage and etc.

```
GNU nano 6.2 /etc/mongod.conf *
# mongod.conf

# for documentation of all options, see:
# http://docs.mongodb.org/manual/reference/configuration-options/

# Where and how to store data.
storage:
  dbPath: /var/lib/mongodb
# engine:
# wiredTiger:
# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /var/log/mongodb/mongod.log

# network interfaces
net:
  port: 27017
  bindIp: 0.0.0.0
# how the process runs
processManagement:
  timeZoneInfo: /usr/share/zoneinfo

#security:
#security:
# authorization: enabled
#operationProfiling:

replication:
  replSetName: "Rcrepl"
#sharding:

## Enterprise-Only Options:
```

G Help	O Write Out	W Where Is	K Cut	T Execute	C Location	M-U Undo
X Exit	R Read File	\ Replace	U Paste	J Justify	_ Go To Line	M-E Redo

DATABASE REPLICATION

First, A key component of MongoDB is replicating, which keeps many copies of your database on several servers to guarantee high availability and fault tolerance. The expert deployment procedure for configuring Rocket-optimized MongoDB replicate is described in this document. Nothing could overcome the redundancy issue without replicating these servers so that if one database server is down, it can be faced with the other server to store the information without interruption. There is not much on the database replication part because everything is included in the file where the main configuration of the Mongoddb is placed. The main path for the path of the mongoddb is in the **/etc/mongod.conf**. Below are the details which should be changed in the attached file path. This is basically pointing one server to the other by name.

❖ Replication:

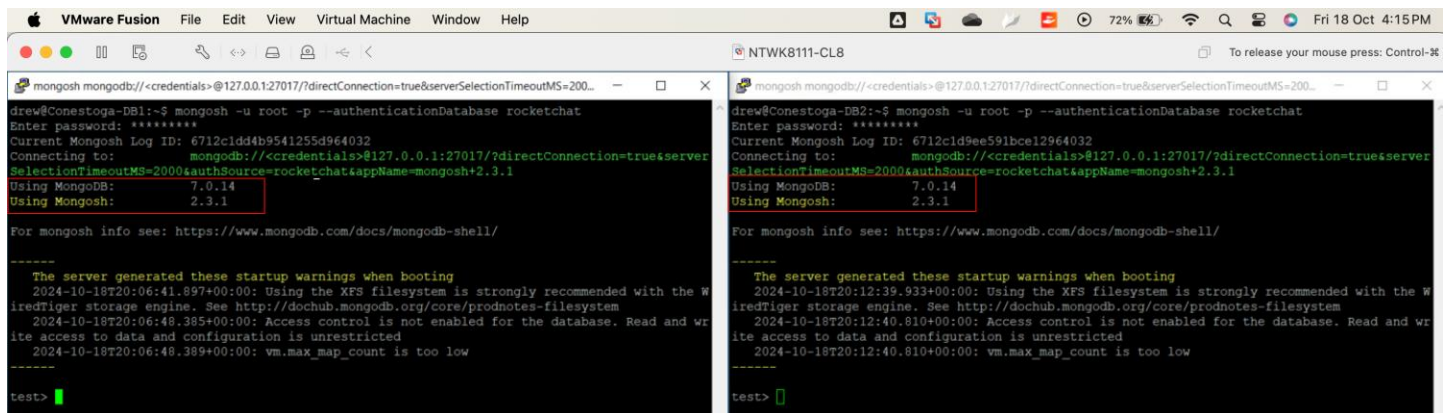
replSetName: "Rcrep1"

- ❖ Once the database is created and the replication name is set, it has to be configured in the database using the following command:

```
mongosh -u root -p - --authenticationDatabase rocketchat
```

root is the user

rocketchat is the database



The image shows two side-by-side terminal windows from a VMware Fusion environment. Both windows are running the MongoDB shell (mongosh) as the 'root' user. The left window shows the command `mongosh -u root -p - --authenticationDatabase rocketchat` and its output, including the MongoDB version (7.0.14) and the shell version (2.3.1). The right window shows the same command and output. Both windows also display startup warnings from the MongoDB server, including information about the XFS filesystem and access control.

- ❖ To initiate the replication, use **rs.initiate()** and **rs.add("fqdn:port")** to add a database server

```
VMware Fusion  File  Edit  View  Virtual Machine  Window  Help
NTWK8111-CL8  To release your mouse press: Control-⌘

mongosh mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=200...

drew@Conestoga-DB1:~$ mongosh -u root -p --authenticationDatabase rocketchat
Enter password: *****
Current Mongosh Log ID: 6712cidd4b9541255d964032
Connecting to:  mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&authSource=rocketchat&appName=mongosh+2.3.1
Using MongoDB:  7.0.14
Using Mongosh:  2.3.1

For mongosh info see: https://www.mongodb.com/docs/mongosh-shell/

-----
The server generated these startup warnings when booting
2024-10-18T20:06:41.897+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2024-10-18T20:06:48.385+00:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
2024-10-18T20:06:48.389+00:00: vm.max_map_count is too low
-----

test> use rocketchat
switched to db rocketchat
rocketchat> rs.initiate()

info2: 'no configuration specified. Using a default configuration for the set',
me: 'Conestoga-DB1.conestoga.local:27017',
ok: 1

RCrepl [direct: other] rocketchat> rs.add("conestoga-db2.conestoga.local:27017")

ok: 1,
'selectionTime': {
  clusterTime: Timestamp({ t: 1729282747, i: 1 }),
  signature: {
    hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAAA', 0),
    keyId: Long('0')
  },
  operationTime: Timestamp({ t: 1729282747, i: 1 })
}
RCrepl [direct: primary] rocketchat>

RCrepl [direct: secondary] rocketchat> rs.conf()
{
  _id: 'RCrepl',
  version: 3,
  term: 1,
  members: [
    {
      _id: 0,
      host: 'Conestoga-DB1.conestoga.local:27017',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long('0'),
      votes: 1
    },
    {
      _id: 1,
      host: 'conestoga-db2.conestoga.local:27017',
      arbiterOnly: false,
      buildIndexes: true,
      hidden: false,
      priority: 1,
      tags: {},
      secondaryDelaySecs: Long('0'),
      votes: 1
    }
  ],
  protocolVersion: Long('1'),
  writeConcernMajorityJournalDefault: true,
  settings: {
    chainingAllowed: true,
    heartbeatIntervalMillis: 2000,
    heartbeatTimeoutSecs: 10,
    electionTimeoutMillis: 10000,
    catchUpTimeoutMillis: -1,
    catchUpTakeoverDelayMillis: 30000,
    getLastErrorModes: {},
    getLastErrorDefaults: { w: 1, wtimeout: 0 },
    replicaSetId: ObjectId('6712c24b7388e97852611325')
  }
}
RCrepl [direct: secondary] rocketchat>
```

```
VMware Fusion  File  Edit  View  Virtual Machine  Window  Help
NTWK8111-CL8  To release your mouse press: Control-⌘

mongosh mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&authSource=rocketchat

RCrepl [direct: secondary] rocketchat> rs.status()

set: 'RCrepl',
date: ISODate('2024-10-18T20:20:56.626Z'),
myState: 2,
term: Long('1'),
syncSourceHost: 'Conestoga-DB1.conestoga.local:27017',
syncSourceId: 0,
heartbeatIntervalMillis: Long('2000'),
majorityVoteCount: 2,
writeMajorityCount: 2,
votingMembersCount: 2,
writableVotingMembersCount: 2,
optimes: {
  lastCommittedOptime: { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
  lastCommittedWallTime: ISODate('2024-10-18T20:20:55.287Z'),
  readConcernMajorityOptime: { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
  appliedOptime: { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
  durableOptime: { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
  lastAppliedWallTime: ISODate('2024-10-18T20:20:55.287Z'),
  lastDurableWallTime: ISODate('2024-10-18T20:20:55.287Z')
},
lastStableRecoveryTimestamp: Timestamp({ t: 1729282806, i: 1 }),
members: [
  {
    _id: 0,
    name: 'Conestoga-DB1.conestoga.local:27017',
    health: 1,
    state: 1,
    stateStr: 'PRIMARY',
    uptime: 109,
    optime: { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
    optimeDurable: { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
    optimeDate: ISODate('2024-10-18T20:20:55.000Z'),
    optimeDurableDate: ISODate('2024-10-18T20:20:55.000Z'),
    lastAppliedWallTime: ISODate('2024-10-18T20:20:55.287Z'),
    lastDurableWallTime: ISODate('2024-10-18T20:20:55.287Z'),
    lastHeartbeat: ISODate('2024-10-18T20:20:55.419Z'),
    lastHeartbeatRecv: ISODate('2024-10-18T20:20:55.420Z'),
    pingMs: Long('0'),
    lastHeartbeatMessage: '',
    syncSourceHost: '',
    syncSourceId: -1,
    infoMessage: '',
    electionTime: Timestamp({ t: 1729282635, i: 2 }),
    electionDate: ISODate('2024-10-18T20:17:15.000Z'),
    configVersion: 3,
    configTerm: 1
  }
],
```

```
mongosh mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&authSource=rocketchat
{
  "_id": 0,
  "name": "Conestoga-DB1.conestoga.local:27017",
  "health": 1,
  "state": 1,
  "stateStr": "PRIMARY",
  "uptime": 109,
  "optime": { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
  "optimeDurable": { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
  "optimeDate": ISODate('2024-10-18T20:20:55.000Z'),
  "optimeDurableDate": ISODate('2024-10-18T20:20:55.000Z'),
  "lastAppliedWallTime": ISODate('2024-10-18T20:20:55.287Z'),
  "lastDurableWallTime": ISODate('2024-10-18T20:20:55.287Z'),
  "lastHeartbeat": ISODate('2024-10-18T20:20:55.419Z'),
  "lastHeartbeatRecv": ISODate('2024-10-18T20:20:55.420Z'),
  "pingMs": Long('0'),
  "lastHeartbeatMessage": "",
  "syncSourceHost": "",
  "syncSourceId": -1,
  "infoMessage": "",
  "electionTime": Timestamp({ t: 1729282635, i: 2 }),
  "electionDate": ISODate('2024-10-18T20:17:15.000Z'),
  "configVersion": 3,
  "configTerm": 1
},
{
  "_id": 1,
  "name": "Conestoga-DB2.conestoga.local:27017",
  "health": 1,
  "state": 2,
  "stateStr": "SECONDARY",
  "uptime": 497,
  "optime": { ts: Timestamp({ t: 1729282855, i: 1 }), t: Long('1') },
  "optimeDate": ISODate('2024-10-18T20:20:55.000Z'),
  "lastAppliedWallTime": ISODate('2024-10-18T20:20:55.287Z'),
  "lastDurableWallTime": ISODate('2024-10-18T20:20:55.287Z'),
  "syncSourceHost": "Conestoga-DB1.conestoga.local:27017",
  "syncSourceId": 0,
  "infoMessage": "",
  "configVersion": 3,
  "configTerm": 1,
  "self": true,
  "lastHeartbeatMessage": ""
},
{
  "ok": 1,
  "clusterTime": {
    "clusterTime": Timestamp({ t: 1729282855, i: 1 }),
    "signature": {
      "hash": Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAAA', 0),

```

To check the status of the replication, use `rs.status()`

```
mongosh mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&authSource=rocketchat
)
RCrepl [direct: primary] rocketchat> cls
RCrepl [direct: primary] rocketchat> rs.status()
{
  "set": "RCrepl",
  "date": ISODate('2024-10-18T20:23:30.994Z'),
  "myState": 1,
  "term": Long('1'),
  "syncSourceHost": "",
  "syncSourceId": -1,
  "heartbeatIntervalMillis": Long('2000'),
  "majorityVoteCount": 2,
  "writeMajorityCount": 2,
  "votingMembersCount": 2,
  "writableVotingMembersCount": 2,
  "optimes": {
    "lastCommittedOptime": { ts: Timestamp({ t: 1729283006, i: 1 }), t: Long('1') },
    "lastCommittedWallTime": ISODate('2024-10-18T20:23:26.717Z'),
    "readConcernMajorityOptime": { ts: Timestamp({ t: 1729283006, i: 1 }), t: Long('1') },
    "appliedOptime": { ts: Timestamp({ t: 1729283006, i: 1 }), t: Long('1') },
    "durableOptime": { ts: Timestamp({ t: 1729283006, i: 1 }), t: Long('1') },
    "lastAppliedWallTime": ISODate('2024-10-18T20:23:26.717Z'),
    "lastDurableWallTime": ISODate('2024-10-18T20:23:26.717Z')
  },
  "lastStableRecoveryTimestamp": Timestamp({ t: 1729282995, i: 1 }),
  "electionCandidateMetrics": {
    "lastElectionReason": "electionTimeout",
    "lastElectionDate": ISODate('2024-10-18T20:17:15.460Z'),
    "electionTerm": Long('1'),
    "lastCommittedOptimeAtElection": { ts: Timestamp({ t: 1729282635, i: 1 }), t: Long('-1') },
    "lastSeenOptimeAtElection": { ts: Timestamp({ t: 1729282635, i: 1 }), t: Long('-1') },
    "numVotesNeeded": 1,
    "priorityAtElection": 1,
    "electionTimeoutMillis": Long('10000'),
    "newTermStartDate": ISODate('2024-10-18T20:17:15.499Z'),
    "wMajorityWriteAvailabilityDate": ISODate('2024-10-18T20:17:15.522Z')
  },
  "members": [
    {
      "_id": 0,
      "name": "Conestoga-DB1.conestoga.local:27017",
      "health": 1,
      "state": 1,
      "stateStr": "PRIMARY",
      "uptime": 1009,
      "optime": { ts: Timestamp({ t: 1729283006, i: 1 }), t: Long('1') },
      "optimeDate": ISODate('2024-10-18T20:23:26.000Z'),
      "lastAppliedWallTime": ISODate('2024-10-18T20:23:26.717Z'),

```

DEPLOYMENT OF UBUNTU HEADLESS FOR ROCKETCHAT

Before RocketChat to be deployed, we must have our Ubuntu system ready to install the application. So we used the version of Ubuntu 24.04 with 4 gb ram and 4 cpu & 50 gb storage space for it. After ubuntu is configured to the login screen of the user credentials. As we have logged in the system start the process by the following steps:

- ❖ Start updating and upgrading the system package list.

```
$ sudo apt update , $ sudo apt upgrade
```

- ❖ After the upgradations install some of the dependencies.

```
$ sudo apt install build-essential
```

- ❖ Make sure that it is pointing to the correct nameserver and Domain.

```
$ sudo nano /etc/resolv.conf
```

- ❖ Connecting the system to the domain and install dependencies

```
$ sudo apt install -y realmd libnss-sss libpam-sss sssd sssd-tools oddjob oddjob-mkhomedir adcli  
samba-common-bin krb5-user
```

- ❖ Join the system as a Kerberos member and to the domain member

```
$ sudo apt kinit Username@DOMAIN-NAME.TLD-EXTENSIONS
```

```
$sudo apt realm join -U Username DOMAIN-NAME-TLD-EXTENSION
```

- ❖ Testing connectivity

```
$ping (website, domain name)
```

DEPLOYMENT OF ROCKETCHAT

First of all, there are some of the prerequisites for the RocketChat application to be deployed. As the proper research technique and information is required to match the perfect version of the Debian ubuntu and version of the application. In this deployment our group decided to move on with the Ubuntu 24.04, which supports well with the version of the latest version of rocketchat. These are the prerequisites:

- ❖ A headless server based (Ubuntu / Debian-based)
- ❖ Required a root user or user with root privileges on server
- ❖ Some knowledge and understanding

Steps for Deployment:

Step 1: Start by installing the dependencies related to the rocketchat.

```
$ sudo apt install gnupg2 gnupg git unzip curl nodejs npm build-essential software-properties-common  
graphicsmagick gcc g++ make net-tools -y
```

Step 2: Configuring the Node.js

Use the curl command to download the nodesource

```
curl -fsSL https://deb.nodesource.com/setup\_14.x
```

Step 3: Install Global NPM Packages

```
$ sudo npm install -g inherits n
```

```
$ sudo n 8.11.3
```

Step 4: Download and Extract Rocket Chat

```
$ curl -L https://releases.rocket.chat/latest/download -o/tmp/rocket.chat.tgz
```

```
$ tar -xvzf /tmp/rocket.chat.tgz -C /tmp
```

Step 5: Move the file to the directory

```
$ sudo mv /tmp/bundle /opt/Rocket.Chat
```

Step 6: Configuring the System Service

```
$ sudo nano /lib/systemd/system/rocketchat.service
```

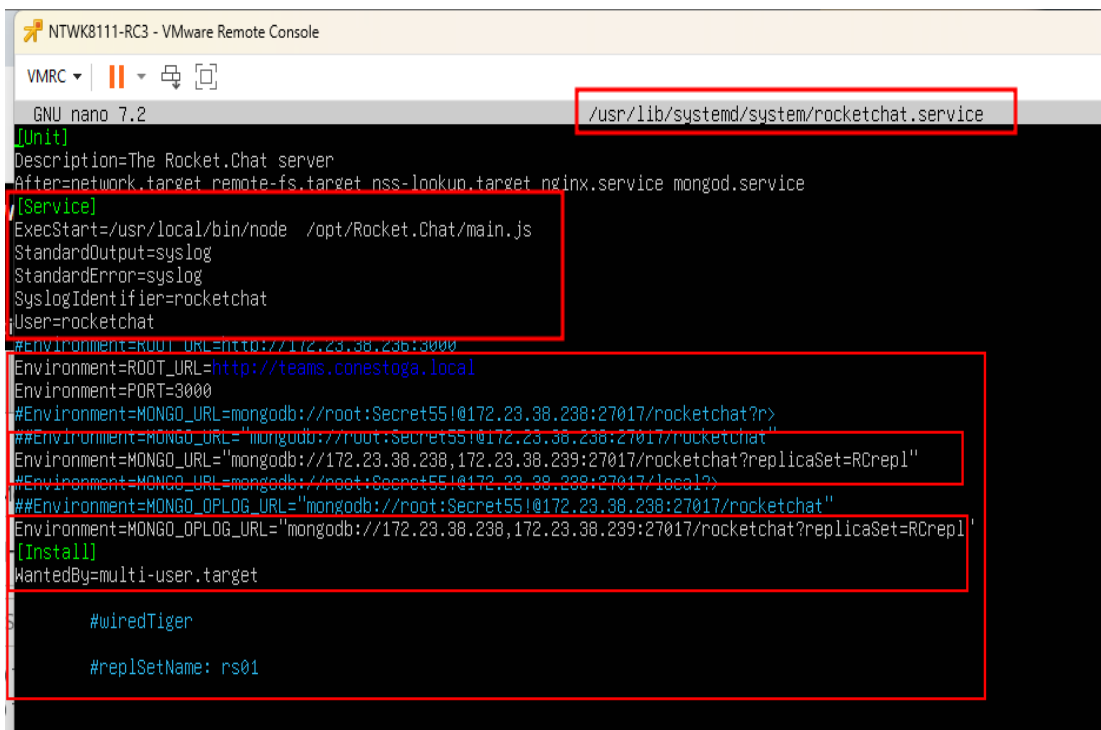
Step 7: Restarting & Enabling Services

```
$ sudo systemctl enable rocket
```

ROCKET CHAT

In this project, we have decided to install Rocket.chat on a headless machine but in a different way. So it can be deployed by using the docker it could be helpful in deploying the container to simplify it. We have deployed it without the docker on Ubuntu Version 24.02. In our group, everyone has each server for RocketChat, and they are pointing to a database. So, the configuration lies for the rocket.chat server is in the path **/usr/lib/systemd/system/rocketchat.service**. Some of the dependencies are important to get this work, like **sudo apt install gnupg2 gnpung git unzip build-essential wget nodejs npm software-properties-common graphicsmagick gcc g++ make net-tools -y**. To deploy Rocket.chat steps are below:

1. Install the required dependencies from above and start configuring.
2. Once the file is configured enable and restart the service with `sudo systemctl enable rocketchat.service`.
3. Open up a web browser and test it by typing the IP Address of the server.



```
NTWK8111-RC3 - VMware Remote Console
VMRC
GNU nano 7.2 /usr/lib/systemd/system/rocketchat.service
[Unit]
Description=The Rocket.Chat server
After=network.target remote-fs.target nss-lookup.target nginx.service mongod.service
[Service]
ExecStart=/usr/local/bin/node /opt/Rocket.Chat/main.js
StandardOutput=syslog
StandardError=syslog
SyslogIdentifier=rocketchat
User=rocketchat
#Environment=ROOT_URL=http://172.23.38.238:3000
Environment=ROOT_URL=http://teams.conestoga.local
Environment=PORT=3000
#Environment=MONGO_URL=mongodb://root:Secret55!@172.23.38.238:27017/rocketchat?r>
##Environment=MONGO_URL="mongodb://root:Secret55!@172.23.38.238:27017/rocketchat"
Environment=MONGO_URL="mongodb://172.23.38.238,172.23.38.239:27017/rocketchat?replicaSet=RCrepl"
#Environment=MONGO_URL=mongodb://root:Secret55!@172.23.38.238:27017/local?
##Environment=MONGO_OPLOG_URL="mongodb://root:Secret55!@172.23.38.238:27017/rocketchat"
Environment=MONGO_OPLOG_URL="mongodb://172.23.38.238,172.23.38.239:27017/rocketchat?replicaSet=RCrepl"
[Install]
WantedBy=multi-user.target

#wiredTiger

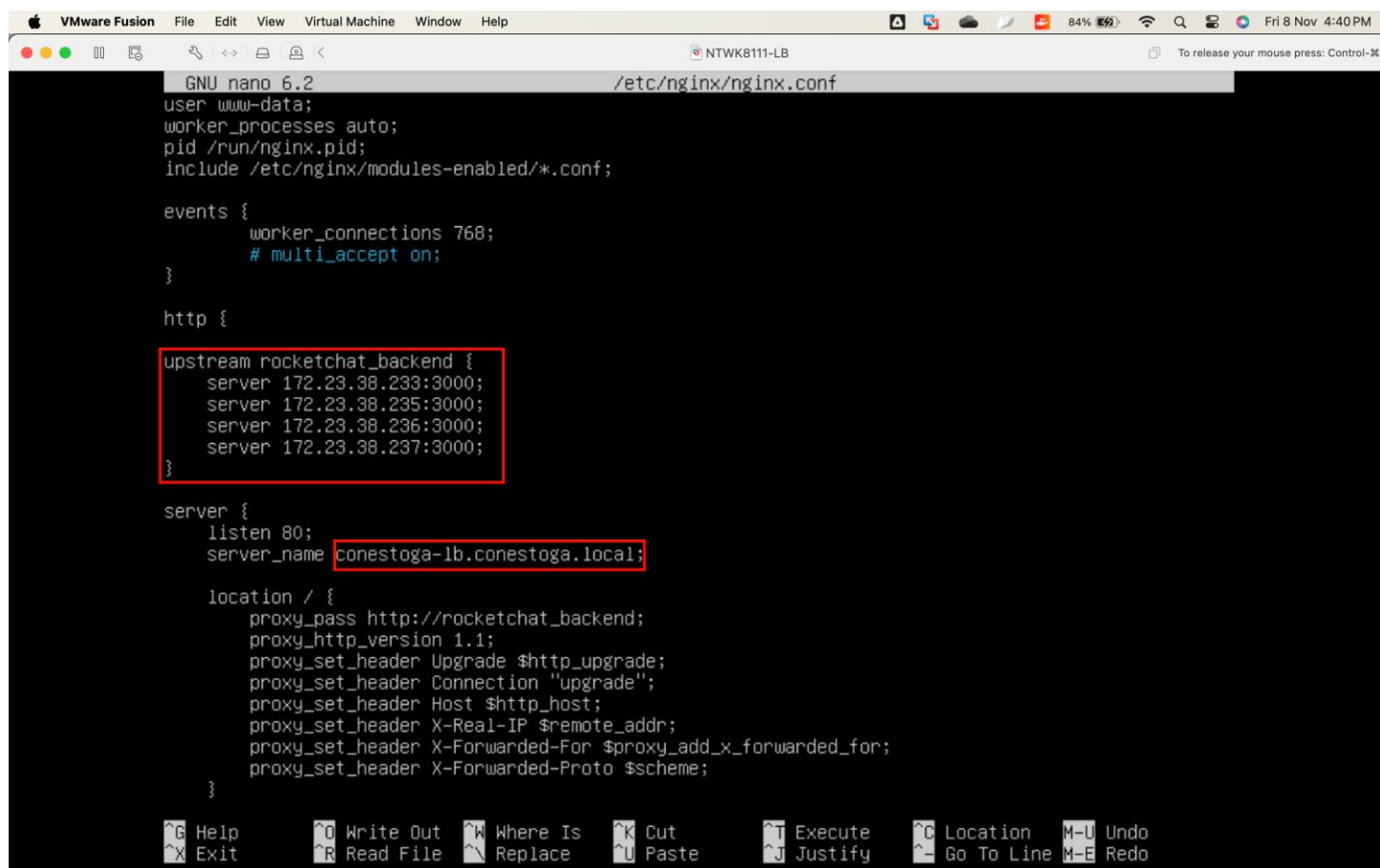
#replSetName: rs01
```

NGINX

Nginx can be configured as a reverse proxy and load balancer in your project. By default, Nginx uses the round-robin load balancing method, where requests are distributed evenly among the listed servers in the `nginx.conf` file in the path `/etc/nginx`.

To create a reverse proxy and a load balancer, these are the steps:

1. Install the nginx application using `apt install nginx`
2. Modify the `nginx.conf` according to the Rocketchat instances
3. Enable the service when it boots typing `systemctl enable nginx.service`
4. Start the service by typing `systemctl start nginx.service`



```
GNU nano 6.2 /etc/nginx/nginx.conf
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
    worker_connections 768;
    # multi_accept on;
}

http {

    upstream rocketchat_backend {
        server 172.23.38.233:3000;
        server 172.23.38.235:3000;
        server 172.23.38.236:3000;
        server 172.23.38.237:3000;
    }

    server {
        listen 80;
        server_name conestoga-lb.conestoga.local;

        location / {
            proxy_pass http://rocketchat_backend;
            proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Connection "upgrade";
            proxy_set_header Host $http_host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
        }
    }
}
```

WEB

To deploy a web server based on Linux, follow these steps:

1. Install Apache application using `apt install apache2`
2. Enable the service when it boots typing `systemctl enable apache2.service`
3. Then start the service using `systemctl start apache2.service`
4. Then, modify the files according to the deployed website. Files are in `/var/www/html/` folder



The screenshot shows a terminal window titled "NTWK8111-WEB" within a VMware Fusion environment. The terminal displays the output of the command `ls -l /var/www/html/`. The output lists various files and directories with their permissions, owners, sizes, and timestamps. The files include `careers.php`, `composer.json`, `composer.lock`, `contactme.php`, `css`, `design.html`, `images`, `index.html`, `js`, `lib`, `LICENSE`, `mailing`, `README.md`, and `vendor`. The `index.html` file is highlighted in blue in the original image.

```
drew@web:~$ ls -l /var/www/html/
total 96
-rw-r--r-- 1 root root 1257 Oct  3 16:09 careers.php
-rw-r--r-- 1 root root  274 Oct  3 16:09 composer.json
-rw-r--r-- 1 root root 3715 Oct  3 16:09 composer.lock
-rw-r--r-- 1 root root  559 Oct  3 16:09 contactme.php
drwxr-xr-x 2 root root 4096 Oct 11 01:27 css
-rw-r--r-- 1 root root 12473 Nov  1 20:39 design.html
drwxr-xr-x 2 root root 4096 Oct 11 01:19 images
-rw-r--r-- 1 root root 29160 Nov  1 20:38 index.html
drwxr-xr-x 2 root root 4096 Oct 11 01:26 js
drwxr-xr-x 10 root root 4096 Oct  3 16:09 lib
-rw-r--r-- 1 root root 1078 Oct  3 16:09 LICENSE
drwxr-xr-x 2 root root 4096 Oct  3 16:09 mailing
-rw-r--r-- 1 root root 1195 Oct  3 16:09 README.md
drwxr-xr-x 4 root root 4096 Oct  3 16:09 vendor
drew@web:~$
```

The main file is `index.html`, and its basic structure is outlined below. Based on this, we created a website with the specified layout and components.

```
<!DOCTYPE html>

<html>

  <head>

    <!-- Meta-information about the page, not displayed to users -->

    <title>Page Title</title>

  </head>

  <body>

    <!-- Content of the page, visible to users -->

    <h1>My First Heading</h1>

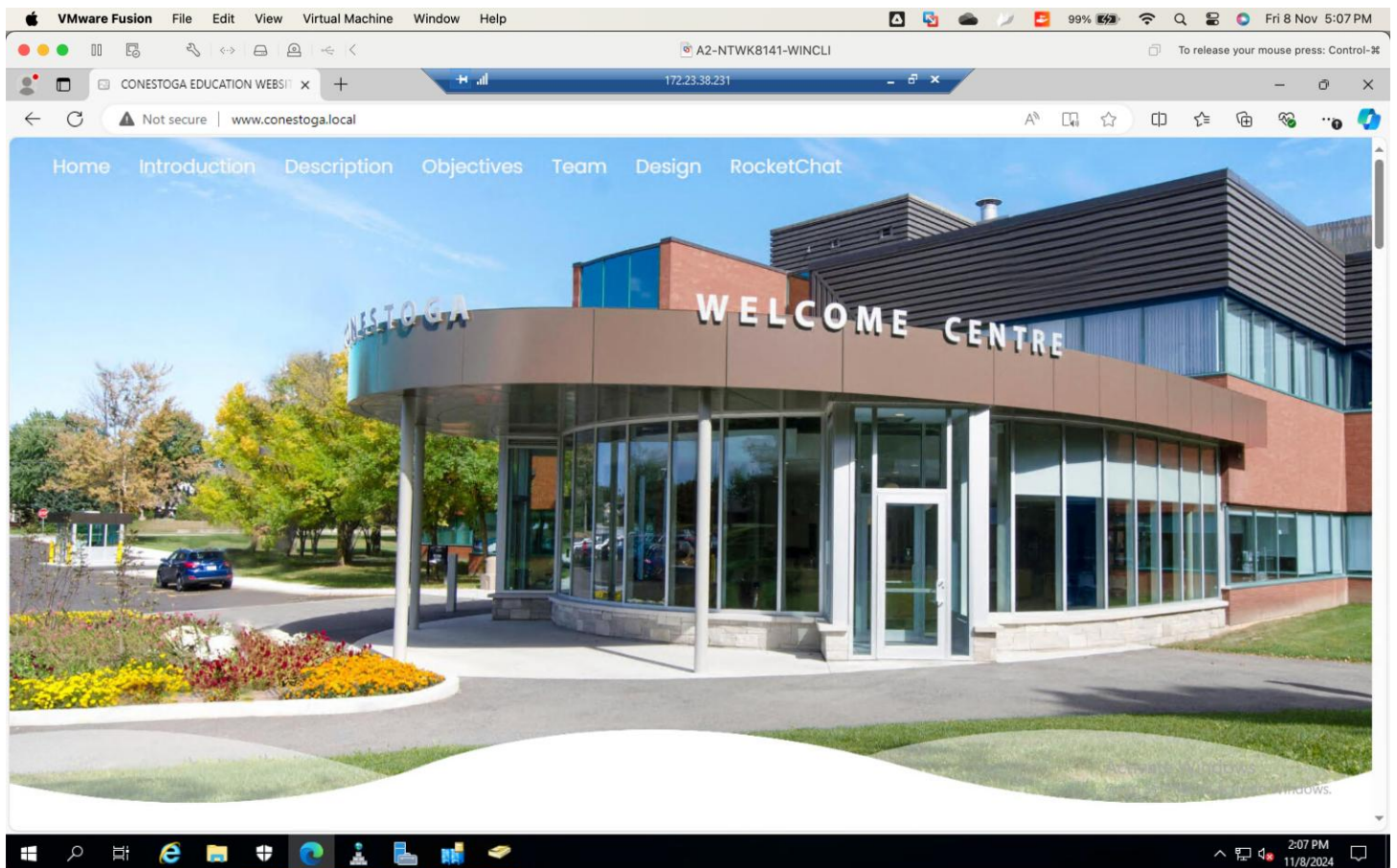
    <p>My first paragraph.</p>

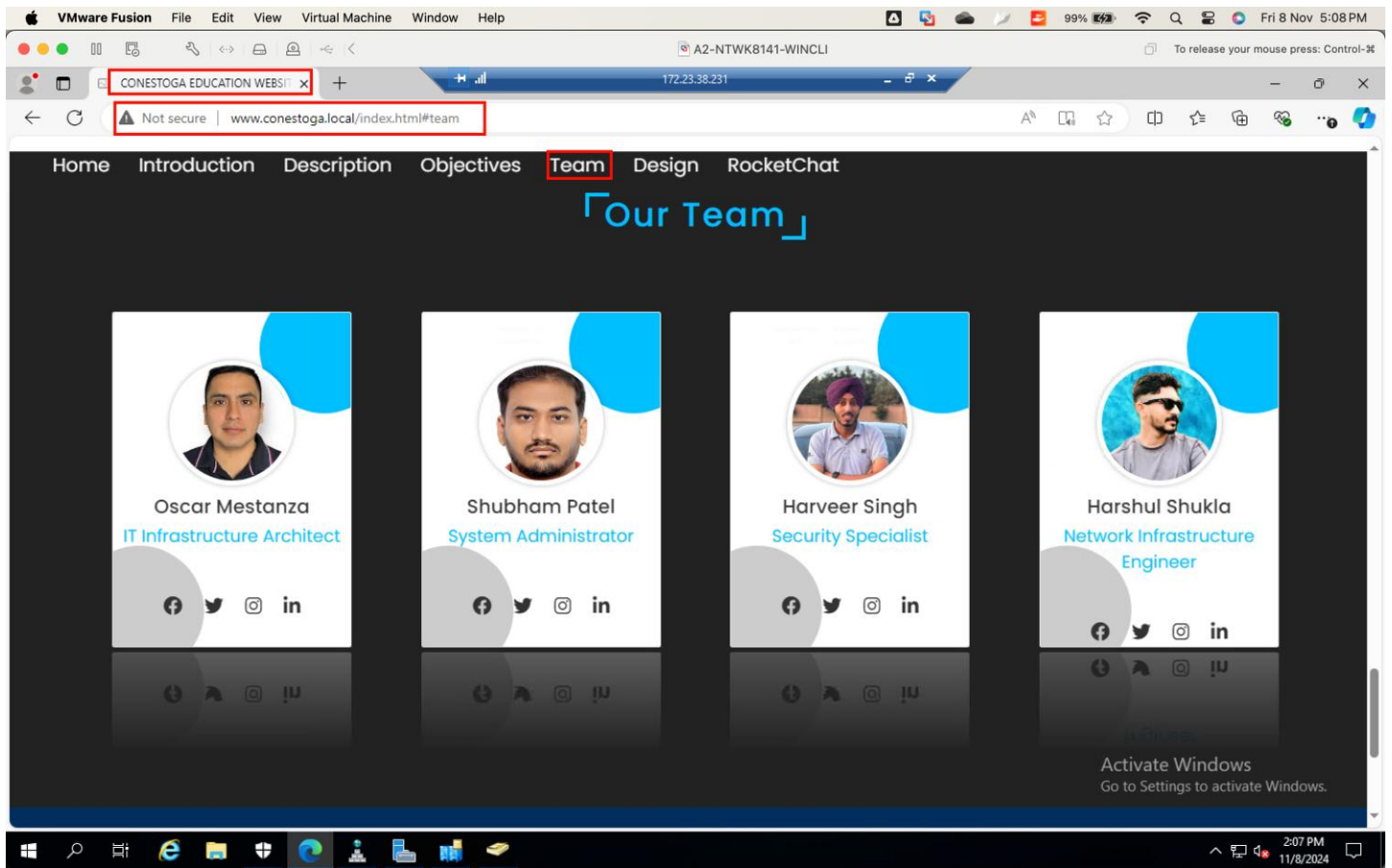
  </body>

</html>
```

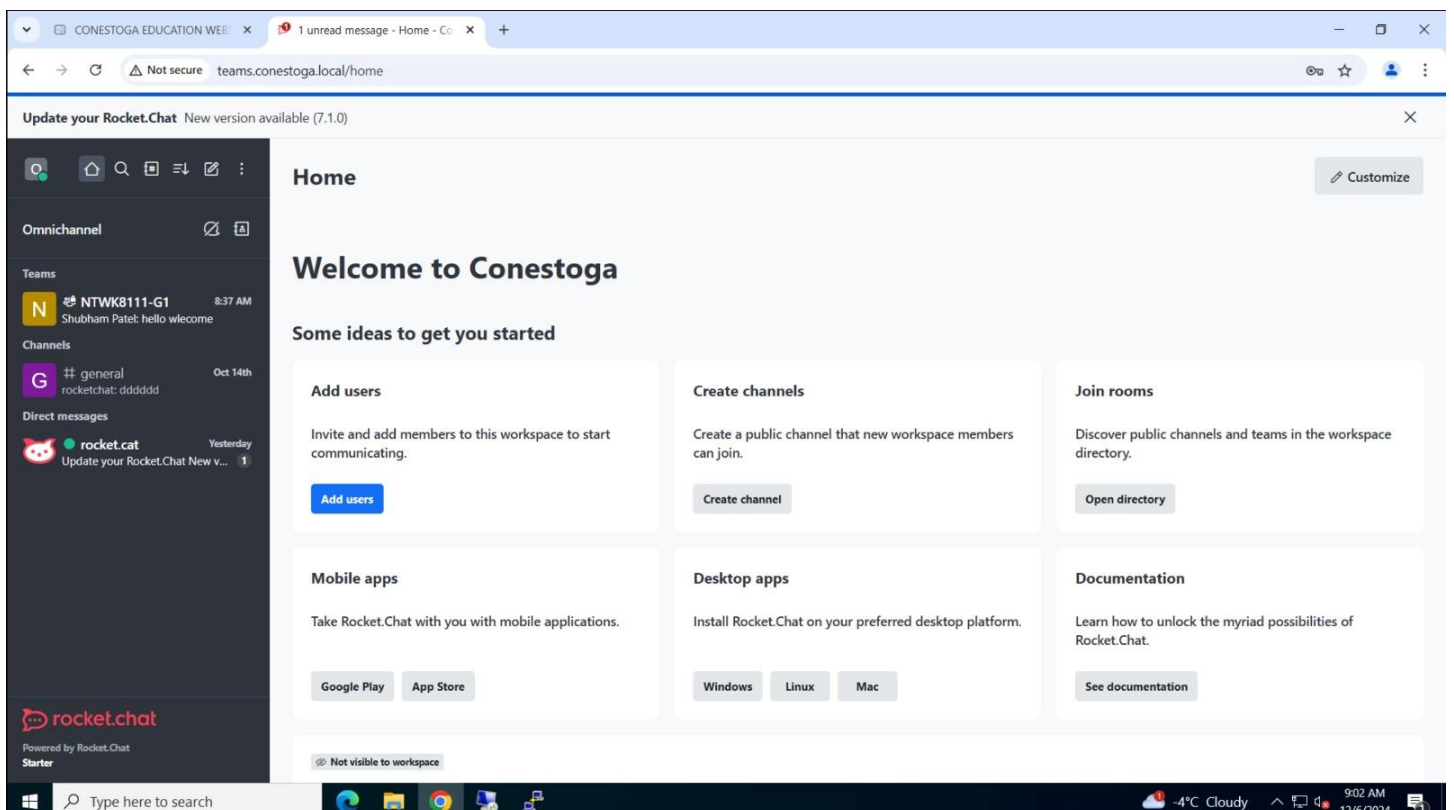
```
VMware Fusion  File  Edit  View  Virtual Machine  Window  Help
NTWK8111-WEB
/var/www/html/index.html
GNU nano 7.2
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CONESTOGA EDUCATION WEBSITE</title>
  <link rel="stylesheet" href="/css/style.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css">
  <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
  <link rel="icon" href="/images/image.png">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.1.3/css/bootstrap.min.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/ionicons/2.0.1/css/ionicons.min.css">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-TX8t27EcRE3e/1hU7zmQxVncDAy5uIKz4rE"
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" integrity="sha384-wvfXpqpZVQG66TAh5PV1GfQNH"
  <link href="/lib/bootstrap/css/bootstrap.min.css" rel="stylesheet">
  <link href="/lib/ionicons/css/ionicons.min.css" rel="stylesheet">
  <link href="/lib/owlcarousel/assets/owl.carousel.min.css" rel="stylesheet">
  <link href="/lib/lightbox/css/lightbox.min.css" rel="stylesheet">
  <script src="/js/main.js"></script>
</head>
<body>
  <!-- <h1>MENU</h1> -->
  <header class="header">
    <!-- <a href="#" class="logo"></a> -->
    <div class="fas fa-bars"></div>
    <nav class="navbar">
      <ul>
        <li><a href="#home">home</a></li>
        <li><a href="#intro">Introduction</a></li>
        <li><a href="#description">Description</a></li>
        <li><a href="#objectives">Objectives</a></li>
        <li><a href="#team">team</a></li>
        <li><a href="design.html">Design</a></li>
        <li><a href="http://teams.conestoga.local">RocketChat</a></li>
        <!-- <li><a href="#faq">FAQ</a></li> -->
      </ul>
    </nav>
  </header>
```

Web page





Rocket.Chat



CONCLUSION

This project highlights the efficient deployment of Rocket.Chat on a Linux server with MongoDB as its database and Active Directory on Windows Server for centralized user management—the integration of Rocket.Chat with Active Directory through LDAP provides the system with seamless authentication, ease of user provisioning, and increased security.

This setup has been tested with a Windows 10 client, and the results have verified that the system works just fine across platforms, making it compatible, reliable, and scalable. This deployment satisfies the needs of modern communication but also showcases how mixing open-source tools with enterprise-grade infrastructure can lead to a powerful, flexible, and user-friendly collaboration platform.

REFERENCES

Deploy with Ubuntu. (n.d.). <https://docs.rocket.chat/docs/deploy-with-ubuntu>

Documentation Group. (n.d.). *Welcome! - The Apache HTTP Server project.* <https://httpd.apache.org/>

Enable and start mongodb on rocketchat. (2021, May 3). MongoDB Developer Community Forums.

<https://www.mongodb.com/community/forums/t/enable-and-start-mongodb-on-rocketchat/105347>

How to install and configure isc-dhcp-server | Ubuntu. (n.d.). Ubuntu. <https://ubuntu.com/server/docs/how-to-install-and-configure-isc-dhcp-server>

nginx documentation. (n.d.). <https://nginx.org/en/docs/>

RocketChat. (n.d.). *Releases · RocketChat/Rocket.Chat.* GitHub.

<https://github.com/RocketChat/Rocket.Chat/releases>

Setting up a RocketChat server on Ubuntu | Ubuntu. (n.d.). Ubuntu. <https://ubuntu.com/tutorials/setup-rocketchat-server-on-ubuntu#2-installation>

Wilson, J. (2023, October 24). *How to Install RocketChat on Ubuntu 22.04.* RoseHosting.

<https://www.rosehosting.com/blog/how-to-install-rocketchat-on-ubuntu-22-04/>