

## Cloud session 19/03/2022

### Let's start

→ Created a new Maven project

→ Ibrahim's teacher shared the pom .xml file with us (below are the dependencies just as a reference)

<dependencies>

```
<dependency>
  <groupId>io.rest-assured</groupId>
  <artifactId>rest-assured</artifactId>
  <version>4.3.0</version>
</dependency>
<dependency>
  <groupId>org.json</groupId>
  <artifactId>json</artifactId>
  <version>20190722</version>
</dependency>
<dependency>
  <groupId>org.testng</groupId>
  <artifactId>testng</artifactId>
  <version>7.1.0</version>
</dependency>
<dependency>
  <groupId>com.google.code.gson</groupId>
  <artifactId>gson</artifactId>
  <version>2.8.6</version>
</dependency>
<dependency>
  <groupId>org.codehaus.jackson</groupId>
  <artifactId>jackson-mapper-asl</artifactId>
  <version>1.9.13</version>
</dependency>
<dependency>
  <groupId>org.seleniumhq.selenium</groupId>
  <artifactId>selenium-java</artifactId>
  <version>3.141.59</version>
</dependency>
<dependency>
```

```

    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.13</version>
</dependency>

<!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
<dependency>
    <groupId>io.github.bonigarcia</groupId>
    <artifactId>webdrivermanager</artifactId>
    <version>4.2.2</version>
</dependency>

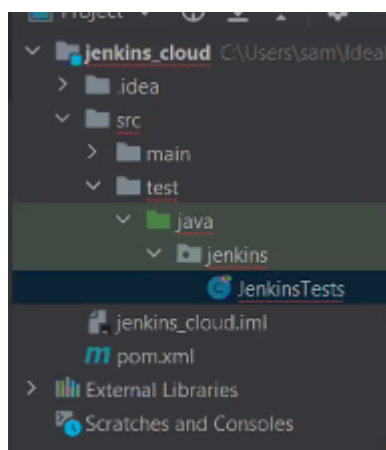
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-surefire-plugin</artifactId>
            <version>3.0.0-M5</version>
        </plugin>
    </plugins>
</build>

```

→ we created a “jenkins” package under the java folder under test

→ we created a “JenkinsTests” class in this package



→ We started typing in the “JenkinsTests” class...

Once ready there, we opened our AWS - as a root user. Then →

Q: Hocam in a team Do all members of the Team use the Jenkins or Aws or just Lead?

A: Everybody uses it.

We created a new Instance → Instances → Launch New Instance → Ubuntu →  
Then:

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

### Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All Instance families Current generation Show/Hide Columns

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GiB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	t2	t2.micro <small>Free tier eligible</small>	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.large	2	8	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.xlarge	4	16	EBS only	-	Moderate	Yes
<input type="checkbox"/>	t2	t2.2xlarge	8	32	EBS only	-	Moderate	Yes

Cancel Previous **Review and Launch** Next: Configure Instance Details

→ Select the free tier

→

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

### Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances: 1 Launch into Auto Scaling Group

Purchasing option: ☐ Request Spot Instances

Network: vpc-037abe3ef5695fb07 (default) Create new VPC

Subnet: No preference (default subnet in any Availability Zone) Create new subnet

Auto-assign Public IP: Use subnet setting (Enable)

Hostname type: Use subnet setting (IP name)

DNS Hostname: ☒ Enable IP name IPv4 (A record) DNS requests  
☒ Enable resource-based IPv4 (A record) DNS requests  
☐ Enable resource-based IPv6 (AAAA record) DNS requests

Placement group: ☐ Add instance to placement group

Capacity Reservation: Open

Domain join directory: No directory

Cancel Previous **Review and Launch** Next: Add Storage

→ Then we just make the size 16

→ Give a name → Then click on the grey Configure button

## Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group  
☐ Select an existing security group

Security group name:

Description:

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	My IP 75.177.25.57/32	e.g. SSH for Admin Desktop
Custom TCP	TCP	8080	My IP 75.177.25.57/32	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom 0.0.0.0/0, ::0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Custom 0.0.0.0/0, ::0	e.g. SSH for Admin Desktop

Add Rule

[Cancel](#) [Previous](#) [Review and Launch](#)

→ Review and Launch

→ Choose an existing key pair or create a new one

→ Then we are done - it can say pending, refresh it and it will be done

For Mac users →


- Open the terminal
- Find the location of your Jenkins file - We type `cd` and space and drag and drop it to the terminal and delete the file extension, so we're in the right location, hit enter
- Then we type `chmod 400 filename.pem`



→ Now back to AWS → on our instance we click on connect:

We see this page when we click on SSH client:


[EC2 Instance Connect](#) | [Session Manager](#) | [SSH client](#) | [EC2 Serial Console](#)

Instance ID

 i-08bd91efa001b5848 (JenkinsNew)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is JenkinsFile.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.  
 `chmod 400 JenkinsFile.pem`
4. Connect to your instance using its Public DNS:  
 `ec2-3-95-18-139.compute-1.amazonaws.com`

Example:

 `ssh -i "JenkinsFile.pem" ubuntu@ec2-3-95-18-139.compute-1.amazonaws.com`

**Note:** In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

→ We copy the ssh bit in Example: and then put it in our terminal and hit enter, it should show something like:

```
1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-31-217:~$
```

→ Then we need to update our tools, type these comment on the terminal and hit the enter

## 1. Update and get ready all tools and packages on Ubuntu Server

- ▶ `sudo apt-get update`
- ▶ `sudo apt-get upgrade`

First - “sudo apt-get update”, then hit enter

Second - “sudo apt-get upgrade”, then hit enter

→ Give it some time to work

→ Next - we need to install the libraries, continue on the terminal:

First command: “sudo apt-get install -y libappindicator1 fonts-liberation” and hit enter

Second command: “wget

[https://dl.google.com/linux/direct/google-chrome-stable\\_current\\_amd64.deb](https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb)” and hit enter

Third command: “sudo dpkg -i google-chrome\*.deb” and hit enter

Forth command: “sudo apt-get install -f” and hit enter

Fifth command: we rerun this one “sudo dpkg -i google-chrome\*.deb” and hit enter

## 1. Install latest Chrome Binary on Ubuntu Server

- ▶ `sudo apt-get install -y libappindicator1 fonts-liberation`
- ▶ `wget https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb`
- ▶ `sudo dpkg -i google-chrome*.deb`
- ▶ `sudo apt-get install -f`

→ we go to google → search for “chromedriver download” and needs to be the same as ours, we need this:

<https://chromedriver.storage.googleapis.com/index.html?path=99.0.4844.51/>

## 2. Install latest Chrome Driver on Ubuntu Server

- ▶ `sudo apt install unzip`
- ▶ `wget https://chromedriver.storage.googleapis.com/99.0.4844.35/chromedriver_linux64.zip`
- ▶ `unzip chromedriver_linux64.zip`
- ▶ `sudo mv chromedriver /usr/bin/chromedriver`

→ Go back to terminal and put the command “sudo apt install unzip” and hit the enter

→ Then this command: “wget

[https://chromedriver.storage.googleapis.com/99.0.4844.35/chromedriver\\_linux64.zip](https://chromedriver.storage.googleapis.com/99.0.4844.35/chromedriver_linux64.zip)

and hit enter

→ Then put this command “unzip chromedriver\_linux64.zip” and hit enter

→ Then put this command “sudo mv chromedriver /usr/bin/chromedriver” and hit enter

Nex stepst:

- 1) We’re installing Java now → put and hit enter “sudo apt-get install default-jdk”
- 2) We’re installing Maven → put and hit enter “sudo apt install maven”
- 3) We can install git (we have it already but just in case) → “sudo apt install git”

# Main tool installations

- ▶ `sudo apt-get install default-jdk`
- ▶ `sudo apt install maven`
- ▶ `sudo apt install git`

Q: Do we need to install it every time in our instances?

A: We install them once, and use them again and again.

Now Jenkins installation:

- 1) Type this and hit enter → `wget -q -O -`

`https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -` - we need to see OK message

- 2) Next: copy exactly as it is, even the single code →

`sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \n/etc/apt/sources.list.d/jenkins.list'`

- 3) Updating our tools with this one → `sudo apt-get update`

- 4) Final comment → `sudo apt-get install jenkins`

→ Next - we go to our Instance:

EC2 > Instances > i-08bd91efa001b5848

**Instance summary for i-08bd91efa001b5848 (JenkinsNew)** [Info](#)



Updated less than a minute ago

Instance ID i-08bd91efa001b5848 (JenkinsNew)	Public IPv4 address 3.95.18.139   <a href="#">open address</a>	Private IPv4 addresses 172.31.31.217
IPv6 address -	Instance state <span>Running</span>	Public IPv4 DNS ec2-3-95-18-139.compute-1.amazonaws.com   <a href="#">open address</a>
Hostname type IP name: ip-172-31-31-217.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-31-217.ec2.internal	Answer private resource DNS name IPv4 (A)
Instance type t2.micro	Elastic IP addresses -	VPC ID vpc-037abe3ef5695fb07
AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. <a href="#">Learn more</a>	IAM Role -	Subnet ID subnet-0f43e78aa5a0c1735

[Details](#) | [Security](#) | [Networking](#) | [Storage](#) | [Status checks](#) | [Monitoring](#) | [Tags](#)

▼ Instance details [Info](#)

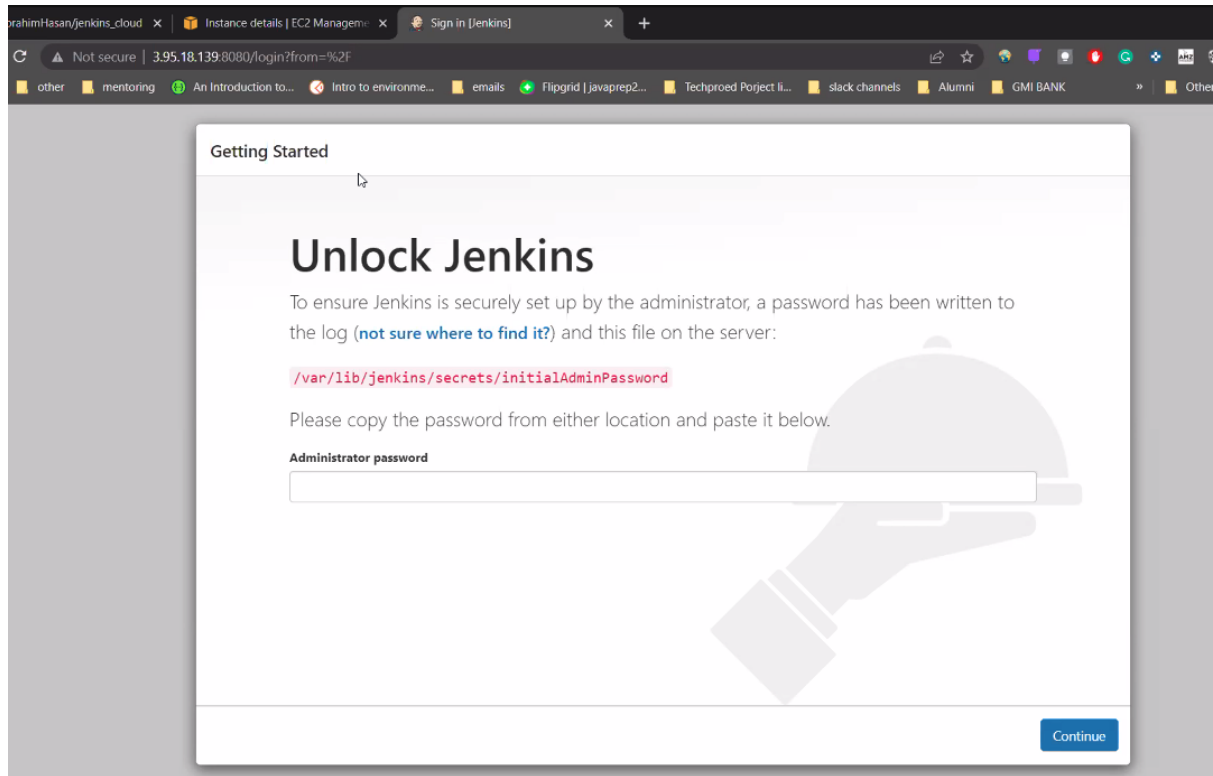
Public IPv4 address

 **3.95.18.139** | [open address](#) 

→ Public IP address → copy it

and paste it to your browser and add the :8080 to it, then hit enter

→ We should see this page:



→ Then we copy this part:

# Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below

→ then add this bit and copy this to the terminal → `sudo cat`

`/var/lib/jenkins/secrets/initialAdminPassword`


→ You'll get your pass on the terminal - then copy and paste it to the pass space on the Unlock Jenkins then choose the left box:



# Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

## Install suggested plugins

 Install plugins the Jenkins community finds most useful.

## Select plugins to install

Select and install plugins most suitable for your needs.

We'll see something like this:

## Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding	** Pipeline: Multibranch
✓ Timestampers	✓ Workspace Cleanup	✓ Ant	✓ Gradle	** Pipeline: Stage Tags Metadata
✓ Pipeline	✓ GitHub Branch Source	✓ Pipeline: GitHub Groovy Libraries	✓ Pipeline: Stage View	** Pipeline: Input Step
✓ Git	✓ SSH Build Agents	Matrix Authorization Strategy	PAM Authentication	** Pipeline: Declarative
LDAP	Email Extension	✓ Mailer		** Lockable Resources
				Pipeline
				** Java JSON Web Token (JJWT)
				** OkHttp
				** GitHub API
				Git
				** GitHub
				GitHub Branch Source
				Pipeline: GitHub Groovy Libraries
				** Pipeline Graph Analysis
				** Pipeline: REST API
				** JavaScript GUI Lib: Handlebars bundle
				** JavaScript GUI Lib: Moment.js bundle
				Pipeline: Stage View
				Git
				SSH Build Agents
				** - required dependency

→ Once done we see this page:

Getting Started

## Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:

Jenkins 2.332.1 [Skip and continue as admin](#) [Save and Continue](#)

→ Put admin everywhere - so we'll never forget it :) Just the e-mail should be an actual one.

Getting Started

## Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:

→ Then we hit the next button, we see this:

Getting Started

# Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various operation of many Jenkins features including email notifications, PR status updates, etc. The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Save password?

☐ Save only on this device

Username

admin

Password

.....

👁

Save

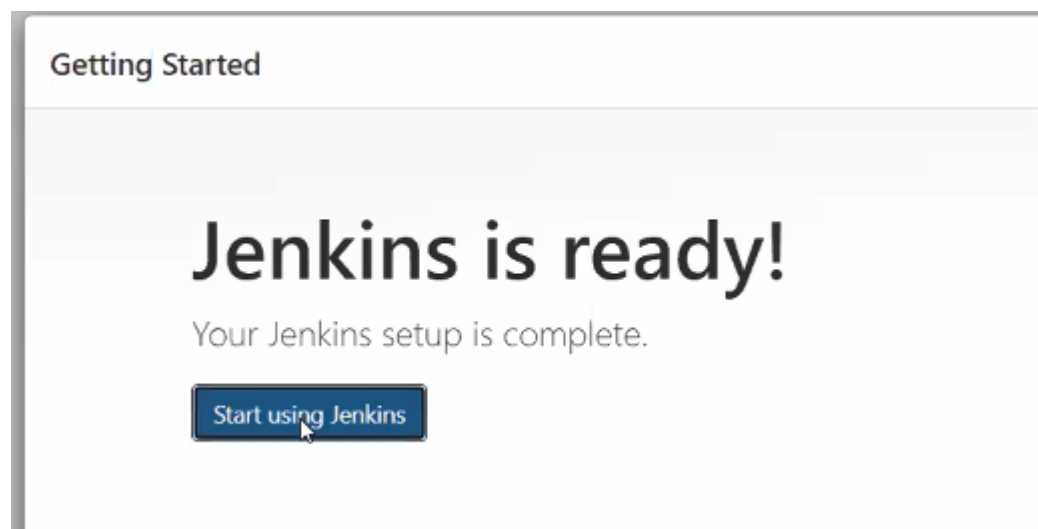
Never

Jenkins 2.332.1

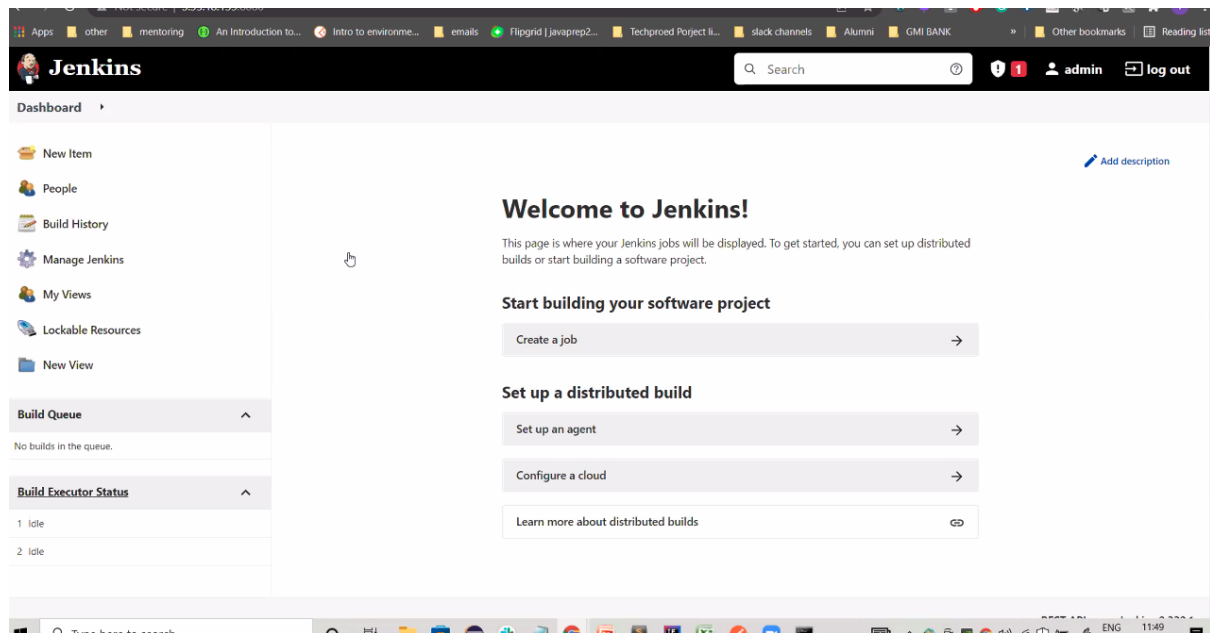
Not now

Save and Finish

→ Then we see this page:



→ And we are ready with our Jenkins setup **FINALLY** 🚀



→ Next - we click on Manage Jenkins →

→ Click on “Global Tool Configuration”

→ we click on “Add JDK” - deselect the default install

→ as a Name, put : “JAVA\_HOME”

→ Under it put this location - same for everyone → /usr/lib/jvm/java-11-openjdk-amd64

→ Git → name can be left as Default

→ path → /usr/lib/git-core/git

→ Hit Apply

→ Click on Add Maven

→ Name: MAVEN\_HOME

→ Under it give a location→ /usr/share/maven

→ Click on Apply and then on Save

→ Click on Dashboard → Click on New Item → Enter a name: JenkinsProject

→ Choose the Freestyle project → hit OK

The we see this page:

Scroll down → Source Code Management → tick Git → URL will the URL from GitHub

→ [https://github.com/HalilIbrahimHasan/jenkins\\_cloud.git](https://github.com/HalilIbrahimHasan/jenkins_cloud.git) (like your own project URL from Git - the one we use for cloning the project)

→ make sure the branch is correct

→ Build → Add Build Step → choose Invoke top-level Maven targets → then give the name MAVEN\_HOME (it will appear on the drop-down) to the Maven Version field  
→ Under it for Goals - write there → clean install  
→ Then hit Apply and then Save

## << Now our project is ready >>

→ We go back to the IJ and run the class again - names changed again :)

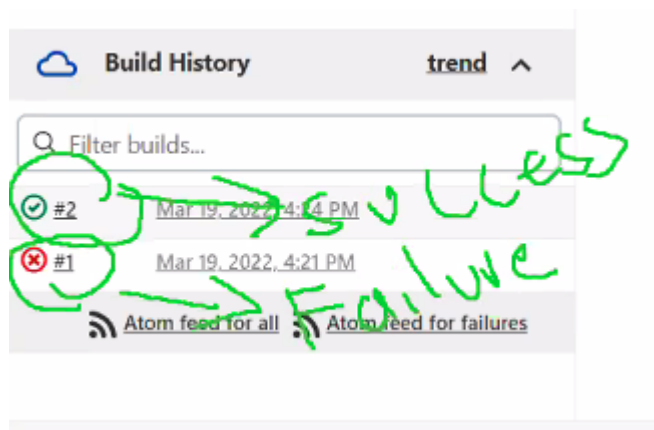
→ Let's see the error message on the cloud →

We're on the project page → click on "Build Now" → starts running our scripts → click on Console Out and you'll see the tests → it fails due to the data change.

→ we rerun the IJ and updated the data with the latest one → then rerun it again → saw the pass message on the IJ →

→ we push our updates → git add. → git commit -m "jenkins updates" → git push

→ then went back to the cloud and rerun it there so we could see the success (Build Now - the second built → Console Out → and we can see the SUCCESS at the bottom of the page 🚀



→ if we click on the "Build Now" this will start a new run, this will be the third one for us.

