Introduction to DevOps Programming

Bhupinder Singh Pawa

GROUP 5:

Pooja Shah	2020hs70005
Meghna Sen	2020hs70035
Rahul Keluskar	2020hs70025
Gaurav Golchaa	2020hs70036
Sneha Agarwal	2020hs70027

Scope

The main aim of this group project was to gather a fair understanding of what UI testing is, how it works and how the various tools like Selenium, Cucumber, Kubernetes, and Jenkins help to automate UI testing. The scope of this project was also to perform a valid demonstration with the given tools, to ensure complete understanding.

Headings

- What is UI Testing and need for it
- Scope of UI Testing
- Manual vs Automated Testing
- TDD vs. BDD
- What is Selenium and WebDriver
- Working of Selenium RC
- Working and Architecture of Selenium WebDriver
- Selenium Grid and its Deployment
- Demo
- Cucumber and why use with Selenium
- Steps to create the Application
- Demo

Softwares Used:

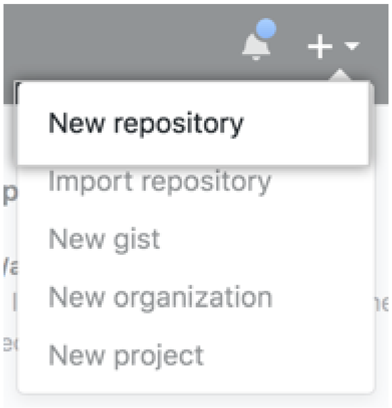
1. Github
2. Maven
3. Selenium
4. Sonarqube
5. Kubernetes
6. Jenkins

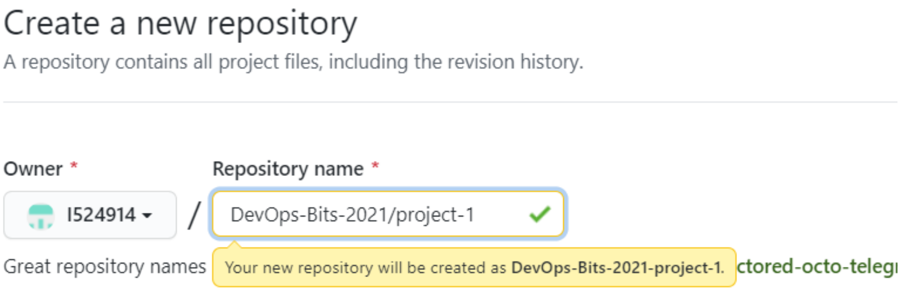
Method/Process

Github

We created the repository using the following steps:

1. We selected the new repository from the upper-right corner and gave the name to our repository





Create a new repository

A repository contains all project files, including the revision history.

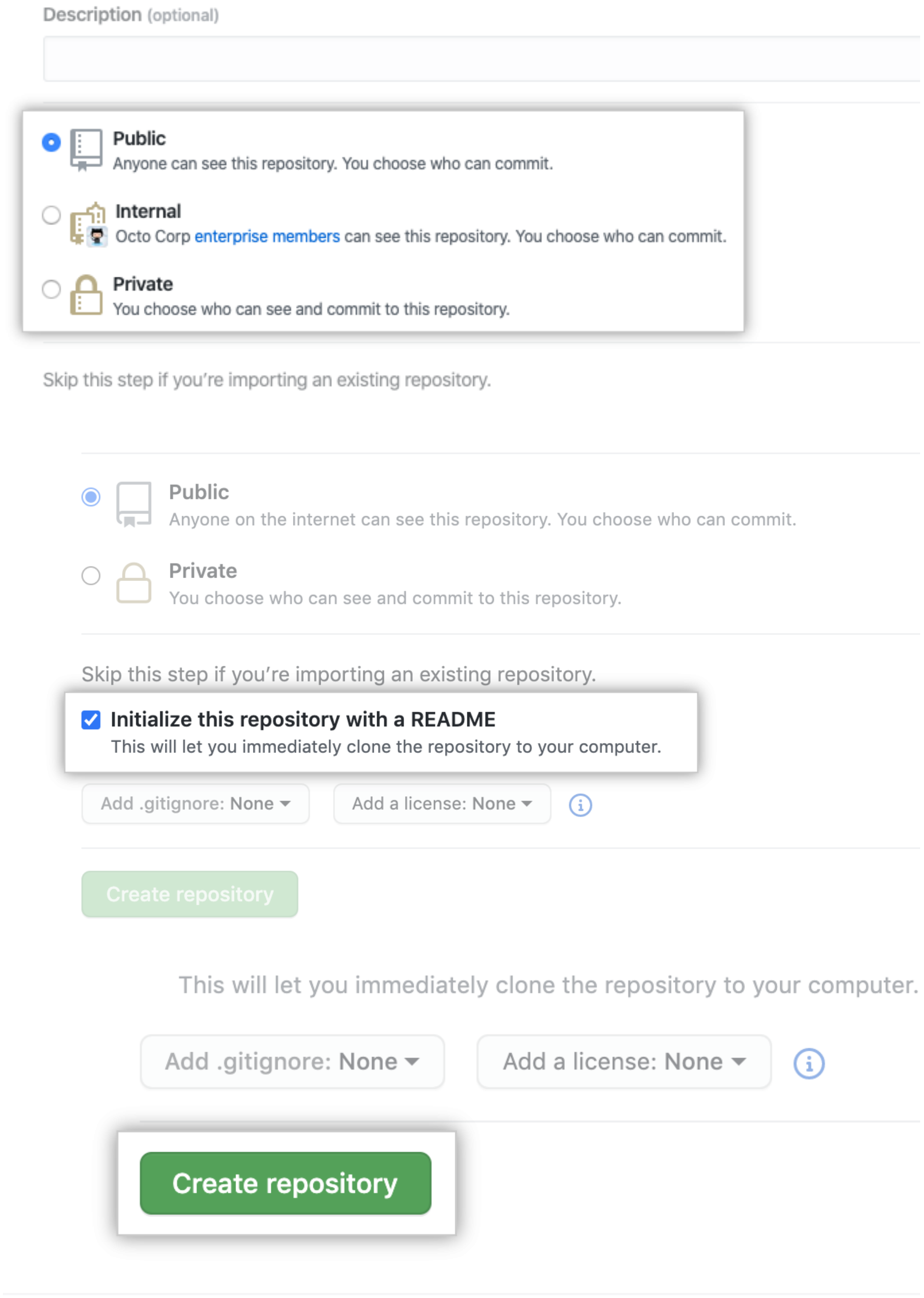
Owner * I524914 / Repository name * DevOps-Bits-2021/project-1 ✓

Great repository names

Your new repository will be created as **DevOps-Bits-2021-project-1.**

ctored-octo-teleg

2. We made the repository publicly visible, and initialized it with a README file.



Description (optional)

● Public
Anyone can see this repository. You choose who can commit.

○ Internal
Octo Corp enterprise members can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☑ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

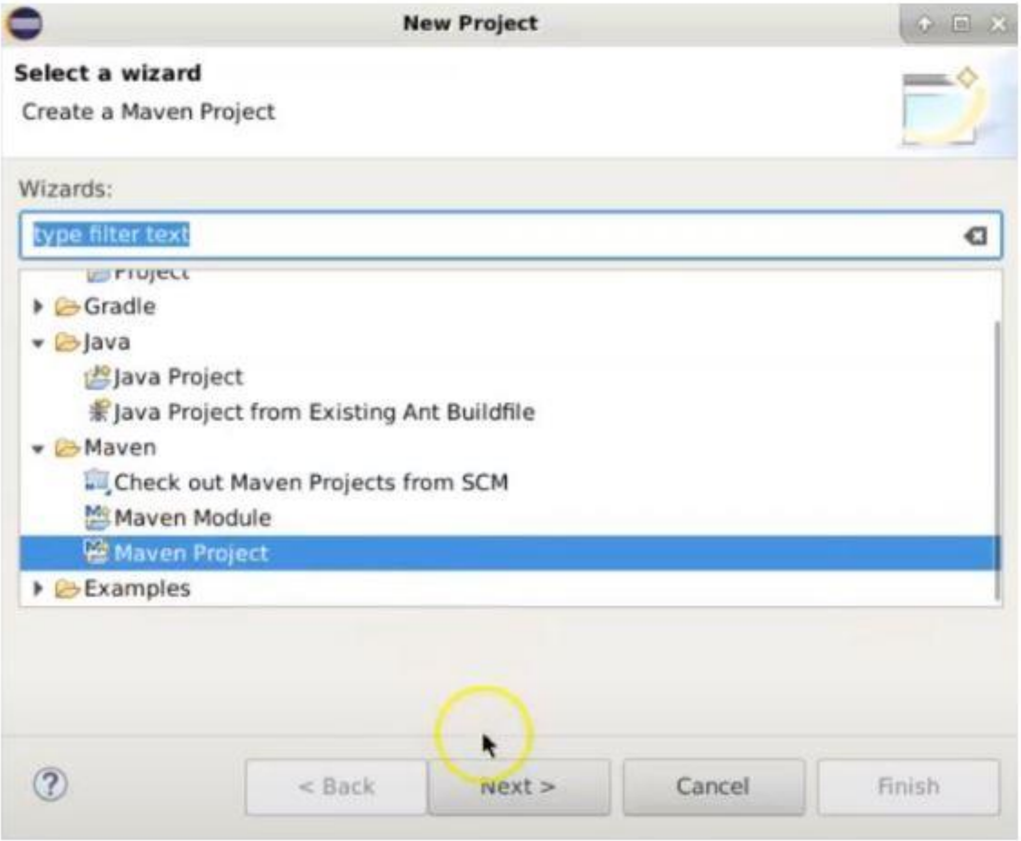
Add .gitignore: None
Add a license: None

This will let you immediately clone the repository to your computer.

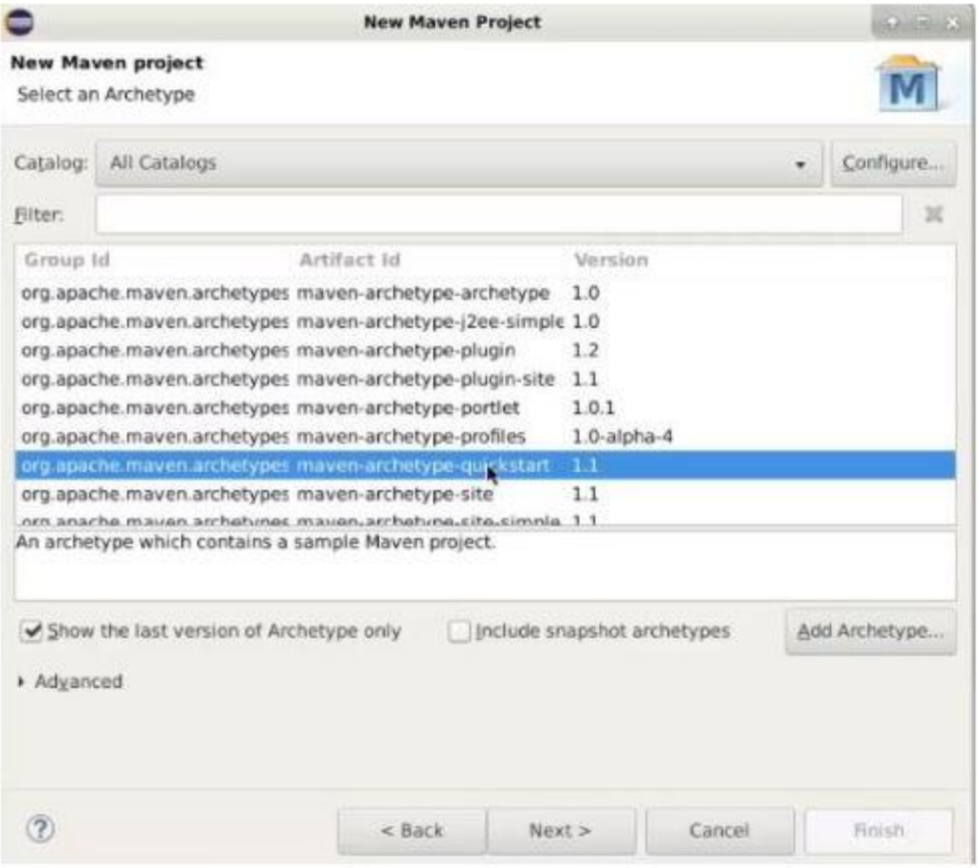
Add .gitignore: None
Add a license: None

Maven Project

We created a Maven project in Eclipse



We then selected a plug-in to add to the project. We then entered the Group ID and the Artifact ID in the next dialog box. After completing the initial installation, the project was ready.



Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-archetype	1.0
org.apache.maven.archetypes	maven-archetype-j2ee-simple	1.0
org.apache.maven.archetypes	maven-archetype-plugin	1.2
org.apache.maven.archetypes	maven-archetype-plugin-site	1.1
org.apache.maven.archetypes	maven-archetype-portlet	1.0.1
org.apache.maven.archetypes	maven-archetype-profiles	1.0-alpha-4
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-site	1.1
org.apache.maven.archetypes	maven-archetype-site-simple	1.1

An archetype which contains a sample Maven project.

☑ Show the last version of Archetype only ☐ Include snapshot archetypes

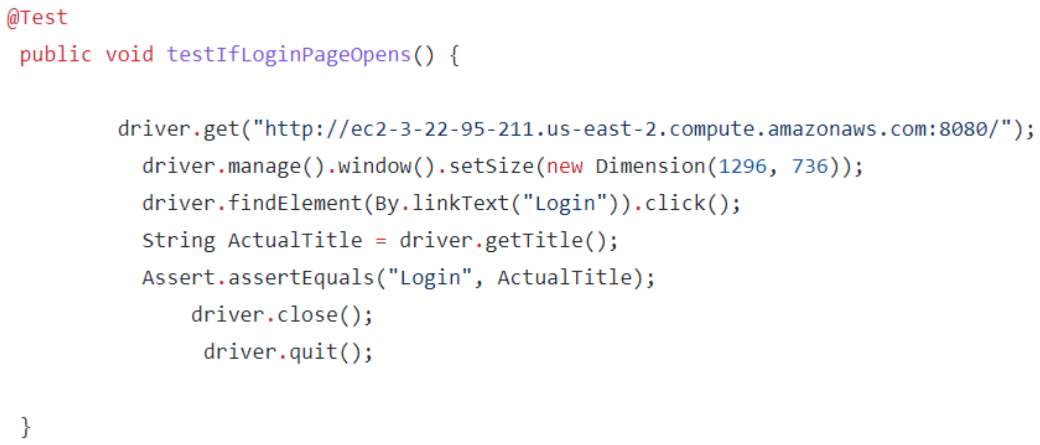
Selenium Web Driver

1. Download of libraries (we used Java, so we downloaded the Java library) and drivers (for the web browser, for interaction)
2. Download of the jar files and addition of them to dependencies in our project
3. Writing of test cases using selenium:
a. Creating an object of the web driver within the class
b. Adding the path of Chrome WebDriver path to project; this helps us to launch the application on our browser

```
public  void setup() {
            System.setProperty("webdriver.chrome.driver", "C:\\Users\\I513949\\Downloads\\chromedriver_win32\\chromedriver.exe");
            driver = new ChromeDriver();
             vars = new HashMap<String, Object>();
              js = (JavascriptExecutor) driver;

}
```

c. Testcase #1: Navigation: Navigates to a website, selects the login button and prints the title.

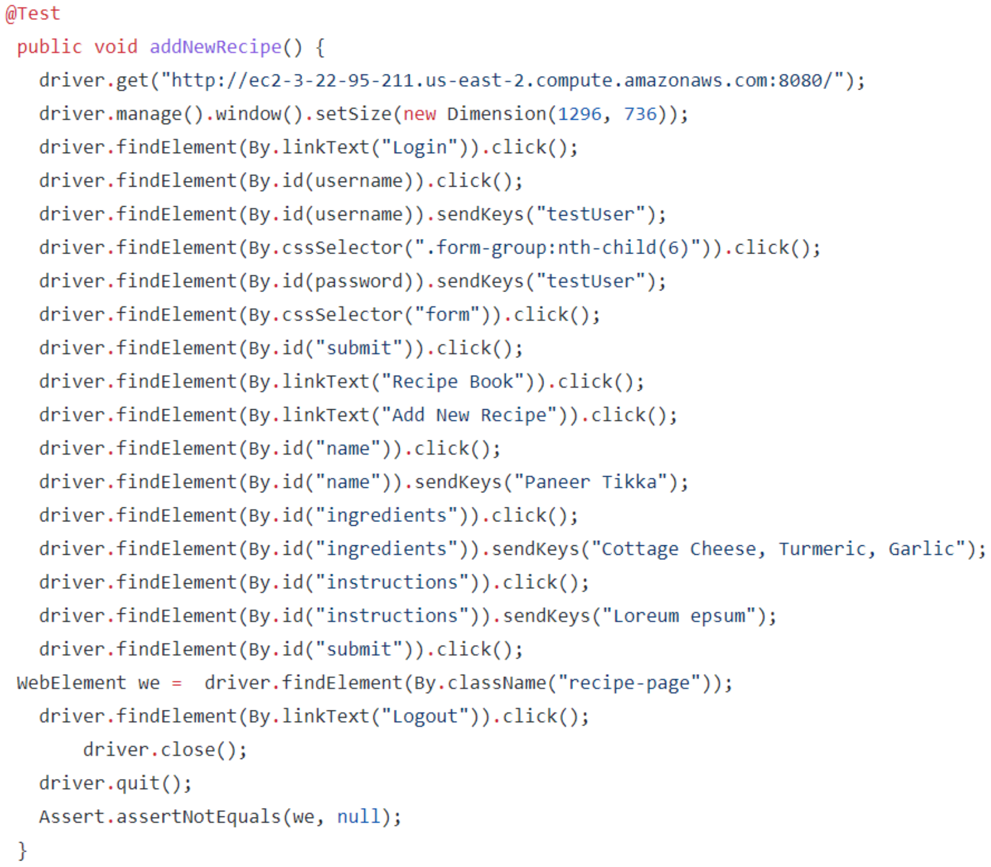


```
@Test
 public void testIfLoginPageOpens() {

        driver.get("http://ec2-3-22-95-211.us-east-2.compute.amazonaws.com:8080/");
          driver.manage().window().setSize(new Dimension(1296, 736));
          driver.findElement(By.linkText("Login")).click();
          String ActualTitle = driver.getTitle();
          Assert.assertEquals("Login", ActualTitle);
              driver.close();
               driver.quit();

 }
```

d. Testcase #2: Add a new recipe



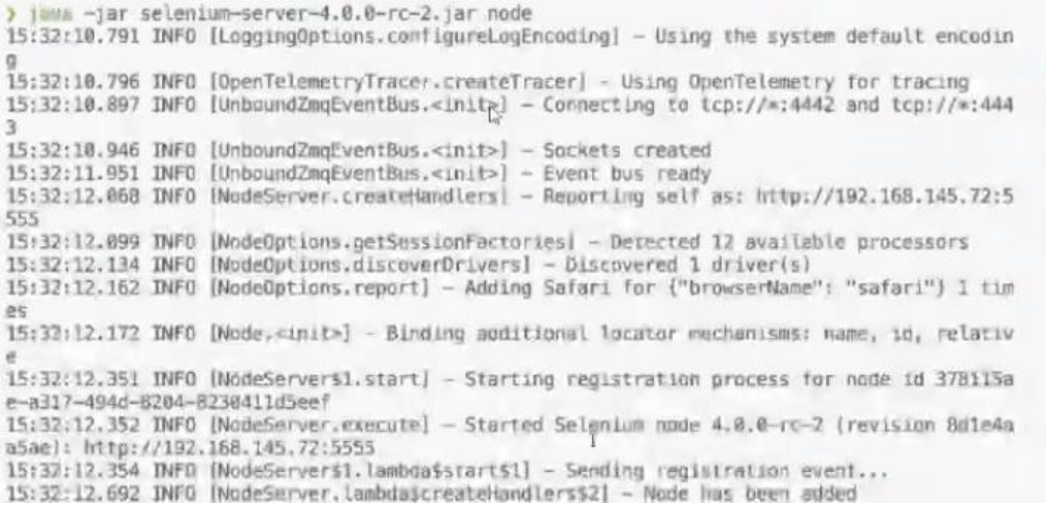
```
@Test
 public void addNewRecipe() {
   driver.get("http://ec2-3-22-95-211.us-east-2.compute.amazonaws.com:8080/");
   driver.manage().window().setSize(new Dimension(1296, 736));
   driver.findElement(By.linkText("Login")).click();
   driver.findElement(By.id(username)).click();
   driver.findElement(By.id(username)).sendKeys("testUser");
   driver.findElement(By.cssSelector(".form-group:nth-child(6)")).click();
   driver.findElement(By.id(password)).sendKeys("testUser");
   driver.findElement(By.cssSelector("form")).click();
   driver.findElement(By.id("submit")).click();
   driver.findElement(By.linkText("Recipe Book")).click();
   driver.findElement(By.linkText("Add New Recipe")).click();
   driver.findElement(By.id("name")).click();
   driver.findElement(By.id("name")).sendKeys("Paneer Tikka");
   driver.findElement(By.id("ingredients")).click();
   driver.findElement(By.id("ingredients")).sendKeys("Cottage Cheese, Turmeric, Garlic");
   driver.findElement(By.id("instructions")).click();
   driver.findElement(By.id("instructions")).sendKeys("Loreum epsum");
   driver.findElement(By.id("submit")).click();
 WebElement we =  driver.findElement(By.className("recipe-page"));
   driver.findElement(By.linkText("Logout")).click();
       driver.close();
   driver.quit();
   Assert.assertNotEquals(we, null);
 }
```

4. Running the above test cases and checking if they are running successfully

Kubernetes and Jenkins

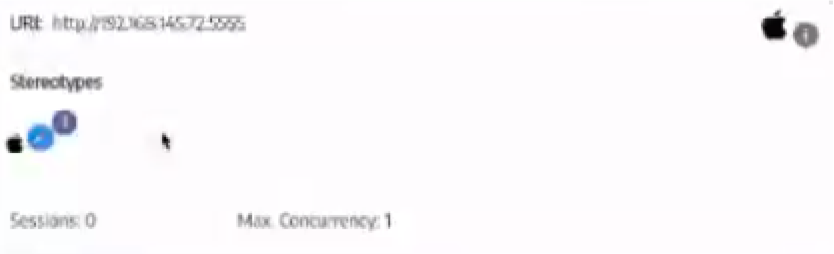
Selenium stand alone jar file was used to make an installation of the Selenium server – we chose it because it is easy as only requires identification as parameter.

We started a node using the following command:



```
> java -jar selenium-server-4.0.0-rc-2.jar node
15:32:10.791 INFO [LoggingOptions.configureLogEncoding] - Using the system default encodin
g
15:32:10.796 INFO [OpenTelemetryTracer.createTracer] - Using OpenTelemetry for tracing
15:32:10.897 INFO [UnboundZmqEventBus.<init>] - Connecting to tcp://*:4442 and tcp://*:444
3
15:32:10.946 INFO [UnboundZmqEventBus.<init>] - Sockets created
15:32:11.951 INFO [UnboundZmqEventBus.<init>] - Event bus ready
15:32:12.068 INFO [NodeServer.createHandlers] - Reporting self as: http://192.168.145.72:5
555
15:32:12.099 INFO [NodeOptions.getSessionFactories] - Detected 12 available processors
15:32:12.134 INFO [NodeOptions.discoverDrivers] - Discovered 1 driver(s)
15:32:12.162 INFO [NodeOptions.report] - Adding Safari for {"browserName": "safari"} 1 tim
es
15:32:12.172 INFO [Node.<init>] - Binding additional locator mechanisms: name, id, relativ
e
15:32:12.351 INFO [NodeServer$1.start] - Starting registration process for node id 378115a
e-a317-494d-8204-8230411d5eef
15:32:12.352 INFO [NodeServer.execute] - Started Selenium node 4.0.0-rc-2 (revision 8d1e4a
a5ae): http://192.168.145.72:5555
15:32:12.354 INFO [NodeServer$1.lambda$start$1] - Sending registration event...
15:32:12.692 INFO [NodeServer.lambda$createHandlers$2] - Node has been added
```

The node is then added:

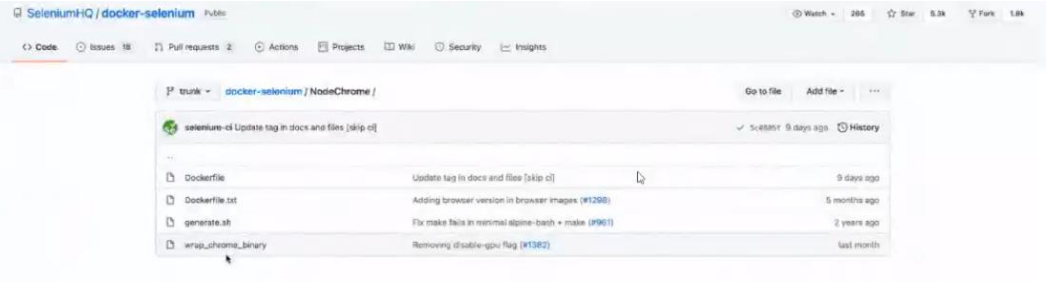


URI: http://192.168.145.72:5555

Stereotypes

Sessions: 0   Max. Concurrency: 1

Also, we used this repository docker-selenium to make a Kubernetes deployment of the Selenium hub.



docker-selenium / NodeChrome /

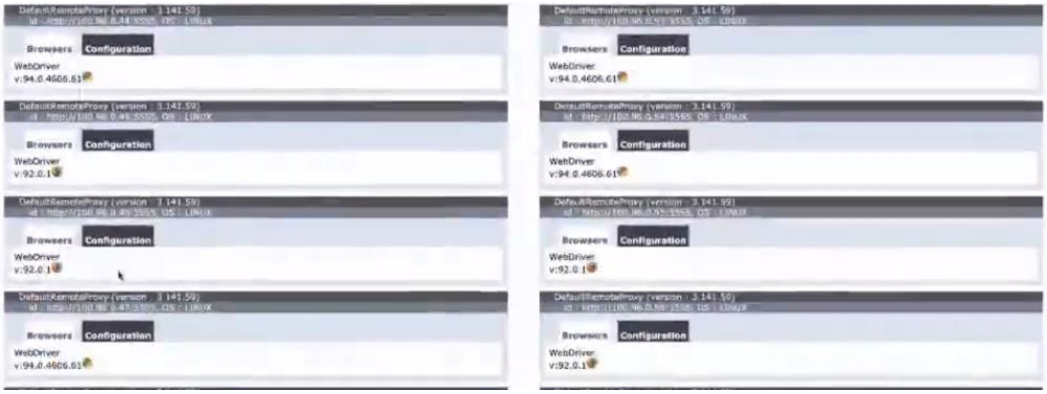
selenium-ci Update tag in docs and files [skip ci]   5c4885f 9 days ago

..		
Dockerfile	Update tag in docs and files [skip ci]	9 days ago
Dockerfile.txt	Adding browser version in browser images (#1298)	5 months ago
generate.sh	Fix make fails in minimal alpine-bash + make (#961)	2 years ago
wrap_chrome_binary	Removing disable-gpu flag (#1382)	last month

We used a Liveness Probe to check if the hub is still on or needs to be scaled or is overloaded.

We have also used LoadBalancer service which generates an external IP that can be used over the internet.

Deployed Grid: (from the Grid Console)



DefaultRemoteProxy (version : 3.141.59)
id : http://100.96.0.44:5555, OS : LINUX
Browsers Configuration
WebDriver
v:94.0.4606.61

DefaultRemoteProxy (version : 3.141.59)
id : http://100.96.0.46:5555, OS : LINUX
Browsers Configuration
WebDriver
v:92.0.1

DefaultRemoteProxy (version : 3.141.59)
id : http://100.96.0.49:5555, OS : LINUX
Browsers Configuration
WebDriver
v:92.0.1

DefaultRemoteProxy (version : 3.141.59)
id : http://100.96.0.47:5555, OS : LINUX
Browsers Configuration
WebDriver
v:94.0.4606.61

DefaultRemoteProxy (version : 3.141.59)
id : http://100.96.0.53:5555, OS : LINUX
Browsers Configuration
WebDriver
v:94.0.4606.61

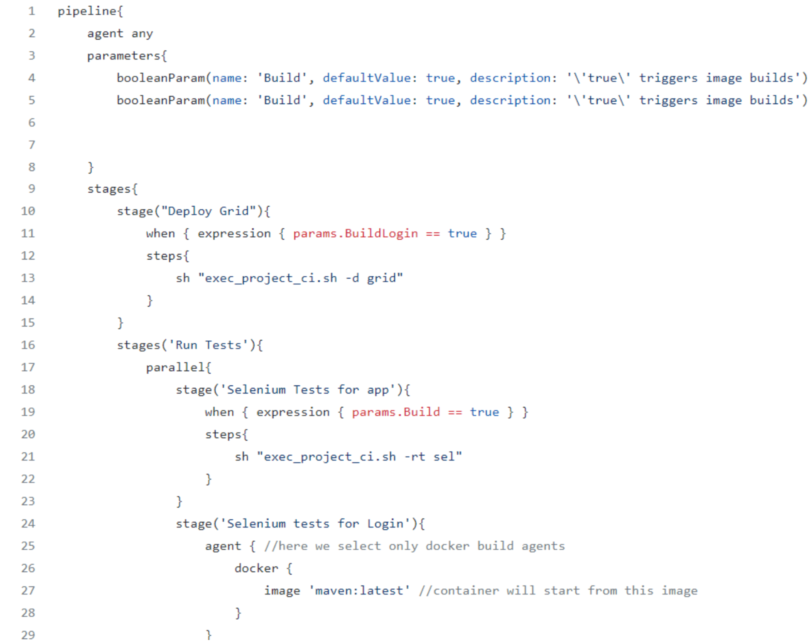
DefaultRemoteProxy (version : 3.141.59)
id : http://100.96.0.54:5555, OS : LINUX
Browsers Configuration
WebDriver
v:94.0.4606.61

DefaultRemoteProxy (version : 3.141.59)
id : http://100.96.0.55:5555, OS : LINUX
Browsers Configuration
WebDriver
v:92.0.1

DefaultRemoteProxy (version : 3.141.59)
id : http://100.96.0.58:5555, OS : LINUX
Browsers Configuration
WebDriver
v:92.0.1

Jenkins: which gave us an overview of the pipeline

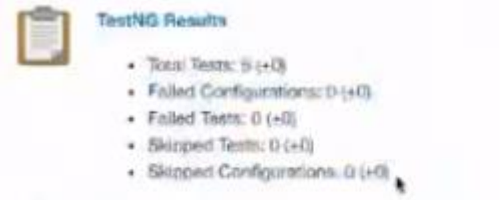
Code:



```
pipeline{
    agent any
    parameters{
        booleanParam(name: 'Build', defaultValue: true, description: '\'true\' triggers image builds')
        booleanParam(name: 'Build', defaultValue: true, description: '\'true\' triggers image builds')


    }
    stages{
        stage("Deploy Grid"){
            when { expression { params.BuildLogin == true } }
            steps{
                sh "exec_project_ci.sh -d grid"
            }
        }
        stages('Run Tests'){
            parallel{
                stage('Selenium Tests for app'){
                    when { expression { params.Build == true } }
                    steps{
                        sh "exec_project_ci.sh -rt sel"
                    }
                }
                stage('Selenium tests for Login'){
                    agent { //here we select only docker build agents
                        docker {
                            image 'maven:latest' //container will start from this image
                        }
                    }
```

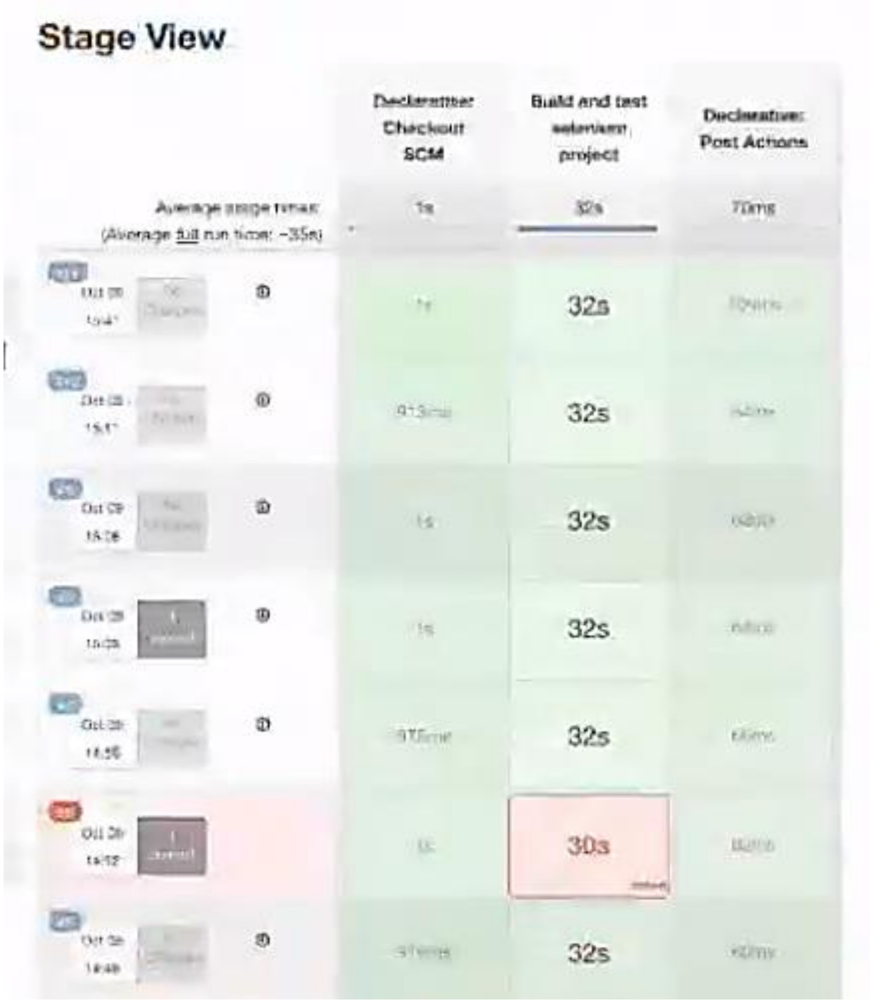
Jenkins report:



TestNG Results

- Total Tests: 5 (+0)
- Failed Configurations: 0 (+0)
- Failed Tests: 0 (+0)
- Skipped Tests: 0 (+0)
- Skipped Configurations: 0 (+0)

Within the branch:



Stage View

	Declarative: Checkout SCM	Build and test selenium project	Declarative: Post Actions
Average stage times: (Average full run time: ~35s)	1s	32s	70ms
[illegible]	1s	32s	[illegible]
[illegible]	913ms	32s	[illegible]
[illegible]	1s	32s	[illegible]
[illegible]	1s	32s	[illegible]
[illegible]	975ms	32s	[illegible]
[illegible]	[illegible]	30s	[illegible]
[illegible]	[illegible]	32s	[illegible]

Critique of the Project

Devops practices followed in the team:

1. Building a collaborative culture which improved coordination between members and promoted teamwork. All the member took full ownership of their individual components in the development process and worked together to complete it. Also, the development and operations were part of everyone's responsibilities irrespective of the specific tasks.

2. Use of agile project management: The team focused on getting feedback and modifying the requirements or solution based on it, thereby adapting and responding to those changes very efficiently. This immediately resulted in a faster deployment.

3. Adoption of continuous integration and delivery (CI/CD):

CI: The team has focused on integrating code changes from the different contributors into one central repository. This allowed us to merge code changes frequently and use automated tools to test the code's correctness. This also made it simpler to detect errors and other qualities issues on tinier amounts of code, rather than a large code set.

CD: The deployment of the software code was done in smaller, more convenient parts, so that they became painless, low-risk events that can be performed at any time. The deployment was hence reduced to simple, predictable activities that can be performed on-demand.

4. Monitoring the right metrics
To measure the effectiveness of a DevOps approach, we tracked the project's performance metrics, such as lead time, mean time to detect, and issue severity. This enabled us to easily monitor when something went wrong with the project.

5. Using the right tools
For our project, and its requirements, we chose the right tools to work with so that the integration, deployment and monitoring went about smoothly.

Challenges and Gaps

1. Complying with all rules of DevOps
2. Monitoring of the pipeline, checking up on bug fixes and sonar issues
3. Continuous and multiple commits leading to conflicts occasionally
4. Work culture of everyone was different as we were all from different teams
5. The whole project was done virtually, hence there was some difficulty in coordination

Enhancements to current practices

1. More effective and regular monitoring of the pipeline
2. Fixing the issues arising immediately, and trying to run multiple tests to prevent the issues in the first place

3. Synchronizing commits amongst team members, and making sure every time someone commits a PR, they check the sonar issues
4. Discussing a base, common work culture amongst the team so the team environment is comfortable and same for everyone

References:

- Ui testing: https://www.browserstack.com/guide/ui-testing-guide
- Tdd vs bdd: https://www.softwaretestinghelp.com/tdd-vs-bdd/
- Selenium alternatives: https://www.scnsoft.com/software-testing/selenium-alternatives
- Webdriver: https://www.browserstack.com/guide/selenium-webdriver-tutorial
- Grid: https://www.youtube.com/watch?v=U1jJsV_YKC8
- Getting started with selenium for chrome: https://sites.google.com/a/chromium.org/chromedriver/getting-started

THE PROCESS

HEADING FIVE

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

HEADING FIVE

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book

Lorem Ipsum is simply dummy text of the printing and typesetting industry.