Requirements should configured in descending **order starting with **Mariadb**

To set Maria DB - Requirements:

- Use any Linux distribution you are familiar with (Centos 7 is recommended)
- As general rule, always run OS updates check and apply if any
- Depending on distribution you are using, it is recommended to install additional package manager enhancement to simplify installation of required packages (Epel in instance of CentOS7)
- Start and enable MariaDB server
- Run secure installation client script: https://mariadb.com/kb/en/mysql_secure_installation/
    - It will ask for **root** password at the start, if this is fresh installation you won't have any. Press enter as instructed and create one (make sure you remember it)
    - Next will come a sequence of options that with **Yes** or **No** questions, with explanations. **Everything but** 'Disallow Remote Root Login' should be answered with **Y (**yes).
        - This is a lab so strict security isn't a priority here. Also, you want to allow all hosts created for this project to be able to 'remotely' access DB as root.
- Next is setting up **DB name** and **users:**
    - You have **root** user created, so use that to login to server
        - Tip: The syntax is: mysql - u <username> -p<password> (i.e -padmin)
    - You don't have any **Database,** yet (because in installation step you removed **sample,** if you followed this instructions to the letter), now you will create one:
        - For purpose of this project, name it **accounts (**Tip: use queries to create Db and mariadb/mysql doc as reference)
    - Next step is to create an **administrator account** (admin) and grant it all privileges(this early in project it's important to note that **root** should be used only as global enforcer - to control config and user-management)
        - **All Privileges Granted** are for Db created in previous step (accounts)
        - Because this may be tricky first time around, the query looks like following: grant all privileges on accounts.* TO 'admin'@'%' identified by 'admin' ;
            - To understand it refer to https://chartio.com/resources/tutorials/how-to-grant-all-privileges-on-a-database-in-mysql/
            - Event though it's covered in above article, instead of using **@locahost,** you use **@'%'** because you want other hosts in virtual network to access Db 'remotely' and not just machine on which Db server is.
        - The next steps are to **FLUSH** privileges (do this every time you apply some changes to privileges)
        - Terminate the current connection.

- **Following** step is feeding newly create Db (accounts) with schema content,
    - To do this clone following git repo: https://github.com/devopshydclub/vprofile-project.git
    - Navigate to the newly created local repo (downloaded folder) **vprofile-project**
    - From here feed empty **accounts** db with src/main/resources/db_backup.sql
        - Tip: mysql -u <account> -p<password> <db name> < src/main/resources/db_backup.sql
        - Use **root** as account
    - If successful, login to **db** (instead of **root,** you can take this opportunity to test **admin)** and run short query to check **tables** (Tip: refer to docs)
    - Close **db** connection and restart **MariaDb** server
- By default **port** 3306 is used by **MySQL** and in the next steps a firewall needs to be configured to allow communication:
    - It doesn't matter which one you will use, be it **UWF, firewalld,iptables**
    - **Although,** firewalld is **recommended due to how easy is to manage zone-management -** spoiler alert, it **automatic :D**
    - Once you accomplish the task with a service/tool of your choice and verify that port 3306 is whitelisted and open, **restart** MariaDB server.

In case you get stuck at any point there is step-by-step doc, that gives a quick rundown of commands to type (**will post it later, or just ping me**).The reason I didn't do simple copy/paste here is because you can do that from any tutorial you get your hands on and still don't understand 10% of the things you did.

The point is to acquire skill and knowledge that you can transfer to other tech stack, instead of always having to google quickest solutions.

To set MemoryCache:

- Use any Linux distribution you are familiar with (Centos 7 is recommended)
- As general rule, always run OS updates check and apply if any
- Depending on distribution you are using, it is recommended to install additional package manager enhancement to simplify installation of required packages (Epel in instance of CentOS7)
- By default **port** 11211 TCP (and as backup port UDP 11111) is used by **Memorycache** and in the next steps a firewall needs to be configured to allow communication:
    - It doesn't matter which one you will use, be it **UWF, firewalld,iptables**
    - **Although,** firewalld is **recommended due to how easy is to manage zone-management -** spoiler alert, it **automatic :D**
    - Once you accomplish the task with a service/tool of your choice and verify that port 11211/tcp, 11111/udp is whitelisted and open
    - Memorycache is different from MariaDb in sense that you will need to assign ports to it through shell
    - Command: memcached -p 11211 -U 11111 -u memcached -d
- I am not going to get into the skeleton of the above is, you can use **man,** or tools like Explainshell to understand it, if not already familiar.
- **Don't** forget to **start and enable** Memcache (tip: use systemctl)

In case you get stuck at any point there is step-by-step doc, that gives a quick rundown of commands to type (**will post it later, or just ping me**).The reason I didn't do simple copy/paste here is because you can do that from any tutorial you get your hands on and still don't understand 10% of the things you did.

The point is to acquire skill and knowledge that you can transfer to other tech stack, instead of always having to google quickest solutions.

To set Rabbitmq - note that installation may vary a lot from linux distro to distro, I personally used CentOS7

- Use any Linux distribution you are familiar with (Centos 7 is recommended)
- As general rule, always run OS updates check and apply if any
- Depending on distribution you are using, it is recommended to install additional package manager enhancement to simplify installation of required packages (Epel in instance of CentOS7)
- **To complete installation part I** you need following - for convenience and established practice, navigate to '/tmp' directory:
    - wget
    - With 'wget' download http://packages.erlang-solutions.com/erlang-solutions-2.0-1.noarch.rpm
    - Upgrade software package dependencies with package manager ('RPM) with downloaded package - > rpm -Uvh <package> (***you must specify where package is, or be in the same directory as the package)**
    - Install 'erlang socat' (quick tip, **use primary packet manager to complete the step:** that can be yum,dnf or apt, depending on linux distro)
- **To complete installation part II**
    - You will use 'curl' to hit **rabbimq** package dependencies repo and in parallel execute the script with bash shell: https://packagecloud.io/install/repositories/rabbitmq/rabbitmq-server/script.rpm.sh | bash
    - Final step, install 'rabbitmq-server' ((quick tip, **use primary packet manager to complete the step:** that can be yum,dnf or apt, depending on linux distro)
- **Enable** and **Start** 'rabbitmq-server' > check the **status (it should be running)**
- Enabling default guest access for remote hosts: Authentication, Authorisation, Access Control — RabbitMQ (if you ask me, you can skip this part, and if you check doc you will see **that they recommend to remove guest account or change it's default password):**
    - By default as you may expected only machine (vm or otherwise) that's running Rabbitmq is allowed to access **default virtual host** by using guest account
    - To allow this remotely:
        - Choose any method you want:
            - Either:
                - If there isn't manually create RabbitMQ config file named **rabbitmq.conf**
                - And inside in sysctl format add **loopback_users**
                    - Chose whatever you want to use:
                        - New format (sysctl -> key = value)
                        - Classic (Erlang terms)
                    - Read the linked doc for more information,although I would recommend getting **familiar with** both, because the new

format (sysctl/ini) doesn't support many options.
    - Or go ahead and automate it with shell scripting (it's fairly simple script)
        - Use **echo** to output the desired string (**option)** and at the same time, store that output inside **rabbitmq.conf (**tip: sh -c 'Echo [string] > /etc/rabbitmq/rabbitmq.conf']
- **Word of warning,** this is just for lab purposes. As mentioned above and in docs, **guest** accounts are a security risk and therefore are best to be removed in prod environment.

- Rabbitmq CLI tool **rabbitmqctl** and communication between inter-nodes is handled over tcp port **25672**
    - It doesn't matter which one you will use, be it **UWF, firewalld,iptables**
    - **Although,** firewalld is **recommended due to how easy is to manage zone-management -** spoiler alert, it **automatic :D**
    - Once you accomplish the task with a service/tool of your choice and verify that port 25672 is whitelisted and open
    - Restart Rabbitmql-server
- One last extra bit, **you can skip** but it's always better to identify problem early than later. As mentioned rabbitmq has CLI tool with which it's shipped, in this step you will use it to test **guest** and **test** account:
    - rabbitmqctl authenticate_user test test
      (should return)
        - Authenticating user "test" ...
        - Success
    - rabbitmqctl authenticate_user guest guest
    - **(should return)**
        - Authenticating user "guest" ...
        - Success
- To understand the CLI tool and it's command use **--help** option.
- If there was a problem, with either of accounts use CLI built in commands of the tool to resolve it - and it's a good way to get used to and familiar with the tool.

In case you get stuck at any point there is step-by-step doc, that gives a quick rundown of commands to type (**will post it later, or just ping me**).The reason I didn't do simple copy/paste here is because you can do that from any tutorial you get your hands on and still don't understand 10% of the things you did.

The point is to acquire skill and knowledge that you can transfer to other tech stack, instead of always having to google quickest solutions.

To set Tomcat - note that installation may vary a lot from linux distro to distro, I personally used CentOS7

- A ton of benefits why setting Tomcat to run with systemd instead of init :Running tomcat with systemd – Hackeriet (***both 'init' and 'systemd' are daemons, but** systemd allows you to manage linux services, where as 'init' **only starts them** and you don't have fine-control over them **) -> if you require proper guidance use this https://www.digitalocean.com/community/tutorials/how-to-install-apache-tomcat-8-on-centos-7 (only Installation steps)**
- After installing you will likely need to open and allow connection over port **8080** choose, repeat with firewall of your choice, what you did earlier for previous components

You don't need to go further than this, configuration of tomcat-apache app container is not the goal of this exercise = only initial setup and startup.

Then next phase is to Build and Deploy artifact:

- Clone branch **java** (git clone -b java https://github.com/steefanSS/DevMentorship.git)
- Navigate to root of the project '**vprofile-project'**
- open src/main/resources/**application.properties (**edit file if you add any custom changes to backend server details - Mysql, Memcache, Rabbimq -ignore elastic search at this phase of the project)
- Build the project (run maven install)
- Deploy artifact:
    - Stop tomcat remove default artifact
    - Copy built artificat from root folder to destination where deleted artificat was (and make sure the name is **ROOT.war)**: cp target/vprofile-v2.war /usr/local/tomcat8/webapps/ROOT.war
    - Start tomcat
    - Make sure to change owner of directory where artifact is to tomcat (chown tomcat.tomcat usr/local/tomcat8/webapps -R)
    - Restart tomact

To set NGINX- note that installation may vary a lot from linux distro to distro, I personally used CentOS7

You can set NGINX the following way (I have speed through this part, because by now you should be able to understand everything going on:

Login to the Nginx vm
$ vagrant ssh web01

Verify Hosts entry, if entries missing update the it with IP and hostnames
cat /etc/hosts

Update OS with latest patches
apt update
apt upgrade
Install nginx
apt install nginx -y

Create Nginx conf file with below content
# vi /etc/nginx/sites-available/vproapp
upstream vproapp {
server app01:8080;
}
server {
listen 80;
location / {
proxy_pass http://vproapp;
}
}

Remove default nginx conf
rm -rf /etc/nginx/sites-enabled/default

Create link to activate website
ln -s /etc/nginx/sites-available/vproapp /etc/nginx/sites-enabled/vproapp

Restart Nginx
systemctl restart nginx

***Expect perhaps simple NGINX configuration, that could be new to you (**I can cover that separately in some more exciting project if you would like or you can research yourself in inpatient :D)