

Secure Code Management Homework

Preda Mihail Irinel, ISM

Setup jenkins

Steps

1. Go to <https://www.jenkins.io/download/> and download Jenkins 2.319.1 LTS for your OS system
2. Open Installer



Fig.1

2. Choose where to install it:

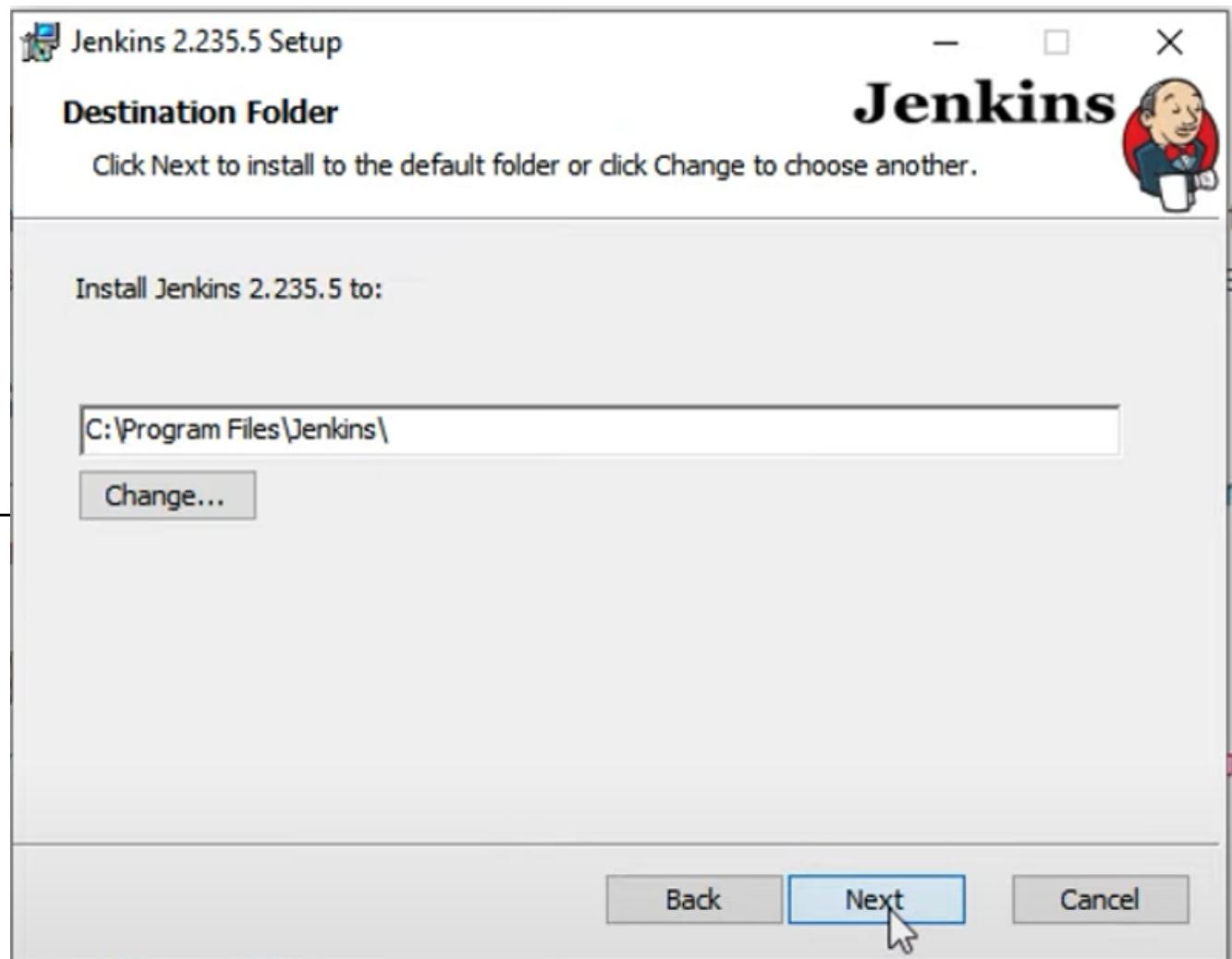


Fig.2

3. Choose Logon type. Select **Run service as LocalSystem**

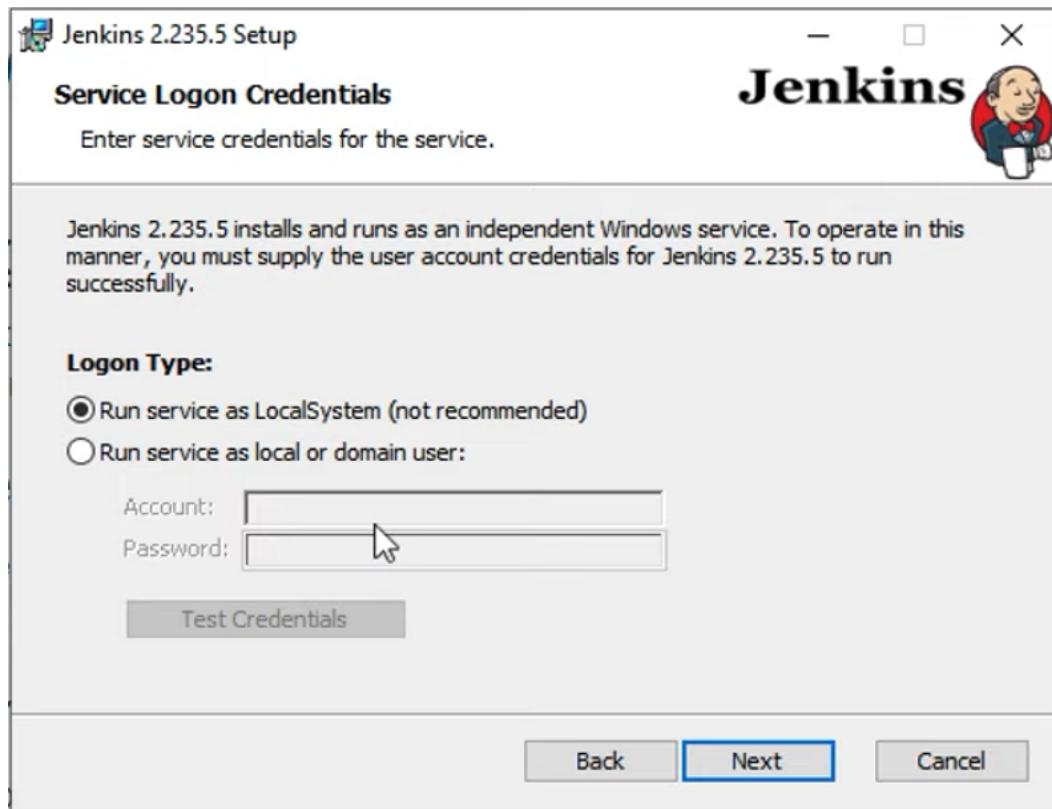


Fig.3

4. Select a port number or leave it as default (8080)

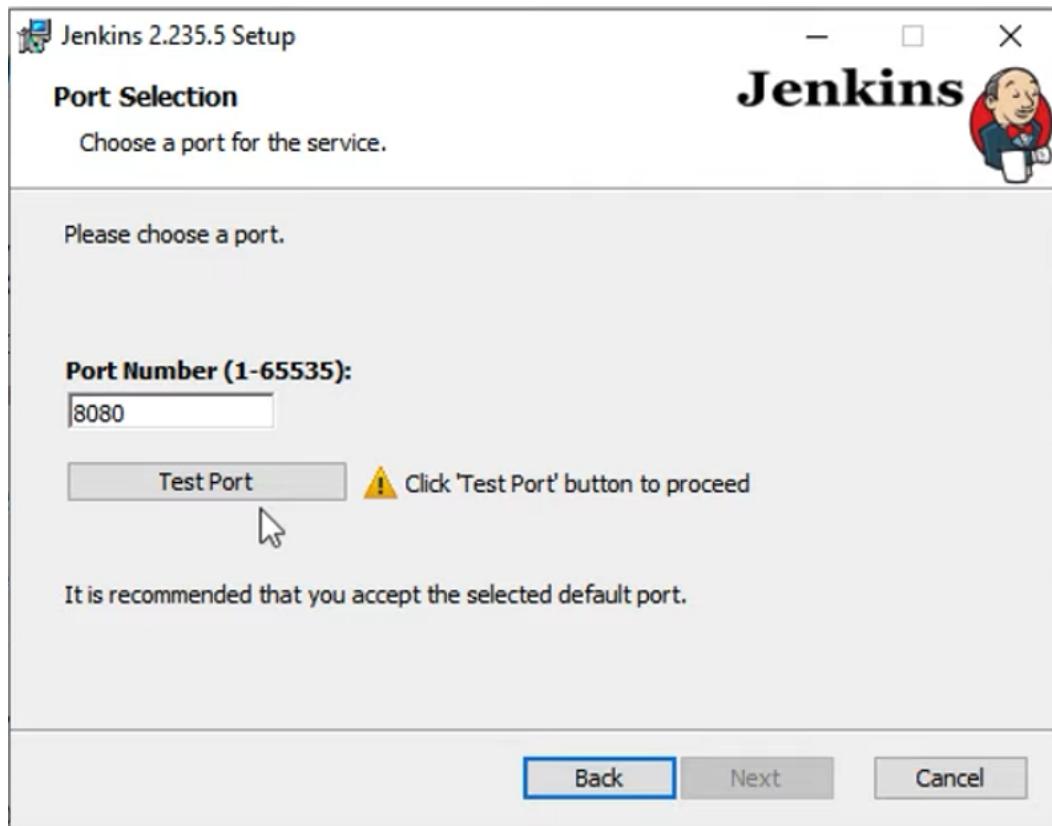


Fig.4

5. Test the port to see if it is available:

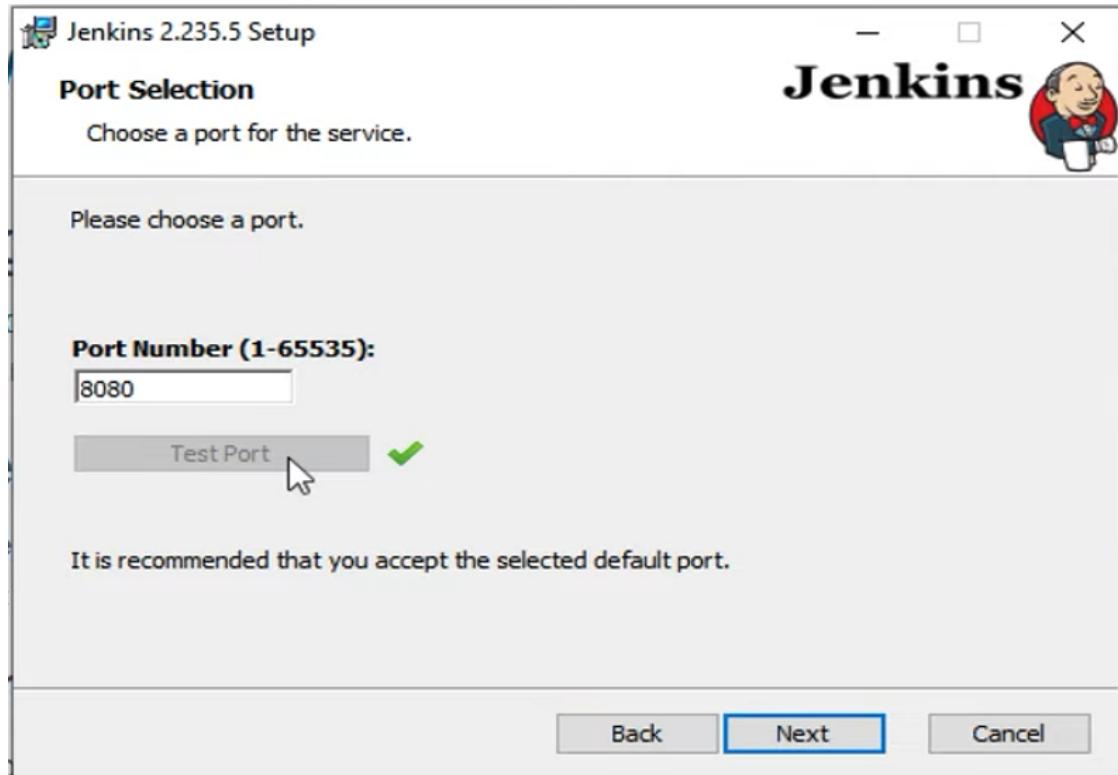


Fig.5

6. Select where your Java JDK is installed:

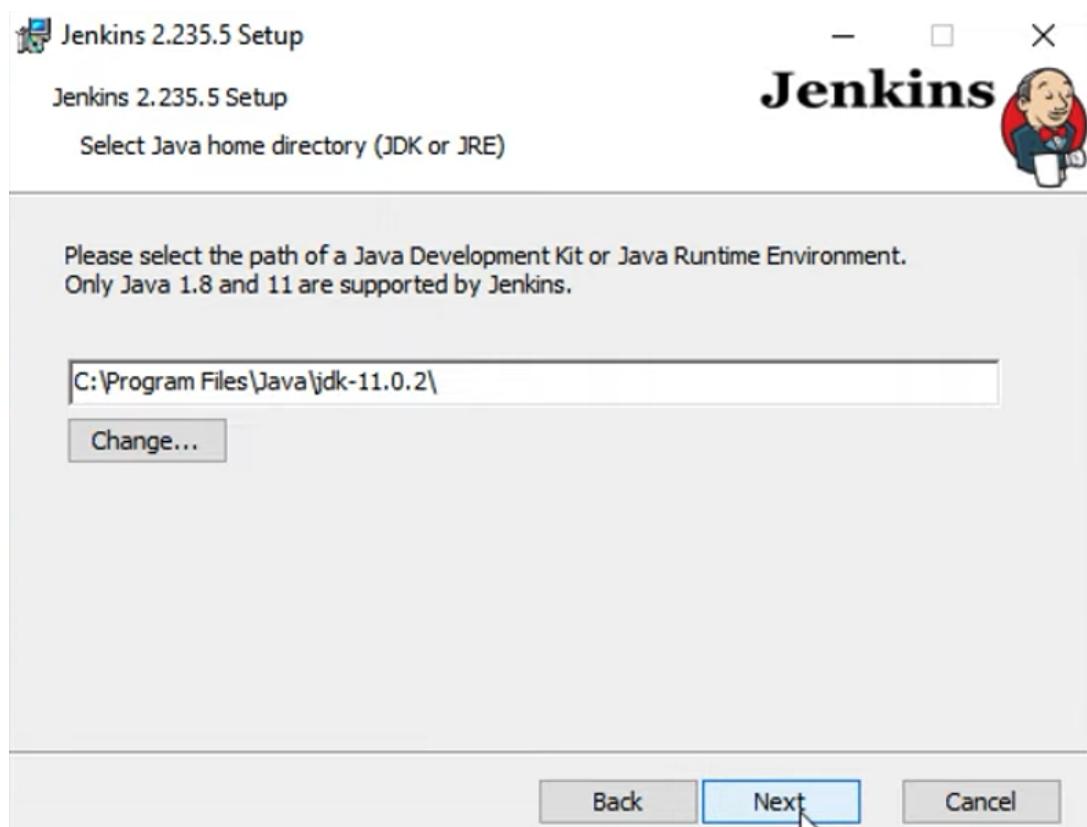
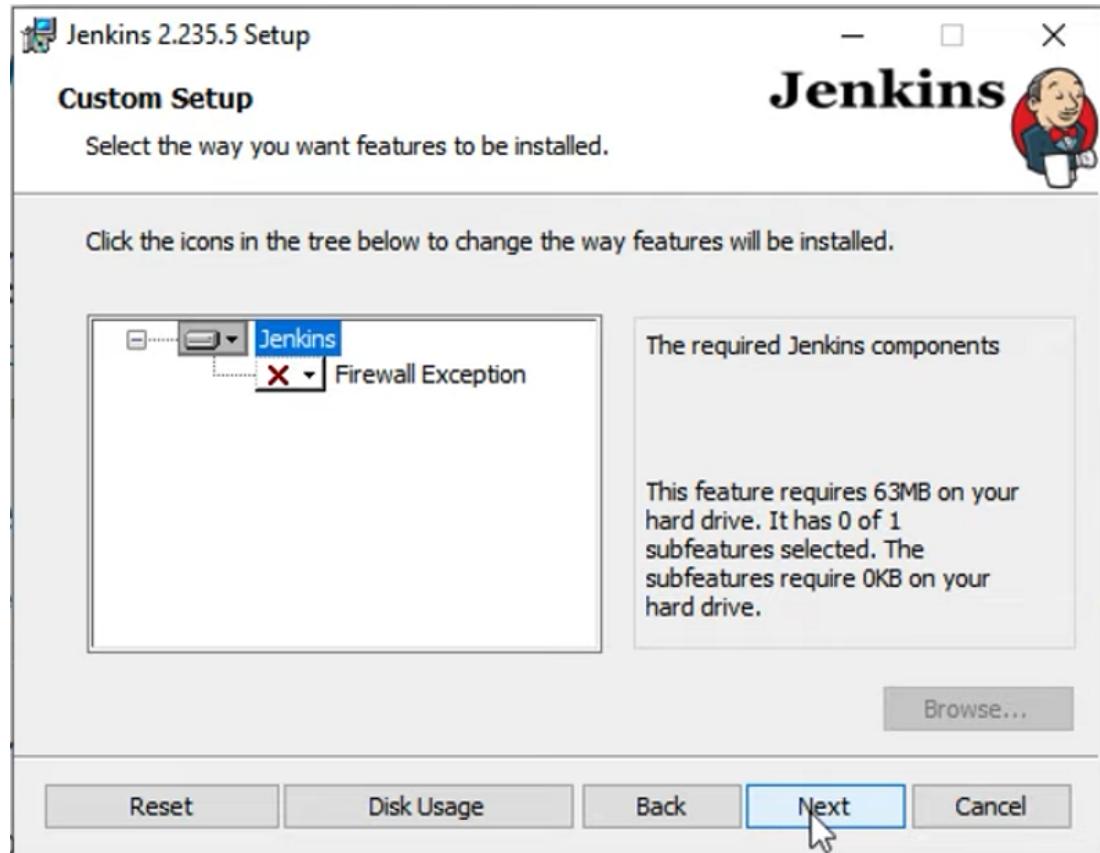


Fig.6

7. Click next and don't change Firewall Exception



8. Install it

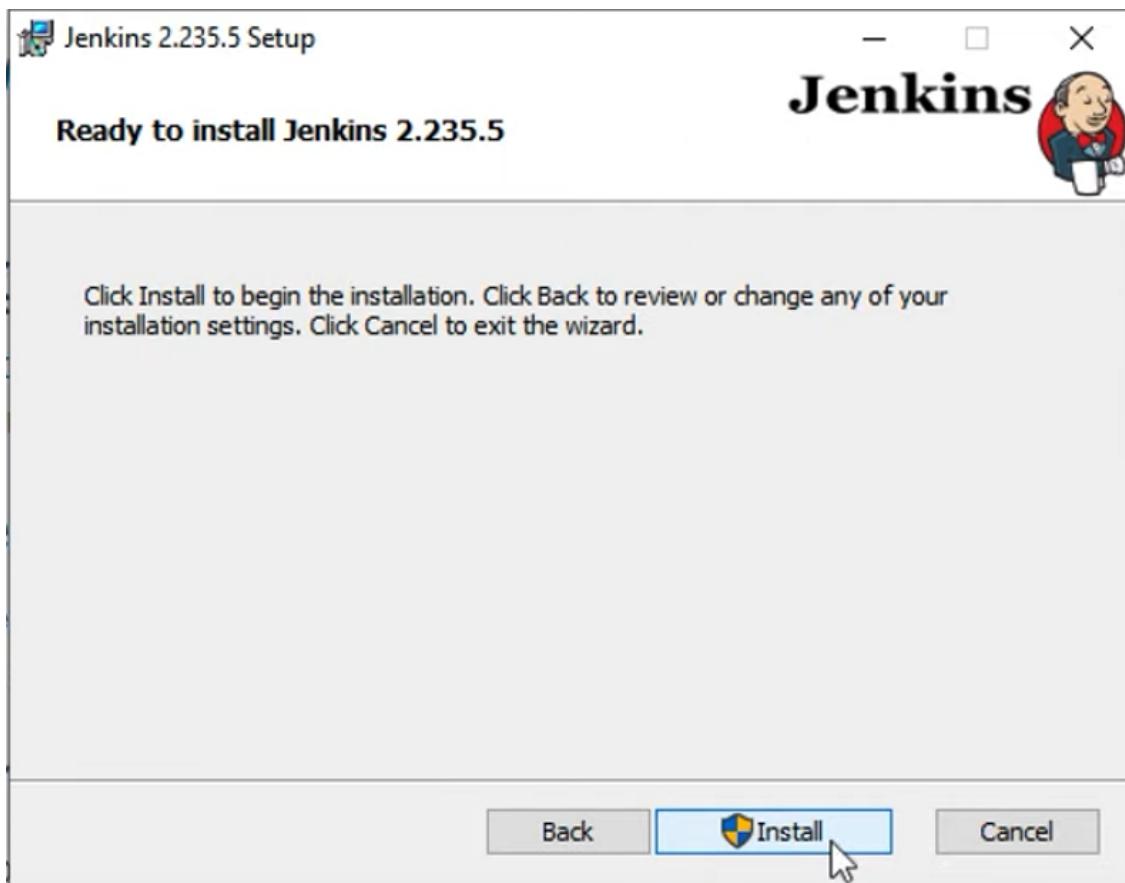


Fig.8

9. After install you need to unlock the Jenkins. Go to http://localhost:{PORT_NUMBER}/ In my case is <http://localhost:8080/>

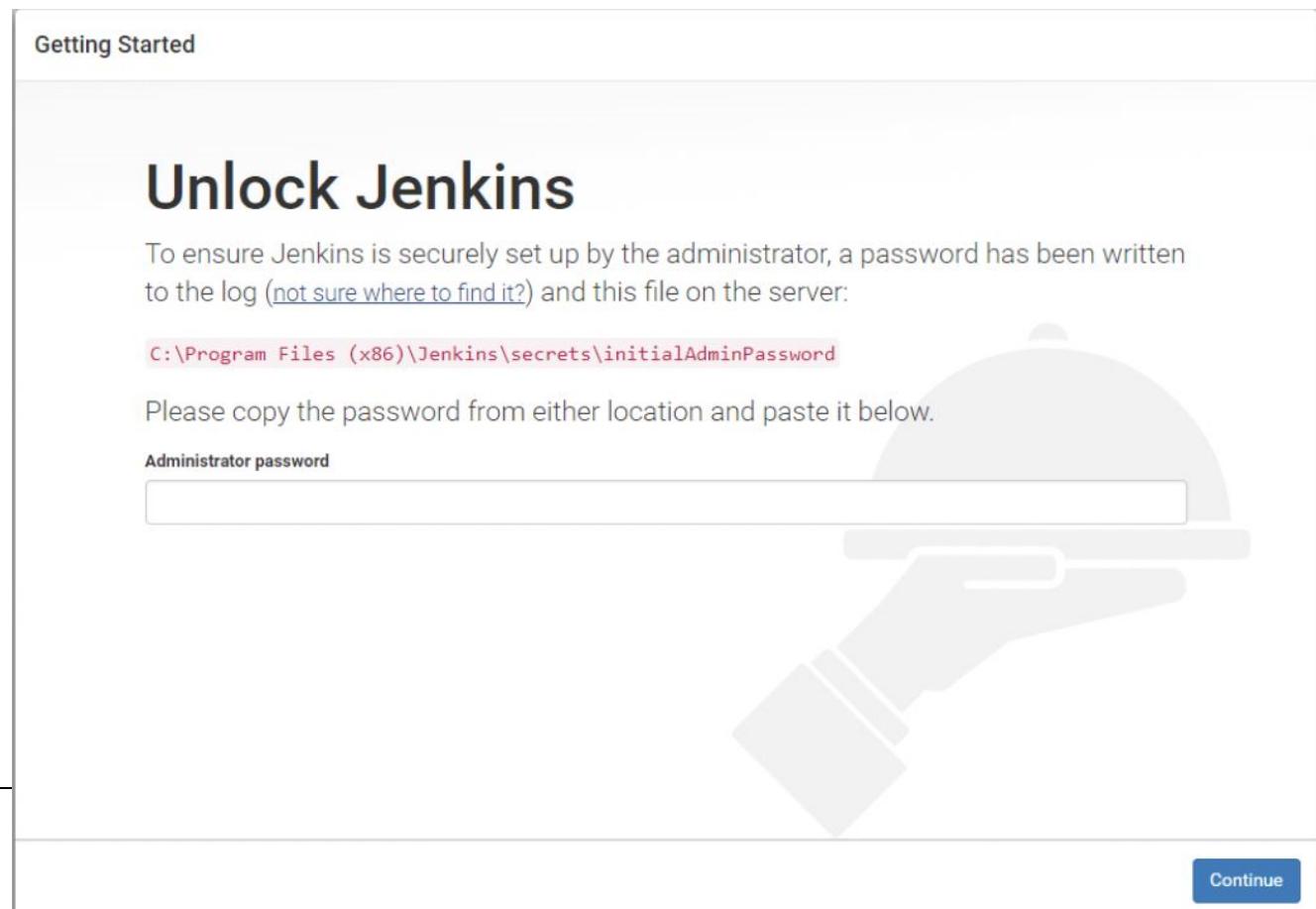


Fig.9

10.1. You will see a window which will require from you an initial administrator passsword For default installation location of the password is [C:\Program Files \(x86\)\Jenkins\secrets\initialAdminPassword](C:\Program Files (x86)\Jenkins\secrets\initialAdminPassword),

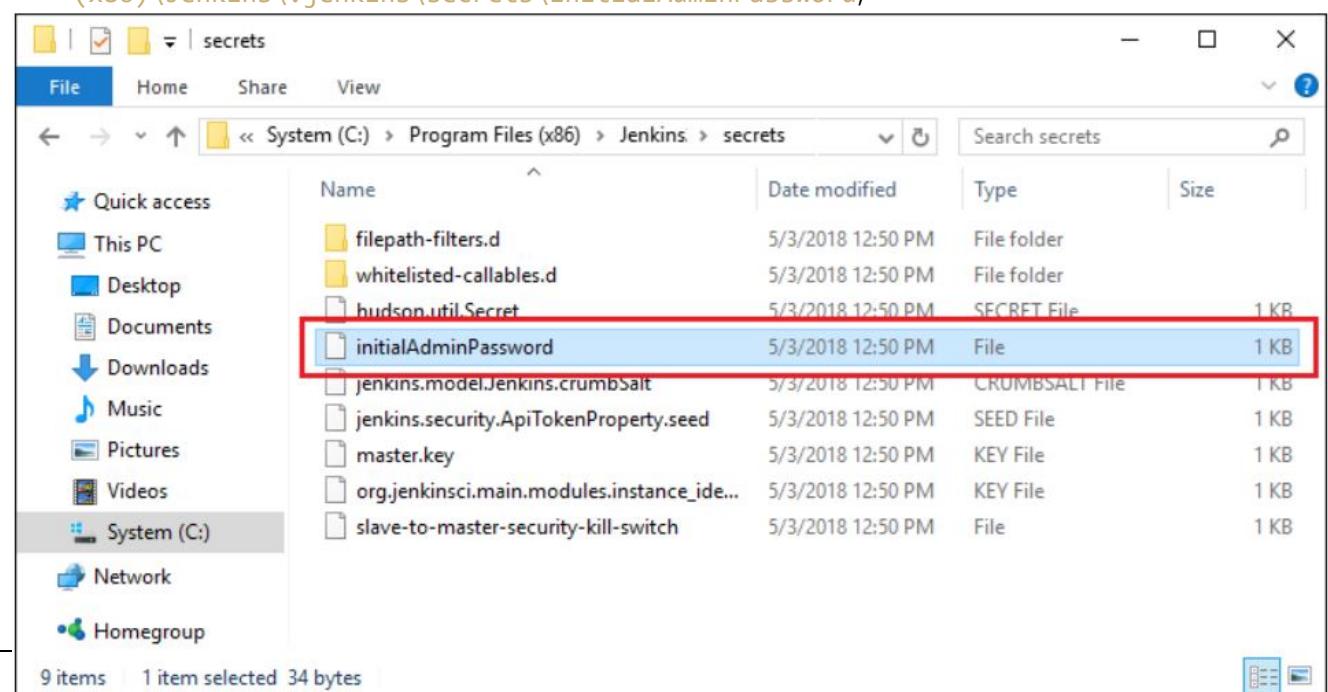


Fig.10

10.2. Copy and past it into the window dialog



A screenshot of a Windows Notepad window titled "initialAdminPassword - Notepad". The window contains the text "47a4a59b97d648ebbf88a6dda119587" and a cursor is visible at the bottom of the text area. The status bar at the bottom shows "Windows (CRLF)", "Ln 1, Col 1", and "100%".

Fig.11

11. After you enter the password you will have 2 options : [Install suggested plugins](#) or [Select plugins to install](#)

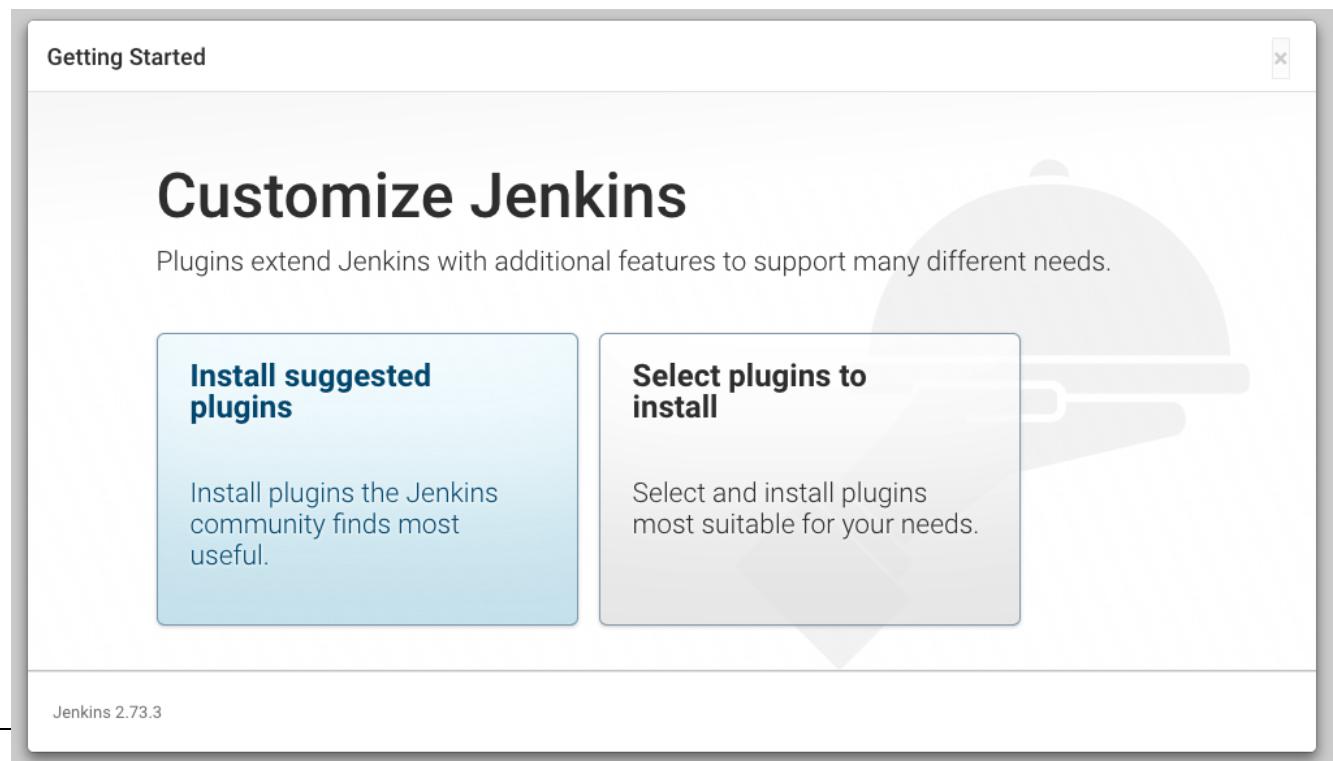


Fig.12

12. After you have chosen, the plugin will start to install and you will see the progress.

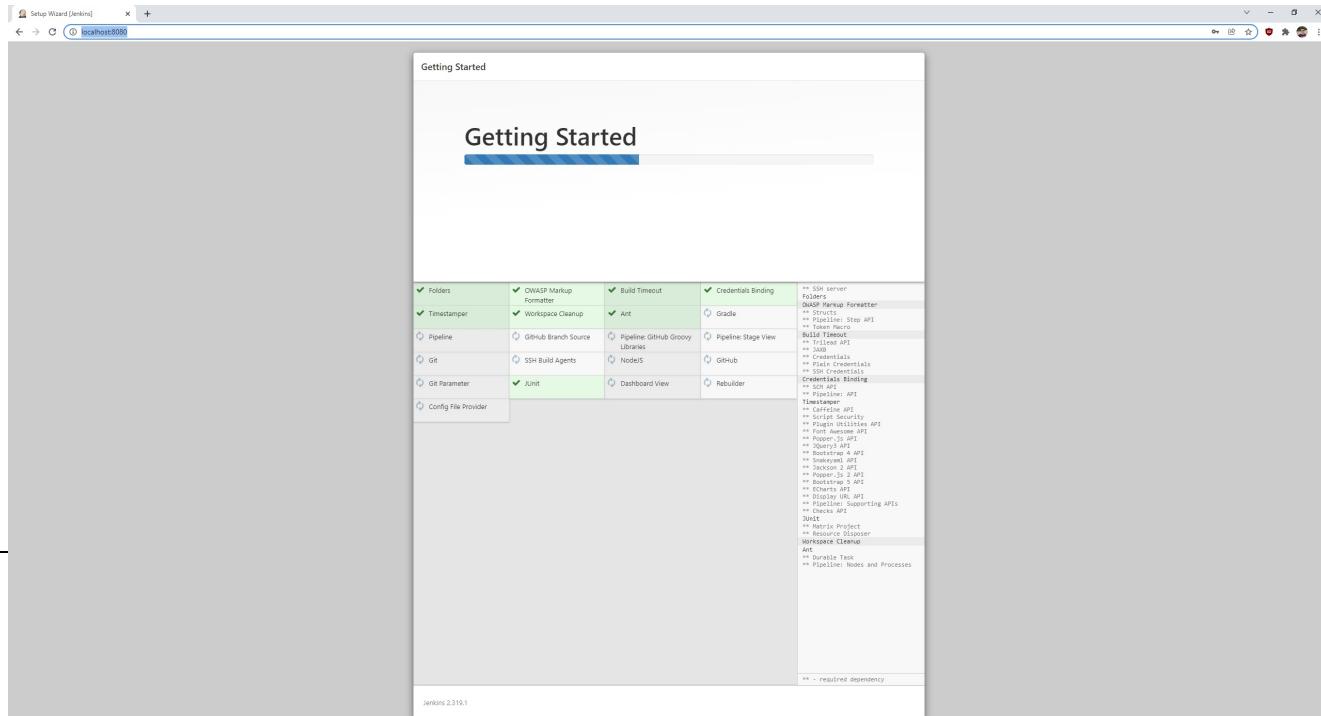


Fig.13

13. After the plugins have been installed you need to create an admin

Username:	<input type="text"/>
Password:	<input type="password"/>
Confirm password:	<input type="password"/>
Full name:	<input type="text"/>
E-mail address:	<input type="text"/>

Jenkins 2.319.1 Skip and continue as admin Save and Continue

Fig.14

14. Set the URL address and press **Save and Finish**

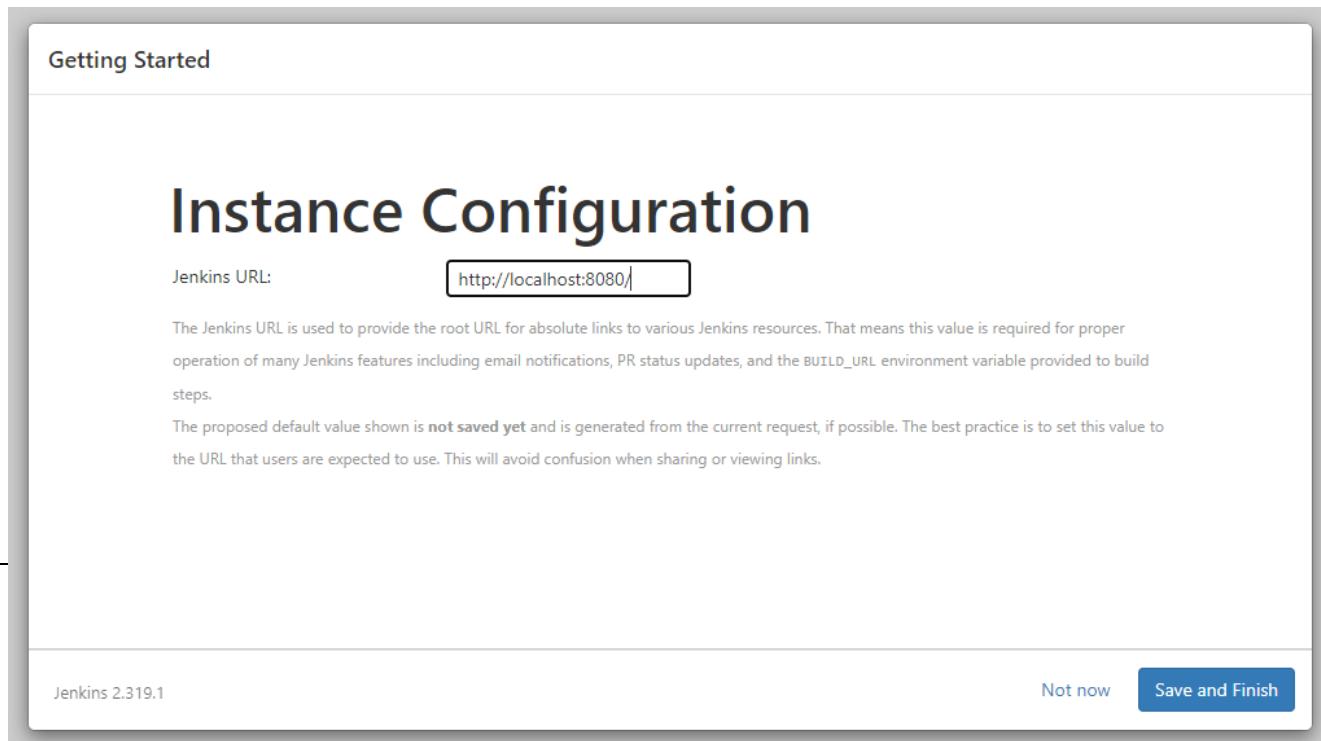


Fig.15

15. Now Jenkins is ready

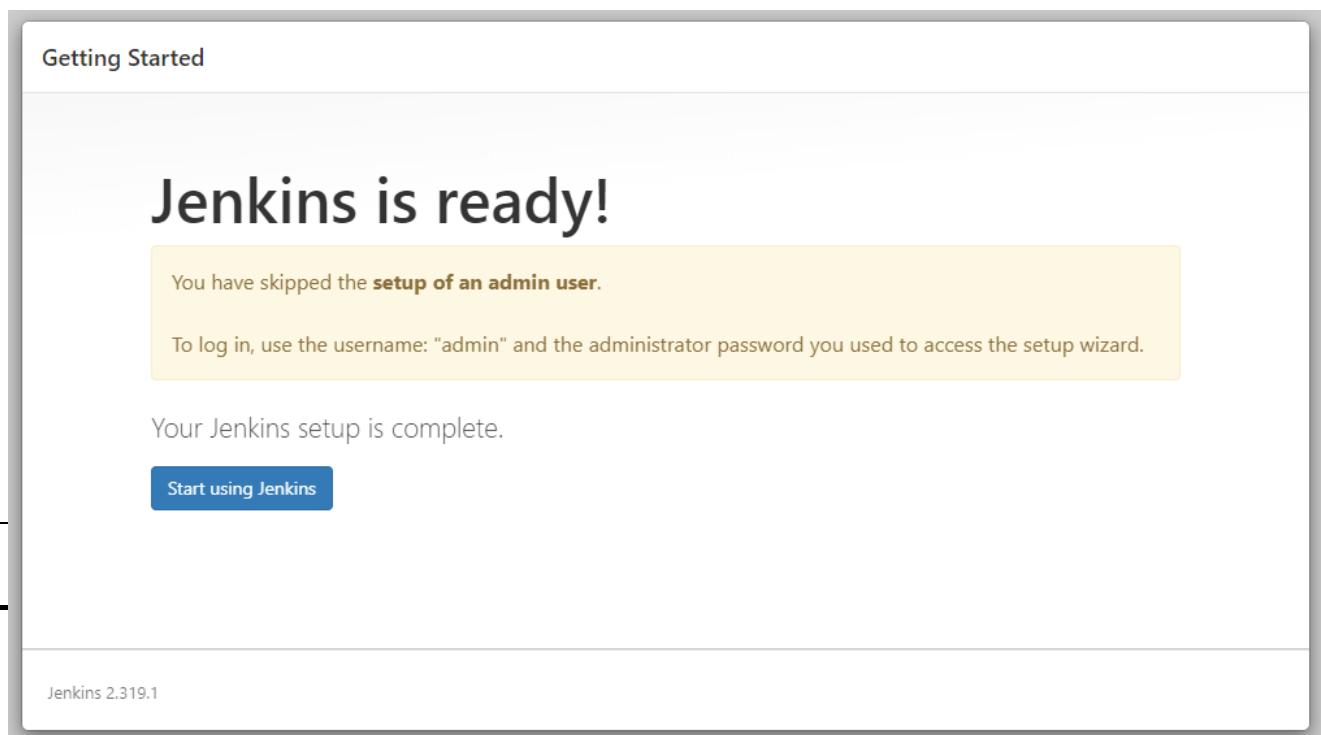


Fig.16

Homework 1

Steps

1. Go to <https://github.com/> and login into your account

2. Create a new repository. Click on **new**

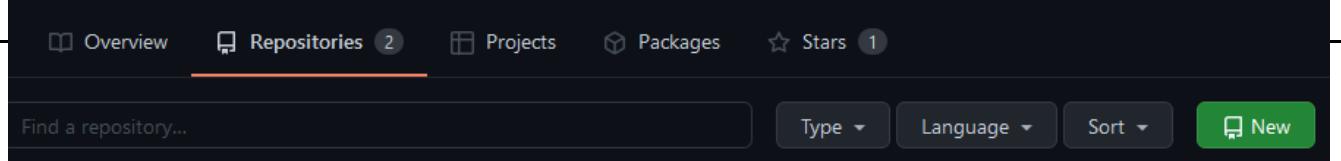


Fig.1

3. Choose a name and leave it as public. Press **Create Repository**

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner * mihaipreda **Repository name *** SQMA_Preda_Mihail_Irinel ✓

Great repository names are short and memorable. Need inspiration? How about [stunning-octo-enigma](#)?

Description (optional) SQMA homework Jenkins

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.

Initialize this repository with:

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

Add a README file This is where you can write a long description for your project. [Learn more](#).

Add .gitignore Choose which files not to track from a list of templates. [Learn more](#).

Choose a license A license tells others what they can and can't do with your code. [Learn more](#).

Create repository

Fig.2

4. Set the origin to your repo

```
Mihai@Mihai MINGW64 /e/ASE/Master/Anul_II/Semestrul_I/Source Code Management/homework/day2/SQMA_Preda_Mihail_Irinel (master)
$ git remote add origin https://github.com/mihaipreda/SQMA_Preda_Mihail_Irinel.git
```

Fig.3

5. Add a new job:

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

[Create a job](#)

Set up a distributed build

[Set up an agent](#)

[Configure a cloud](#)

[Learn more about distributed builds](#)

REST API Jenkins 2.319.1

Fig.4

6. Enter Job Name

Enter an item name

SQMA_Preda_Mihail_Irinel_Jenkins_1
» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

OK

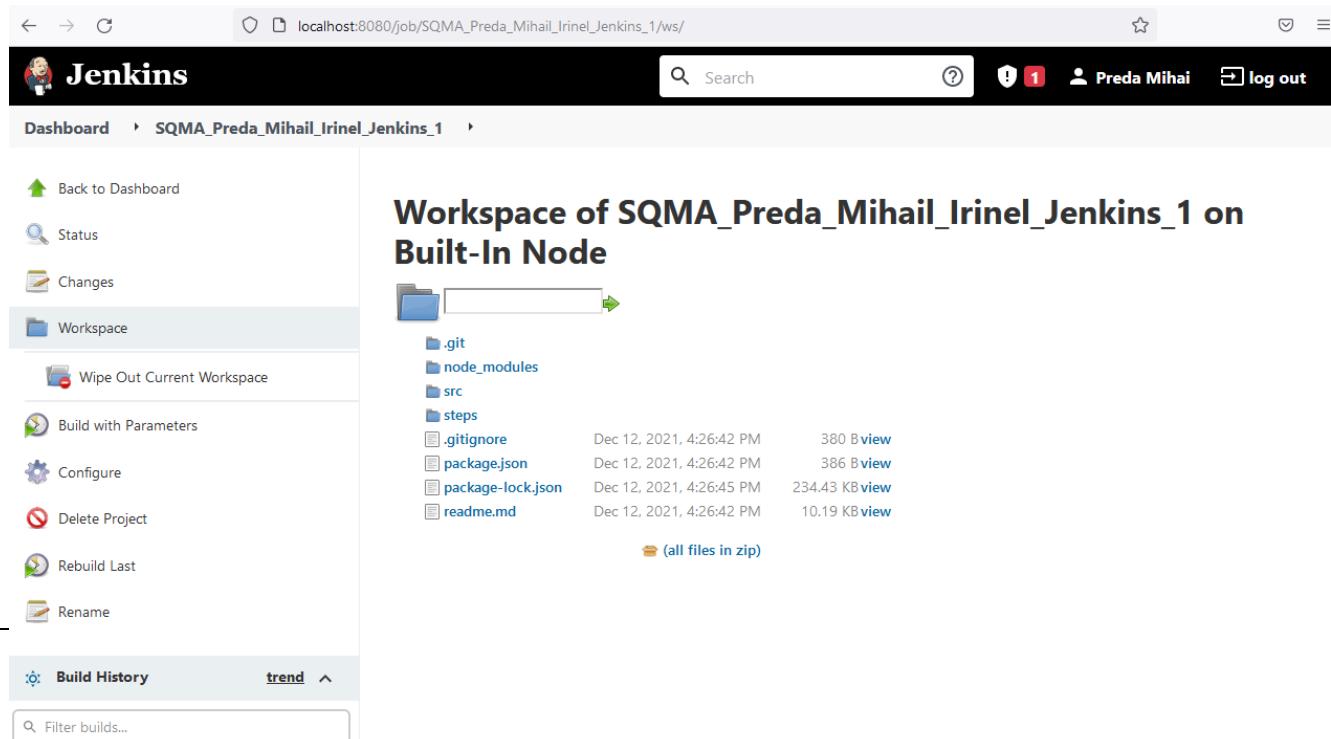
Fig.5

7. Enter a description. Go to **Source Code Management**, check **Git**, put the link to the repository and click **Save**

The screenshot shows the Jenkins job configuration page for 'SQMA_Preda_Mihail_Irinel_Jenkins_1'. The 'General' tab is selected. In the 'Description' field, the text 'Jenkins Job 1 ~~SQMA~~ homework' is entered. Under 'Source Code Management', the 'Git' option is selected, and the 'Repository URL' is set to 'https://github.com/mihailpreda/SQMA_Preda_Mihail_Irinel.git'. The 'Save' button at the bottom is highlighted with a green border.

Fig.6

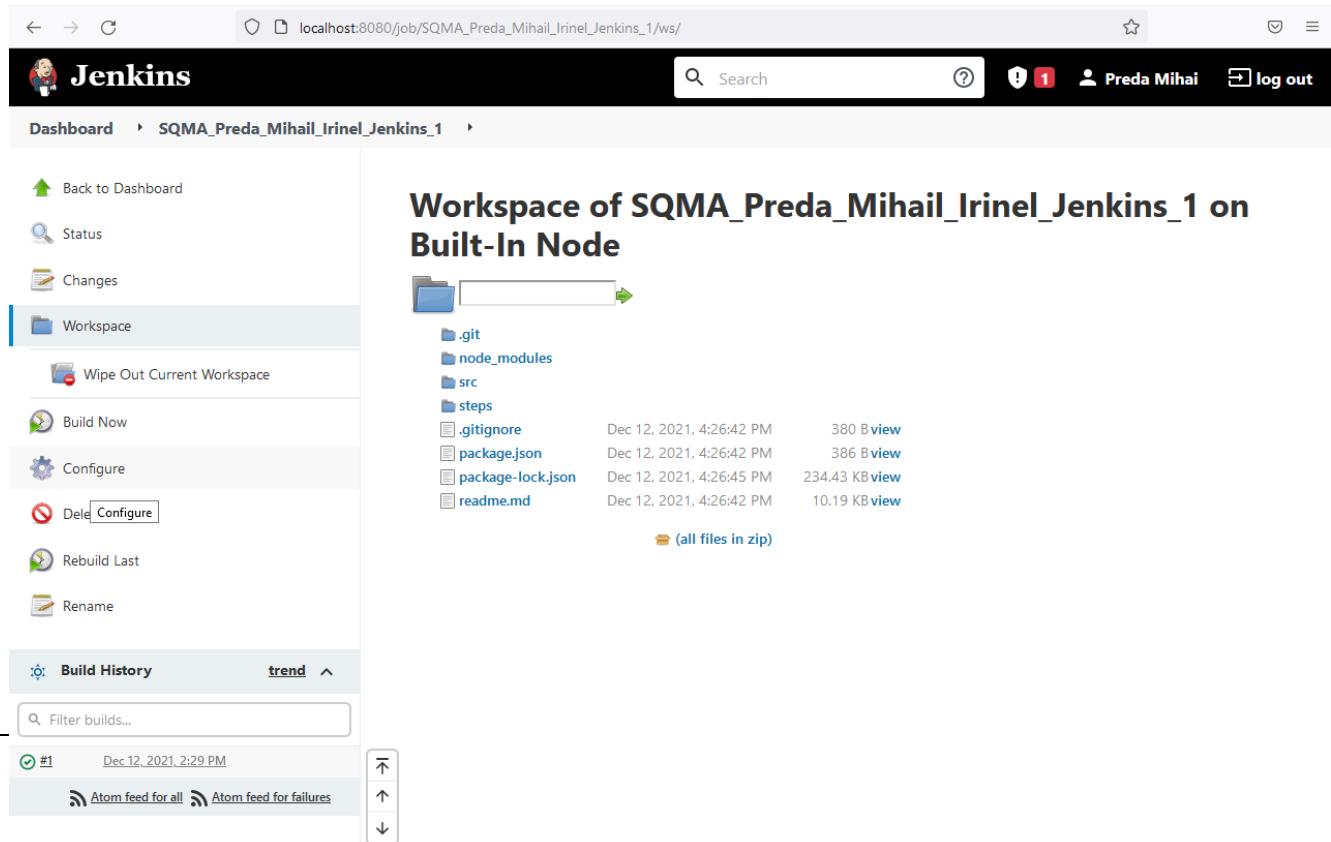
8. If everything is okey you will seein the workspace that the Jenkins successfully downloaded the files from Github.



The screenshot shows the Jenkins workspace interface for the job 'SQMA_Preda_Mihail_Irinel_Jenkins_1'. The left sidebar has 'Workspace' selected. The main area displays the contents of the workspace, which include a '.git' folder, 'node_modules', 'src', 'steps', '.gitignore', 'package.json', 'package-lock.json', and 'readme.md'. Each file is listed with its last modified date, size, and a 'view' link. A 'zip' button is also present.

Fig.7

9. Then go to **Configure**



This screenshot is similar to Fig.7, showing the Jenkins workspace for the same job. However, the 'Configure' option in the sidebar is now highlighted with a blue border. The rest of the interface is identical, showing the workspace contents and build history.

Fig.8

10. Go to **General** and tick **This project is parameterized**. After that you put the parameters that you have. In my case I have 2 test suits, each one is run by specifying `npm run testA` or `npm run testB`

The screenshot shows the Jenkins job configuration interface for a job named "SQMA_Preda_Mihail_Irinel_Jenkins_1". The "General" tab is selected. Under "Rebuild options", the checkbox "This project is parameterized" is checked. A "Choice Parameter" section is expanded, showing a parameter named "whichTest" with two choices: "testA" and "testB". There is also a "Description" field which is currently empty. At the bottom of the configuration page, there are "Save" and "Apply" buttons.

Fig.9

11. Then go to **Build**, add a **Windows batch command** and add the following line `npm install && npm run %whichTest%`

The screenshot shows the Jenkins job configuration page for 'SQMA_Preda_Mihail_Irinel_Jenkins_1'. The 'Build Environment' tab is selected. Under 'Build', there is a step titled 'Execute Windows batch command' with the command: `npm install && npm run %whichTest%`. The 'Post-build Actions' section is also visible.

Fig.10

12. Click on Build with parameters

The screenshot shows the Jenkins interface for the project "SQMA_Preda_Mihail_Irinel_Jenkins_1". The left sidebar contains links for Back to Dashboard, Status, Changes, Workspace, Build with Parameters (which is selected), Configure, Delete Project, Rebuild Last, Rename, and Build History. The Build History section lists builds from #10 down to #2, with build #10 being the most recent. The main content area displays the project name, a brief description ("Jenkins Job 1 SQMA homework"), and links for Workspace and Recent Changes. It also includes sections for Permalinks and a list of recent builds.

Project SQMA_Preda_Mihail_Irinel_Jenkins_1

Jenkins Job 1 SQMA homework

[Edit description](#) [Disable Project](#)

Workspace **Recent Changes**

Permalinks

- Last build (#10), 10 min ago
- Last stable build (#10), 10 min ago
- Last successful build (#10), 10 min ago
- Last failed build (#5), 22 min ago
- Last unsuccessful build (#5), 22 min ago
- Last completed build (#10), 10 min ago

Build History [trend](#)

#	Build Number	Date
10	#10	Dec 12, 2021, 2:48 PM
9	#9	Dec 12, 2021, 2:48 PM
8	#8	Dec 12, 2021, 2:40 PM
7	#7	Dec 12, 2021, 2:39 PM
6	#6	Dec 12, 2021, 2:39 PM
5	#5	Dec 12, 2021, 2:37 PM
4	#4	Dec 12, 2021, 2:36 PM
3	#3	Dec 12, 2021, 2:35 PM
2	#2	Dec 12, 2021, 2:34 PM

[Atom feed for all](#) [Atom feed for failures](#)

Fig.11

13. Choose **testA**, then press **Build**.

The screenshot shows the Jenkins web interface for the project "SQMA_Preda_Mihail_Irinel_Jenkins_1". The left sidebar contains navigation links: Back to Dashboard, Status, Changes, Workspace, Build with Parameters, Configure, Delete Project, Rebuild Last, and Rename. The main content area is titled "Project SQMA_Preda_Mihail_Irinel_Jenkins_1". It displays the message "This build requires parameters: whichTest" followed by a dropdown menu set to "testA". A large blue "Build" button is visible. Below this, the "Build History" section is shown, listing ten builds from Dec 12, 2021, at various times. Builds #10, #9, #8, #7, #6, #4, and #3 are marked with green checkmarks, indicating success. Builds #5, #2, and #1 are marked with red X's, indicating failure. At the bottom of the history table are links for "Atom feed for all" and "Atom feed for failures".

#	Build Number	Timestamp	Status
1	#10	Dec 12, 2021, 2:48 PM	Success
2	#9	Dec 12, 2021, 2:48 PM	Success
3	#8	Dec 12, 2021, 2:40 PM	Success
4	#7	Dec 12, 2021, 2:39 PM	Success
5	#6	Dec 12, 2021, 2:39 PM	Failure
6	#5	Dec 12, 2021, 2:37 PM	Failure
7	#4	Dec 12, 2021, 2:36 PM	Success
8	#3	Dec 12, 2021, 2:35 PM	Failure
9	#2	Dec 12, 2021, 2:34 PM	Failure

Fig.12

14. Click on the latest done build, go to **Console Output** and see the results :

The screenshot shows the Jenkins interface for a job named "SQMA_Preda_Mihail_Irinel_Jenkins_1". The current build number is #11. On the left, there's a sidebar with various build-related links: Back to Project, Status, Changes, **Console Output** (which is selected and highlighted in blue), View as plain text, Edit Build Information, Delete build '#11', Parameters, Git Build Data, Rebuild, and Previous Build. The main content area is titled "Console Output" with a green checkmark icon. It displays the command-line output of the build process. The output shows the build was started by user "Preda Mihai" and ran as SYSTEM. It details the cloning of the repository from GitHub, the execution of npm install, and the running of Jest tests. The test results indicate 1 skipped, 1 passed, and 4 total tests.

```
Started by user Preda Mihai
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\SQMA_Preda_Mihail_Irinel_Jenkins_1
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\SQMA_Preda_Mihail_Irinel_Jenkins_1.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/mihailpreda/SQMA_Preda_Mihail_Irinel.git # timeout=10
Fetching upstream changes from https://github.com/mihailpreda/SQMA_Preda_Mihail_Irinel.git
> git.exe --version # timeout=10
> git --version # 'git' version 2.34.1.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/mihailpreda/SQMA_Preda_Mihail_Irinel.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision ea5f0885fb7a428540a1ccb035d12c1d5c0b6bff (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f ea5f0885fb7a428540a1ccb035d12c1d5c0b6bff # timeout=10
Commit message: "updated"
> git.exe rev-list --no-walk ea5f0885fb7a428540a1ccb035d12c1d5c0b6bff # timeout=10
[SQMA_Preda_Mihail_Irinel_Jenkins_1] $ cmd /c call C:\Windows\TEMP\jenkins16172269540097053974.bat

C:\ProgramData\Jenkins\.jenkins\workspace\SQMA_Preda_Mihail_Irinel_Jenkins_1>npm install && npm run testA

up to date, audited 327 packages in 823ms

24 packages are looking for funding
  run 'npm fund' for details

found 0 vulnerabilities

> sqma_preda_mihail_irinel@1.0.0 testA
> jest -t CalculatorAdditionSubtraction

PASS src/CalculatorAdditionSubtraction.test.js

Test Suites: 1 skipped, 1 passed, 1 of 2 total
Tests:       2 skipped, 2 passed, 4 total
Snapshots:   0 total
Time:        0.378 s, estimated 1 s
Ran all test suites with tests matching "CalculatorAdditionSubtraction".
Finished: SUCCESS
```

Fig.13

15. Click on Build with parameters

The screenshot shows the Jenkins interface for the project "SQMA_Preda_Mihail_Irinel_Jenkins_1". The left sidebar contains navigation links: Back to Dashboard, Status, Changes, Workspace, Build with Parameters (which is selected and highlighted in blue), Configure, Delete Project, Rebuild Last, Rename, and Build History. The Build History section lists the last ten builds, with build #10 being the most recent successful one. The main content area displays the project name "Project SQMA_Preda_Mihail_Irinel_Jenkins_1" and its description "Jenkins Job 1 SQMA homework". It includes links to Workspace and Recent Changes, and buttons for Edit description and Disable Project. Below the workspace link, there is a section titled "Permalinks" with a list of build links.

Build	Date
#10	Dec 12, 2021, 2:48 PM
#9	Dec 12, 2021, 2:48 PM
#8	Dec 12, 2021, 2:40 PM
#7	Dec 12, 2021, 2:39 PM
#6	Dec 12, 2021, 2:39 PM
#5	Dec 12, 2021, 2:37 PM
#4	Dec 12, 2021, 2:36 PM
#3	Dec 12, 2021, 2:35 PM
#2	Dec 12, 2021, 2:34 PM

Permalinks

- Last build (#10), 10 min ago
- Last stable build (#10), 10 min ago
- Last successful build (#10), 10 min ago
- Last failed build (#5), 22 min ago
- Last unsuccessful build (#5), 22 min ago
- Last completed build (#10), 10 min ago

Fig.14

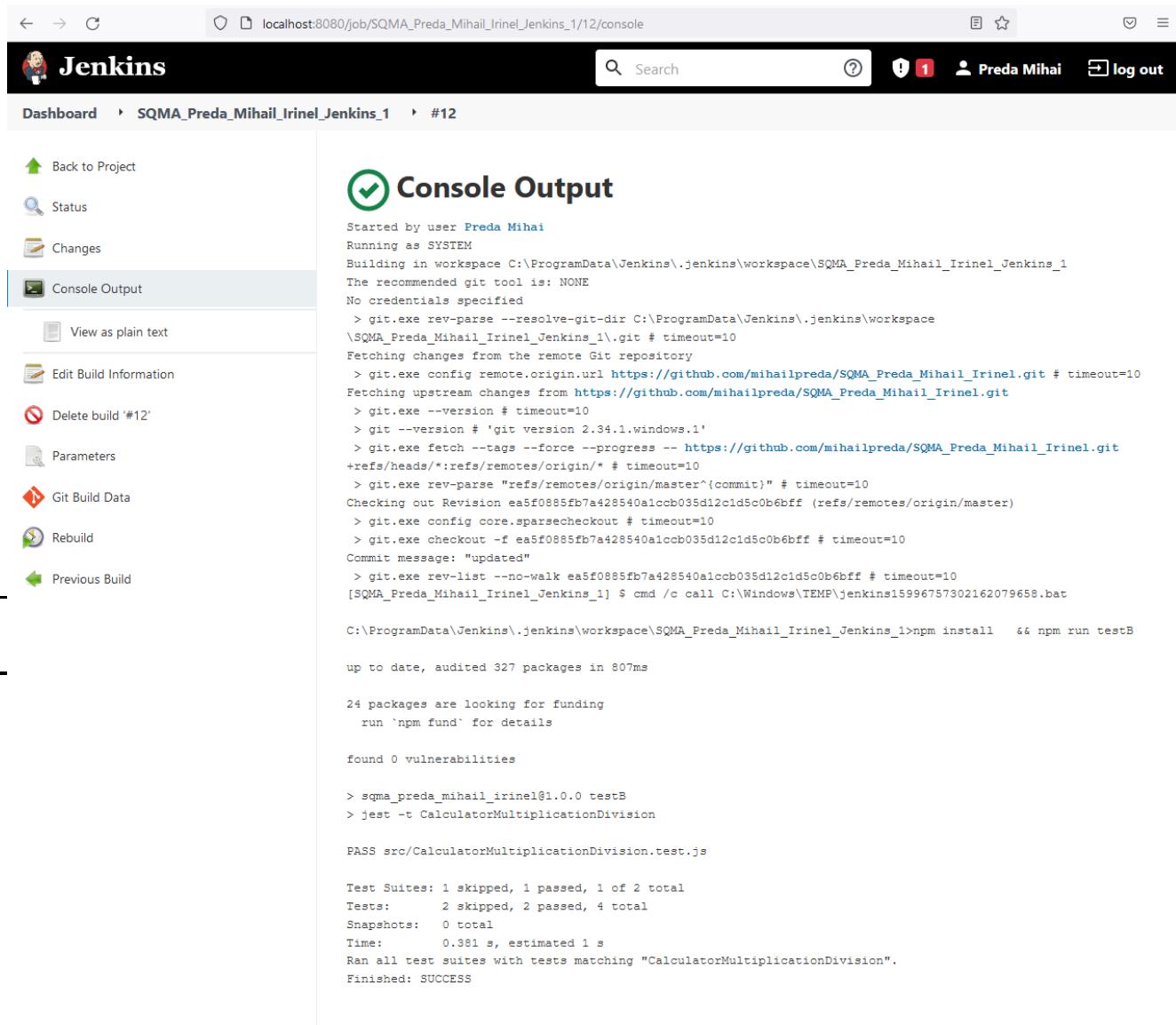
16. Choose **testB**, then press **Build**.

The screenshot shows the Jenkins interface for the project "SQMA_Preda_Mihail_Irinel_Jenkins_1". The left sidebar contains navigation links: Back to Dashboard, Status, Changes, Workspace, Build with Parameters, Configure, Delete Project, Rebuild Last, and Rename. The main content area displays the project name "Project SQMA_Preda_Mihail_Irinel_Jenkins_1". It indicates that this build requires parameters, specifically "whichTest", which is currently set to "testB". A large blue "Build" button is present. Below this, the "Build History" section lists previous builds, each with a status icon (green checkmark for successful, red X for failed) and the build number. The most recent build, #11, was successful and ran on Dec 12, 2021, at 3:00 PM. The build history also includes entries for builds #10, #9, #8, #7, #6, #5, #4, #3, and #2, with #5, #3, and #2 failing. At the bottom of the history section are links for "Atom feed for all" and "Atom feed for failures".

Build Number	Status	Date
#11	Success	Dec 12, 2021, 3:00 PM
#10	Success	Dec 12, 2021, 2:48 PM
#9	Success	Dec 12, 2021, 2:48 PM
#8	Success	Dec 12, 2021, 2:40 PM
#7	Success	Dec 12, 2021, 2:39 PM
#6	Success	Dec 12, 2021, 2:39 PM
#5	Failure	Dec 12, 2021, 2:37 PM
#4	Success	Dec 12, 2021, 2:36 PM
#3	Failure	Dec 12, 2021, 2:35 PM
#2	Failure	Dec 12, 2021, 2:34 PM

Fig.15

17. Click on the latest done build, go to **Console Output** and see the results :



The screenshot shows the Jenkins interface for a job named "SQMA_Preda_Mihail_Irinel_Jenkins_1". The "Console Output" tab is selected in the left sidebar. The main content area displays the build log for build #12. The log shows the following command-line output:

```
Started by user Preda Mihai
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\SQMA_Preda_Mihail_Irinel_Jenkins_1
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\SQMA_Preda_Mihail_Irinel_Jenkins_1\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/mihailpreda/SQMA_Preda_Mihail_Irinel.git # timeout=10
Fetching upstream changes from https://github.com/mihailpreda/SQMA_Preda_Mihail_Irinel.git
> git.exe --version # timeout=10
> git --version # 'git' version 2.34.1.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/mihailpreda/SQMA_Preda_Mihail_Irinel.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision ea5f0885fb7a428540a1ccb035d12c1d5c0b6bff (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f ea5f0885fb7a428540a1ccb035d12c1d5c0b6bff # timeout=10
Commit message: "updated"
> git.exe rev-list --no-walk ea5f0885fb7a428540a1ccb035d12c1d5c0b6bff # timeout=10
[SQMA_Preda_Mihail_Irinel_Jenkins_1] $ cmd /c call C:\Windows\TEMP\jenkins15996757302162079658.bat

C:\ProgramData\Jenkins\.jenkins\workspace\SQMA_Preda_Mihail_Irinel_Jenkins_1>npm install && npm run testB
up to date, audited 327 packages in 807ms

24 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

> sqma_preda_mihail_irinel@1.0.0 testB
> jest -t CalculatorMultiplicationDivision

PASS src/CalculatorMultiplicationDivision.test.js

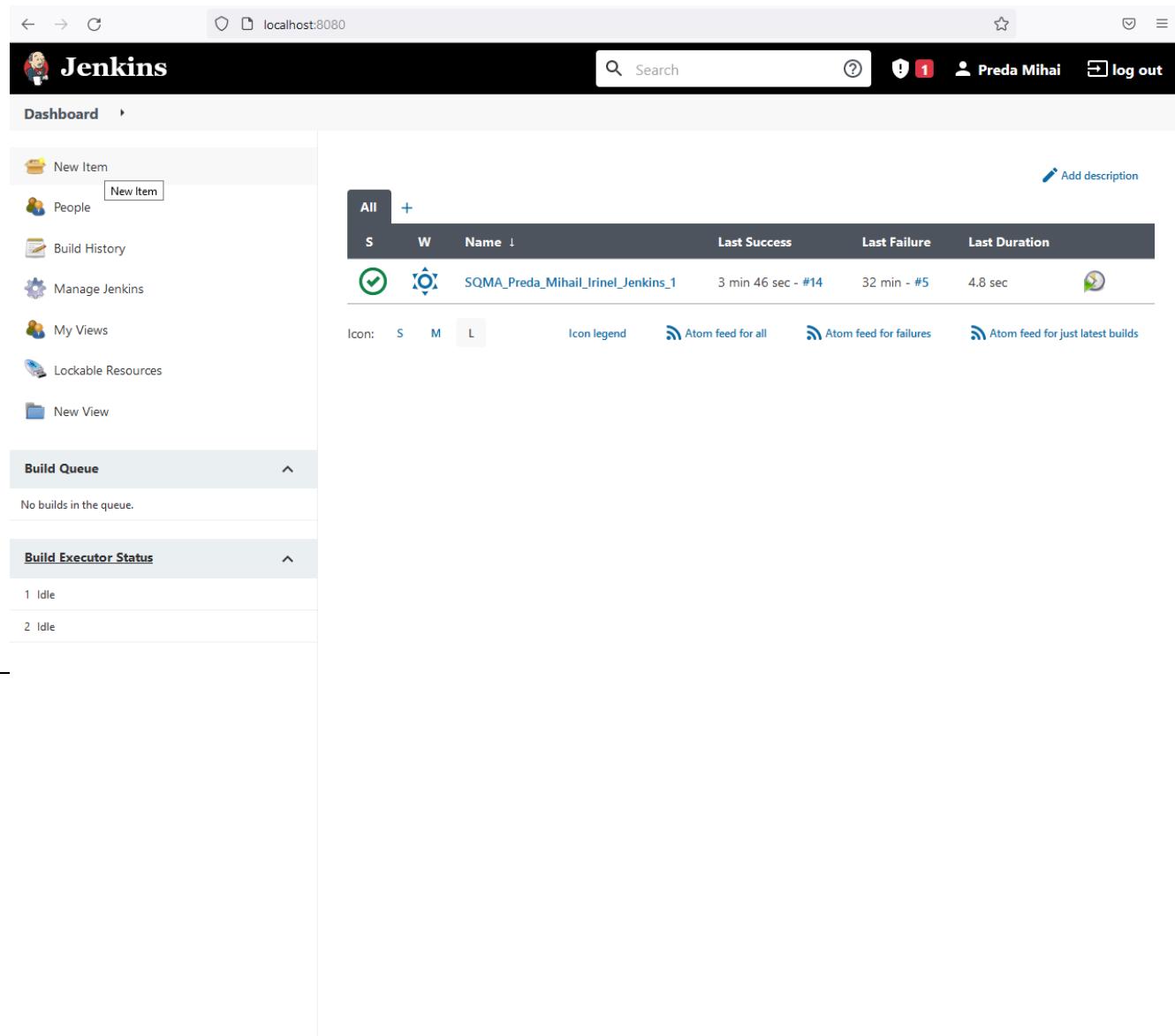
Test Suites: 1 skipped, 1 passed, 1 of 2 total
Tests:       2 skipped, 2 passed, 4 total
Snapshots:  0 total
Time:        0.381 s, estimated 1 s
Ran all test suites with tests matching "CalculatorMultiplicationDivision".
Finished: SUCCESS
```

Fig.16

Homework 2

Steps

1. From **Dashboard**, click on **New Item**



The screenshot shows the Jenkins dashboard at localhost:8080. The left sidebar contains links for 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', 'Lockable Resources', and 'New View'. The main area displays a table of build items. The first item in the table is 'SQMA_Preda_Mihail_Irinel_Jenkins_1', which has a green checkmark icon, indicating it is successful. The table includes columns for 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. Below the table, there are icons for 'S' (Stable), 'M' (Medium), and 'L' (Large), an 'Icon legend', and three RSS feed links: 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds'.

Fig.1

2. Enter a name for the item, select **Pipeline** and press **OK**

The screenshot shows the Jenkins interface for creating a new item. At the top, there's a navigation bar with links for 'Dashboard' and 'All'. The main area has a title 'Enter an item name' and a text input field containing 'SQMA_Preda_Mihail_Irinel_Jenkins_2_Pipeline', which is noted as a 'Required field'. Below this, there are several project types listed with their icons and descriptions:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**: Creates a set of multibranch project subfolders by scanning for repositories.

At the bottom of the dialog, there's a note: 'If you want to create a new item from other existing, you can use this option:' followed by a 'Copy from' button and a 'Type to autocomplete' input field. A large blue 'OK' button is at the bottom right.

Fig.2

3. Add a description

The screenshot shows the Jenkins Pipeline configuration page for a job named "SQMA_Preda_Mihail_Irinel_Jenkins_2_Pipeline". The "General" tab is selected. In the "Description" field, the text "SOMA homework 2 Pipeline that will run all tests" is entered. Below the description, there are sections for "Build Triggers" and "Advanced Project Options". The "Build Triggers" section includes options like "Build after other projects are built", "Build periodically", and "GitHub hook trigger for GITScm polling". The "Advanced Project Options" section has a "Pipeline" tab selected, which contains "Save" and "Apply" buttons.

Fig.3

4. Scroll down to **Pipeline** and Add a **Pipeline script**. To do this click on **Pipeline Syntax**

The screenshot shows the Jenkins configuration interface for a project named "SQMA_Preda_Mihail_Irinel_Jenkins_2_Pipeline". The "Build Triggers" tab is selected, displaying various trigger options like "Build after other projects are built" and "GitHub hook trigger for GITScm polling". Below it, the "Pipeline" section is expanded, showing the "Definition" tab with "Pipeline script" selected. A Groovy script editor window displays the following code:

```
script
  1
  2
```

Below the script editor, there are checkboxes for "Use Groovy Sandbox" (which is checked) and "Pipeline Syntax". At the bottom, there are "Save" and "Apply" buttons.

Fig.4

5. Now, select **Snippet Generator** from the left handside, then in the main page at **Sample step** select **build: Build a job**. After that, write the project to be build. Select with what parameter do you want the job and then click on **Generate Pipeline Script**

The screenshot shows the Jenkins Pipeline Syntax Snippet Generator interface. On the left, a sidebar menu includes 'Back', 'Snippet Generator' (which is selected and highlighted in blue), and other links like 'Declarative Directive Generator', 'Declarative Online Documentation', 'Steps Reference', 'Global Variables Reference', 'Online Documentation', 'Examples Reference', and 'IntelliJ IDEA GDSL'. The main content area has a title 'Overview' with a descriptive paragraph about the Snippet Generator. Below it is a section titled 'Steps' under 'Sample Step'. A dropdown menu shows 'build: Build a job'. Underneath are fields for 'Project to Build' (containing 'SQMA_Preda_Mihail_Irinel_Jenkins_1'), 'Wait for completion' (checked), 'Propagate errors' (checked), and 'Quiet period' (empty). A 'Parameters' section shows 'whichTest' with a dropdown menu containing 'testA'. At the bottom is a large blue button labeled 'Generate Pipeline Script'. Below the button, a code preview window displays the generated Jenkins Pipeline script: `build job: 'SQMA_Preda_Mihail_Irinel_Jenkins_1', parameters: [string(name: 'whichTest', value: 'testA')]`. The bottom of the page features a 'Global Variables' section with a note about unsupported features.

Fig.5

The screenshot shows the Jenkins Pipeline Syntax Snippet Generator interface. At the top, there's a navigation bar with links for 'Dashboard', 'SQMA_Preda_Mihail_Irinel_Jenkins_2_Pipeline', and 'Pipeline Syntax'. On the right side of the header, there are icons for search, help, notifications (with 1 notification), user profile, and 'log out'.

The main content area has a sidebar on the left with a 'Back' button and several links: 'Snippet Generator', 'Declarative Directive Generator', 'Declarative Online Documentation', 'Steps Reference', 'Global Variables Reference', 'Online Documentation', 'Examples Reference', and 'IntelliJ IDEA GDSL'. The 'Snippet Generator' link is currently active, indicated by a grey background.

The main panel is titled 'Overview' and contains the following information:

- A descriptive text about the Snippet Generator: "This **Snippet Generator** will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click **Generate Pipeline Script**, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)"
- A section titled 'Steps' with a 'Sample Step' example. The step selected is 'build: Build a job'. The configuration includes:
 - 'build'
 - 'Project to Build': 'SQMA_Preda_Mihail_Irinel_Jenkins_1'
 - Checkboxes for 'Wait for completion' and 'Propagate errors'
 - 'Quiet period': An empty input field
 - 'Parameters': 'whichTest': 'testB'
- A large blue button labeled 'Generate Pipeline Script'.
- The generated Pipeline Script code:

```
build job: 'SQMA_Preda_Mihail_Irinel_Jenkins_1', parameters: [string(name: 'whichTest', value: 'testB')]
```

Below the 'Steps' section, there's a 'Global Variables' section with a note: "There are many features of the Pipeline that are not steps. These are often exposed via global variables, which are not supported by the snippet generator. See the [Global Variables Reference](#) for details."

Fig.6

6. Copy the generated codes into the **Pipeline Script** and press **Save**.

The screenshot shows the Jenkins Pipeline configuration page for a project named "SQMA_Preda_Mihail_Irinel_Jenkins_2_Pipeline". The "Build Triggers" tab is selected. Under "Build Triggers", there are several options: "Disable Rebuilding for this job", "This project is parameterized", "Throttle builds", "Build after other projects are built", "Build periodically", "GitHub hook trigger for GITScm polling", "Poll SCM", "Disable this project", "Quiet period", and "Trigger builds remotely (e.g., from scripts)". The "Pipeline" tab is also visible, showing the pipeline script:

```
1 build job: 'SQMA_Preda_Mihail_Irinel_Jenkins_1', parameters: [string(name: 'whichTest', value: 'testA')]
2 build job: 'SQMA_Preda_Mihail_Irinel_Jenkins_1', parameters: [string(name: 'whichTest', value: 'testB')]
```

At the bottom of the pipeline section, there is a checkbox for "Use Groovy Sandbox" which is checked. Below the pipeline section, there are "Save" and "Apply" buttons.

Fig.7

7. From left handside menu click on **Build Now**

The screenshot shows the Jenkins interface for the pipeline "SQMA_Preda_Mihail_Irinel_Jenkins_2_Pipeline". The left sidebar has a "Build Now" button highlighted. The main content area displays the "Stage View" which says "This Pipeline has run successfully, but does not define any stages. Please use the `stage` step to define some stages in this Pipeline." Below it is the "Permalinks" section with a bulleted list of build history. The "Build History" table shows two builds: #1 (Dec 12, 2021, 3:17 PM) and #2 (Dec 12, 2021, 3:33 PM). There are also links for "Atom feed for all" and "Atom feed for failures".

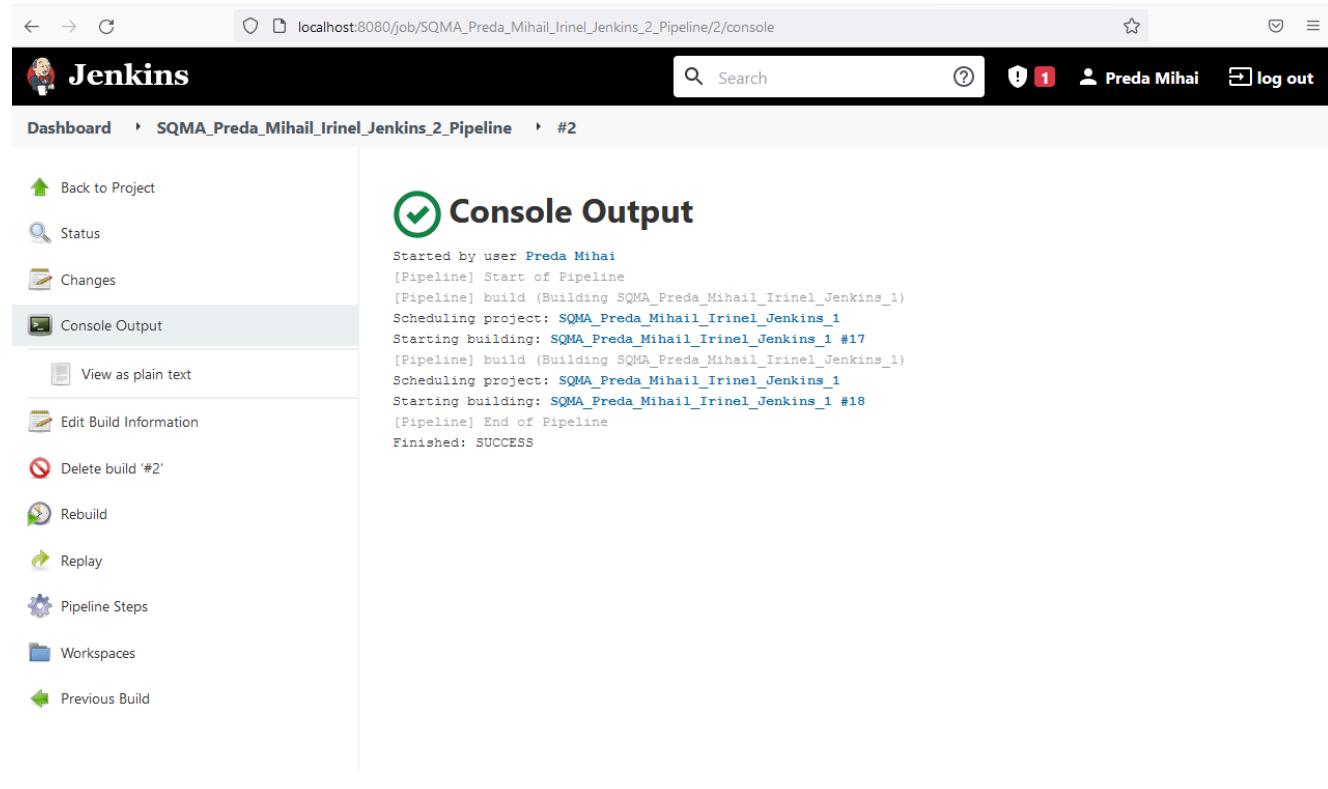
Fig.8

8. To see Pipeline result click on latest done build

This screenshot is similar to Fig.8, showing the Jenkins interface for the same pipeline. However, the "Build Now" button in the sidebar is now unhighlighted, indicating the build has been triggered. The "Build History" table shows the latest build #2 (Dec 12, 2021, 3:33 PM) is the most recent completed build.

Fig.9

9. Then click on **Console Output** to see the build



The screenshot shows the Jenkins interface for a pipeline job named 'SQMA_Preda_Mihail_Irinel_Jenkins_2_Pipeline'. The left sidebar has a 'Console Output' item selected. The main content area is titled 'Console Output' with a green checkmark icon. It displays the build log:

```
Started by user Preda Mihai
[Pipeline] Start of Pipeline
[Pipeline] build (Building SQMA_Preda_Mihail_Irinel_Jenkins_1)
Scheduling project: SQMA_Preda_Mihail_Irinel_Jenkins_1
Starting building: SQMA_Preda_Mihail_Irinel_Jenkins_1 #17
[Pipeline] build (Building SQMA_Preda_Mihail_Irinel_Jenkins_1)
Scheduling project: SQMA_Preda_Mihail_Irinel_Jenkins_1
Starting building: SQMA_Preda_Mihail_Irinel_Jenkins_1 #18
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Fig.10

Project setup

```
npm install
```

Project run

```
npm start
```

Project test

```
npm test
```

Project individual test suites

```
npm run testA
npm run testB
```