Devops

@ vivek_mukvanu

① A DOCKER project :

name : web application setup / stack

Information :

↳ multi tier web application stack.

↳ local setup

↳ ↳ Automated

↳ ↳ repetable

↳ ↳ IAAC

↳ TOOLS :

① Hypervisor : vmware

② Automation : vagrant

③ cli : git Bash

④ IDE : code editor.

OBJective :

↳ VM Automation locally.

↳ Baseline for upcomming project.

↳ Real world project setup locally.
[For R & D]

Architecture designe :

① services :

→ nginx → memcached

→ Tomcat → mySQL

→ RabbitmQ

email : vivek.mukvanu.ims@gmail.com

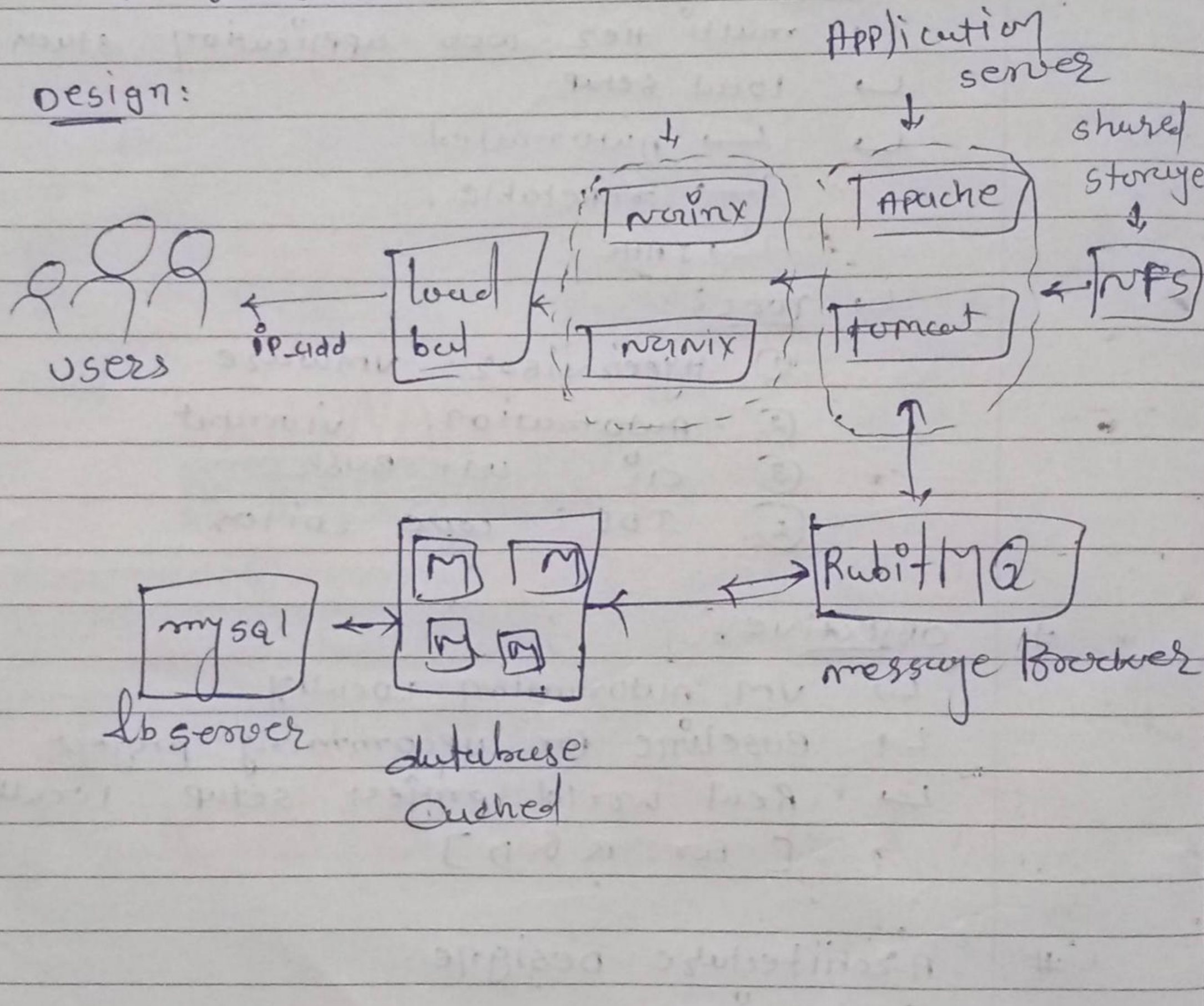
② Automation:

→ vimrunt

→ vimwure

→ git Bash

Design:



flow of execution :

1. setup tools
2. done source code
3. cd into vigrant dir
4. Bring up VM
5. validate
6. setup all services
   a. Mysql
   b. memcached
   c. Rabbit MQ
   d. Tomcat
   e. Nginx
   F. APP Build
7. verify from BROWSER :

started

Prerequisite tools :

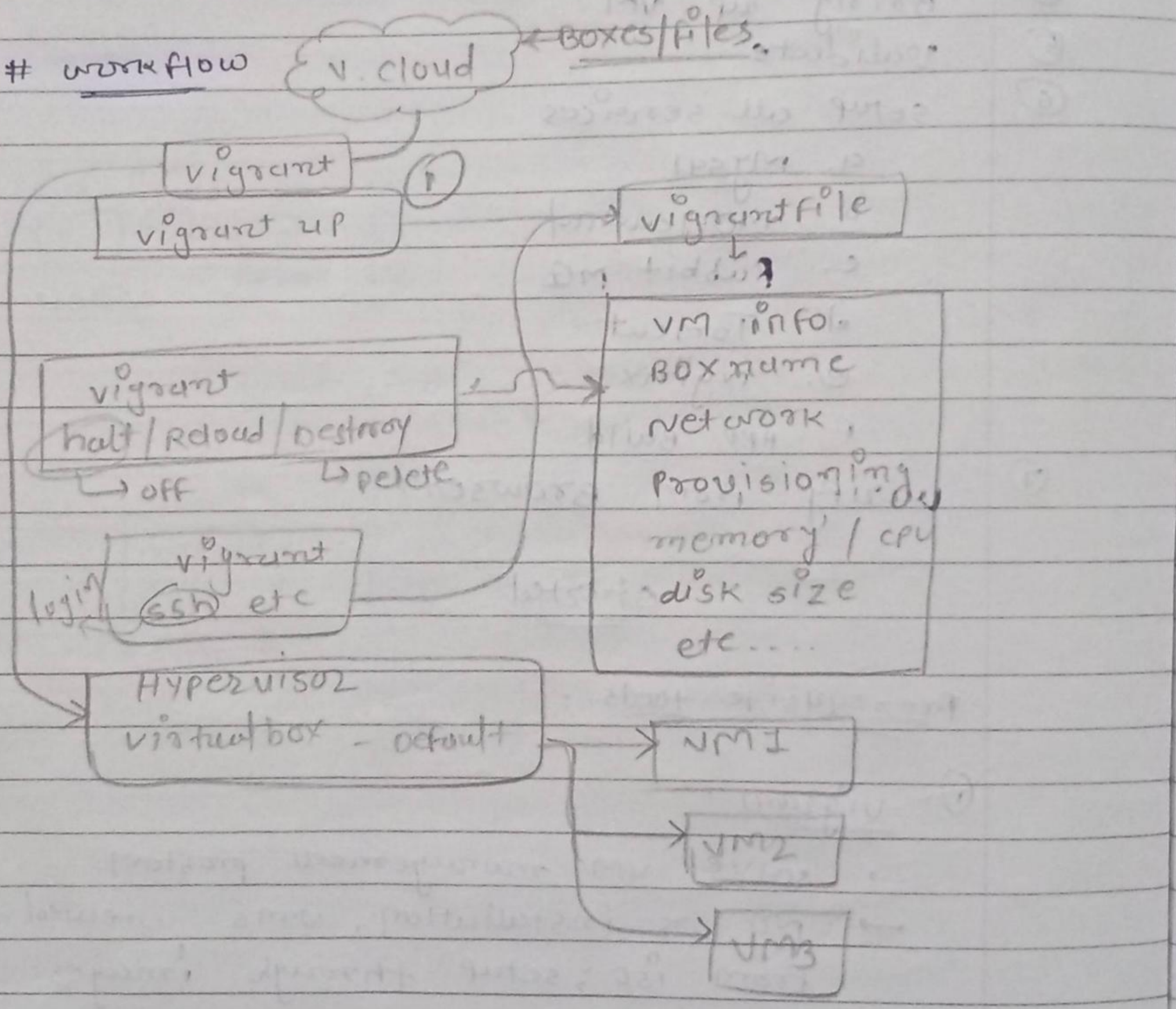
1. **vigrant :**

→ solve VM management problem
→ NO OS installation, VMs created from iso ; setup through images called 'BOXES'.
→ images / Boxes available in vigrant cloud.
→ manage VM's through file → 'vigrantfile'

→ VM changes Automatic through file
→ command to manage VM's
→ allows provisioning / executing command / script.

NOTE : works with type 2 Hypervisor.
: non-production.

workflow



⇒ install vigrant tools.
→ " create dir → vigrantfile → vigrant up → ssh "

t) After installation of VM's get ready for set up services.

① database (mariadb-db-server) : MYSQL

→ login 2 dbo1 :

vagrant ssh db01 (vagrant file location=from)
sudo -i # switching to ROOT user
yum update -y # update latest packeges
for newly provisioning os.

Database_pass = "admin123"
↳ set-up env veriable for use of
setting up db password.
↳ echo "Database_pass='admin123'" > /etc/profile

source /etc/profile
↳ now variable become permnent.

set-up Repository:
yum install epel-Release -y

yum install git mariadb-server -y
↳ installing packages.

systemctl start mariadb
" enable "
↳ start/enable service.

mysql_secure_installation
↳ command install set-up parameters.

- set current Root password [Y/n] : Y
  : admin123
  : admin123
- Remove anonymous user : yes
- disallow Root login : n
- Remove Test databases : Y
- Reload priviledges : Y
- There you go " mysql Root pass set-up"

mysql -u root -P
password : Admin123
MariaDB [(none)] > exit
} test the DB login.

git clone (-b) local-setup https://github.com/devopshydclub/vprofile-project.git
(-b) → (Specify Branch name.)

mysql -u Root -P"$Database_pass" -e "create database accounts"
accounts → name of db
} "creating our database."

mysql -u Root -P"$Database_pass" -e "grant all priviledges on accounts.* TO 'admin'@'appol' identified by 'admin123'
↳ setting-up user account for db from (appol)
(host)

mysql -u root -p"$database_pass" accounts < db-backup.sql

↳ "Runs all query from file to account database."

mysql -u root -p"$database_pass" -e "FLUSH PRIVILEDGES"

mysql -u root -p"$database_pass"

↳ verify now:

↳ accounts database should created!

↳ tables should created.

↳ exit

systemctl Restart mariadb

↳ Restart mariadb-server:

systemctl start firewalld

" enable "

firewall-cmd --zone=public --add-port=3306/tcp --permanent

firewall-cmd --reload

systemctl restart mariadb

↳ "if centos/RHEL uses firewall so allow port 3306 on it"

(2) cached service for database :

→ why use ?

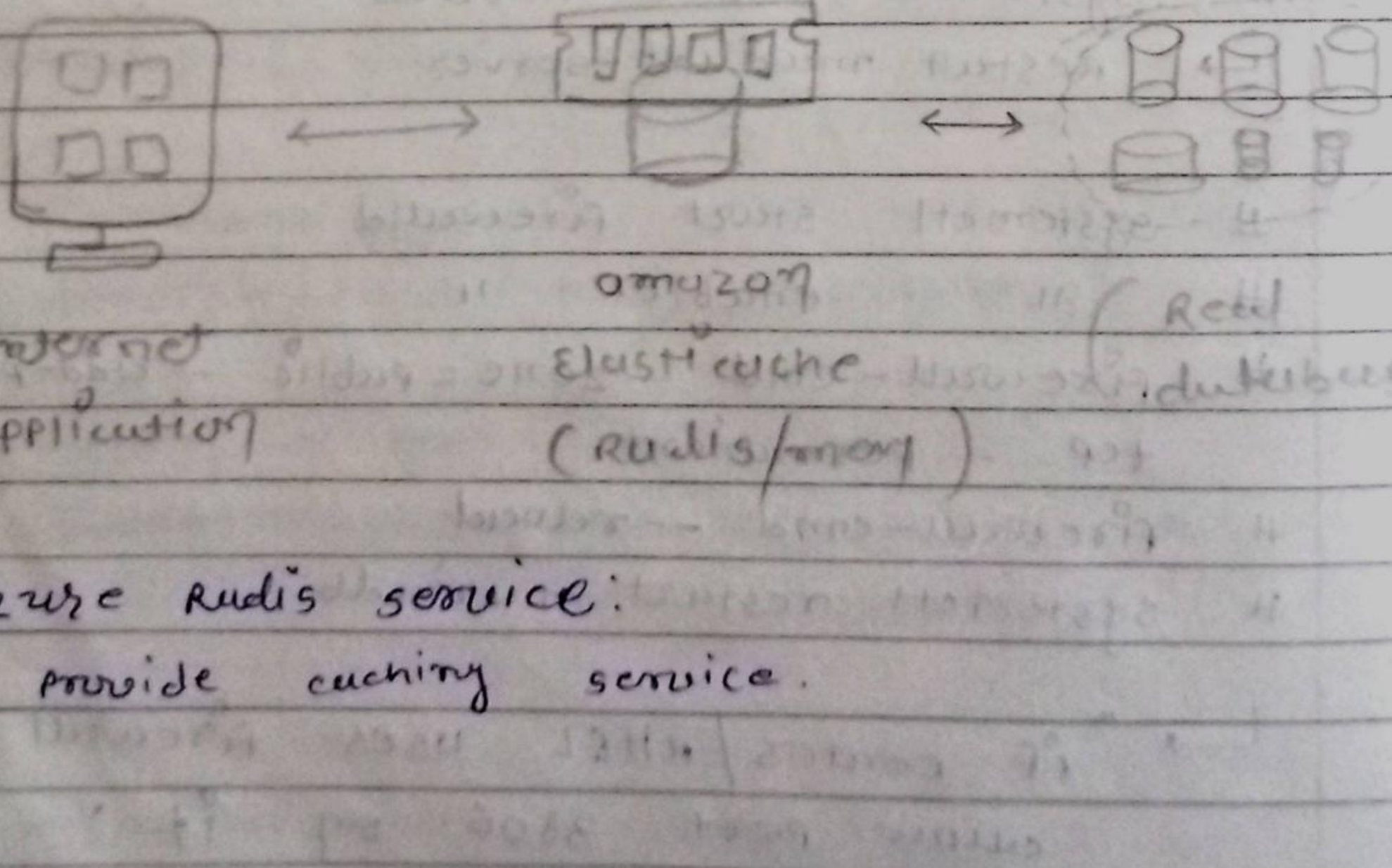
↳ cache is in-memory cache service supporting flexible, real-time use cases

↳ accelerate application and database performance.

- AWS Elasticache :

↳ compatible with Redis and memcache

↳ microsecond latency, scalable,



- Azure Redis service :

→ Provide caching service.

PAGE
DATE

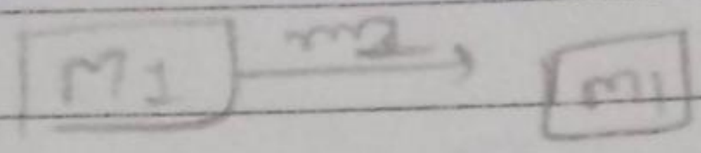
memcache setup :

```
$ vagrant ssh mc01
$ sudo -i
$ yum update -y
$ yum install epel-Release -y
$ yum install memcached -y
$ systemctl start memcached
$    "      enable   "
$ memcached -p 11211 -U 11111 -u memcached -d
                    ↓          ↓
                  tcp port   UDP port
    ↳ start listening on ports (tcp / UDP).

$ ss -tunlp | grep 11211
  ↳ verify port number.

$ Exit
```

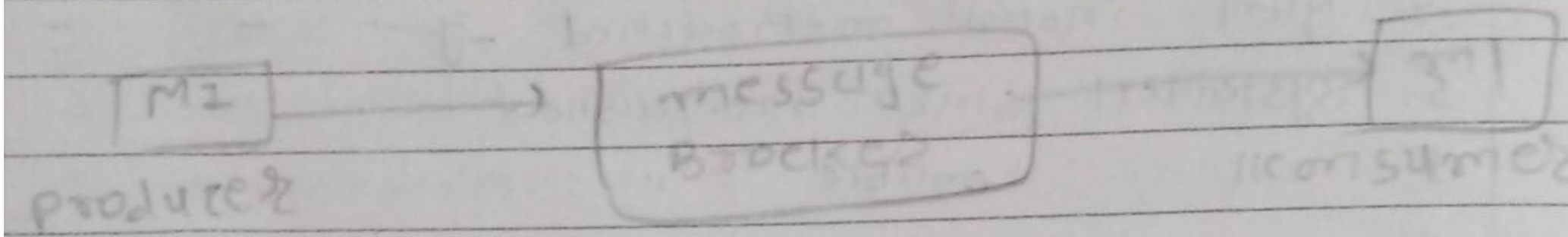
RABBIT MQ

message Broker :

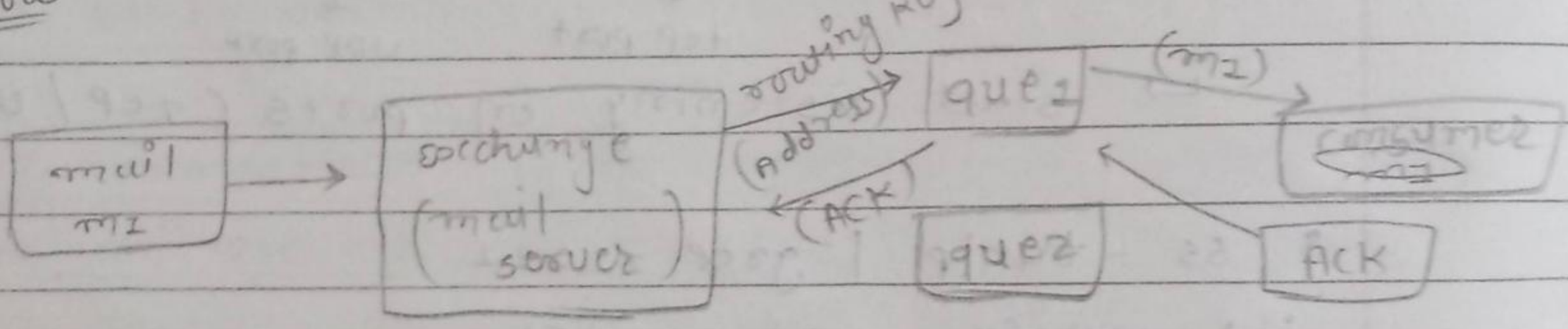
→ transport message from 1 place to another



AMQP → Advance message queing protocol. (0-1-9



flow



exchange type :

① Direct , ③ TOPIC , ⑤ Default

② fanout , ④ Header ,

(~~direct ex. but no routing key~~)

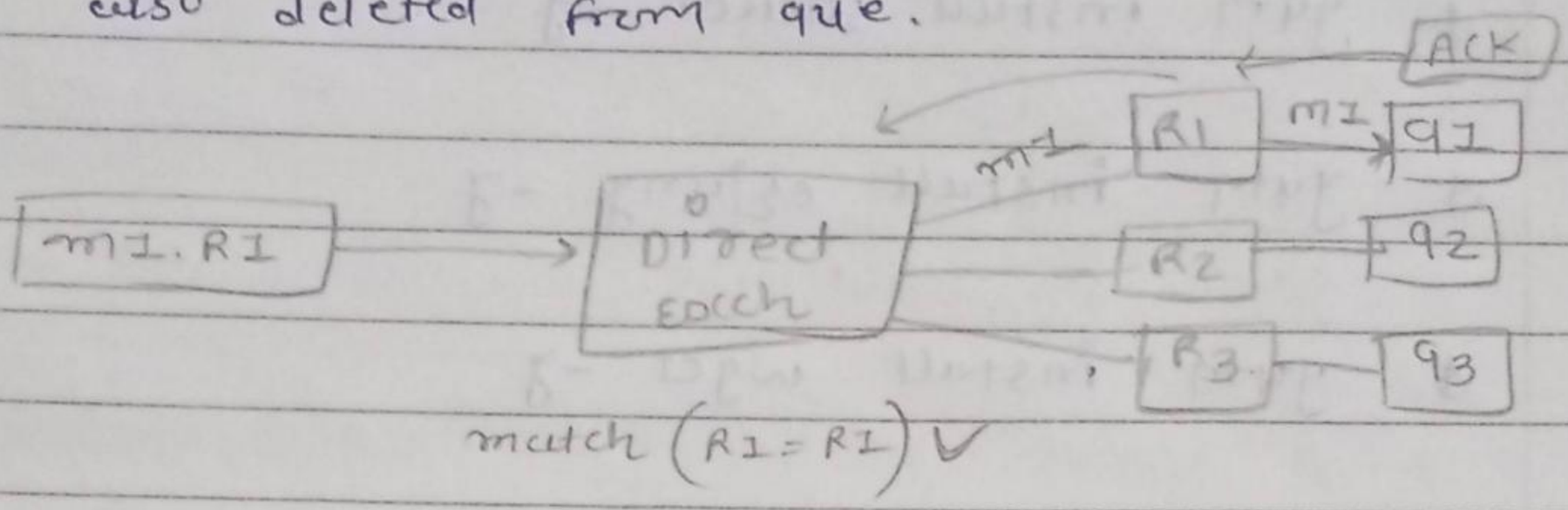
① Default exchage :

↳ when you don't bind any (que) with exchange that (que) directly bind with default exchange

↳ Routing key will same as (que) name.

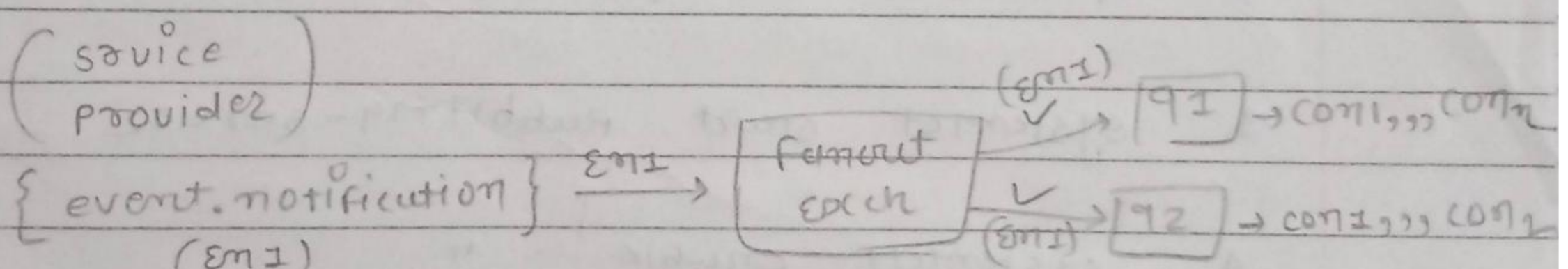
ii) Direct exchange :

→ it has Routing key associated with exchange
→ match key to forward message
→ Ack will be sent After message arrived also deleted from que.



iii) fanout Exchange :

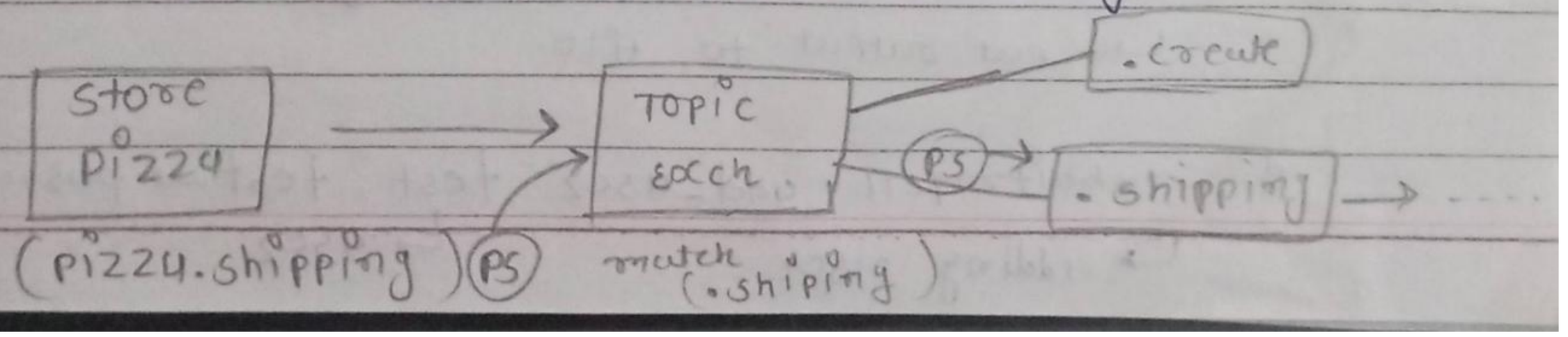
→ no Routing key OR empty key
→ message forwarded to all que which Bound with exchange server.



(iv) TOPIC Exchange :

→ match pattern from Binding key.

Exc: 1 *.india , pizza.shipping
Exc: 2 india.* , pizza.inventory



commands:

$ yum install epel-release -y
↳ Before install update package manager
$ yum update -y

$ yum install socket -y

$ yum install erlang -y

$ yum install wget -y

$ wget url-Rabbitmq-3.6.10-1.el7.noarch.RPM.
↳ download RPM package
"https://www.rabbitmq.com/releases/rabbitmq-server/v3.6.10

$ rpm --import url-of-rpm-key.

$ rpm -ivh package_name_of.RPM-

$ systemct start Rubbitmq-server

$ systemctl enable "

$ " status "

$ echo "[{rubbit, [{loopback_users, []}]}]." >
/etc/rubbitmq/rubbitmq.config
↳ put output to file.

$ rabbitmqctl add_user test test → password.
↳ adding user ↳ user

$ rabbitmqctl set_user_tags test administrator

↳ set superuser privileges.

test → user
administrator → role

$ systemctl restart rabbitmq-server

Application setup :- (Note: refer pdf file)

$ login to appo1 server :

$ yum install java-1.8.0-openjdk -y
↳ prerequisite for tomcat server.

$ yum install git maven wget -y

$ cd /tmp/ # changing dir to /opt/tomcat
$ wget https://archive.apache.org/dist/tomcat/tomcat-8/v8.5.37/bin/apache-tomcat-8.5.37.tar.gz
-O tomcatbin.tar.gz
↓ "version of tomcat"
↓ "Package name on system"

$ tar xzvf tomcatbin.tar.gz

$ useradd --home-dir /usr/local/tomcat8 --shell /sbin/nologin tomcat
↓ ↓ ↓
(user can't login) username Homedir

$ rsync -avzh /tmp/apache-tomcat-8.5.37/ /usr/local/tomcat8
↳ synchronize both directory.

$ chown (-R) tomcat.tomcat /usr/local/tomcat8

-R → Recursively.
tomcat → user owner
tomcat → group owner
/usr/local/tomcat8 → homedir

$ create service file :
$ vi /etc/systemd/system/tomcat.service

[Unit]
Description = tomcat
After = network.target

[Service]
User = tomcat
WorkingDirectory = /usr/local/tomcat8
Environment = JRE_HOME=/usr/lib/jvm/jre
" JAVA_HOME = "
" = CATALINA_HOME = /usr/local/tomcat8
" = CATALINA_BASE = "
ExecStart = /usr/local/tomcat8/bin/catalina.sh
ExecStop = /usr/local/tomcat8/shutdown.sh

[Install]
WantedBY = multi-user.target

$ systemctl daemon-Reload
$ " start tomcat
$ " enable "
$ " status "

Allow into firewall :

* systemctl start firewalld
* systemctl enable firewalld
* firewall-cmd --zone=Public --add-port=8080/tcp --permanent
* firewall-cmd --Reload.

5 Doploy Application :-

git clone https://github.com/vivekmalkvaal/DevOps-complete-web-application-deployment-from-scratch.git

cd DevOps-Tab..

vim src/main/resources/application.properties

↳ put Backend server detailes.

↳ open Readme.md.

Build code:

$ mvn → Build the scrach code

$ mvn install

→ Deploy artifact:

systemctl stop tomcat ; sleep 100

rm -rf /usr/local/tomcat8/webapp/ROOT*

copy artifact → /usr/local/tomcat/webapp/ (Here)

systemctl start tomcat ; sleep 300

chown tomcat:tomcat /usr/local/tomcat8/webapp -R

systemctl Restart tomcat

(6) NGINX setup :

login into nginx machine.

cat /etc/host → verify all hosts ip

apt update && apt upgrade

apt install nginx -y

vi /etc/nginx/sites-available/APPlication1

↳ ~~server~~

upstream application1 {

server.APP01:8080;

}

server {

listen 80;

location / {

proxy-pass http://application1;

}

}

→ Remove default nginx configuration :

rm -rf /etc/nginx/sites-enable/default

→ create link to Activate website :

ls -s /etc/nginx/sites-available/application1

/etc/nginx/sites-enabled/application1

systemctl restart nginx

→ copy IP address and puste on browser,

→ start using APP :) OR

Trouble shoot !! :(