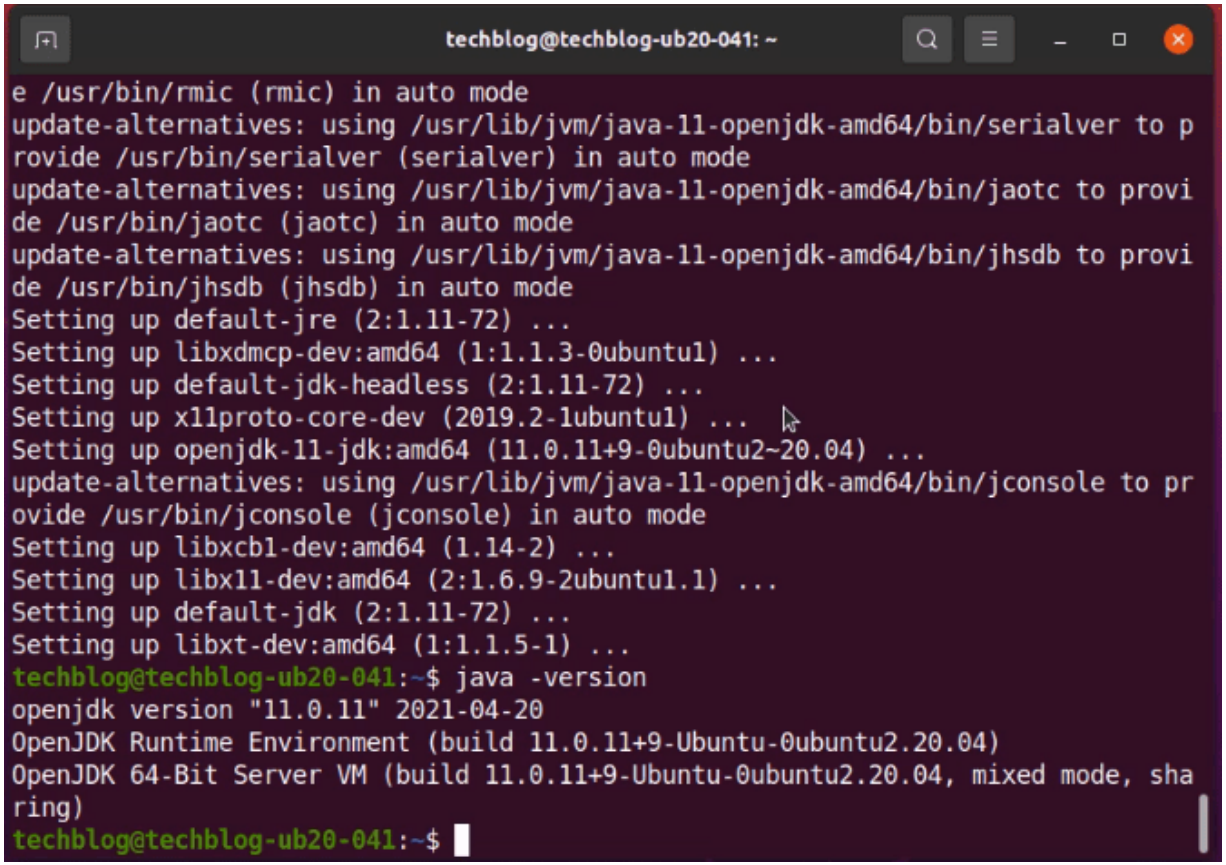
Step 1 – Install Java

You must have JRE (Java runtime environment) installed on your system. Tomcat 10 required to have JRE 8 or higher version installed on your system. Use the following command to install OpenJDK to fulfill the requirements.

```
sudo apt update

sudo apt install default-jdk
```



Check the current active Java version:

```
java -version
```

Step 2 – Create Tomcat User

We recommended to run Tomcat server with a dedicated user account. Create a new user, which is recommended for security purposes mainly for production deployments.
To create a new account, type:

```
sudo useradd -m -d /opt/tomcat -U -s /bin/false tomcat
```

The above command will create a user and group with the name "`tomcat`" in your system.

Step 3 – Install Tomcat 10

The Apache Tomcat development team releases the latest version of Tomcat from time to time. So it will be good to check to download the latest Tomcat version from the official download server. Use the below command to download Tomcat 10.

```
wget https://mirrors.estointernet.in/apache/tomcat/tomcat-10/v10.0.5/bin/apache-tomcat-10.0.5.tar.gz
```

Once the download completed, extract the downloaded archive and copy all content to the tomcat home directory.

```
sudo tar xzvf apache-tomcat-10.0.5.tar.gz -C /opt/tomcat --strip-components=1
```

Next, set the proper file permissions.

```
sudo chown -R tomcat:tomcat /opt/tomcat/

sudo chmod -R u+x /opt/tomcat/bin
```

You have now the latest Tomcat application on your system.

Step 4 – Create Tomcat User

Now, configure your tomcat with user accounts to secure access to admin/manager pages. To do this, edit **conf/tomcat-users.xml** file in your editor and paste the following code inside <tomcat-users> </tomcat-users> tags. We recommend changing the password in the below configuration with high secured password.

```
sudo nano /opt/tomcat/conf/tomcat-users.xml
```

Add the following values. Make sure to change the password for admin and manager access.

```
<!-- user manager can access only manager section -->

<role rolename="manager-gui" />

<user username="manager" password="Your_Password" roles="manager-gui" />

<!-- user admin can access manager and admin section both -->

<role rolename="admin-gui" />

<user username="admin" password="Your_Password" roles="manager-gui,admin-gui" />
```

Save file and close.

Step 5 – Enable Remote Tomcat Access

The default Tomcat manager and host-manager applications are accessible for localhost only. To allow access these pages from the remote system, you need to modify the following configuration files.
You can either allow specific remote system or allow all. Edit the `context.xml` file for the manager and host manager application:

```
sudo nano /opt/tomcat/webapps/manager/META-INF/context.xml
```

Comment out the section added for IP address restriction to allow connections from anywhere.

```
<Context antiResourceLocking="false" privileged="true" >

  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"

                   sameSiteCookies="strict" />

  <!-- <Valve className="org.apache.catalina.valves.RemoteAddrValve"
allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" /> --> ... </Context>
```

Similarly edit `context.xml` for host-manager application in the text editor:

```
sudo nano /opt/tomcat/webapps/host-manager/META-INF/context.xml
```

Comment out the same section to allow connections from anywhere.

```
<Context antiResourceLocking="false" privileged="true" >

  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"

                   sameSiteCookies="strict" />

  <!--<Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" /> -->
  ...
</Context>
```

Save all files and close them.

Step 6 – Create a Tomcat Systemd Unit File

Tomcat provides bash scripts to start, stop service. But, to make it simpler, create a startup script to manage Tomcat as systemd service. Let's create a tomcat.service file with the following content:

```
sudo nano /etc/systemd/system/tomcat.service
```

```
[Unit]

Description=Tomcat

After=network.target



[Service]

Type=forking



User=tomcat

Group=tomcat



Environment="JAVA_HOME=/usr/lib/jvm/java-1.11.0-openjdk-amd64"

Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom"
```

```
Environment="CATALINA_BASE=/opt/tomcat"

Environment="CATALINA_HOME=/opt/tomcat"

Environment="CATALINA_PID=/opt/tomcat/temp/tomcat.pid"

Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"



ExecStart=/opt/tomcat/bin/startup.sh

ExecStop=/opt/tomcat/bin/shutdown.sh



[Install]

WantedBy=multi-user.target
```

Reload the systemd daemon service to load newly create files.

```
sudo systemctl daemon-reload
```

Now, start the Tomcat application for the first time.

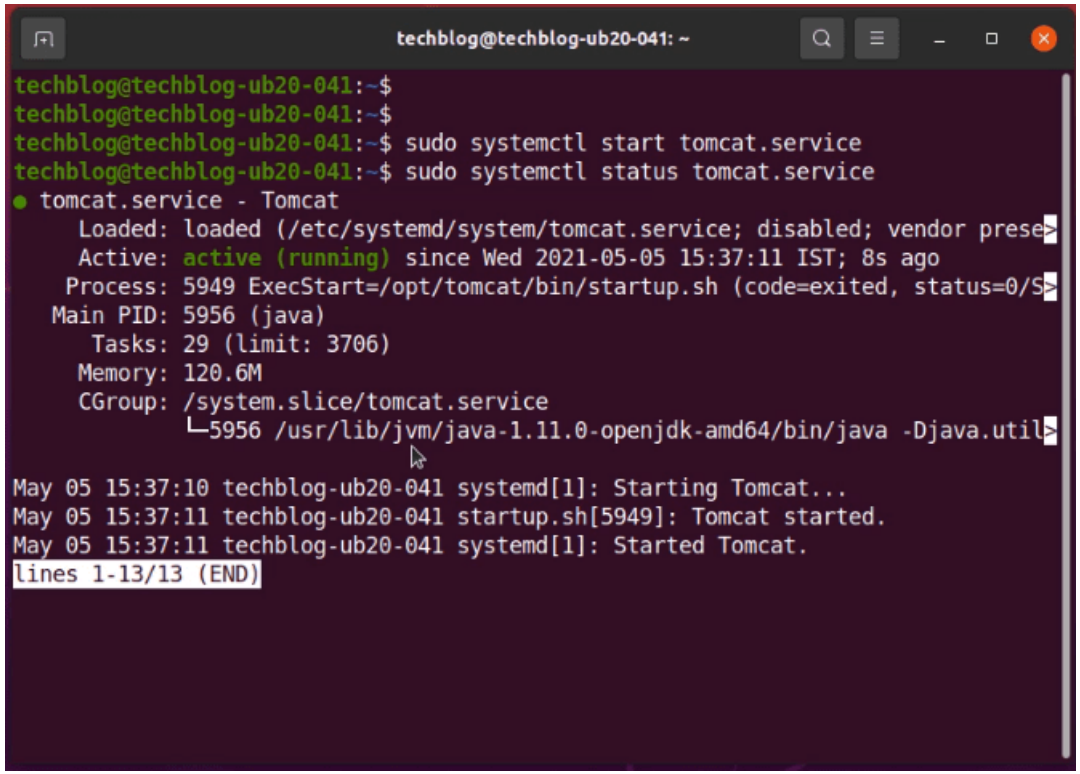
```
sudo systemctl start tomcat.service
```

Next, enable the tomcat service to auto start for subsequent system boots. This is more important for the production deployments.

```
sudo systemctl enable tomcat.service
```

As of now, the tomcat application is running on your system. You can verify the service status by executing the command as below. Make sure the status is showing "`active (running)`".

```
sudo systemctl status tomcat.service
```

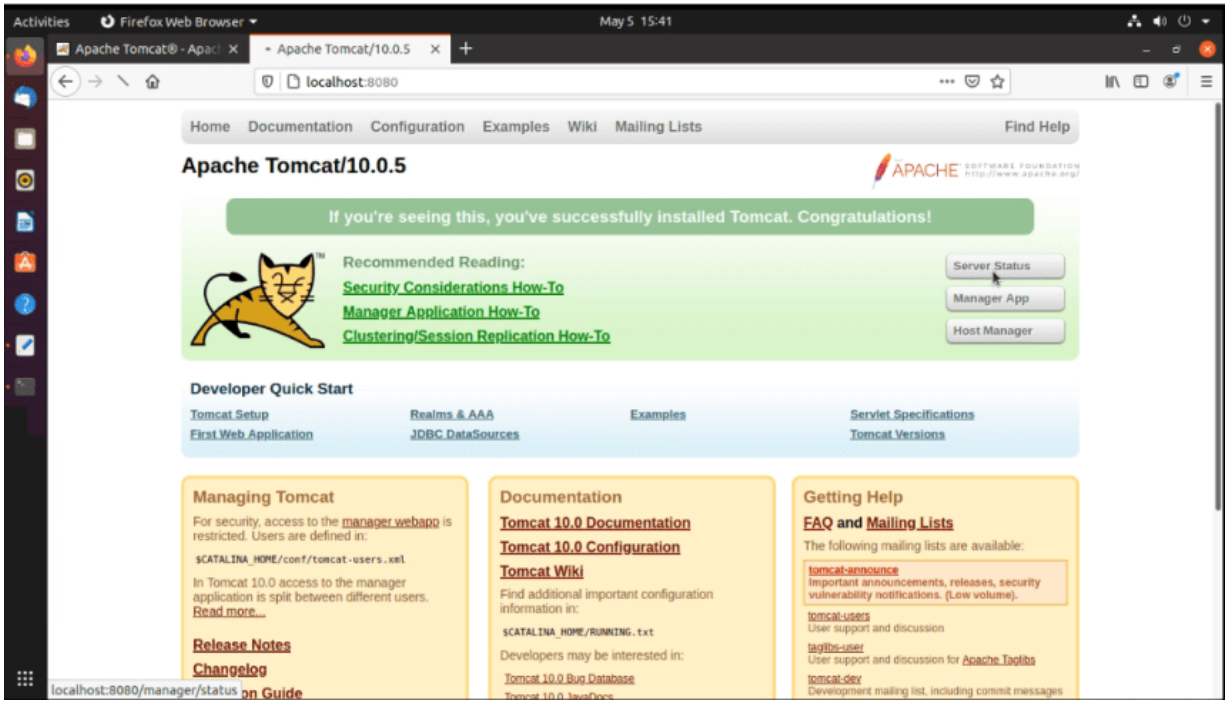


Step 7 – Access the Tomcat Web Interface

The default Tomcat server runs on port 8080. As you have configured Tomcat on your system, you can access the web interface from your system. You can access tomcat interfaces by entering your server's IP address or a domain name pointed to that server, followed by port 8080 in your browser:

```
http://localhost:8080/
```

You will see the page like below:



Step 8 – Configure DNS for Tomcat Server

At a terminal prompt, enter the following command to install dns:

--→ `sudo apt install bind9`

A very useful package for testing and troubleshooting DNS issues is the `dnsutils` package. Very often these tools will be installed already, but to check and/or install `dnsutils` enter the following:

--→ `sudo apt install dnsutils`

Primary Server

In this section BIND9 will be configured as the Primary server for the domain `example.com`. Simply replace `example.com` with your FQDN (Fully Qualified Domain Name).

Forward Zone File

To add a DNS zone to BIND9, turning BIND9 into a Primary server, first edit `/etc/bind/named.conf.local`:

```
zone "example.com" {
    type master;
    file
"/etc/bind/db.example.com";
};
```

Now use an existing zone file as a template to create the `/etc/bind/db.example.com` file:

--→ `sudo cp /etc/bind/db.local /etc/bind/db.example.com`

Edit the new zone file `/etc/bind/db.example.com` and change `localhost.` to the FQDN of your server, leaving the additional `.` at the end. Change `127.0.0.1` to the nameserver's IP Address and `root.localhost` to a valid email address, but with a `.` instead of the usual `@` symbol, again leaving the `.` at the end. Change the comment to indicate the domain that this file is for.

Create an *A record* for the base domain, `example.com`. Also, create an *A record* for `ns.example.com`, the name server in this example:

```
;
; BIND data file for example.com
;
$TTL    604800
@       IN      SOA     example.com. root.example.com. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL

@       IN      NS      ns.example.com.
@       IN      A       192.168.1.10
@       IN      AAAA    ::1
ns      IN      A       192.168.1.10
```

Once you have made changes to the zone file BIND9 needs to be restarted for the changes to take effect:

```
-->    sudo systemctl restart bind9.service
```

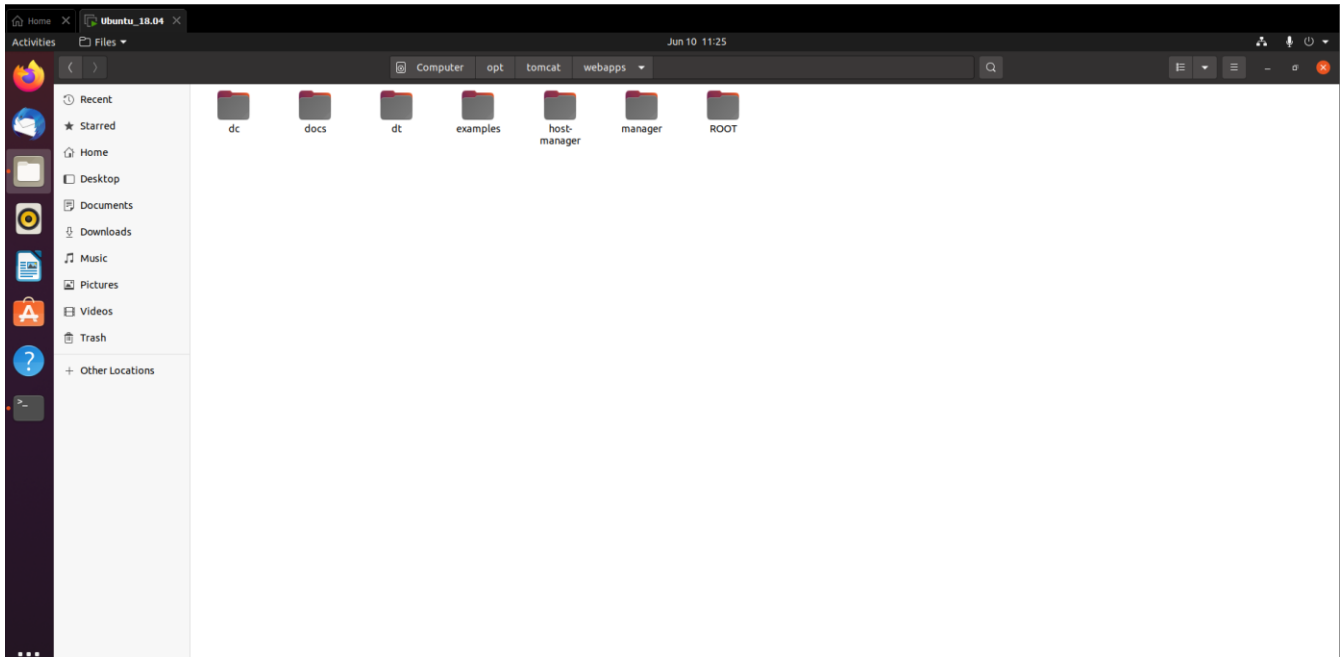
Simulate the DNS

Open the **/etc/hosts** file through root privilege and add following entry

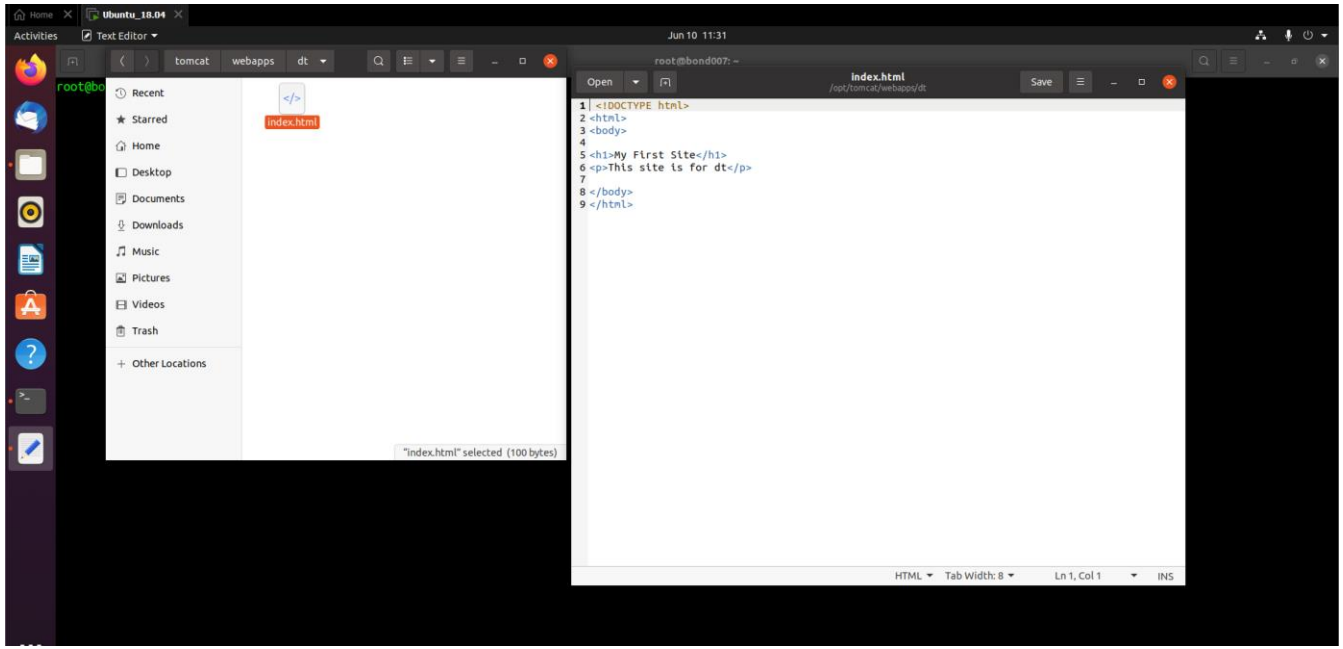
192.168.80.245 bond007.delta.local
192.168.80.245 dc.delta.local
192.168.80.245 dt.delta.local

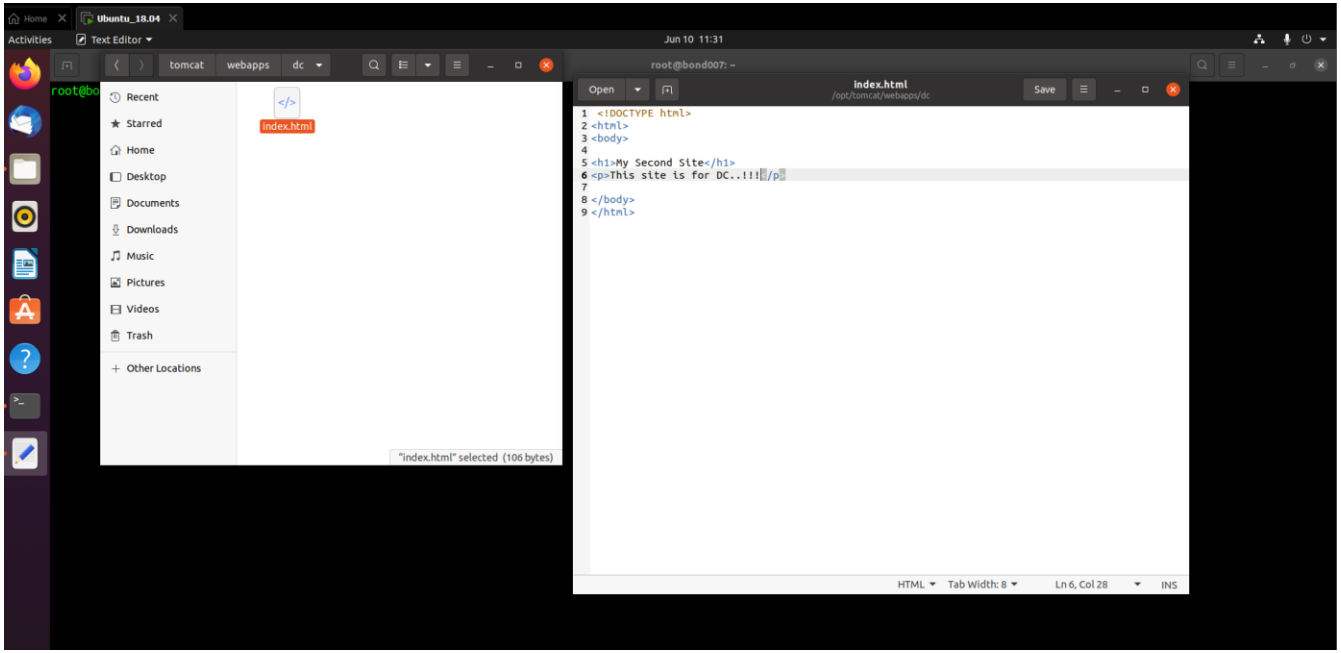
Step 9 A] – How to Run Multiple Web Applications in One Apache Tomcat Server with the same port

Create two folders for two different websites: - e.g., dc and dt



Create two different HTML files and put them in a dc and dt folder





And Browse the 3 Websites with same port

http://dt.delta.local:8080/

http://dt.delta.local:8080/dc/

http://dt.delta.local:8080/dt/