

EXPERIMENT NO 6

Aim : Implement SSL Web Security method.

Theory :

SSL stands for Secure Sockets Layer. SSL is the standard security technology for establishing an encrypted link between a web server and a browser. This link ensures that all data passed between the web server and browsers remain private and integral. SSL is an industry standard and is used by millions of websites in the protection of their online transactions with their customers. To be able to create an SSL connection a web server requires an SSL Certificate. When you choose to activate SSL on your web server you will be prompted to complete a number of questions about the identity of your website and your company. Then web server then creates two cryptographic keys - a Private Key and a Public Key. The Public Key does not need to be secret and is placed into a Certificate Signing Request (CSR) - a data file also containing your details. You should then submit the CSR.

During the SSL Certificate application process, the Certification Authority will validate your details and issue an SSL Certificate containing your details and allowing you to use SSL. Your web server will match your issued SSL Certificate to your Private Key. Your web server will then be able to establish an encrypted link between the website and your customer's web browser. The complexities of the SSL protocol remain invisible to your customers. Instead their browsers provide them with a key indicator to let them know they are currently protected by an SSL encrypted session - the lock icon in the lower right-hand corner, clicking on the lock icon displays your SSL Certificate and the details about it. All SSL Certificates are issued to either companies or legally accountable individuals.

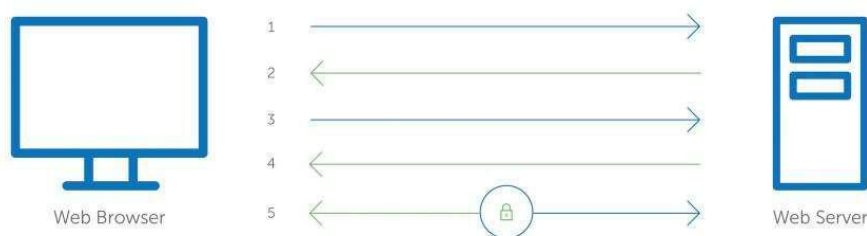


Fig 1 : SSL establishment process

Figure 1 describes

1. Browser connects to a web server (website) secured with SSL (https). Browser requests that the server identify itself.
2. Server sends a copy of its SSL Certificate, including the server's public key.
3. Browser checks the certificate root against a list of trusted CAs and that the certificate is unexpired, unrevoked, and that its common name is valid for the website that it is connecting to. If the browser trusts the certificate, it creates, encrypts, and sends back a symmetric session key using the server's public key.
4. Server decrypts the symmetric session key using its private key and sends back an acknowledgement encrypted with the session key to start the encrypted session.
5. Server and Browser now encrypt all transmitted data with the session key. SSL consists of :
 - It contains your domain name, your company name, your address, your city, your state and your country.
 - It will also contain the expiration date of the Certificate and details of the Certification Authority responsible for the issuance of the Certificate.
 - When a browser connects to a secure site it will retrieve the site's SSL Certificate and check that it has not expired, it has been issued by a Certification Authority the browser trusts, and that it is being used by the website for which it has been issued.
 - If it fails on any one of these checks the browser will display a warning to the end user letting them know that the site is not secured by SSL.

Objective:

Create a SSL connection between web server and the client.

- IDE: Netbeans 8.0.1 version
- Java Development Kit (JDK): Version 8
- Glassfish server: 4.1.2

- Operating System : Ubuntu 16.04

Step 1:

Install all required packages / IDE

- Install netbeans

```
$ sudo add-apt-repository ppa:vajdics/netbeans-installer
```

```
$ sudo apt update
```

```
$ sudo apt install netbeans-installer
```

```
$ Install Glassfish server 4.1.2
```

Download <https://download.oracle.com/glassfish/4.1.2/release/index.html> and extract in home directory

Step 2:

Configure Glassfish server and create a certificate

1. Open Terminal and change directory to glassfish domain config

```
$ cd glassfish-4.1.1/glassfish/domains/domain1/config/
```

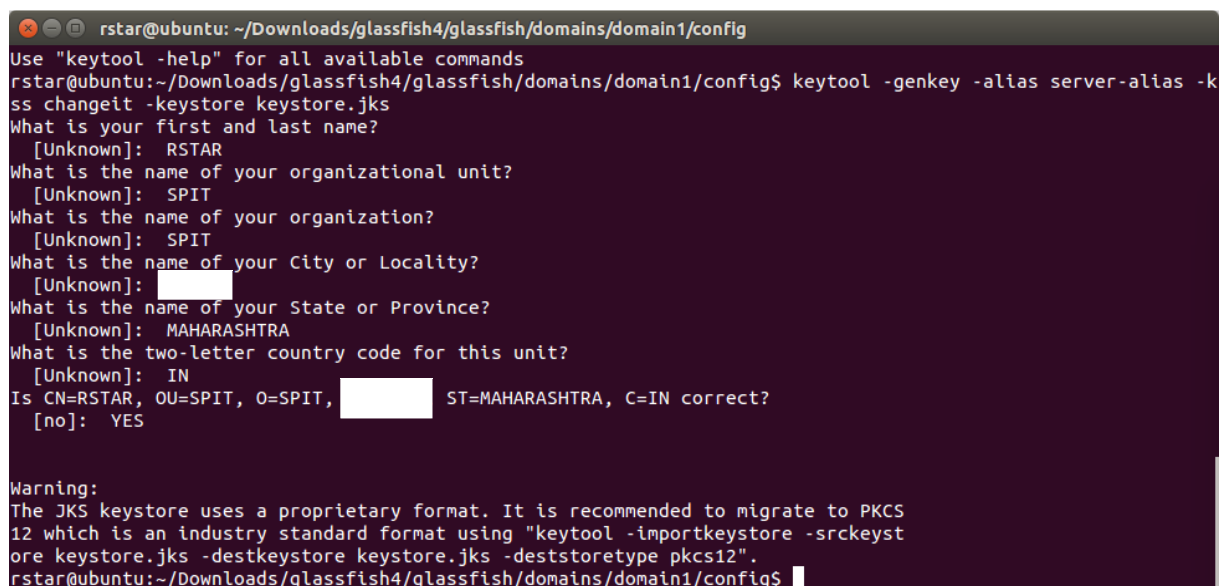
2. Use keytool command to generate the certificates.

Keytool is a key and certificate management utility provided by Java SE 8

Run the following commands as follows:

1. Generate new keys

```
$ keytool -genkey -alias server-alias -keyalg RSA -keypass changeit -storepass changeit -keystore keystore.jks
```



```
rstar@ubuntu: ~/Downloads/glassfish4/glassfish/domains/domain1/config
Use "keytool -help" for all available commands
rstar@ubuntu:~/Downloads/glassfish4/glassfish/domains/domain1/config$ keytool -genkey -alias server-alias -k
ss changeit -keystore keystore.jks
What is your first and last name?
[Unknown]: RSTAR
What is the name of your organizational unit?
[Unknown]: SPIT
What is the name of your organization?
[Unknown]: SPIT
What is the name of your City or Locality?
[Unknown]: 
What is the name of your State or Province?
[Unknown]: MAHARASHTRA
What is the two-letter country code for this unit?
[Unknown]: IN
Is CN=RSTAR, OU=SPIT, O=SPIT, ST=MAHARASHTRA, C=IN correct?
[no]: YES

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS
12 which is an industry standard format using "keytool -importkeystore -srckeyst
ore keystore.jks -destkeystore keystore.jks -deststoretype pkcs12".
rstar@ubuntu:~/Downloads/glassfish4/glassfish/domains/domain1/config$
```

2. Export the generated key certificate to server.cer

```
$ keytool -export -alias server-alias -storepass changeit -file server.cer -keystore  
keystore.jks
```

```
rstar@ubuntu: ~/Downloads/glassfish4/glassfish/domains/domain1/config
rstar@ubuntu:~/Downloads/glassfish4/glassfish/domains/domain1/config$ keytool -export -alias server-alias -s
keystore keystore.jks
Certificate stored in file <server.cer>

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry stan
ystore -srckeystore keystore.jks -destkeystore keystore.jks -deststoretype pkcs12".
rstar@ubuntu:~/Downloads/glassfish4/glassfish/domains/domain1/config$
```

3. Add the certificate to the trusted store file

```
$ keytool -import -v -trustcacerts -alias server-alias -file server.cer -keystore cacerts.jks -  
keypass changeit -storepass changeit
```

```
rstar@ubuntu: ~/Downloads/glassfish4/glassfish/domains/domain1/config
rstar@ubuntu:~/Downloads/glassfish4/glassfish/domains/domain1/config$ keytool -import -v -trustcacerts -alia
s server-alias -file server.cer -keystore cacerts.jks -keypass changeit -storepass changeit
Owner: CN=RSTAR, OU=SPIT, O=SPIT, ST=MAHARASHTRA, C=IN
Issuer: CN=RSTAR, OU=SPIT, O=SPIT, ST=MAHARASHTRA, C=IN
Serial number: 7c15bd6a
Valid from: Sun Apr 07 07:27:41 PDT 2019 until: Sat Jul 06 07:27:41 PDT 2019
Certificate fingerprints:
    MD5: 23:D4:A5:5B:C6:EE:60:57:28:02:44:53:05:2C:12:3F
    SHA1: BA:A5:F0:92:92:2A:DA:E4:C4:D6:71:95:A5:4A:B6:C3:18:03:ED:B7
    SHA256: AA:F3:3D:B5:68:81:F0:0C:2C:37:9C:32:EA:67:56:5A:CA:E2:65:0E:CA:09:5D:24:90:DA:E2:40:7E:CF:9
F:26
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 62 C9 C8 76 F9 70 09 FA 16 8E 4B B9 4F 84 83 1F b..v.p....K.O...
0010: 2D 68 5C 44 -h\D
]
]

Trust this certificate? [no]: YES
Certificate was added to keystore
[Storing cacerts.jks]
rstar@ubuntu:~/Downloads/glassfish4/glassfish/domains/domain1/config$
```

4. Verify if the certificate was successfully added into the keystore.

```
$ keytool -list -v -keystore keystore.jks
```

```
rstar@ubuntu: ~/Downloads/glassfish4/glassfish/domains/domain1/config

rstar@ubuntu:~/Downloads/glassfish4/glassfish/domains/domain1/config$ keytool -list -v -keystore keystore.jks
Enter keystore password:

***** WARNING WARNING WARNING *****
* The integrity of the information stored in your keystore *
* has NOT been verified! In order to verify its integrity, *
* you must provide your keystore password. *
***** WARNING WARNING WARNING *****

Keystore type: jks
Keystore provider: SUN

Your keystore contains 3 entries

Alias name: glassfish-instance
Creation date: Mar 23, 2017
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=localhost-instance, OU=GlassFish, O=Oracle Corporation, L=Santa Clara, ST=California, C=US
Issuer: CN=localhost-instance, OU=GlassFish, O=Oracle Corporation, L=Santa Clara, ST=California, C=US
Serial number: 76a8bbae
Valid from: Thu Mar 23 16:20:42 PDT 2017 until: Sun Mar 21 16:20:42 PDT 2027
Certificate fingerprints:
    MD5:  4E:D4:CF:22:3D:77:3F:B0:1C:0F:94:64:B7:9B:F2:06
    SHA1: 60:64:1C:FE:E8:61:74:30:03:39:B9:90:9D:43:98:68:07:B7:86:63
```

5. Validate if the certificate was successfully added into the trust store.

\$ keytool -list -keystore cacerts.jks

```
rstar@ubuntu: ~/Downloads/glassfish4/glassfish/domains/domain1/config

rstar@ubuntu:~/Downloads/glassfish4/glassfish/domains/domain1/config$ keytool -list -keystore cacerts.jks
Enter keystore password:

***** WARNING WARNING WARNING *****
* The integrity of the information stored in your keystore *
* has NOT been verified! In order to verify its integrity, *
* you must provide your keystore password. *
***** WARNING WARNING WARNING *****

Keystore type: jks
Keystore provider: SUN

Your keystore contains 77 entries

digicertassuredidrootca, Jan 7, 2008, trustedCertEntry,
Certificate fingerprint (SHA1): 05:63:B8:63:0D:62:D7:5A:BB:C8:AB:1E:4B:DF:B5:A8:99:B2:4D:43
trustcenterclass2caii, Jan 7, 2008, trustedCertEntry,
Certificate fingerprint (SHA1): AE:50:83:ED:7C:F4:5C:BC:8F:61:C6:21:FE:68:5D:79:42:21:15:6E
thawtepremiumserverca, Dec 2, 2009, trustedCertEntry,
Certificate fingerprint (SHA1): E0:AB:05:94:20:72:54:93:05:60:62:02:36:70:F7:CD:2E:FC:66:66
swissignplatinumg2ca, Aug 13, 2008, trustedCertEntry,
Certificate fingerprint (SHA1): 56:E0:FA:C0:3B:8F:18:23:55:18:E5:D3:11:CA:E8:C2:43:31:AB:66
swissignsilverg2ca, Aug 13, 2008, trustedCertEntry,
Certificate fingerprint (SHA1): 9B:AA:E5:9F:56:EE:21:CB:43:5A:BE:25:93:DF:A7:F0:40:D1:1D:CB
thawteserverca, Dec 2, 2009, trustedCertEntry,
Certificate fingerprint (SHA1): 9F:AD:91:A6:CE:6A:C6:C5:00:47:C4:4E:C9:D4:A5:0D:92:D8:49:79
equifaxsecurebusinessca1, Jul 18, 2003, trustedCertEntry,
Certificate fingerprint (SHA1): DA:40:18:8B:91:89:A3:ED:EE:AE:DA:97:FE:2F:9D:F5:B7:D1:8A:41
utnuserfirstclientauthemailca, May 2, 2006, trustedCertEntry,
Certificate fingerprint (SHA1): B1:72:B1:A5:6D:95:F9:1F:E5:02:87:E1:4D:37:EA:6A:44:63:76:8A
```

So now the certificate is available both in the keystore and truststore. The keystore contains the private key of the server while truststore contains the CA certificates or the public keys only. This is a cleaner demarcation of the certificates and the keys where private keys can be

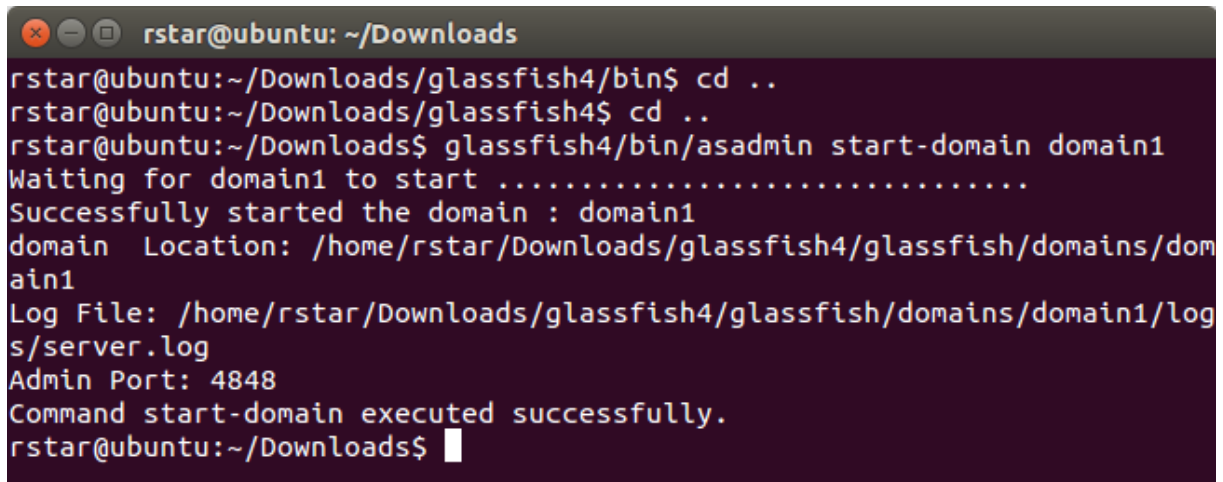
kept in more secured environment in the keystores but public keys can be kept in more accessible option in the truststore.

However in this example since we do not have a CA certificate the server certificate is stored in the trusted store.

Step 3. Setup Glassfish Server

1. Start glassfish server

```
$ glassfish-4.1.1/bin/asadmin start-domain  
domain1
```

A terminal window titled 'rstar@ubuntu: ~/Downloads' shows the following commands and output:

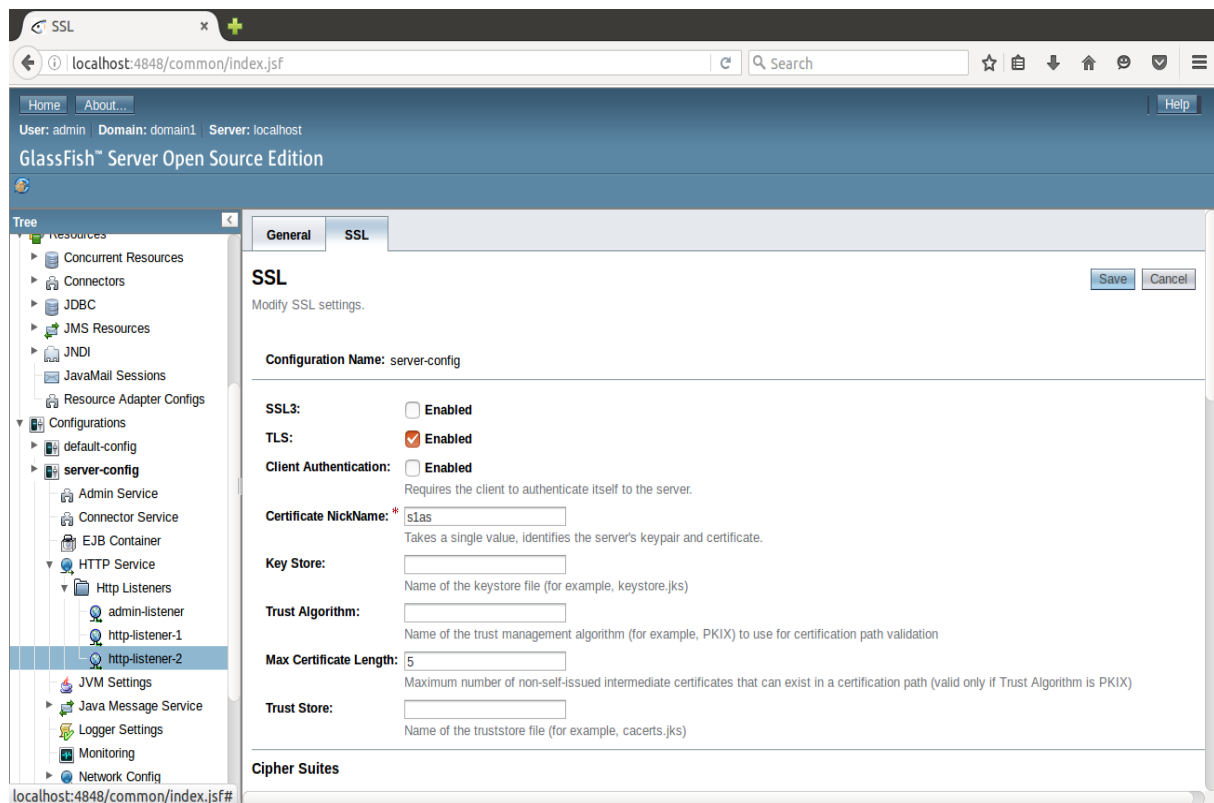
```
rstar@ubuntu:~/Downloads/glassfish4/bin$ cd ..  
rstar@ubuntu:~/Downloads/glassfish4$ cd ..  
rstar@ubuntu:~/Downloads$ glassfish4/bin/asadmin start-domain domain1  
Waiting for domain1 to start .....  
Successfully started the domain : domain1  
domain Location: /home/rstar/Downloads/glassfish4/glassfish/domains/domain1  
Log File: /home/rstar/Downloads/glassfish4/glassfish/domains/domain1/logs/server.log  
Admin Port: 4848  
Command start-domain executed successfully.  
rstar@ubuntu:~/Downloads$
```

Go To Localhost : 4848 on web browser

Click Configurations -> server-config -> HTTP Service -> http-listeners-2

Http-Listeners-2 denotes the secured HTTPS port 8181

Click the SSL tab and modify the Certificate Nick-name to “server-alias” as per the certificate we have created above.



2. Restart glassfish server

\$ glassfish-4.1.1/bin/asadmin restart-domain domain1

3. Stop glassfish server

\$ glassfish-4.1.1/bin/asadmin stop-domain domain1

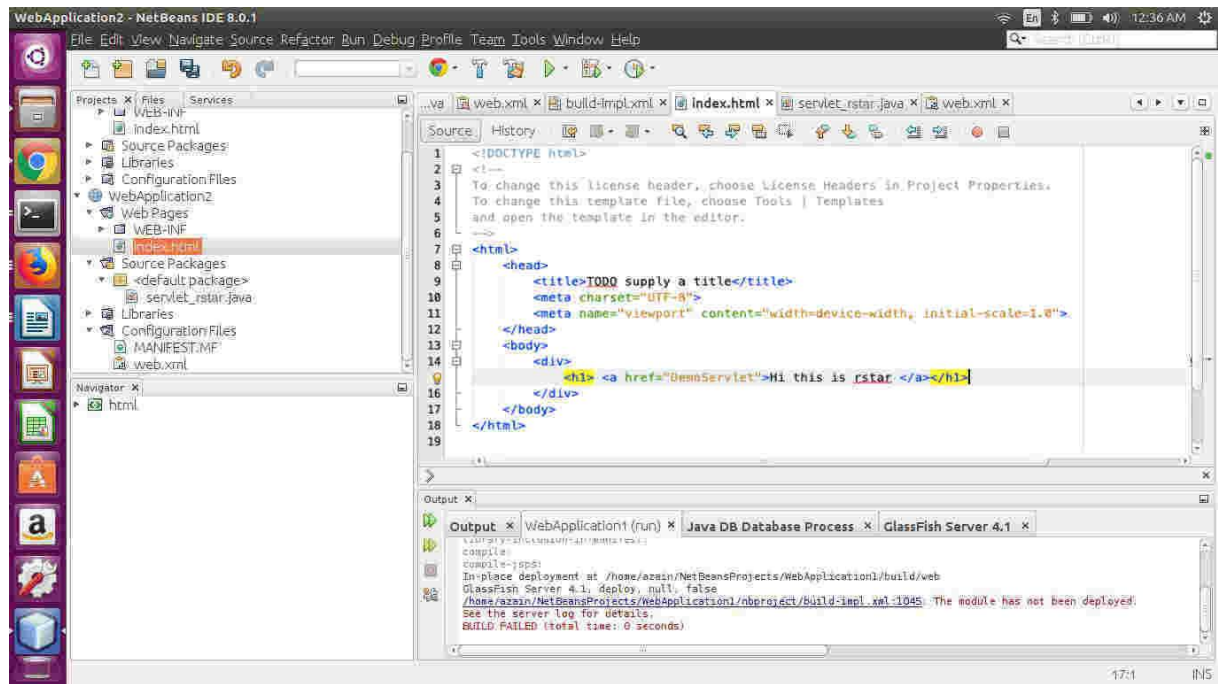
Shown above

4. Develop and Deploy Java application in Netbeans 8.2

Create web application in java.

Using Reference : <https://www.javatpoint.com/creating-servlet-in-netbeans-ide>

A servlet creation can be done by adding servlet by a Right Click on Default Packages under source packages.



- The created Servlet : servlet_rstar can be seen under default packages. (No need to change any code in this file. i.e Servlet file)
- Make changes in index.html under Web Page

On line No 14.

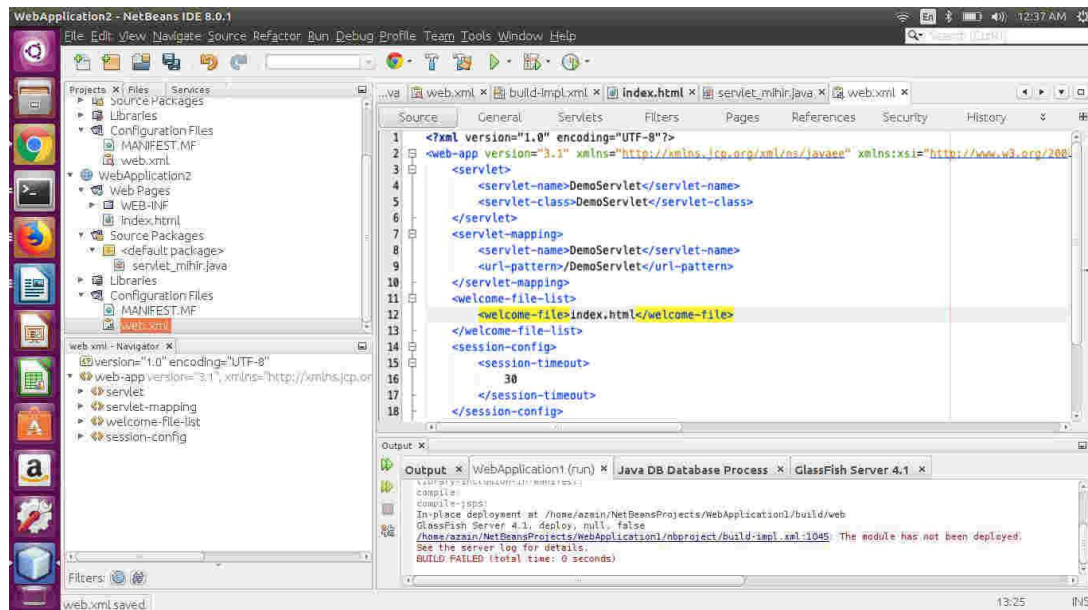
Add `<h1> hi this is rstar`

`</h1>`

Where “servlet_rstar” is the servlet created.

- Add Welcome file List in Web.xml

To create a link between the application and its first web page i.e. Index.html

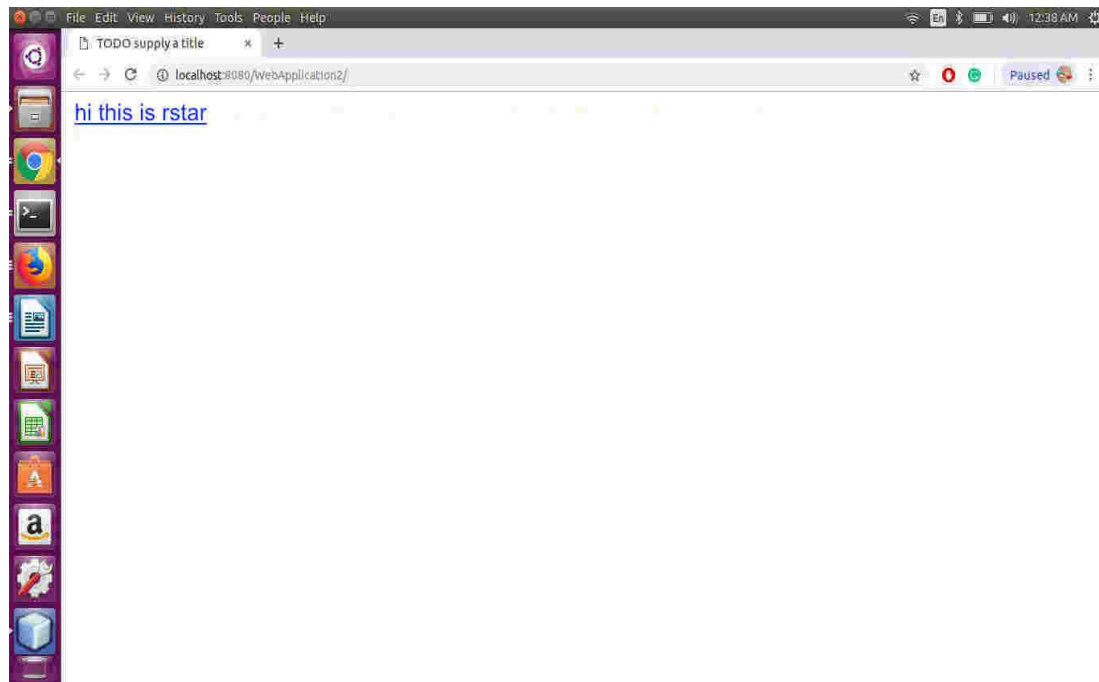


Run and Deploy the application by clicking on the play button on the toolbar.

http://localhost:8080/project_rstar/ / will appear on web browser.

i.e. http://localhost:8080/<project_name>

- The Following web page will appear. Clicking on this link will redirect to servlet web page.



Connect SSL certificate and Web application <https://howtodoinjava.com/tomcat/how-to-configure-tomcat-with-ssl-or-https/> Change the web.xml in netbeans configurations

The only change required is in web.xml where the transport-guarantee will be changed from none to confidential.

Add to web.xml the following code

```
<security-constraint>

    <web-resource-collection>

        <web-resource-name>View    Secure    URLs</web-resource-
        name> <url-pattern>/*</url-pattern>

    </web-resource-collection>

    <user-data-constraint>

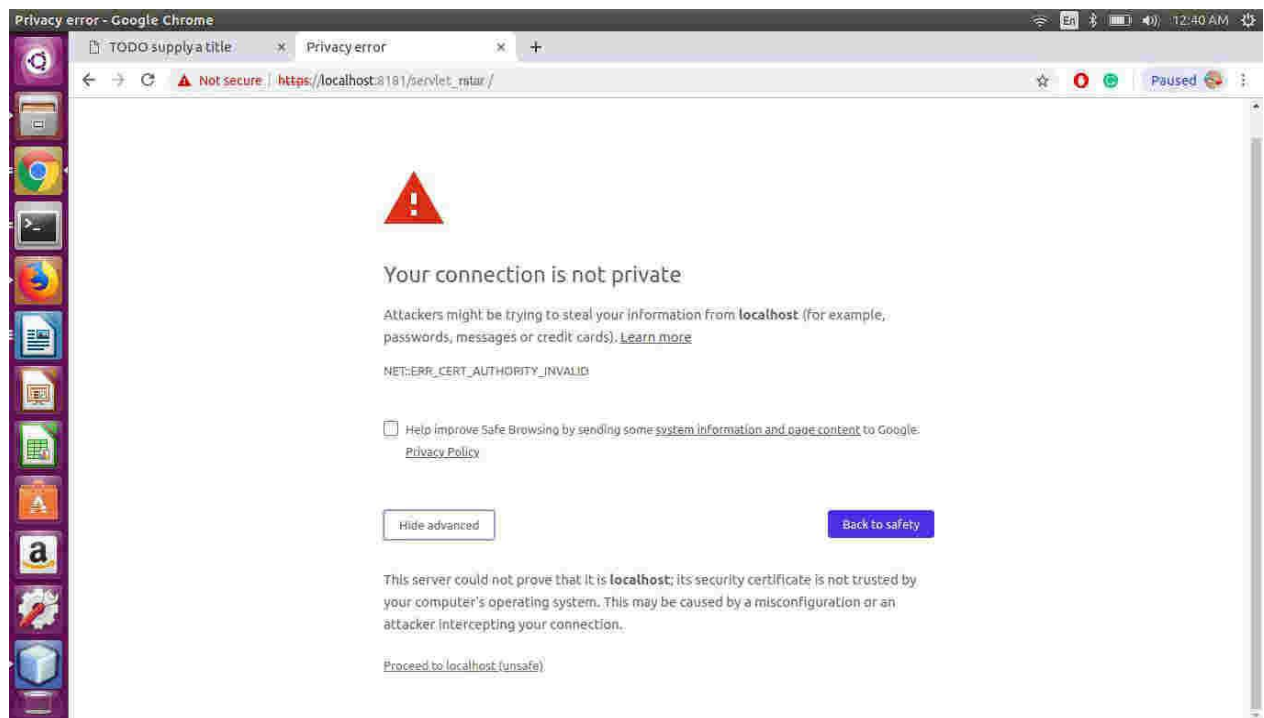
        <transport-guarantee>CONFIDENTIAL</transport-
        guarantee>

    </user-data-constraint>

</security-constraint>
```

- Explanation : The url pattern is set to /* so any page/resource from your application is secure (it can be only accessed with https). The transport-guarantee tag is set to CONFIDENTIAL to make sure your app will work on SSL.

Now try to access the application using https://localhost:8443/<project_name>/. This will show the certificate information in browser.



Conclusion:

From the above experiment we have studied how to generate secure certificates using keytool on open source glassfish server and a web application. The certificate produced by the server are self signed one instead of a certificate from CA(Certification Authority)the browsers gives a warning message. This is because browser's truststore does not contain these certificates.