Spring boot in eclipse MARS :

In Eclipse, go to Help > Install New Software

Use the update url

http://dist.springsource.org/snapshot/GRECLIPSE/e4.5/ for

Eclipse Mars (4.5.x).

Select the

– Groovy Compiler 2.4 Feature and

– Groovy-Eclipse Feature.

They both give us the Groovy support we need. Next.

Optionally, Install YAML support

In Eclipse, again go to Help > Install New Software

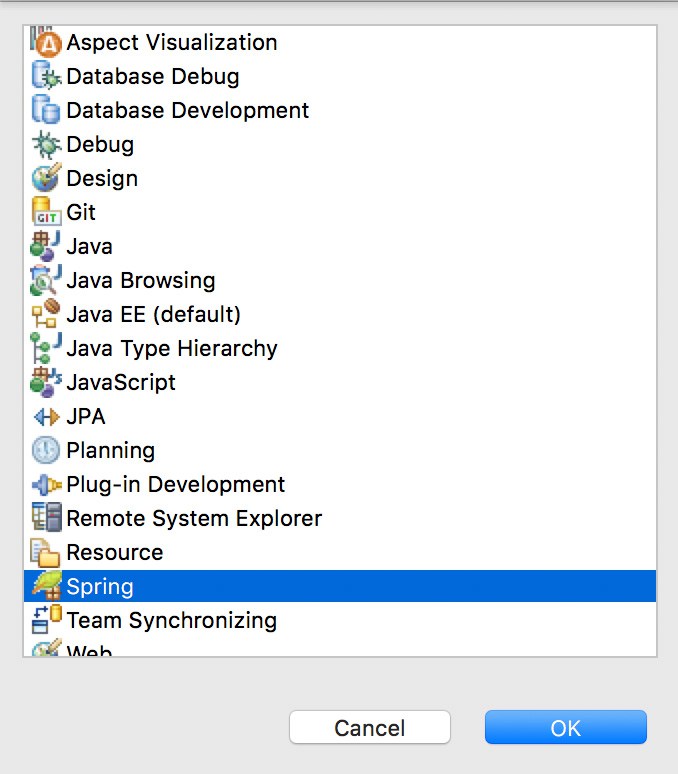
Use the SpringSource Update Site for Eclipse 4.5 update url

http://dist.springsource.com/release/TOOLS/update/e4.5/

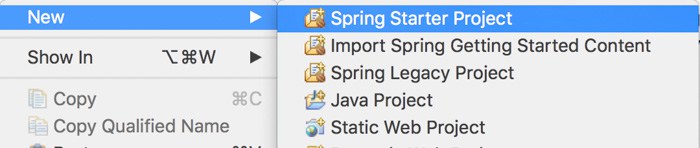
Select and install Spring IDE Core

Creating spring boot application in eclipse mars :

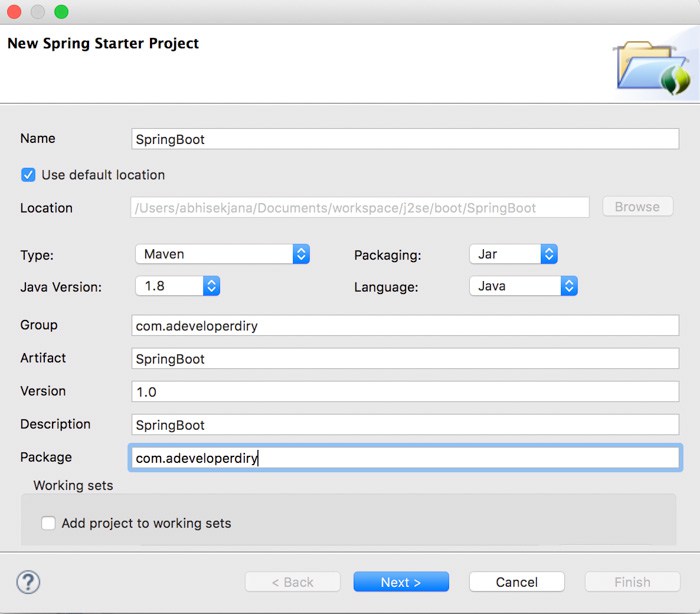
* open Spring perspective.



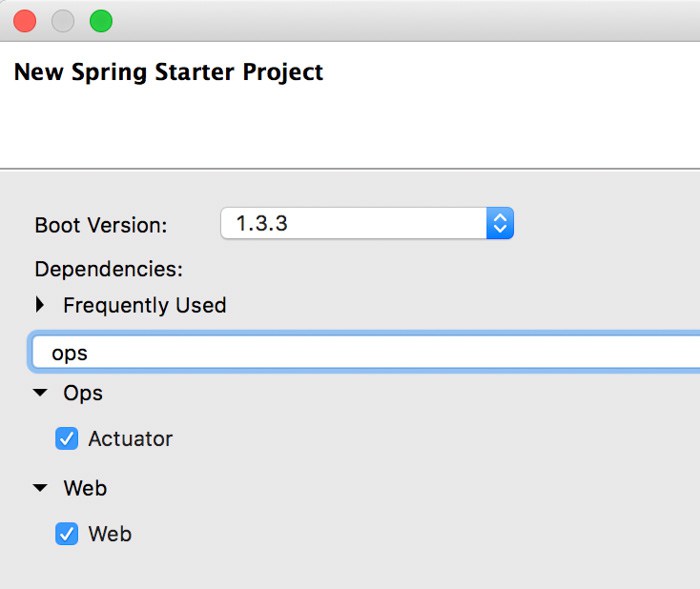
Create a New -> Spring Starter Project.



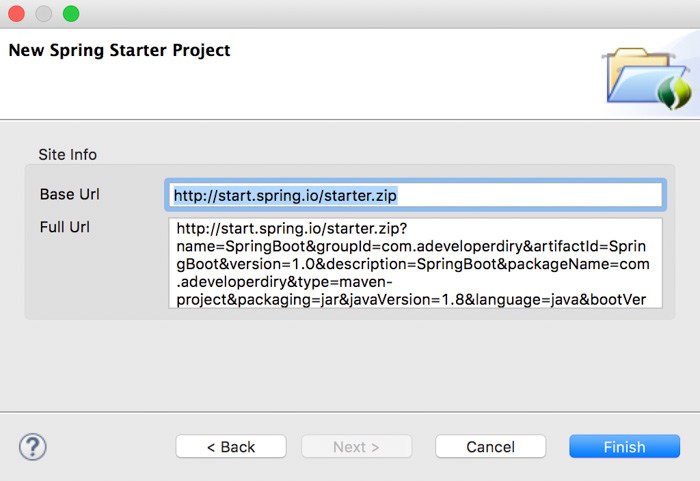
Enter the following details. Click Next.



Select Web and Actuator. Click on Next.



Click on Finish.



This should create the Spring Boot Project in Eclipse.

Running the app :

Mvn spring-boot:run

This will open a web browser to localhost:8080

To gracefully exit the application hit ctrl-c

To create an executable jar we need to add the spring-boot-maven-plugin to our pom.xml

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

Then “your-project”-1.0-snapshot.jar it is around in high size.If you want to peek inside ,you can use jar tvf:

jar tvf target/“your-project”-1.0-snapshot.jar

To run that application ,use the java –jar command:

java –jar target/“your-project”-1.0-snapshot.jar

with Spring Boot, you can create standalone applications that use an embedded server, making them 100% runnable applications.

the SpringApplication singleton class in the main method that will execute the application. The run method call accepts two parameters—the class that actually contains the annotated @SpringBootApplication annotation and the application’s arguments.

Spring Boot has many features that make it suitable for:

• Cloud Native Applications that follow the 12 factor patterns (developed by the Netflix

engineering team at http://12factor.net/ )

• Productivity increases by reducing time of development and deployment

• Enterprise-production-ready Spring applications

• Non-functional requirements, such as the Spring Boot Actuator (a module that

brings metrics, health checks, and management easily) and embedded containers

for running web applications (such as Tomcat, Undertow, Jetty, etc.)

• The SpringApplication class. I showed you that in a Java Spring Boot application,

the main method executes this singleton class. This particular class provides a

convenient way to initiate a Spring application.

• Spring Boot allows you to create applications without requiring any XML

configuration. Spring Boot doesn’t generate code.

• Spring Boot provides a fluent builder API through the SpringApplicationBuilder

singleton class that allows you to create hierarchies with multiple application

contexts.

• The ApplicationArguments interface. Spring Boot allows you to access any

application arguments. This is useful when you want to run your application with

some parameters. For example, you can use --debug mylog.txt or --audit=true

and have access to those values.

• Spring Boot allows you to execute code after the application has started. The only

thing you need to do is implement the CommandLineRunner interface and provide

the implementation of the run(String ...args) method. A particular example is to

initialize some records in a database as it starts or check on some services and see if

they are running before your application starts.

• Spring Boot allows you to externalize configurations by using an

application.properties or application.yml file.

• You can add administration-related features, normally through JMX. You do this simply by enabling the **spring.application.admin.enabled** property in the **application.properties or application.yml files**.

• Spring Boot allows you to have profiles that will help your application run in different environments.

UNIX OSs: Linux, OS X, and Solaris

There are a lot of tools that can help you install the Spring Boot CLI. If you are using any UNIX environment,

including Linux, OS X, or Solaris, you can use a very good tool named SDKMAN. You can find it at

http://sdkman.io/ . Open a terminal window and execute the following:

**$ curl -s get.sdkman.io | bash**

After it finishes, you can execute the following line to run the sdk command:

**$ source "$HOME/.sdkman/bin/sdkman-init.sh"**

Then make sure that the sdk command is working by executing this line:

**$ sdk version**

SDKMAN 3.2.4

Next, it’s time to install the Spring Boot CLI, which you do by executing this command:

**$ sdk install springboot**

Once the CLI is installed, you can check if everything went okay by executing this request:

**$ spring --version**

Spring CLI v1.3.2.RELEASE

You should get the latest version of Spring Boot; in my case it’s release 1.3.2. Now you are ready to start

using the Spring Boot CLI on a UNIX system.

You can use the same s dk command to install Groovy and Gradle. You can install those two by

executing: $ sdk install groovy and $ sdk install gradle .

**$ ruby -e "$(curl -fsSL** [**https://raw.githubusercontent.com/Homebrew/install/master/install**](https://raw.githubusercontent.com/Homebrew/install/master/install)**)"**

You can then execute the following command to install Spring Boot:

**$ brew tap pivotal/tap**

**$ brew install springboot**

**$ ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/linuxbrew/go/install)"**

Then execute the same commands from above:

**$ brew tap pivotal/tap**

**$ brew install springboot**

Windows OS

If you are a Windows user or you don’t want to use the previous methods, you can download the ZIP binary

distribution and uncompress it. These are the links of release 1.3.2:

• http://repo.spring.io/release/org/springframework/boot/spring-bootcli/

1.3.2.RELEASE/spring-boot-cli-1.3.2.RELEASE-bin.zip

• http://repo.spring.io/snapshot/org/springframework/boot/spring-boot-cli/

These links are the binary versions, but if you wonder where those links are coming from, you can

find them here: https://docs.spring.io/spring-boot/docs/current/reference/html/gettingstarted-

installing-spring-boot.html#getting-started-manual-cli-installation . You must have

the JAVA\_HOME variable set (pointing to your Java SDK) and the SPRING\_HOME variable pointing to where

you uncompress the binary distribution. Also make sure to set up your PATH variable, which includes the

% SPRING\_HOME%\bin path (or, if you are using UNIX, it’s $SPRING\_HOME/bin) . By setting these variables to the environment, you will have access to the spring.bat or spring scripts.

**$ spring init --build gradle myapp – this will create gradle based build application structure**

the Spring Boot project structure created when you execute the **spring init** command**. If you want to add more features—such as web, JPA, and Maven projects—you can execute the**

**following command:**

**$ spring init -dweb,data-jpa,h2,thymeleaf --build maven myapp --force**

This command will create a Spring Boot Maven project and will include all the necessary dependencies

in the pom.xml file to run a Spring Boot web application. It will include libraries to handle web files (this will

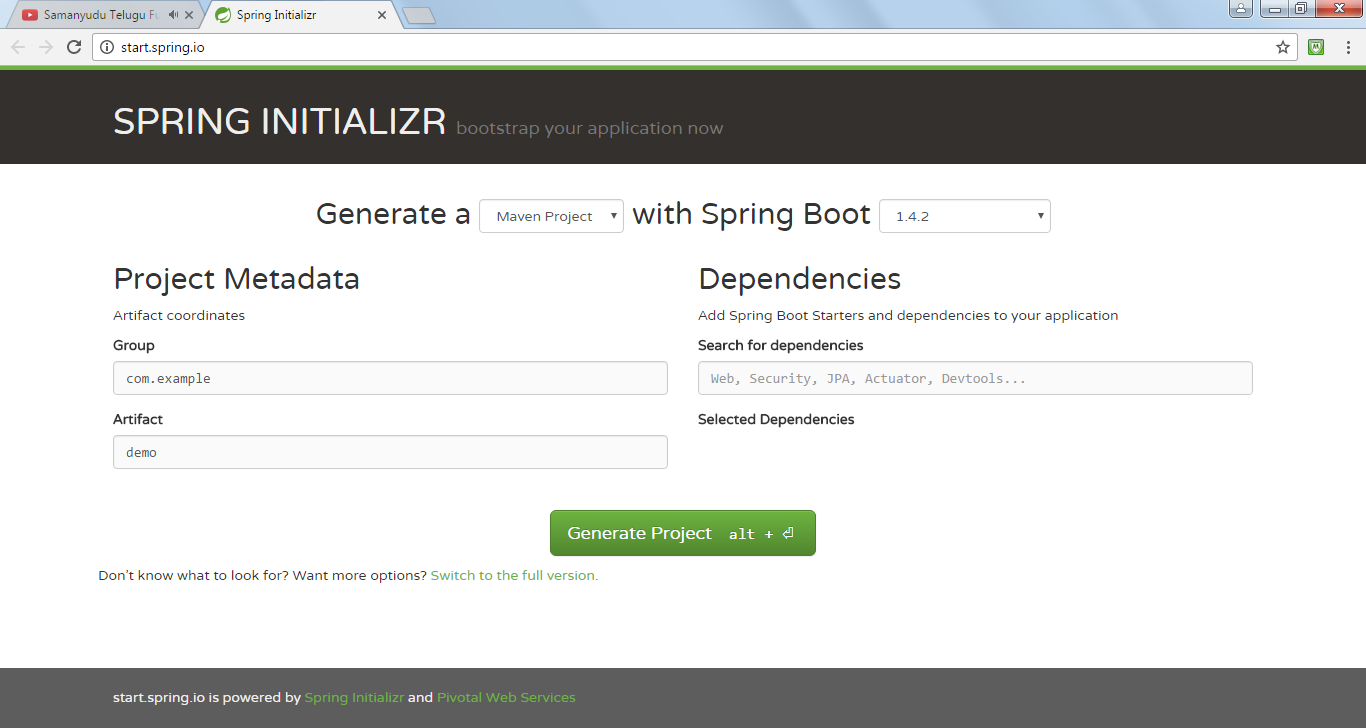
include the embedded Tomcat server), persistence ( data-jpa ), the H2 database engine ( h2 ), and a viewer

engine ( thymeleaf ). **You need to use --force to override the previous myapp directory or you can change**

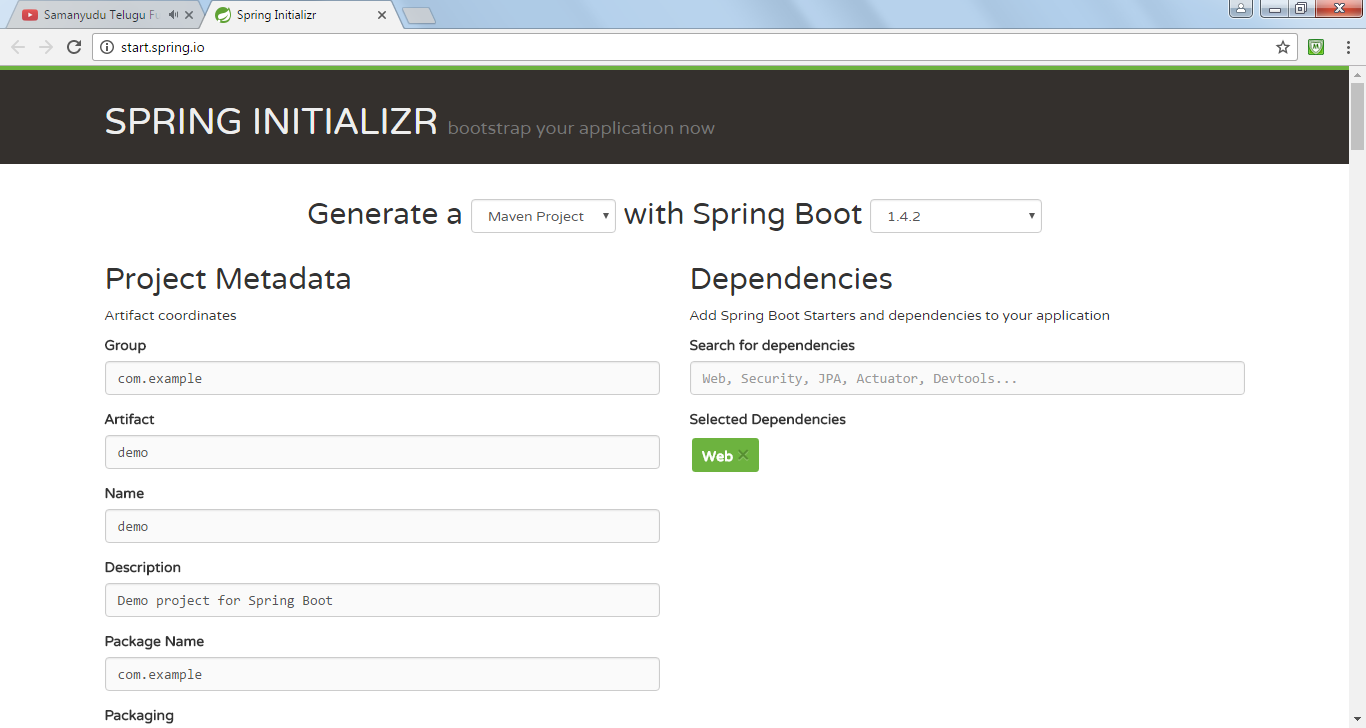
**the name.**

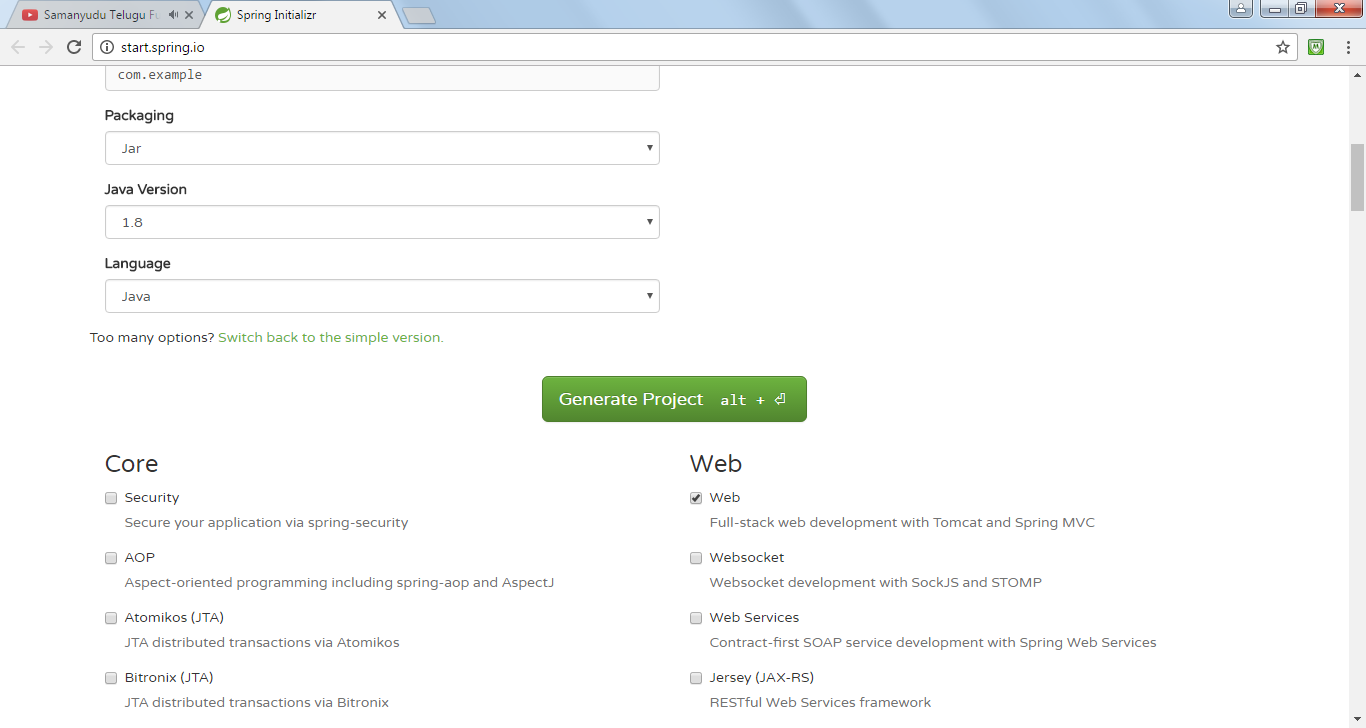
**Spring Initializer :**

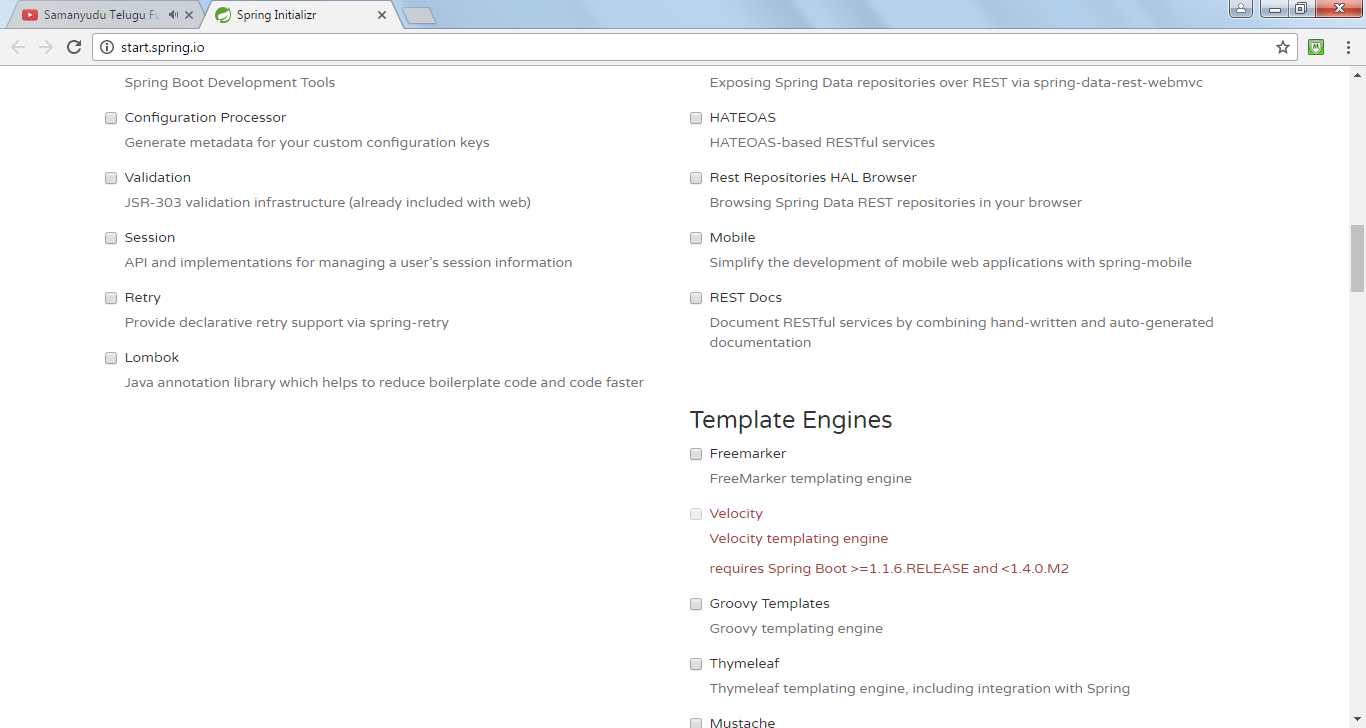
*Simple view of the Spring Initializr ( http://start.spring.io )*

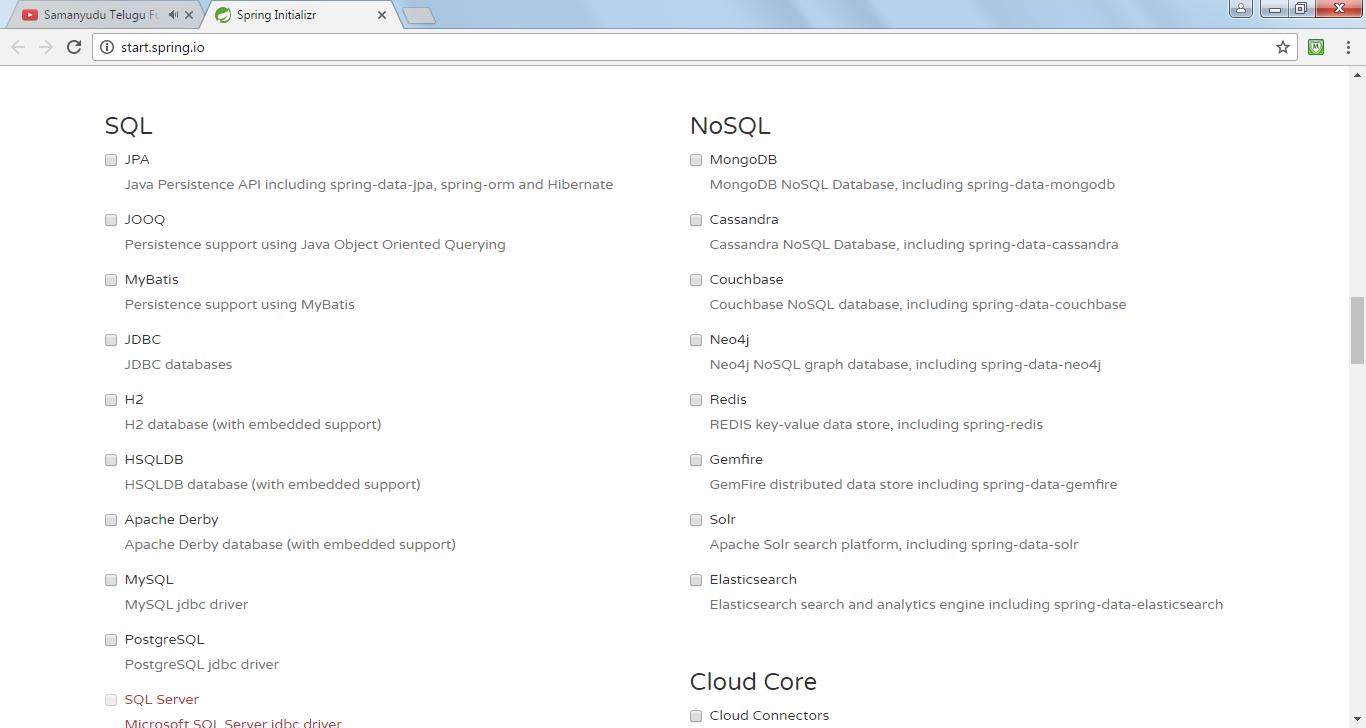
****

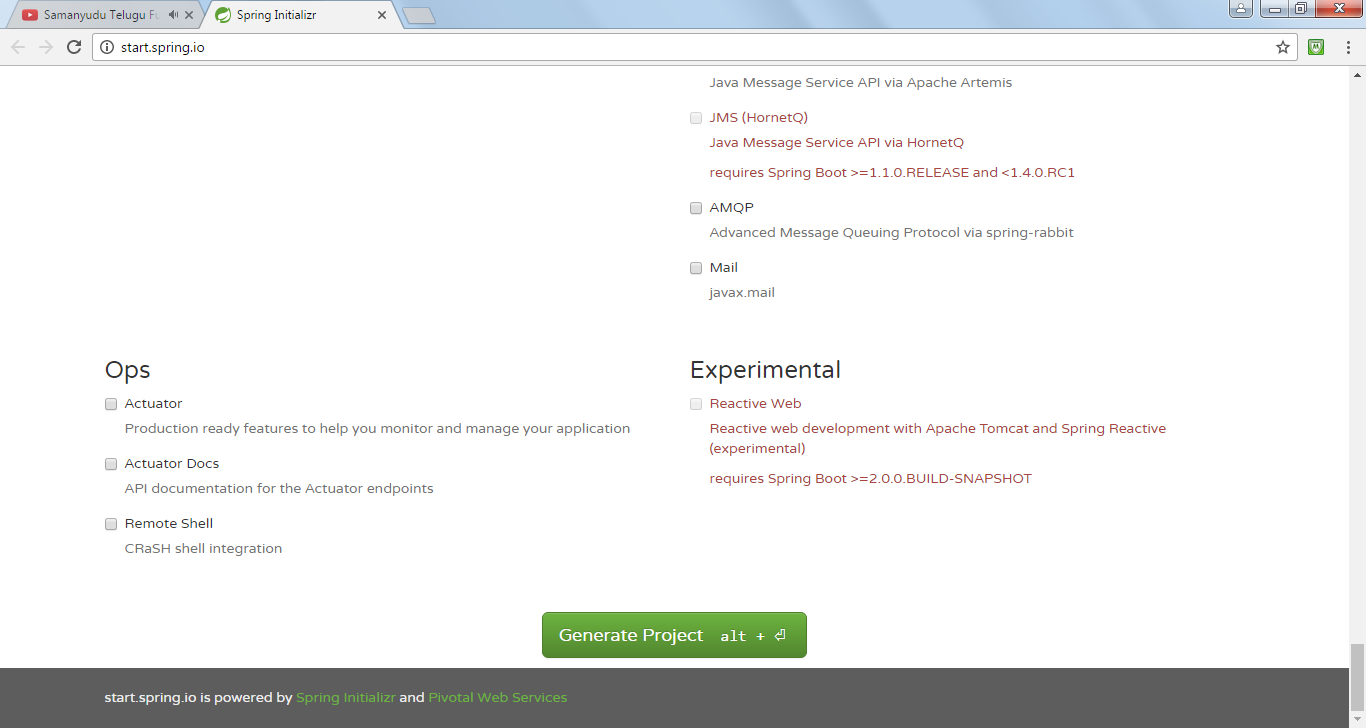
Click on The Switch to the full version











After you select the features you want to use, click the Generate Project button to get the ZIP file that contains your project.

Using the Spring Initializr with UNIX cURL

The Spring Initializr can be accessed using the UNIX cURL command because at the end it is a web service

and it exposes a RESTful API. So, for example, if you wanted to create a simple project that contains just the

minimum files, you could execute the following command:

**$ curl -s https://start.spring.io/starter.zip -o myapp.zip**

**This command will create a myapp.zip** file that contains all the structure for the Spring Boot app. And

by default it contains a Maven project with its pom.xml file and a Maven wrapper.

Spring Boot Using Spring Tool Suite (STS)

If you are already using the Eclipse IDE , you can install the STS as a plugin or download it at

https://spring.io/tools/sts/all