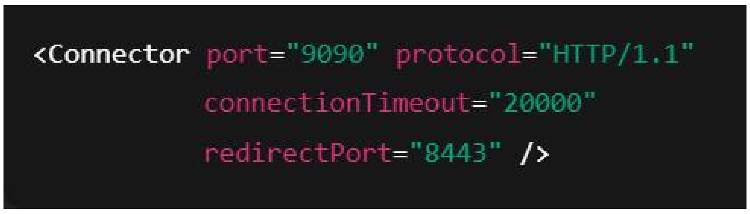
- To install Jenkins in Ubuntu:
  - You need to install Java because *Jenkins is written in Java.*
  - Its not a native program (like .exe or .bin), rather it's a **.war** file (Java Web Application Archive).
- server.xml
  - whatever port you give in the <Connector … >, will server your website.
  - <ip>:<port>
  - you can give protocol inside the <Connector …>
    - HTTP/1.1 : simplest, default, good for small apps.
    - NIO : better for many concurrent connections.
    - AJP : for reverse proxy setups (Apache/Nginx → Tomcat).
    -
    - **HTTP/1.1** : Default HTTP connector using the blocking I/O (BIO) or NIO implementation. Handles normal HTTP requests.
    - **org.apache.coyote.http11.Http11NioProtocol** : Non-blocking I/O (NIO) HTTP connector. Better for handling many simultaneous connections efficiently.
    - **org.apache.coyote.http11.Http11Nio2Protoco** : Uses Java NIO2 (asynchronous I/O). Advanced, can scale better for high-load servers.
    - **org.apache.coyote.http11.Http11AprProtocol** : Uses APR/native libraries for maximum performance. Requires Tomcat Native library installed.
    - **AJP/1.3** : Connects Tomcat to a web server like Apache HTTPD using AJP protocol (common in production).
  - Sometimes, you might see **redirectPort** inside Connector:



```
<Connector port="9090" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

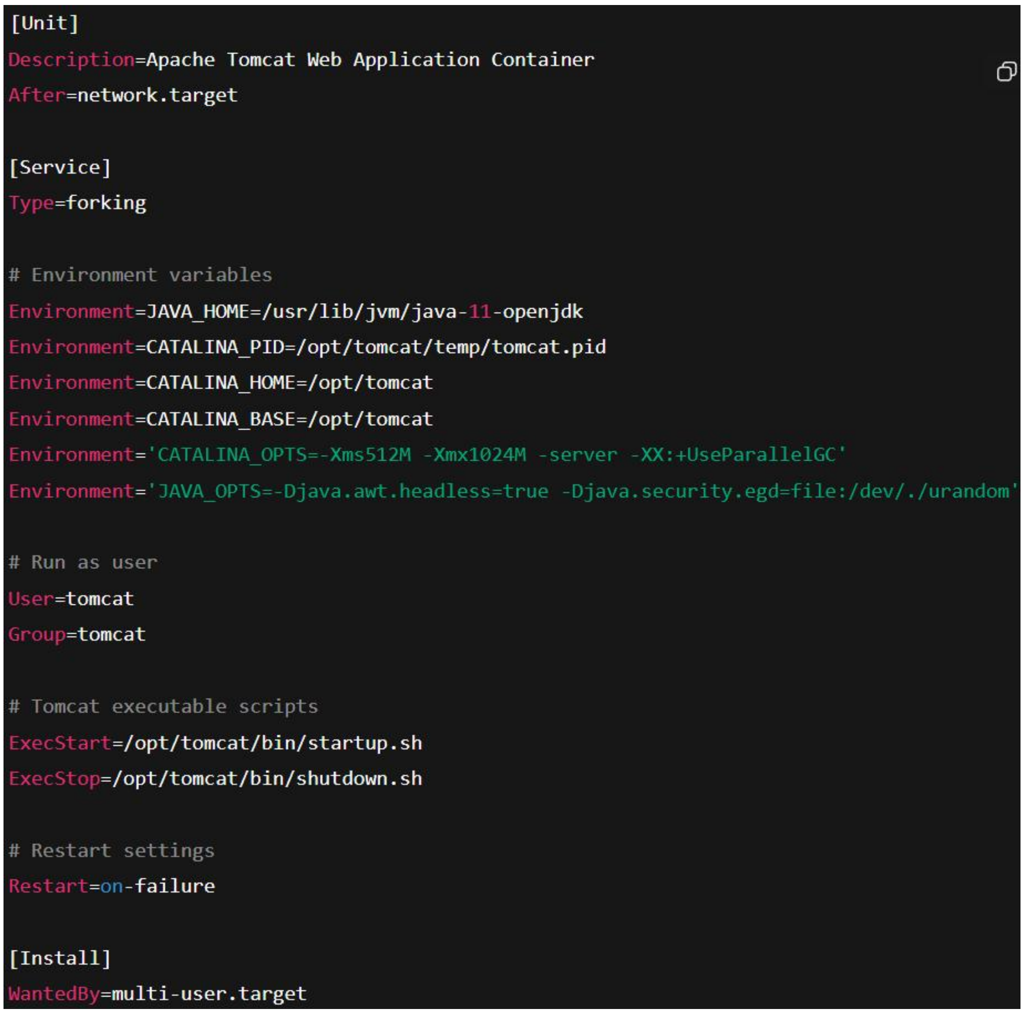
    -
    - **redirectPort="8443"** → If a request comes in on 9090 that requires HTTPS, Tomcat will automatically redirect it to port 8443, where you would typically have an HTTPS connector configured.

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
           maxThreads="200"
           scheme="https" secure="true" SSLEnabled="true"
           keystoreFile="conf/keystore.jks" keystorePass="changeit"
           clientAuth="false" sslProtocol="TLS"/>
```

- So in short: redirectPort points to the HTTPS port that handles secure traffic when needed.

- Process to host a single web app in apache tomcat:
  - Go to **/tmp/** directory (optional) and download the **tar.gz** file of tomcat of whatever version you want.
  - Extract that file using **tar -xzvf <tar file name>** command.
  - Now create a folder which will serve the web site (usually we take inside **/opt/** directory)
    - I created **/opt/tomcat/**
  - Copy all the files inside the extracted file into this path i.e. to **/opt/tomcat/**
  - Now create one user for tomcat (it is not necessary; even you can run Tomcat as **root** or **any other** user; but for security point of view, it's a good practice to create a dedicated user for running **tomcat**)
    - I created one **tomcat**
    - **useradd -r -m -U -d /opt/tomcat -s /bin/false tomcat** **(in RPM based linux)**
    - Here **/opt/tomcat** is the home directory of the user **tomcat**.
      - It is not required. For consistency like every user should have a home directory, it is created. (but completely optional)
  - Now, make the user **tomcat** owner of **/opt/tomcat/** directory along with all the *sub-folders* and *files* inside it.
    - **chown -R tomcat:tomcat /opt/tomcat**
  - There are **shell** scripts to run or shutdown tomcat inside the **bin** folder i.e. **/opt/tomcat/bin**
    - You need to give executable permission to all those files.
    - **chmod +x /opt/tomcat/bin/*.sh**
  - Now, delete everything present inside the directory **/opt/tomcat/webapps/** and copy the **.war** (build file) file into this and rename that as **ROOT.war**.
  - You can run the file **startup.sh** to run tomcat, or better create a **tomcat.service** file for this.
    - Create the **tomcat.service** file inside **/etc/systemd/system/** directory.
    - You can see in the file, there is a property called **User** and **Group**.
      - If you have created the user, give that against that User and Group field.
    - If you are creating multiple **tomcat** to host multiple websites independently, then also the Environment variable **names** (not the values) mentioned inside the **tomcat.service** file will not be changed.
      - This variable will be limited to this service only

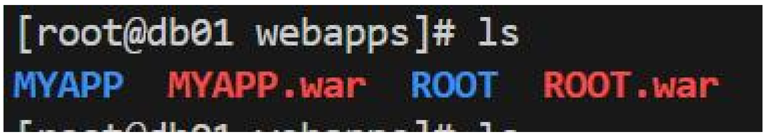
    - Even if any other service is using same variable, that'll not interfere with this.
    - Means, these are local to the service.
    - Just there values needs to be changed like directories and all.
    - Also, select the proper **openjdk** version.



```
[Unit]
Description=Apache Tomcat Web Application Container
After=network.target

[Service]
Type=forking

# Environment variables
Environment=JAVA_HOME=/usr/lib/jvm/java-11-openjdk
Environment=CATALINA_PID=/opt/tomcat/temp/tomcat.pid
Environment=CATALINA_HOME=/opt/tomcat
Environment=CATALINA_BASE=/opt/tomcat
Environment='CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC'
Environment='JAVA_OPTS=-Djava.awt.headless=true -Djava.security.egd=file:/dev/./urandom'

# Run as user
User=tomcat
Group=tomcat

# Tomcat executable scripts
ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/opt/tomcat/bin/shutdown.sh

# Restart settings
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

- Now,
  - **systemctl daemon reload**
  - **systemctl start tomcat**
- After strting **tomcat**, it'll extract that **ROOT.war** and one folder called **ROOT** will be created there.
- Now, the website is hosted. You can access it with the port (mentioned inside the **server.xml**) file. (default: 8080)

- Multiple tomcats to host multiple web-apps independently:
    - In my case, I created the following folders
        - **/opt/tomcat_v1**
        - **/opt/tomcat_v2**
    - Then follow the steps as previous, copy the respective web-apps build file (**.war**) into the respective folder's **webapps/** directories.
    - Now, you need to configure the ports to which the web-apps will listen:
        - **/opt/tomcat_v{*}/conf/server.xml** , The following ports should be unique for all the **tomcat** files:
            - <Server **port**="8005" shutdown="SHUTDOWN"> (**port** (shutdown port))
            - <Connector **port**="8080" protocol="HTTP/1.1" connectionTimeout="20000" **redirectPort**="8443" maxParameterCount="1000" /> (**port** and **redirectPort**)
    - Also, inside the **tomcat_v{*}.service** files will contain the proper **path values.**
    - One user can be used for all the **tomcat instances**, but better to **create different user for different instances.**

- The above case was for hosting different websites in different **tomcat** instances independently.
  - But if you want to host all web-apps in same **tomcat** instance i.e. all the web-apps will be listening to **same port**.
  - Just copy the **.war** files of all the web-apps and paste those inside **webapps/** folder.
  - Then in browser **http://<ip>:<port>/<filename>** , you can access the website.
    - **<filename>.war** will decide the route of the webapps.
  - 

```
[root@db01 webapps]# ls
MYAPP  MYAPP.war  ROOT  ROOT.war
```

    - In this case,
      - http://<IP>:<Port>/ : it'll serve the web-app having name as ROOT.
      - http://<IP>:<Port>/MYAPP : It'll serve the web-app having name as MYAPP.

- **ss -tulnp**
  - ss → socket statistics tool (modern replacement for netstat).
  - Options:
    - -t → show TCP sockets.
    - -u → show UDP sockets.
    - -l → show only listening sockets (services waiting for connections).
    - -n → show ports as numbers (skip DNS/service name resolution).
    - -p → show process using the socket (requires root).