



## 1. AWS Deployment 60% (/m/59/5447)

EC2 Set Up (/m/)

MySQL and Data Export (/m/)

Apache Set Up (/m/)

Spring Boot Set Up (/m/)

JDK and systemd

Chapter Survey

## Spring Boot Set Up

When a HTTP request comes into our EC2 server, Apache will receive the request and use reverse proxy to forward the request to our Spring Boot application running on port 9090. In this tab, we will secure copy our application into our EC2 instance and use systemd to run our application.

- For our reverse proxy to work, we are going to use the Apache JServ Protocol. In your Spring Boot application, add the following code:

com.codingdojo.auth.AuthApplication.java

```
package com.codingdojo.auth;
import org.apache.catalina.connector.Connector;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.web.embedded.tomcat.TomcatServletWebServerFactory;
import org.springframework.context.annotation.Bean;
@SpringBootApplication
public class AuthApplication {
    public static void main(String[] args) {
        SpringApplication.run(AuthApplication.class, args);
    }
    @Bean
    public TomcatServletWebServerFactory servletContainer() {
        TomcatServletWebServerFactory tomcat = new TomcatServletWebServerFactory();
        Connector ajpConnector = new Connector("AJP/1.3");
        ajpConnector.setPort(9090);
        ajpConnector.setSecure(false);
        ajpConnector.setAllowTrace(false);
        ajpConnector.setScheme("http");
        tomcat.addAdditionalTomcatConnectors(ajpConnector);
        return tomcat;
    }
}
```

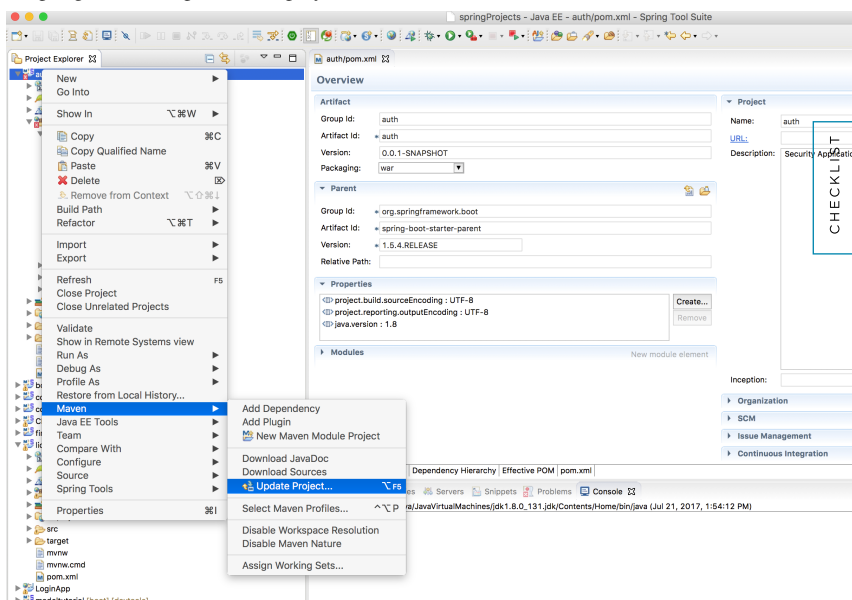
- Next, we need to package our project into a **war** file.

- Go to your 'pom.xml' file and Overview tab. If your packaging is jar, change it to war.

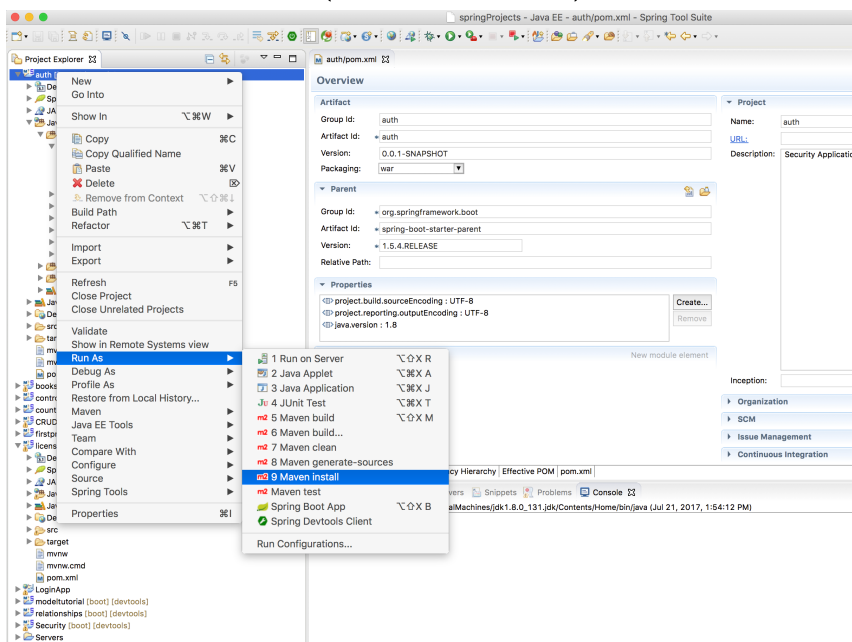
- Run Maven -> Update Project



## Spring Boot Set Up | AWS Deployment



- Run Run As -> Maven Install (this will create the war file)



3. STS will build a war file and save it inside the target directory. In this example, the full directory is **/Users/eduardobaik/Desktop/springProjects/auth/target/auth-0.0.1-SNAPSHOT.war**.

- Navigate to said directory in your terminal and secure copy the war file into the home. You will need your pem file path and your public ip address. For example:

```
scp -i ~/Desktop/springProject.pem auth-0.0.1-SNAPSHOT.war ubuntu@34.228.244.112:~/
```



BACK TO TRACKS

The screenshot shows two terminal windows. The top window is titled 'Desktop — -bash — 113x27' and shows a user named 'eduardobaik' at 'Eduardos-MBP' using 'scp' to transfer a file named 'auth-0.0.1-SNAPSHOT.war' from their local machine to a remote server. The bottom window is titled 'Desktop — ubuntu@ip-172-31-9-165: ~ — ssh -i springProject.pem ubu' and shows the user 'ubuntu' at 'ip-172-31-9-165' using 'pwd' to confirm the directory and 'ls' to list files, showing the file 'auth-0.0.1-SNAPSHOT.war' has been successfully transferred.

4. Let's create a folder for our application inside of the '/var' directory.

```
sudo mkdir /var/springApp
sudo mv ~/auth-0.0.1-SNAPSHOT.war /var/springApp/
```

5. Now, we need to tell Apache to proxy requests to our application.

- Set up proxy

```
sudo a2enmod proxy
sudo a2enmod proxy_ajp
```

- Open our virtual host conf file.

```
cd /etc/apache2/sites-available
sudo vim 000-default.conf
```

- Add the proxy configuration at the bottom. Make sure to have it run on port 9090.

[BACK TO TRACKS](#)

```
<VirtualHost *:80>
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
ProxyPass / ajp://localhost:9090/
ProxyPassReverse / ajp://localhost:9090/
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
:wq
```

CHECKLIST

#### 6. Restart Apache.

```
sudo service apache2 restart
```

[Privacy Policy](#)[To repor](#)[◀ PREVIOUS \(/M/59/5447/34106\)](#)