<epam>

Continuous integration / Continuous delivery (deployment). Lection 1.

Prerequisites

- Software development methodologies and techniques
- Virtualization and containerization
- Version Control Systems
- Operating systems administration
- Networking

Continuous integration

IF

IT_business.bloods == time && money

THEN

IT_business.heart_pumping = CI / CD

FORMALY

“CI/CD bridges the gaps between development and operation activities and teams by enforcing automation in building, testing and deployment of applications. DevOps practices involve continuous development, continuous testing, continuous integration, continuous deployment and continuous monitoring of software applications throughout its development life cycle. The CI/CD practice, or CI/CD pipeline, forms the backbone of DevOps operations.”

https://en.wikipedia.org/wiki/CI/CD

FORMALY

“**DevOps** is a set of practices that combines software development (*Dev*) and IT operations (*Ops*). It aims to shorten the systems development life cycle and provide continuous delivery with high software quality. DevOps is complementary with Agile software development. Several DevOps aspects came from the Agile methodology.”

https://en.wikipedia.org/wiki/DevOps

Software Development Life Cycle

1. **Identify the Current Problems (Requirement Analysis)**

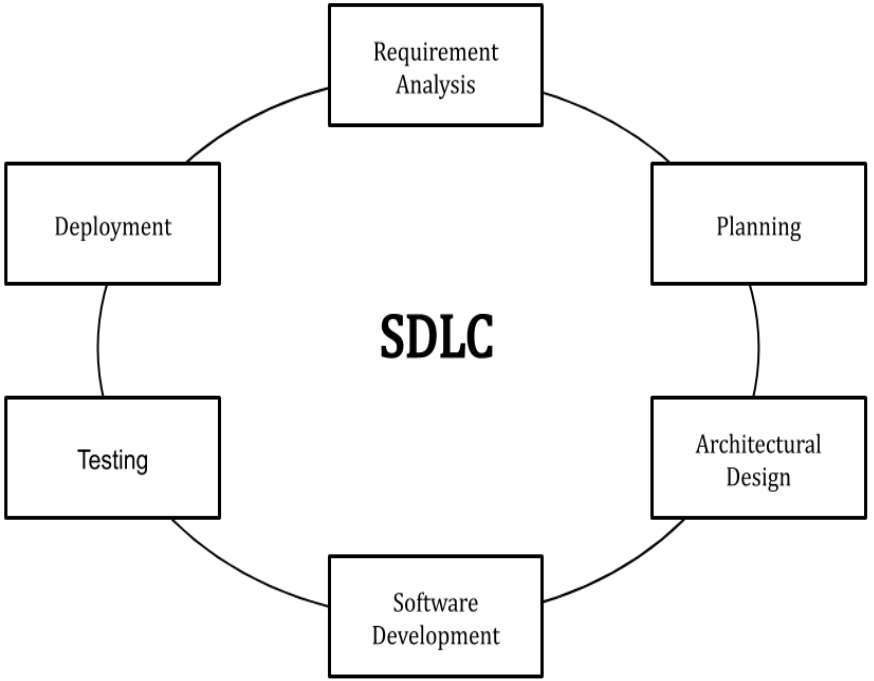
“What are the current problems?” This stage of the SDLC means getting input from all stakeholders, including customers, salespeople, industry experts, and programmers. Learn the strengths and weaknesses of the current system with improvement as the goal.

2. **Plan**

“What do we want?” In this stage of the SDLC, the team determines the cost and resources required for implementing the analyzed requirements. It also details the risks involved and provides sub-plans for softening those risks.

3. **Design**

“How will we get what we want?” This phase of the SDLC starts by turning the software specifications into a design plan called the Design Specification. All stakeholders then review this plan and offer feedback and suggestions. It’s crucial to have a plan for collecting and incorporating stakeholder input into this document. Failure at this stage will almost certainly result in cost overruns at best and the total collapse of the project at worst.



https://stackify.com/what-is-sdlc/

4. Build

“Let’s create what we want.” At this stage, the actual development starts. It’s important that every developer sticks to the agreed blueprint. Also, make sure you have proper guidelines in place about the code style and practices.

5. Code Test

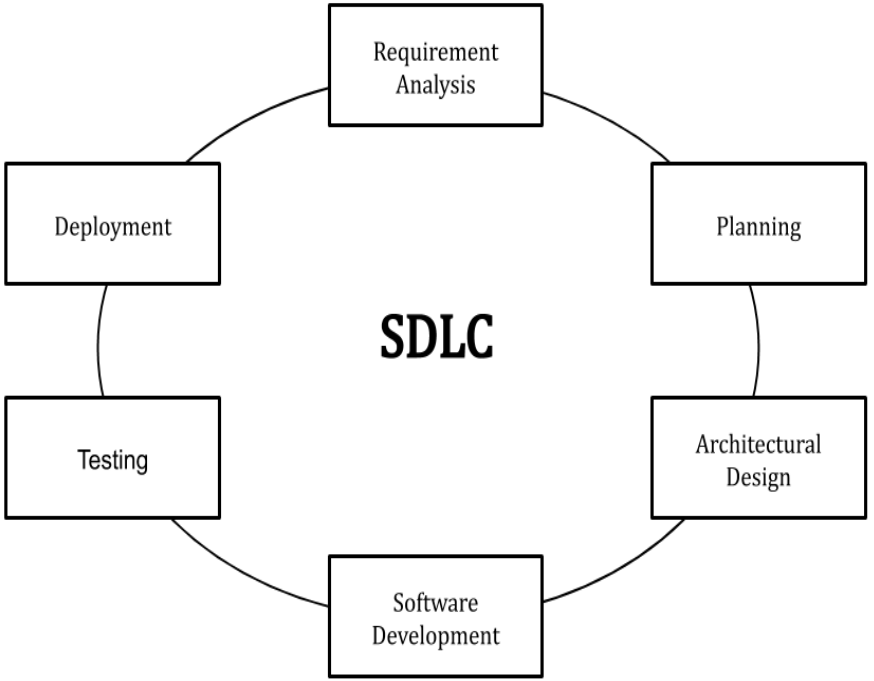
“Did we get what we want?” In this stage, we test for defects and deficiencies. We fix those issues until the product meets the original specifications.

In short, we want to verify if the code meets the defined requirements.

6. Software Deployment

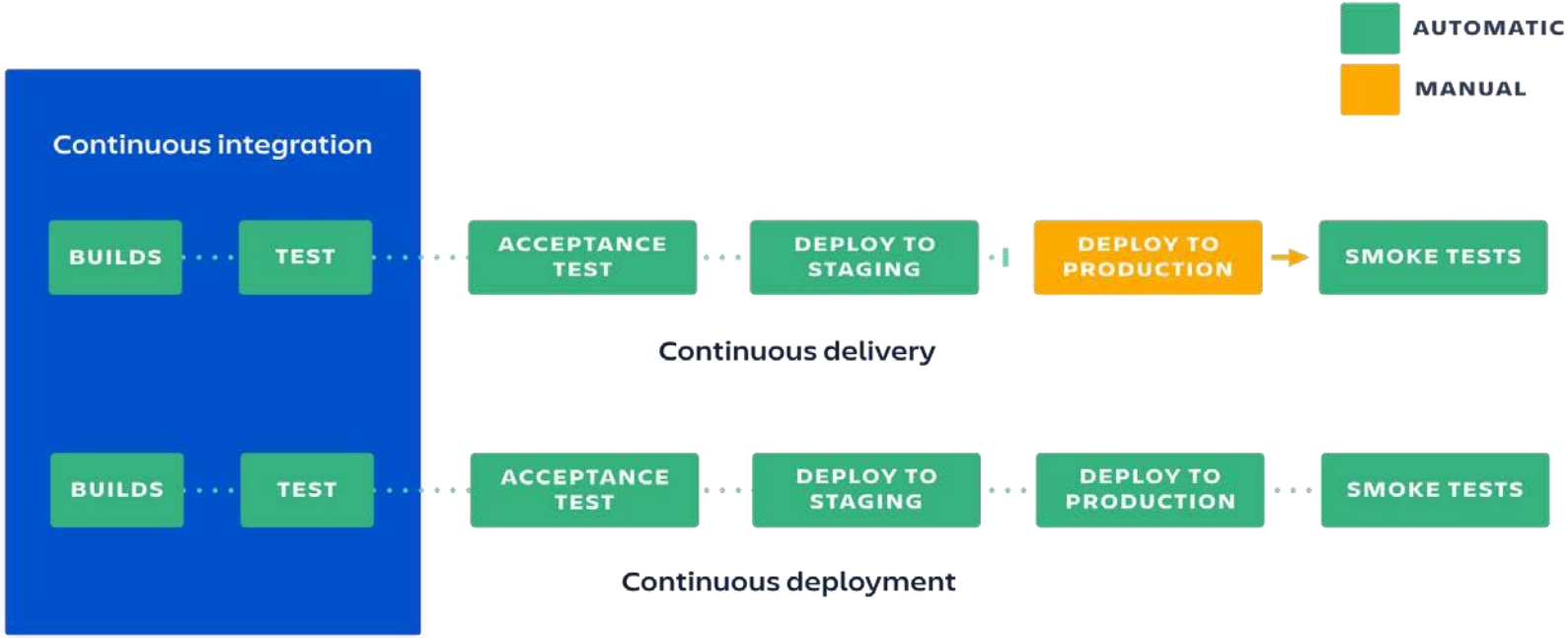
“Let’s start using what we got.” At this stage, the goal is to deploy the software to the production environment so users can start using the product. However, many organizations choose to move the product through different deployment environments such as a testing or staging environment.

This allows any stakeholders to safely play with the product before releasing it to the market. Besides, this allows any final mistakes to be caught before releasing the product.



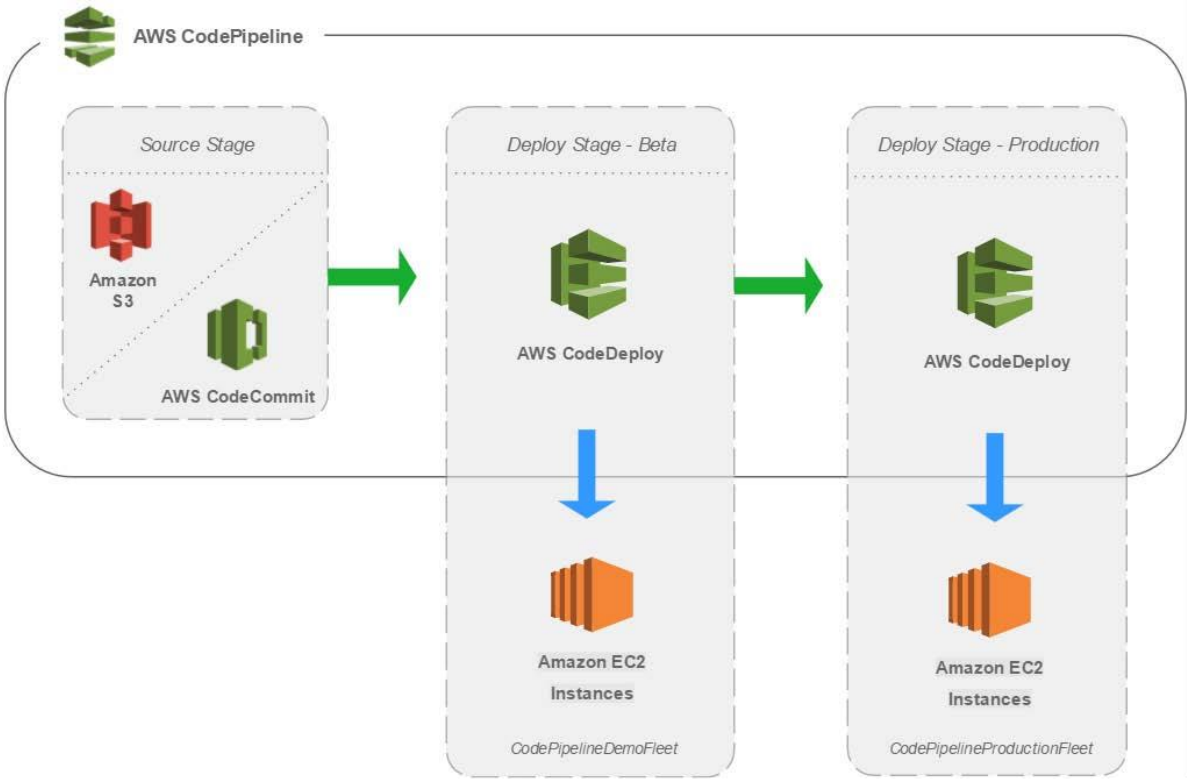
https://stackify.com/what-is-sdlc/

Atlassian vision

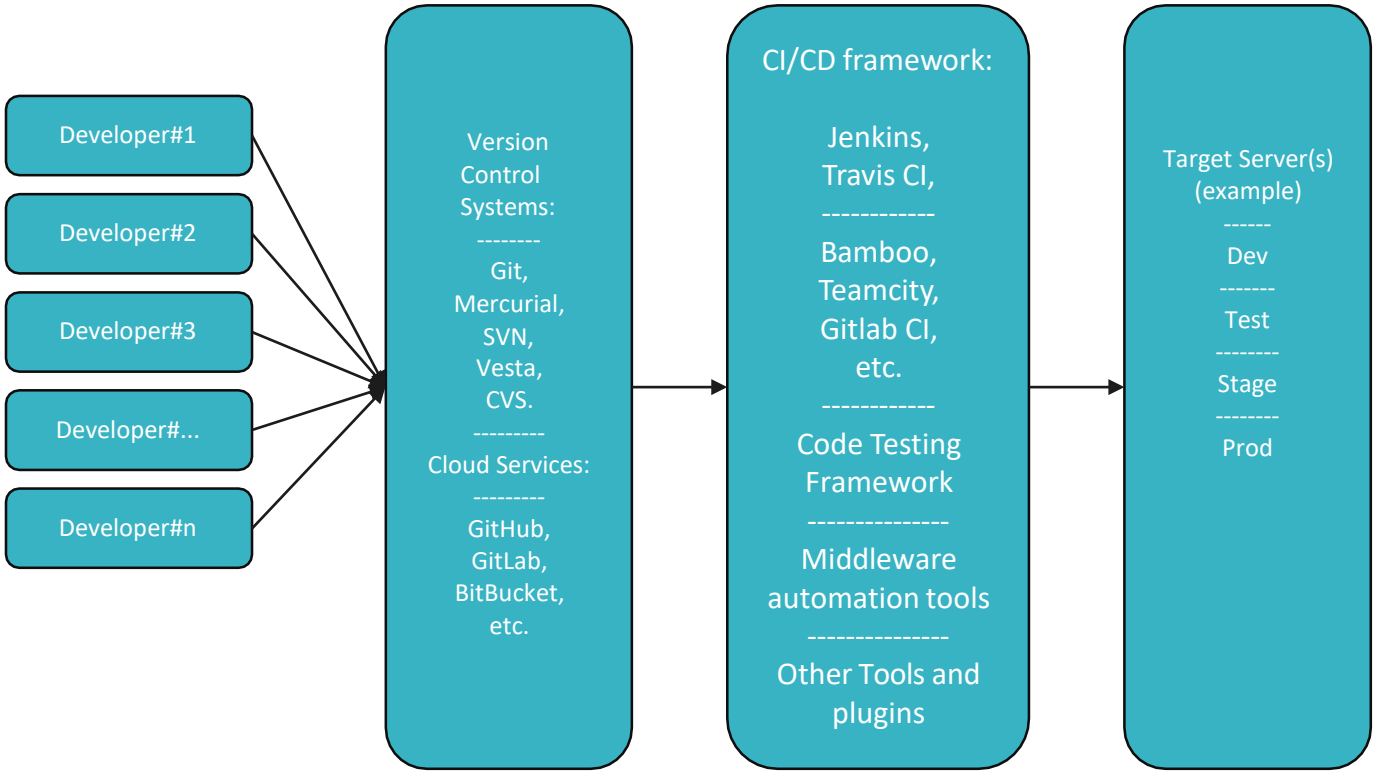


RedHat vision

AWS vision


AWS CodePipeline
Source Stage
Amazon S3
AWS CodeCommit
Deploy Stage - Beta
AWS CodeDeploy
Amazon EC2 Instances
CodePipelineDemoFleet
Deploy Stage - Production
AWS CodeDeploy
Amazon EC2 Instances
CodePipelineProductionFleet

CI/CD pipeline



Prerequisites

- VirtualBox ver. 6+
- Installed Ubuntu 20.04 with SSH server (server edition) with “Network”-> “Adapter” set to “Bridged Adapter”
- Clone(-s) of Installed Ubuntu 20.04 with SSH server (server edition) with “Network”-> “Adapter” set to “Bridged Adapter” (full clone with reinitialized MAC- addresses)

SSH client :

https://mobaxterm.mobatek.net/download.html (for Windows)

or

https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html (Windows, Linux, but simple)

or

•https://www.termius.com/

(Windows, Linux, Mac)

Installing Jenkins

Linux

Jenkins installers are available for several Linux distributions.

- Debian/Ubuntu
- Fedora
- Red Hat / CentOS

Prerequisites

Minimum hardware requirements:

- 256 MB of RAM
- 1 GB of drive space (although 10 GB is a recommended minimum if running Jenkins as a Docker container)

Recommended hardware configuration for a small team:

- 4 GB+ of RAM
- 50 GB+ of drive space

Comprehensive hardware recommendations:

- Hardware: see the Hardware Recommendations page

Software requirements:

- Java: see the Java Requirements page
- Web browser: see the Web Browser Compatibility page
- For Windows operating system: Windows Support Policy

https://www.jenkins.io/doc/book/installing/linux/

Jenkins (https://jenkins.io/)

- https://jenkins.io/download/

Downloading Jenkins

Jenkins is distributed as WAR files, native packages, installers, and Docker images. Follow these installation steps:

1. Before downloading, please take a moment to review the **Hardware and Software requirements** section of the User Handbook.
2. Select one of the packages below and follow the download instructions.
3. Once a Jenkins package has been downloaded, proceed to the **Installing Jenkins** section of the User Handbook.
4. You may also want to verify the package you downloaded. Learn more about verifying Jenkins downloads.

Download Jenkins 2.289.3 LTS for:

- Generic Java package (.war)
  SHA-256: 996dfd29d5f933546af9e9f77c29b371fb0627b8266b6c9f134ac2e0f1248b87
- Docker
- Ubuntu/Debian
- CentOS/Fedora/Red Hat
- Windows
- openSUSE
- FreeBSD
- Gentoo
- macOS
- OpenBSD

Download Jenkins 2.307 for:

- Generic Java package (.war)
  SHA-256: e1f73c7fc6060dfb0eca64ac8aaed36d91c747219d35eff3653889ddf8edff2c
- Docker
- Ubuntu/Debian
- CentOS/Fedora/Red Hat
- Windows
- openSUSE
- Arch Linux
- FreeBSD
- Gentoo
- macOS
- OpenBSD
- OpenIndiana Hipster

Jenkins Debian packages

Jenkins Debian Packages

This is the Debian package repository of Jenkins to automate installation and upgrade. To use this repository, first add the key to your system:

```
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
```

Then add a Jenkins apt repository entry:

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

Update your local package index, then finally install Jenkins:

```
sudo apt-get update
sudo apt-get install jenkins
```

The apt packages were signed using this key:

```
pub   rsa4096 2020-03-30 [SC] [expires: 2023-03-30]
      62A9756BFD780C377CF24BA8FCEF32E745F2C3D5
uid                      Jenkins Project
sub   rsa4096 2020-03-30 [E] [expires: 2023-03-30]
```

You will need to explicitly install a Java runtime environment, because Oracle's Java RPMs are incorrect and fail to register as providing a java dependency. Thus, adding an explicit dependency requirement on Java would force installation of the OpenJDK JVM.

- 2.164 (2019-02) and newer: Java 8 or Java 11
- 2.54 (2017-04) and newer: Java 8
- 1.612 (2015-05) and newer: Java 7

With that set up, the Jenkins package can be installed with:
See the installation guide for more information, including how Jenkins is run and where the configuration is stored, etc.

https://pkg.jenkins.io/debian-stable/
https://www.jenkins.io/doc/book/installing/linux/#debianubuntu

Jenkins Debian packages

If an error is reported, “jenkins : Depends: daemon but it is not installable”, add the "universe" apt repository of community maintained free and open source software for Ubuntu by executing this command after `sudo apt-get update`:

```
sudo add-apt-repository universe
```

This package installation will:

- Setup Jenkins as a daemon launched on start. See `/etc/init.d/jenkins` for more details.
- Create a ‘jenkins’ user to run this service.
- Direct console log output to the file `/var/log/jenkins/jenkins.log`. Check this file if you are troubleshooting Jenkins.
- Populate `/etc/default/jenkins` with configuration parameters for the launch, e.g `JENKINS_HOME`
- Set Jenkins to listen on port 8080. Access this port with your browser to start configuration.

If your `/etc/init.d/jenkins` file fails to start Jenkins, edit the `/etc/default/jenkins` to replace the line `----HTTP_PORT=8080----` with `----HTTP_PORT=8081----` Here, "8081" was chosen but you can put another port available.

https://www.jenkins.io/doc/book/installing/linux/#weekly-release

Jenkins Debian packages

Installation of Java

Jenkins requires Java in order to run, yet certain distributions don't include this by default and some Java versions are incompatible with Jenkins.

There are multiple Java implementations which you can use. OpenJDK is the most popular one at the moment, we will use it in this guide.

To install the Open Java Development Kit (OpenJDK) run the following:

- Update the repositories

```
sudo apt update
```

- search of all available packages:

```
sudo apt search openjdk
```

- Pick one option and install it:

```
sudo apt install openjdk-11-jdk
```

- Confirm installation:

```
sudo apt install openjdk-11-jdk
```

- checking installation:

```
java -version
```

- the result must be something like:

```
openjdk version "11.0.9.1" 2020-11-04
OpenJDK Runtime Environment (build 11.0.9.1+1-post-Debian-1deb10u2)
OpenJDK 64-Bit Server VM (build 11.0.9.1+1-post-Debian-1deb10u2, mixed mode, sharing)
```

https://www.jenkins.io/doc/book/installing/linux/#installation-of-java

Use SSH clients to copy-paste all the followings commands(except#)

1. wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
2. sudo nano /etc/apt/sources.list
3. #Add next string after last row in the editing file

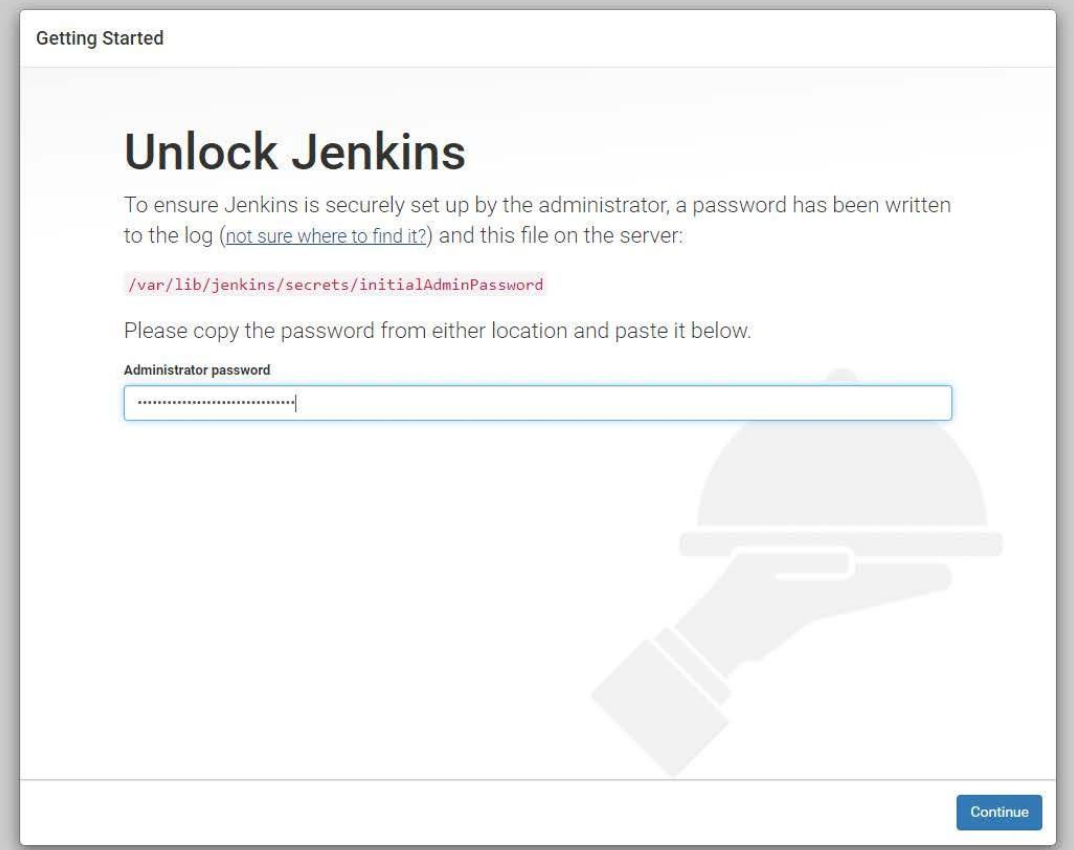
deb https://pkg.jenkins.io/debian-stable binary/

#And save changes

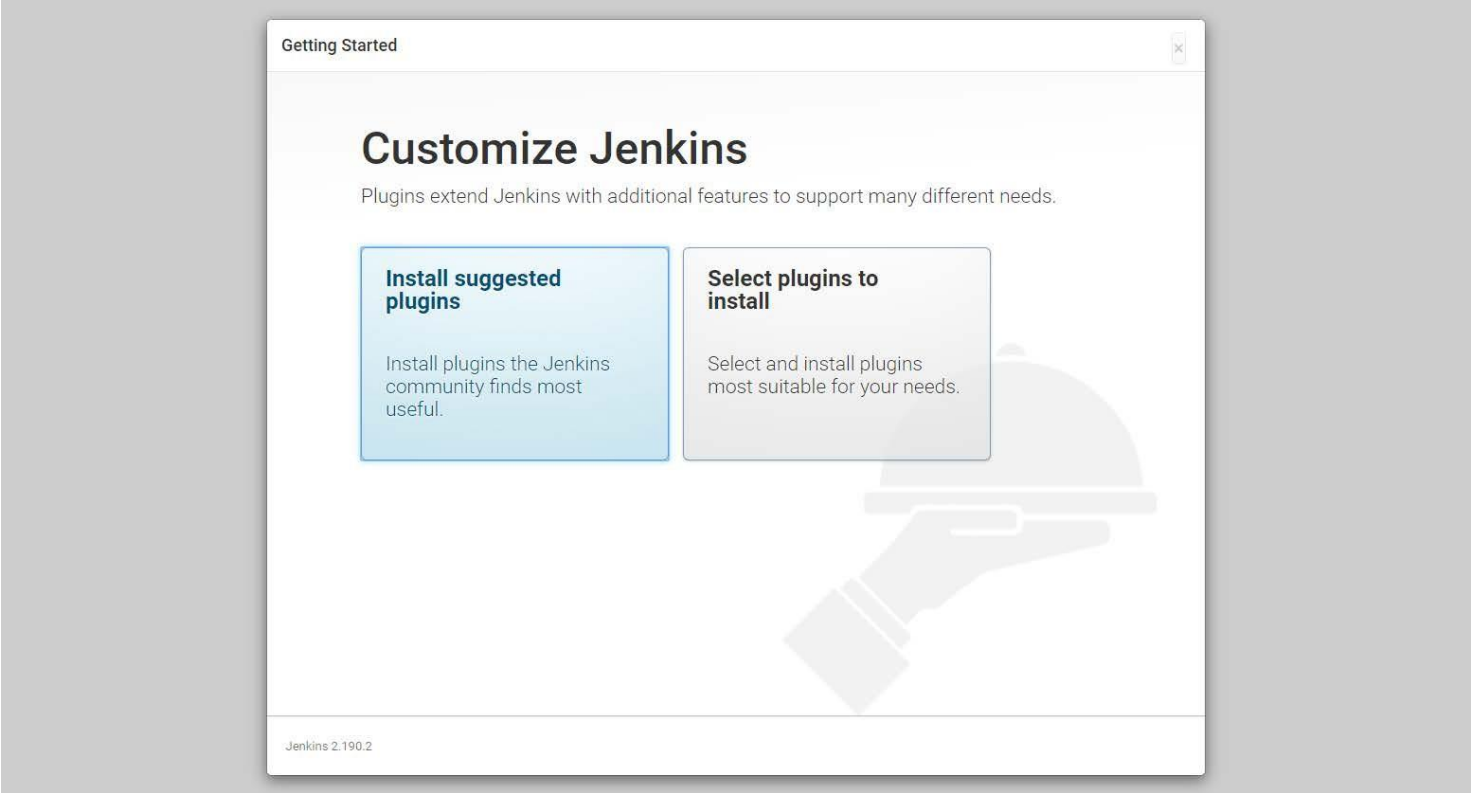
4. sudo apt-get update
5. java -version
6. sudo apt-get install openjdk-8-jdk
7. sudo apt-get install jenkins
8. service jenkins status
9. sudo cat /var/lib/jenkins/secrets/initialAdminPassword

#The output of this command should be entered in input field on the next slide

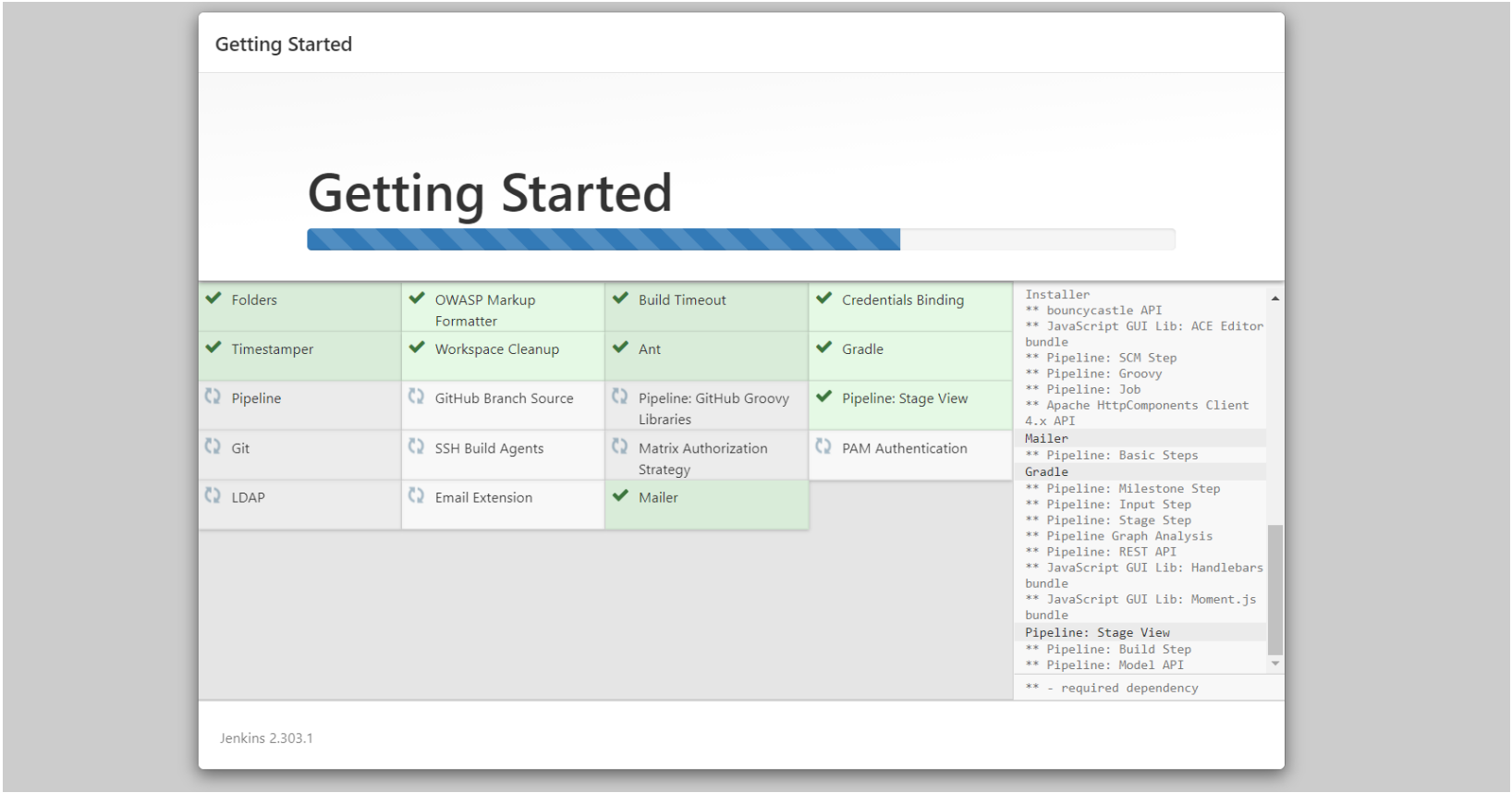
Unlocking Jenkins



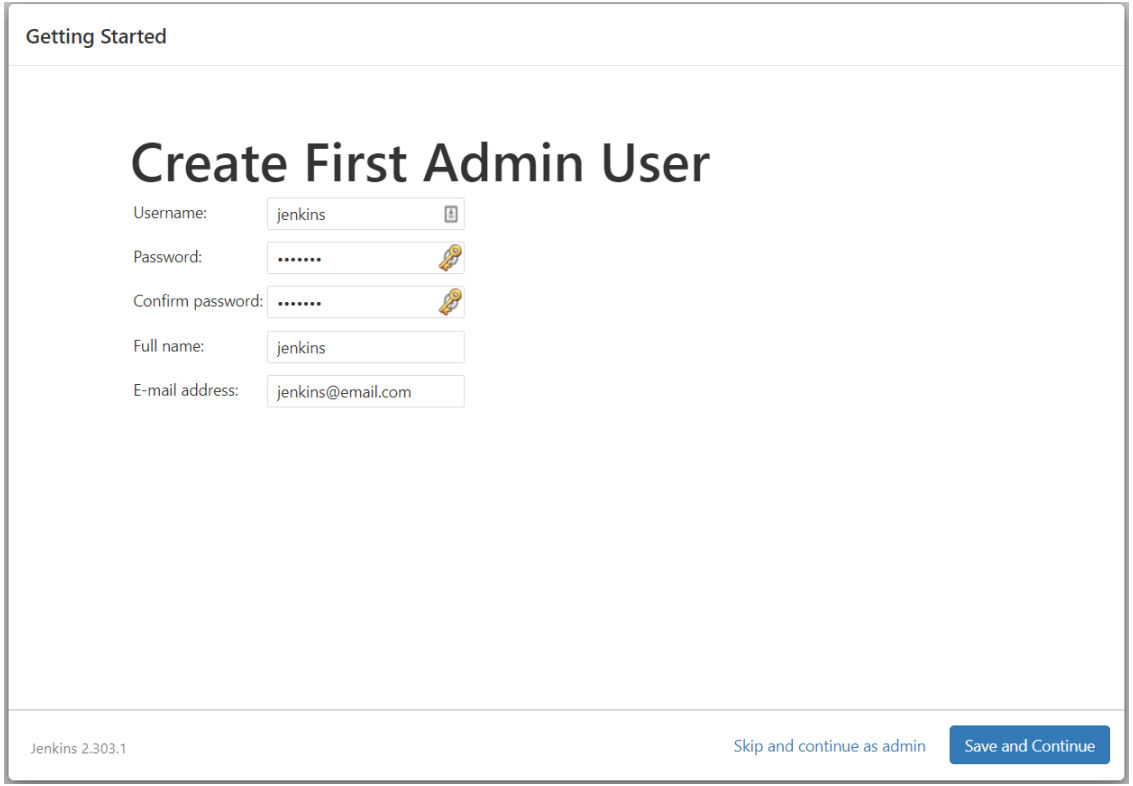
Expand functionality of Jenkins with plugins



Suggested Plugins installation

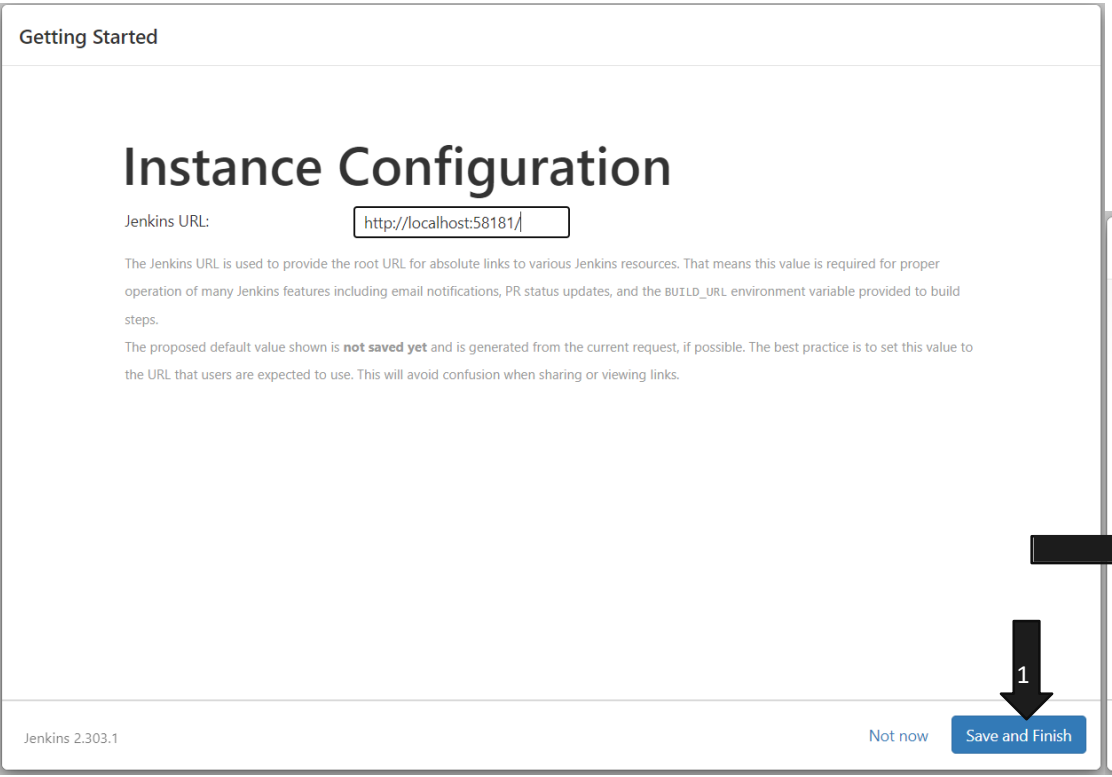


Create user with admin privileges



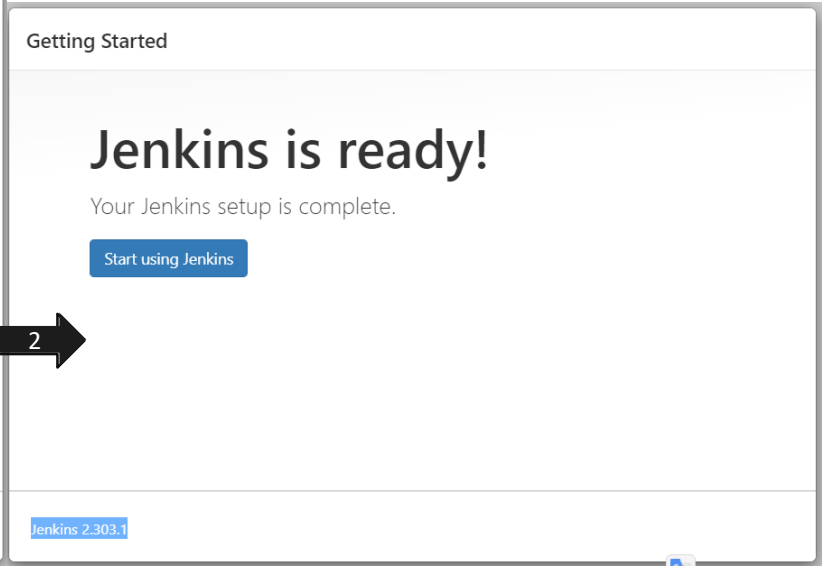
Language will be set to English as default using plugin Locale (later)

Jenkins instance configuration



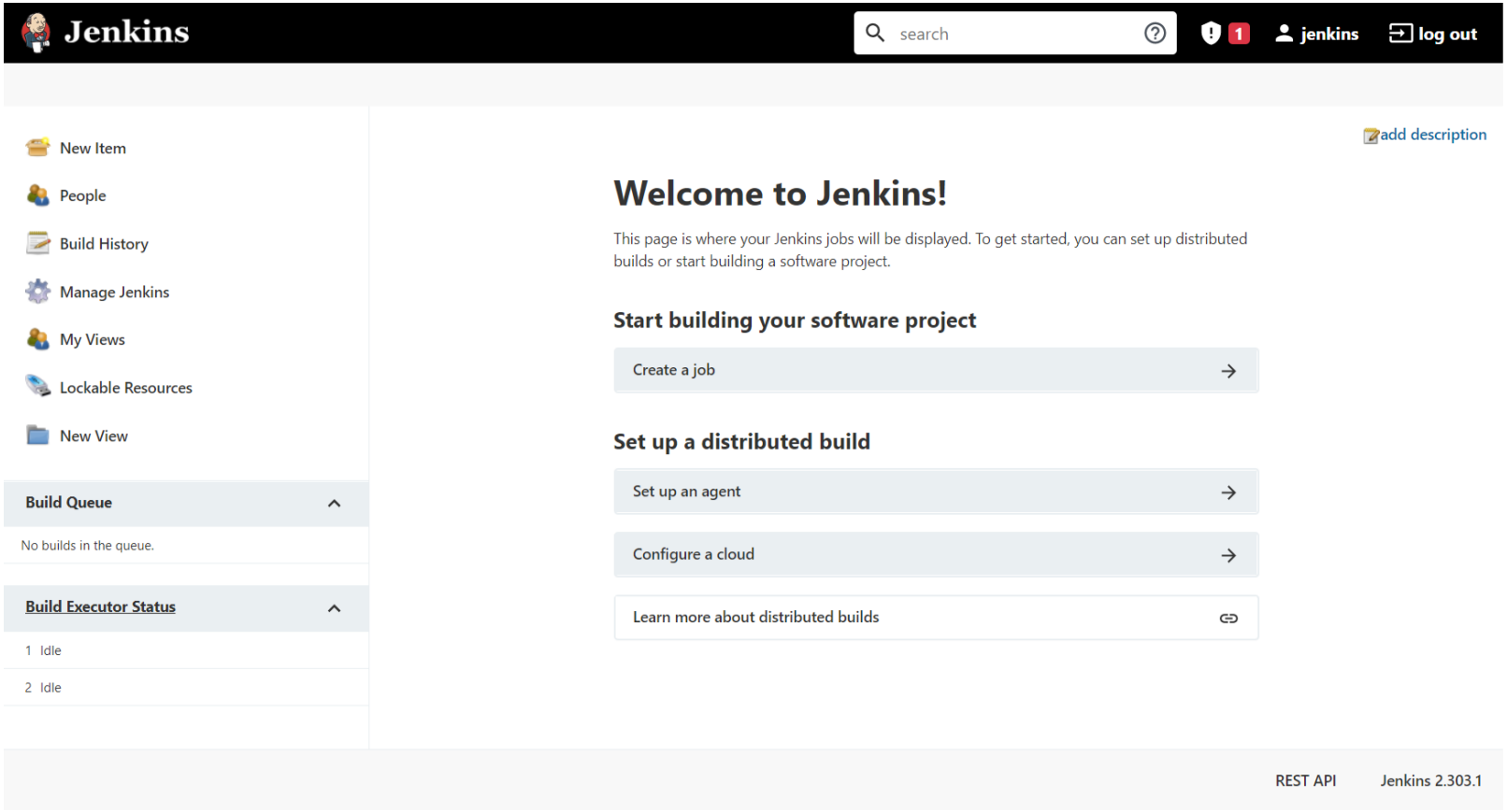
1

2

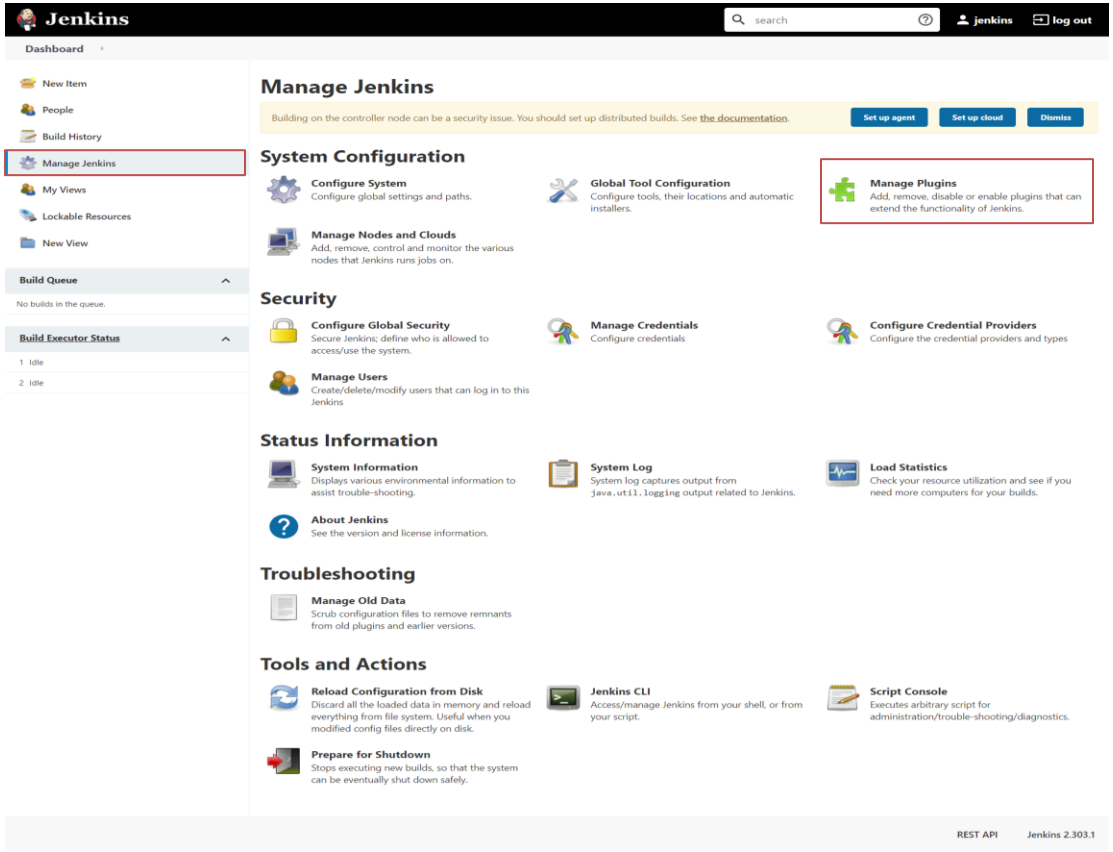


"Grey" or behind NAT IP makes some difficulties. We'll talk about it.

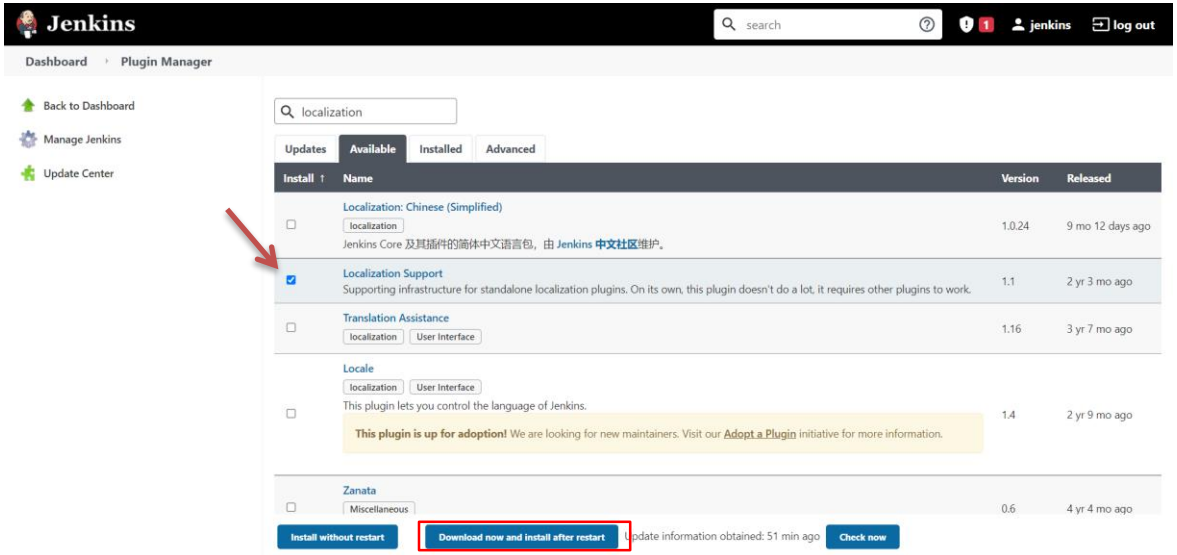
First look at Jenkins GUI. Let’s make it in English

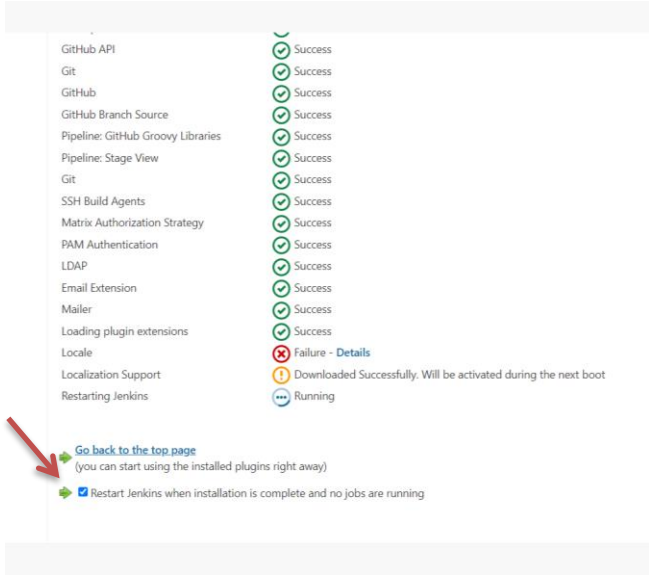


Install first plugin manually

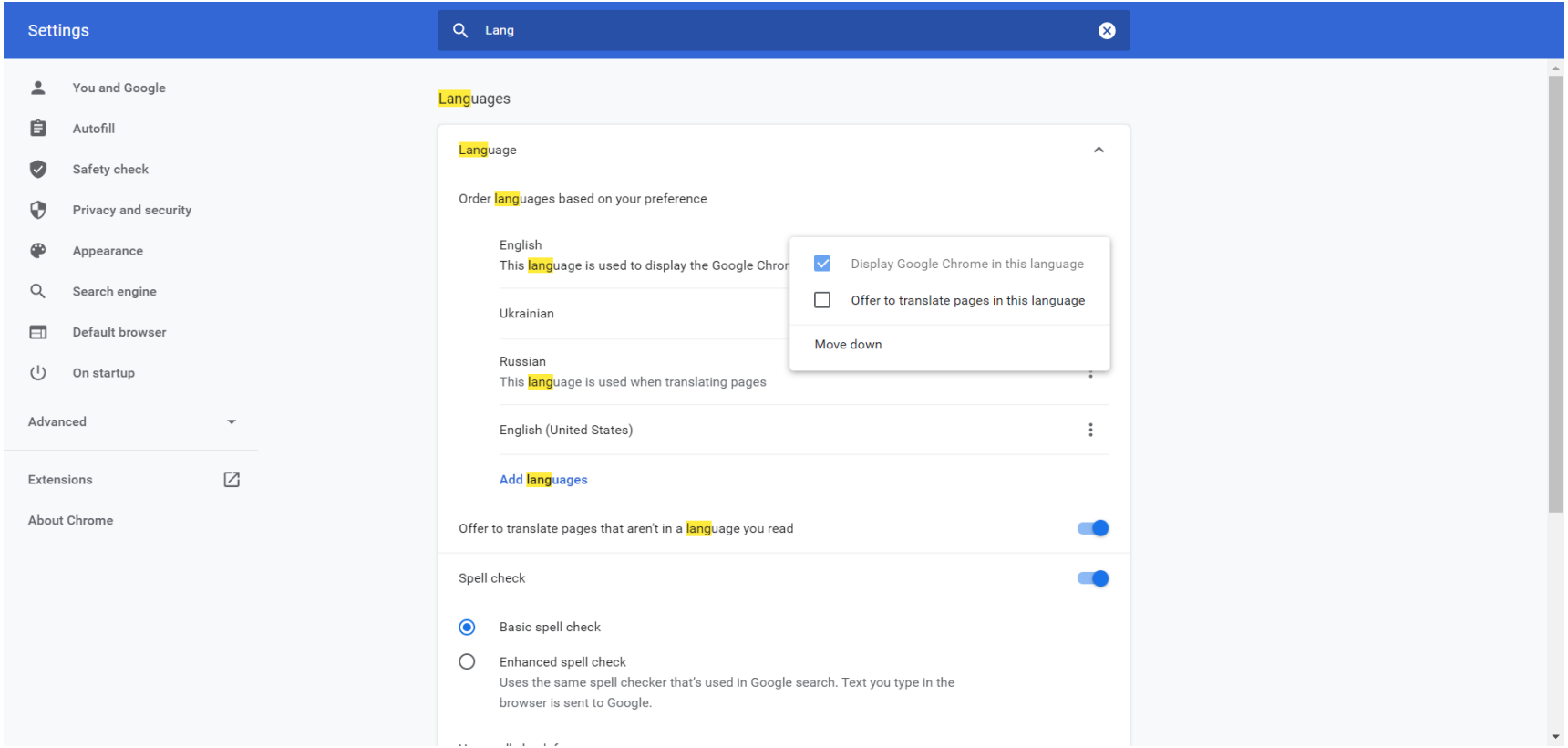


Plugin Locale installation

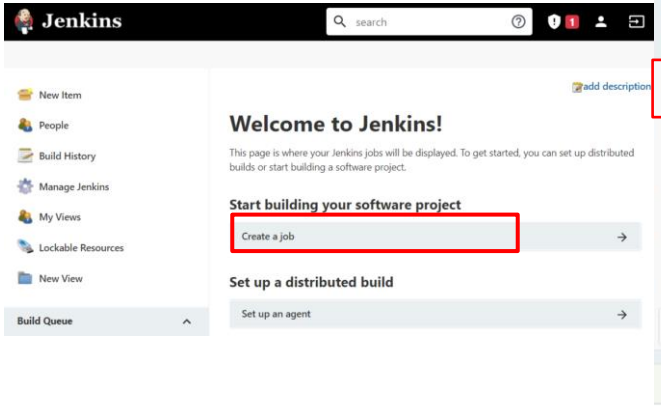


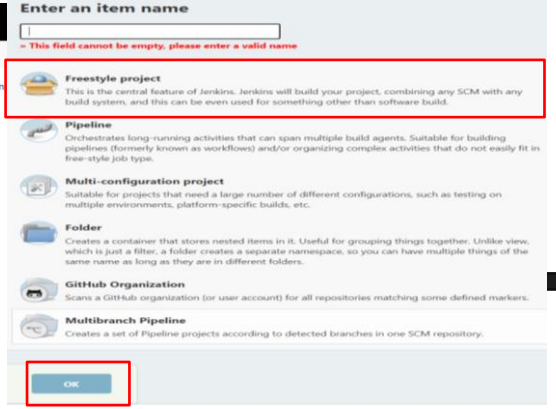


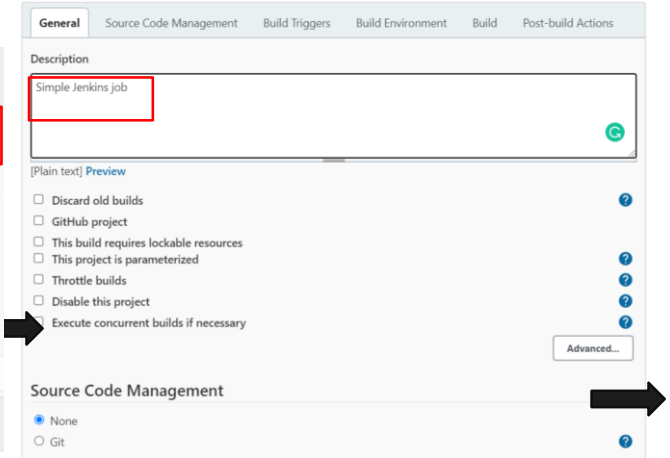
Chrome Language setting

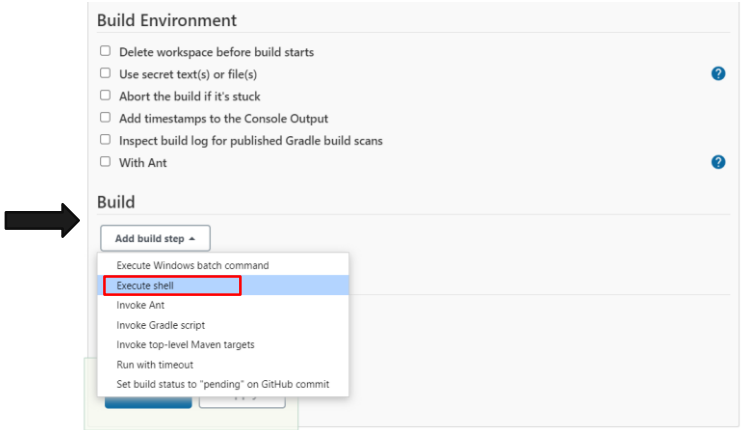


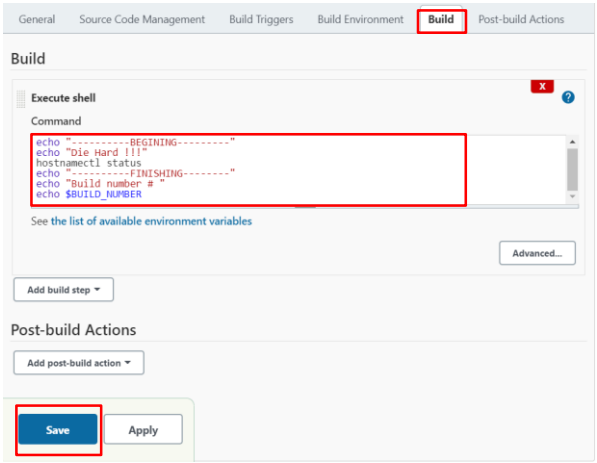
Simple job example in Jenkins



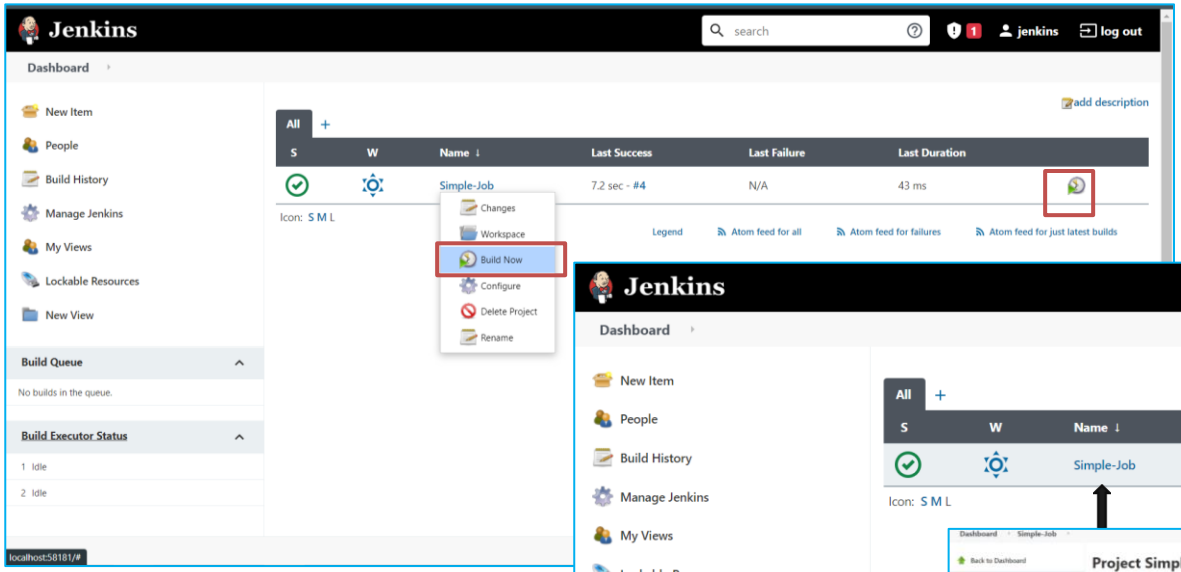


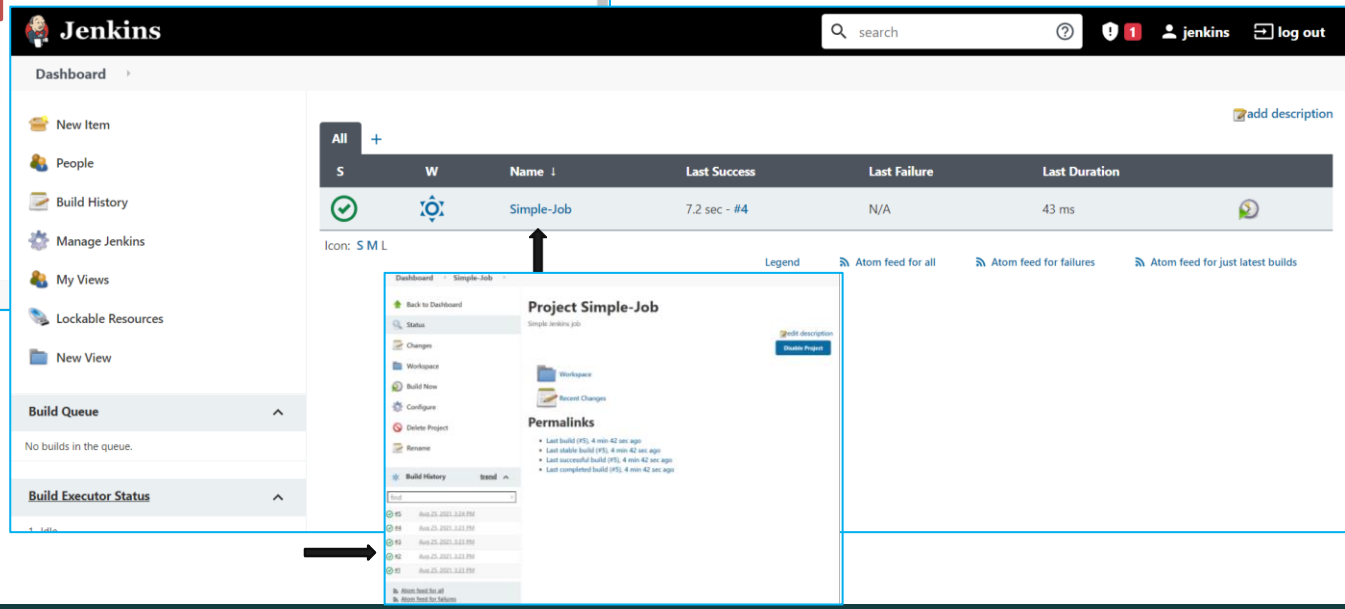




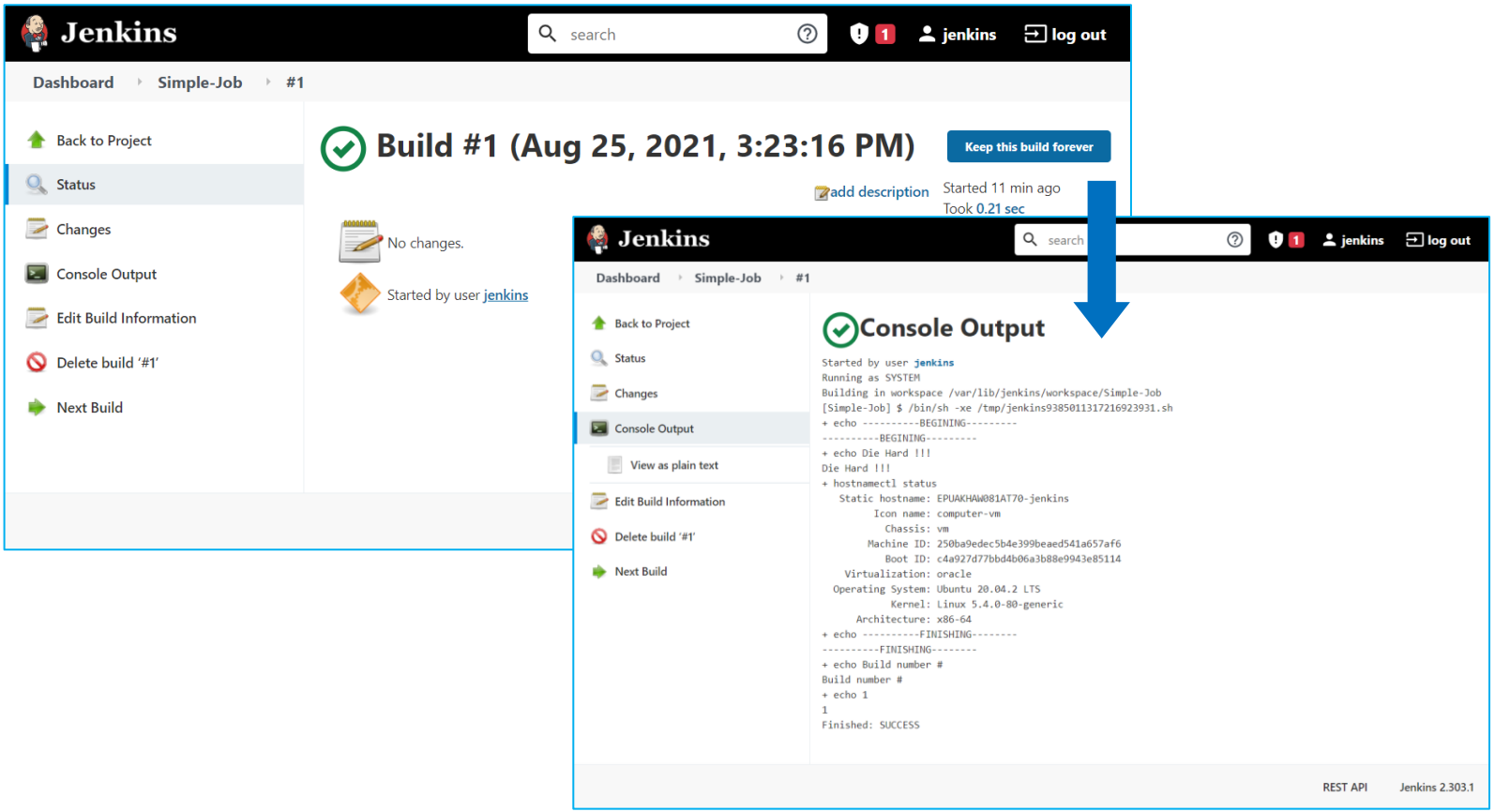


Simple job example in Jenkins

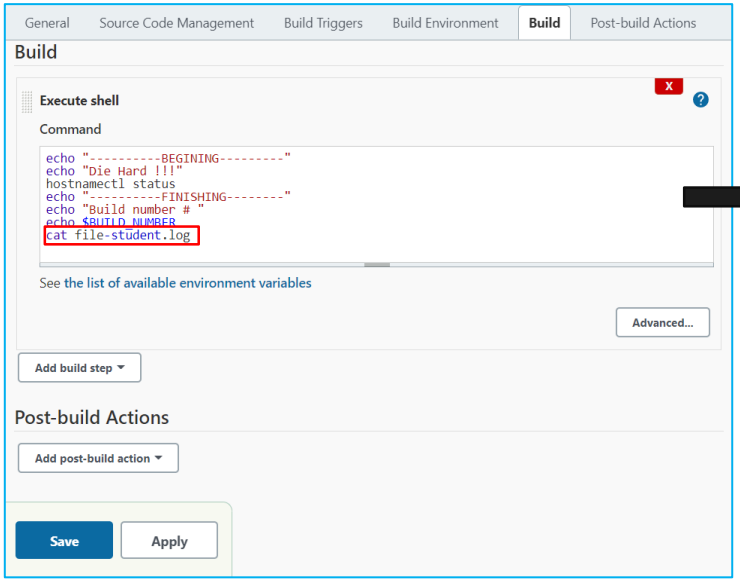


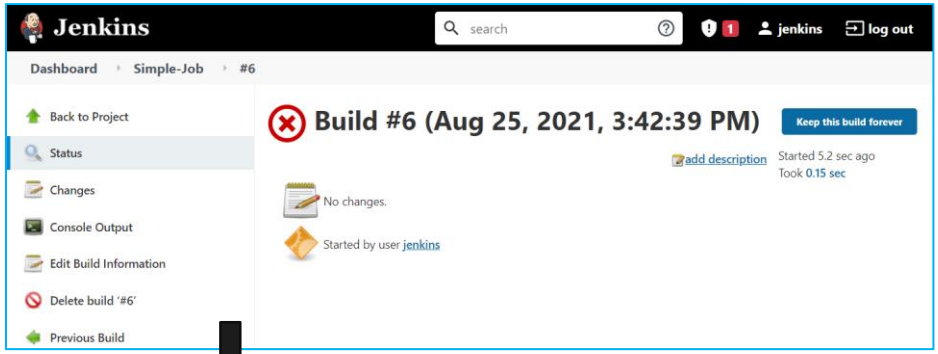


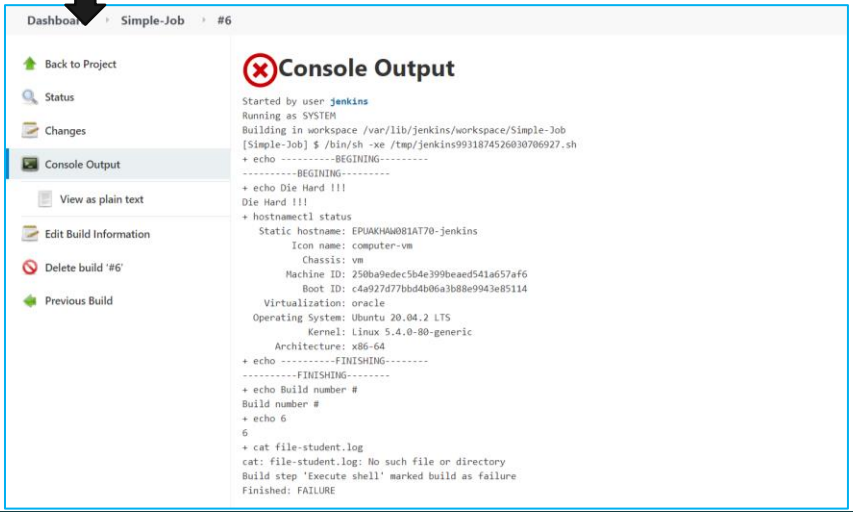
Simple job example in Jenkins (Finished: SUCCESS)



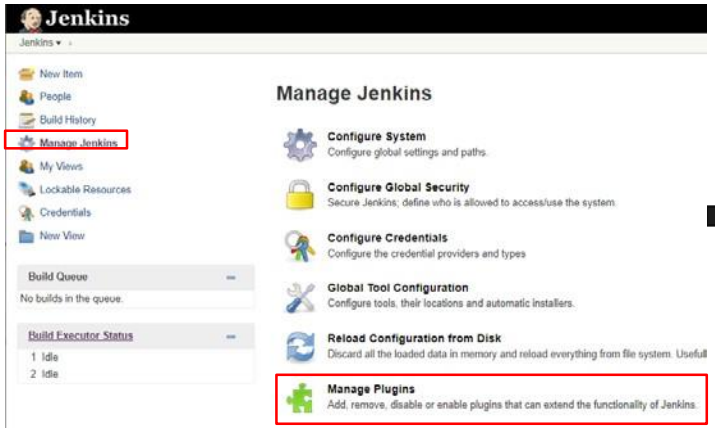
Simple job example in Jenkins (Finished: FAILED)

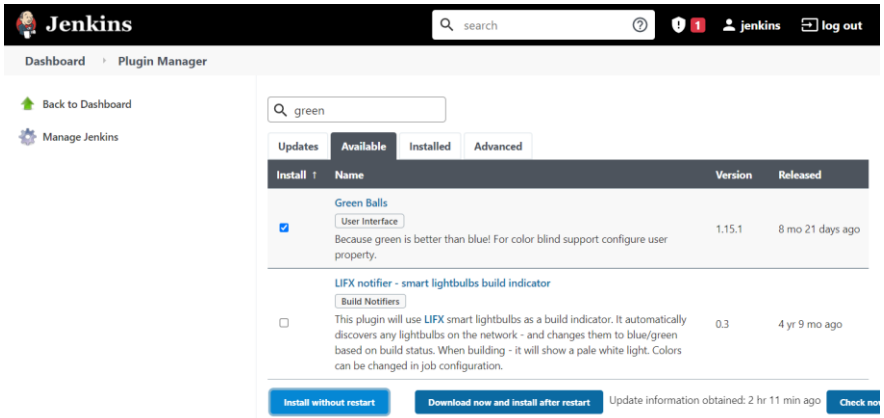


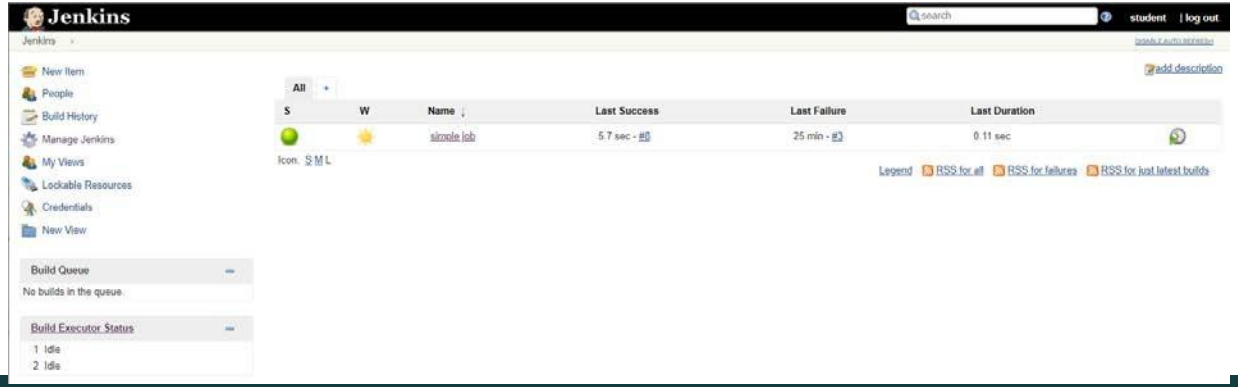




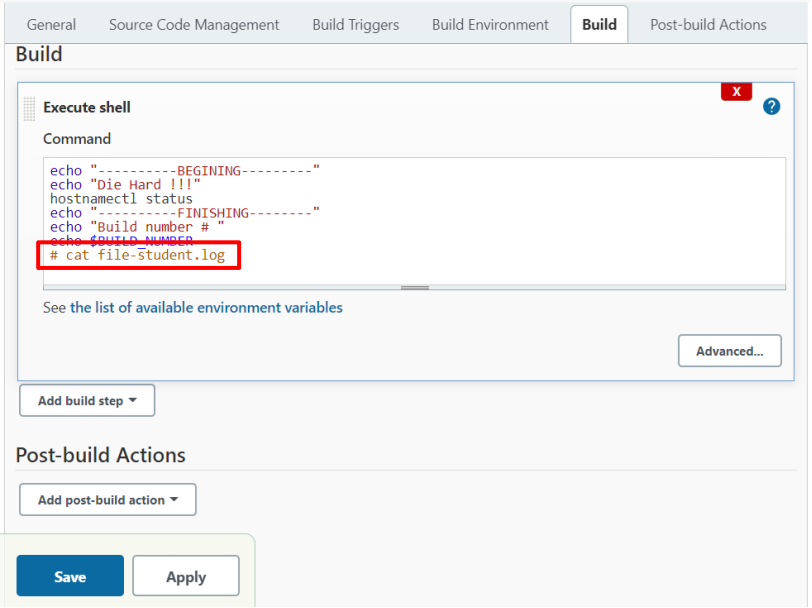
Frequently used plugins make GUI better: Green Balls (green better than blue :-))

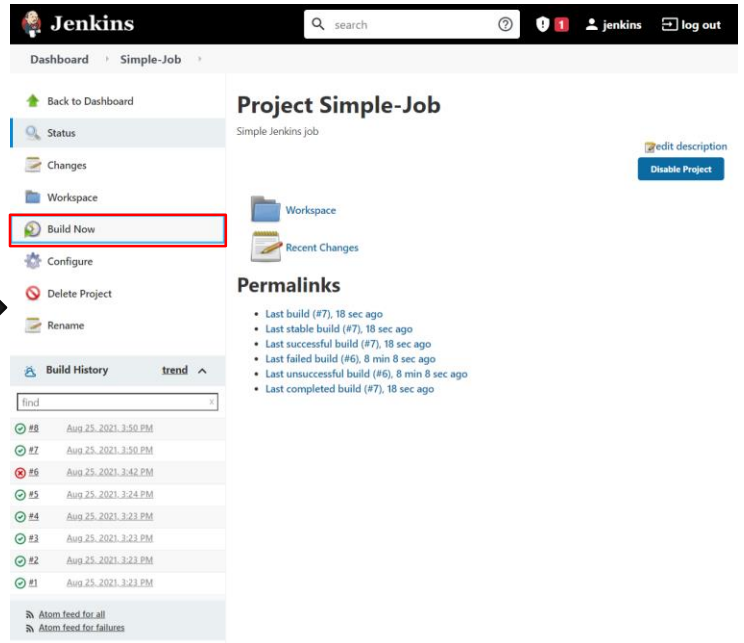




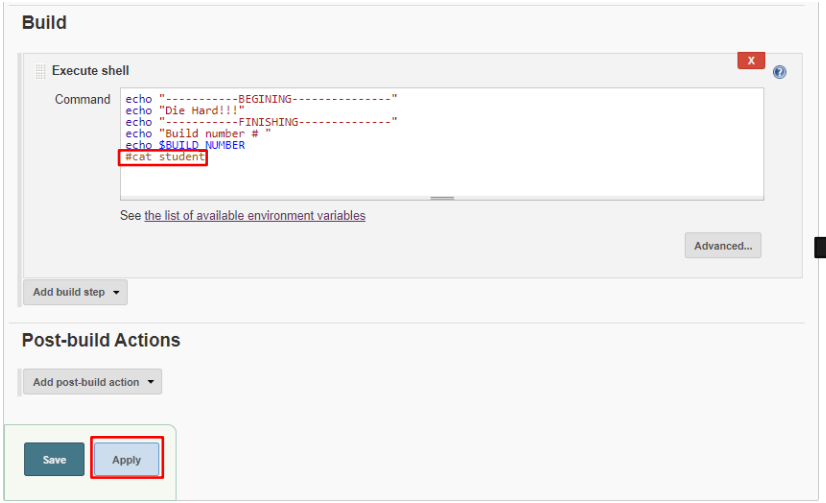


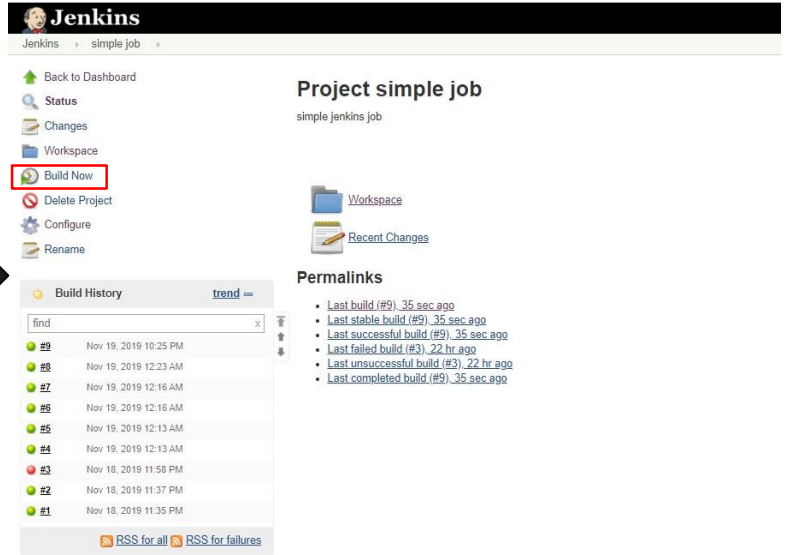
Simple job example in Jenkins (Finished: SUCCESS) with GreenBalls plugin enabled





Simple job example in Jenkins (Finished: SUCCESS) with GreenBalls plugin enabled





References

- https://jenkins.io/doc/book/installing/
- https://jenkins.io/doc/tutorials/
- https://www.javatpoint.com/jenkins
- https://www.edureka.co/blog/jenkins-tutorial/
- https://wiki.jenkins.io/display/JENKINS/Home

Q & A


Thank you!