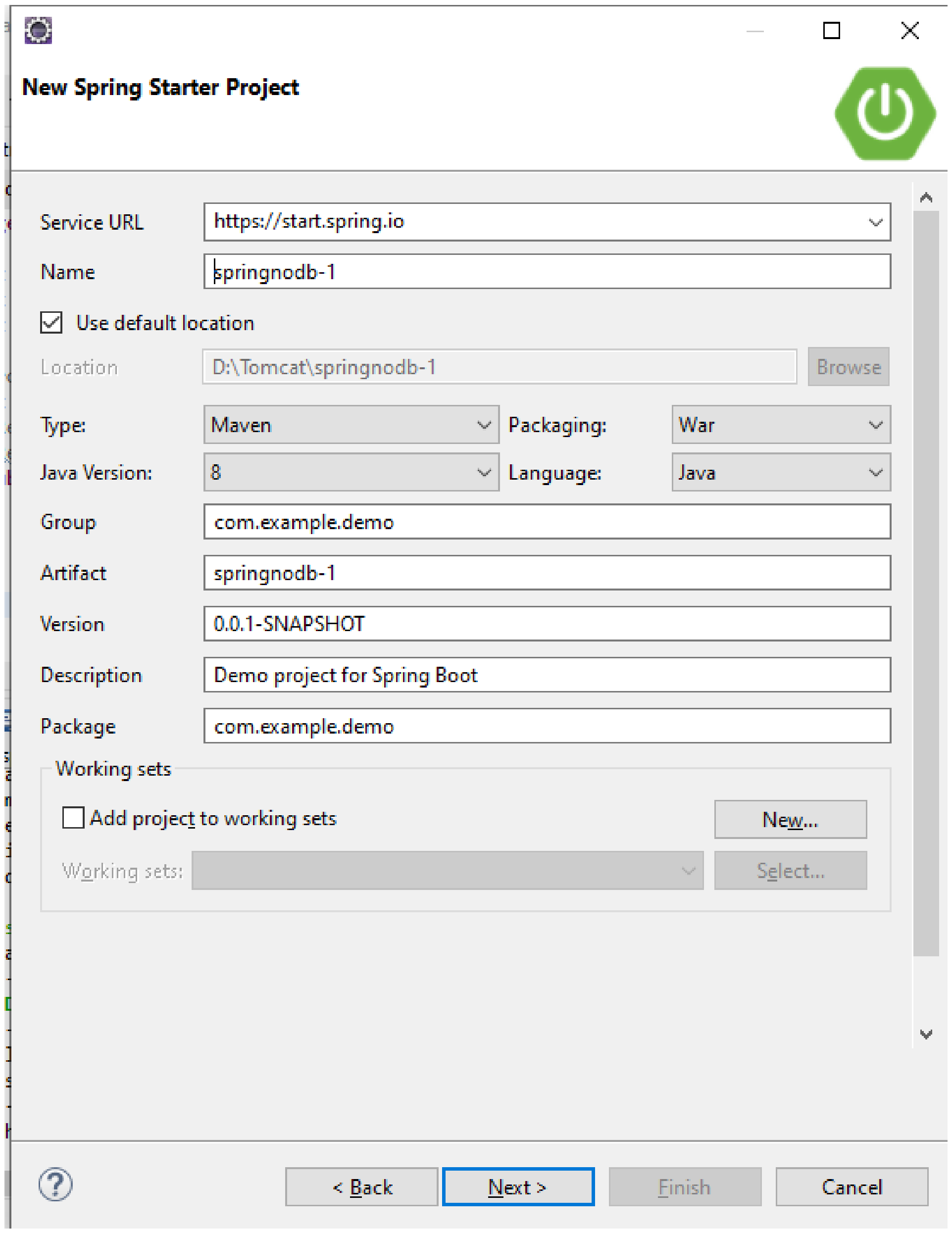
SOURCE CODE:

Let’s go to sprint initializer page and quickly bootstrap a simple spring boot web application.
Open https://start.spring.io/ and fill up details such as Group, Artifact, Name and dependency.


New Spring Starter Project

Service URL	https://start.spring.io
Name	springnodb-1
☑ Use default location	
Location	D:\Tomcat\springnodb-1
Type:	Maven
Packaging:	War
Java Version:	8
Language:	Java
Group	com.example.demo
Artifact	springnodb-1
Version	0.0.1-SNAPSHOT
Description	Demo project for Spring Boot
Package	com.example.demo

Working sets
☐ Add project to working sets
Working sets:

Please note that I have added a single dependency Spring Web as you can see in the above screenshot.

After filling up details and selecting web dependency, click on Generate
This will download the zipped project into your local system.

Unzip the downloaded project and import the project into your IDE. I will be importing to eclipse for this tutorial.

Click on File -> Import -> Maven -> Existing Maven Project

Click Next

Click on Browse and navigate to your unzipped project folder and select that. Selecting that should show the pom.xml file as shown in below screenshot.

Click on Finish to the project into the IDE.

Now, we have the project imported into the IDE. Let’s add a Rest Controller into our spring boot project so that we can test it.

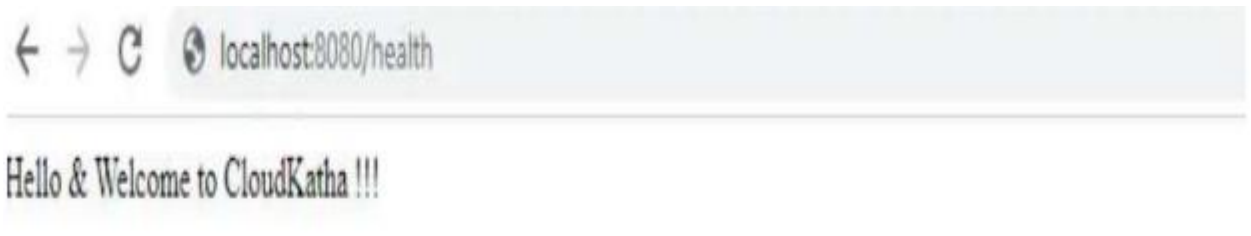
Click on your project and create a controller class with the endpoint of /health. We will use minimal code like the below snippet.

```
package com.cloudkatha.demo.controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
@RestController
public class TestController { @GetMapping("/health")
public String health() {
return "Hello & Welcome to CloudKatha !!!";
(// Here we can give any statement I wish to use this statement)
}
}
```

Before we test it, let’s build the project using mvn clean install

Once the build is successful, let’s run the project locally.

Right-click on the class annotated with @SpringBootApplication and Run as Java Application.



Hello & Welcome to CloudKatha !!!

Our application is running fine locally and you can access it like localhost:8080/health.

Our application is working perfectly in the local environment. Let’s prepare the final jar file and keep it handy in a folder as we will need the jar file during the deployment.

Run maven build to package the application as a fat jar.

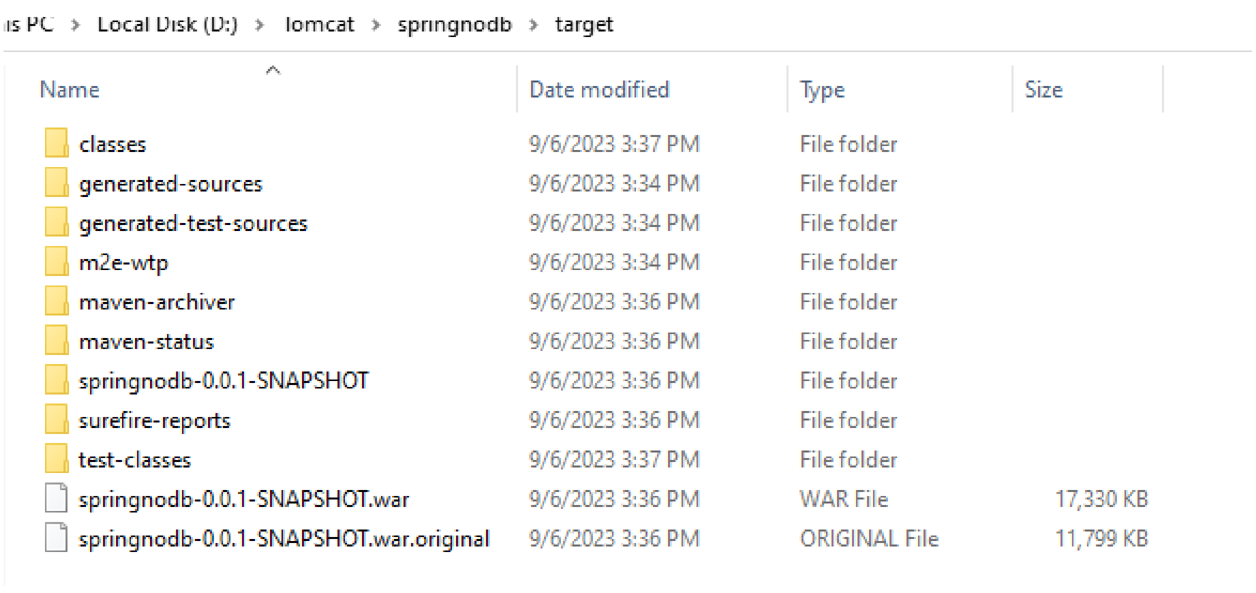
mvn clean install

Once your build is successful, you will find the jar file in the target folder.

Copy the jar file and keep it in your system at a convenient location.

Before we deploy our spring boot app, we need to launch an EC2 instance.

Let’s keep the KeyPair and the Jar to be deployed into the same folder as below



is PC > Local Disk (D:) > Tomcat > springnodb > target

Name	Date modified	Type	Size
classes	9/6/2023 3:37 PM	File folder	
generated-sources	9/6/2023 3:34 PM	File folder	
generated-test-sources	9/6/2023 3:34 PM	File folder	
m2e-wtp	9/6/2023 3:34 PM	File folder	
maven-archiver	9/6/2023 3:36 PM	File folder	
maven-status	9/6/2023 3:36 PM	File folder	
springnodb-0.0.1-SNAPSHOT	9/6/2023 3:36 PM	File folder	
surefire-reports	9/6/2023 3:36 PM	File folder	
test-classes	9/6/2023 3:37 PM	File folder	
springnodb-0.0.1-SNAPSHOT.war	9/6/2023 3:36 PM	WAR File	17,330 KB
springnodb-0.0.1-SNAPSHOT.war.original	9/6/2023 3:36 PM	ORIGINAL File	11,799 KB

Please note that If you are using a Linux machine you can use scp command out of the box. But if you are using windows, It doesn't come with scp client and you must install one before you can copy file to EC2.

So, If you are a Linux user, continue the below steps
Open the terminal and navigate to the directory where you have kept KeyPair and Jar File.

I used here Cloud9 IDE with Linux instance for development so I will use the below command to copy my files using SCP.

Navigate to the folder containing your jar and keypair.
cd /path/to/folder

Prepare the copy command as per the below syntax
scp -i ./DemoKeyPair.pem . <username>@<public-ip or

DNS>:/pathwhere/you/needto/copy

After putting the details, the command result is like below.

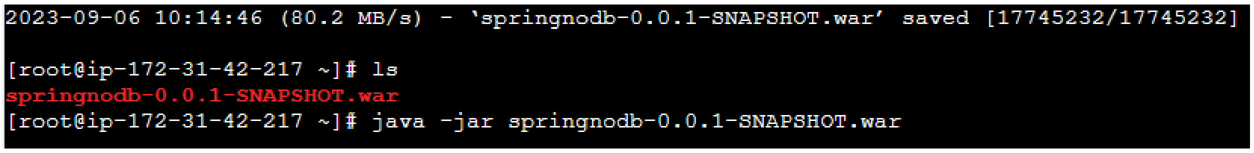
scp -i ./DemoKeyPair.pem ./demo-0.0.1-SNAPSHOT.jar ec2-user@ec2-34-240-45-168.eu-west-1.compute.amazonaws.com:-

Please note that before you can run the above command you must set the permissions of your private key file so that only you can read it. If you don't set

the permission then you can not connect to your instance using keypair.
chmod 400 DemoKeyPair.pem

Type the above command in your terminal and hit enter.
Now permission on your private key is set and you can run your copy command.

scp -i ./DemoKeyPair.pem ./demo-0.0.1-SNAPSHOT.jar ec2-user@ec2-34-240-45-168.eu-west-1.compute.amazonaws.com:

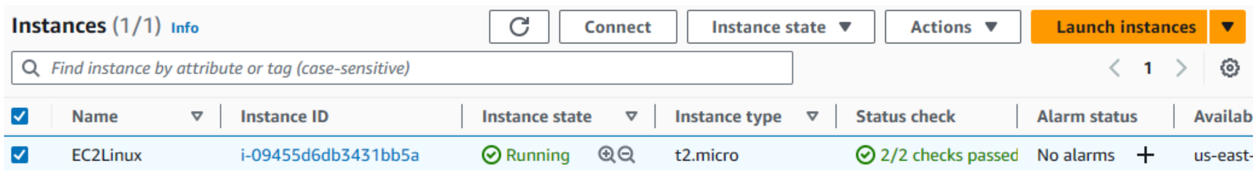


```
2023-09-06 10:14:46 (80.2 MB/s) - 'springnodb-0.0.1-SNAPSHOT.war' saved [17745232/17745232]

[root@ip-172-31-42-217 ~]# ls
springnodb-0.0.1-SNAPSHOT.war
[root@ip-172-31-42-217 ~]# java -jar springnodb-0.0.1-SNAPSHOT.war
```

As confirmed by ls command, you can see that jar is copied to the home directory of my EC2 instance.

Note: Please notice the tilda(~) sign at the end. It means the jar file is being copied to the home directory of the instance



☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availab
☑	EC2Linux	i-09455d6db3431bb5a	✓ Running	t2.micro	✓ 2/2 checks passed	No alarms +	us-east-