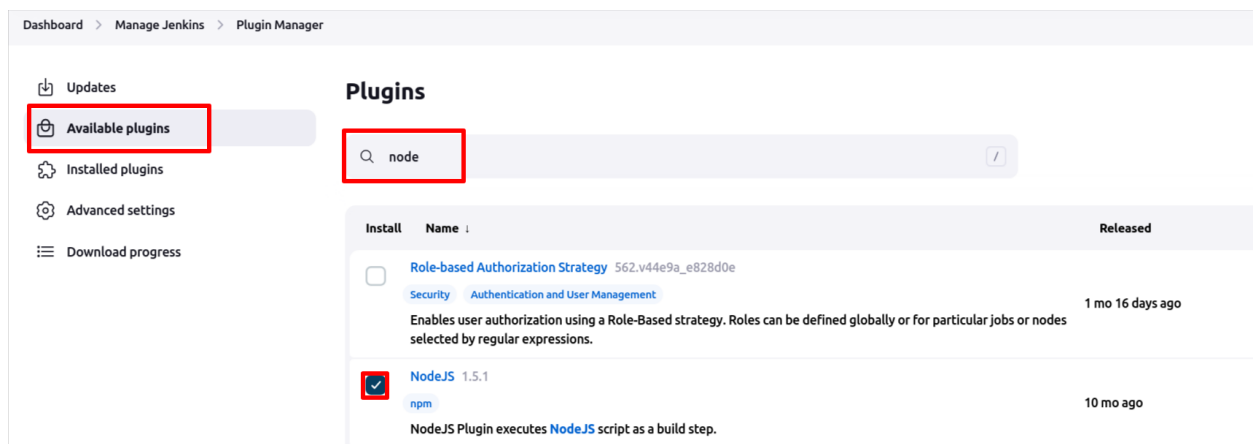




Lab 4S. Creating A Pipeline for the NodeJS Result Application (Solution)

Install the **NodeJS** plugin for Jenkins.



From **Jenkins > System Configuration > Global Tool Configuration > NodeJS Installation**, add a NodeJS installation with version 8.9.0.

NodeJS

NodeJS installations

List of NodeJS installations on this system

Add NodeJS

Name the installation exactly **NodeJS 19.0.1** as the name will be referred to later in the pipeline.

The screenshot shows a configuration window for installing NodeJS. At the top, there's a title bar 'NodeJS' with a close button. Below it, the 'Name' field is set to 'NodeJS 19.0.1'. A checkbox labeled 'Install automatically' is checked. Underneath, there's a section titled 'Install from nodejs.org' with a 'Version' dropdown menu also set to 'NodeJS 19.0.1'. Below the dropdown, there's a note: 'For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail'. A checkbox for 'Force 32bit architecture' is present and unchecked. At the bottom, there's a section for 'Global npm packages to install' with a text input field and a note: 'Specify list of packages to install globally – see npm install -g. Note that you can fix the packages version by using the syntax'.

Create a new job inside the **instavote** folder called **result-build**, this time as a freestyle project.



Enter an item name

result-build

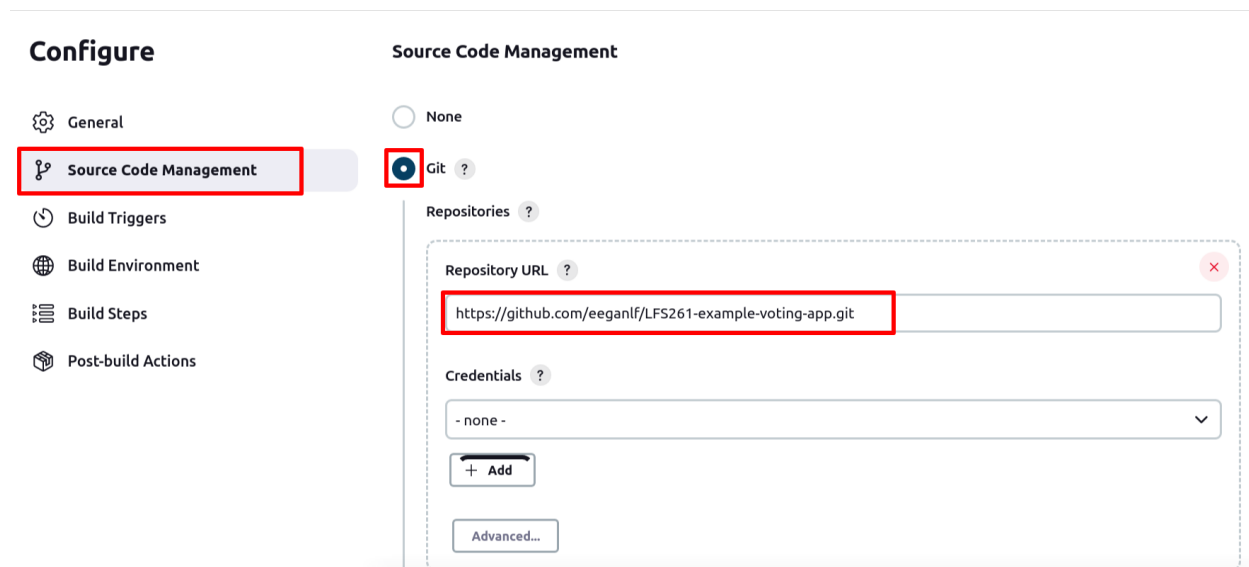
Required field

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.

Paste in the **example-voting-app** Git repository that you forked in the **Source Code Management** section.



Configure

Source Code Management

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

None

Git

Repositories

Repository URL

https://github.com/eeganlf/LFS261-example-voting-app.git

Credentials







- none -

+ Add

Advanced...

Define the build trigger as PollSCM with an interval of 2 mins by pasting `H/2 * * * *` into the **Schedule** input box.







Configure

-  General
-  Source Code Management
-  **Build Triggers**
-  Build Environment
-  Build Steps
-  Post-build Actions

Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts) ?
 - ☐ Build after other projects are built ?
 - ☐ Build periodically ?
 - ☐ GitHub hook trigger for GITScm polling ?
 - ☒ Poll SCM ?
- Schedule ?
- H/2 * * * ***
- Would last have run at Saturday, November 12, 2022 at 10:29:12 PM Coordinated Universal Time; would next run at Saturday, November 12, 2022 at 10:31:12 PM Coordinated Universal Time.
- ☐ Ignore post-commit hooks ?

In the **Build Environment** choose to provide NodeJS configurations with the version you have selected in the **Global Tool Configuration**.

-  General
-  Source Code Management
-  Build Triggers
-  **Build Environment**
-  Build Steps
-  Post-build Actions

Build Environment

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s) ?
- ☐ Provide Configuration files ?
- ☐ Add timestamps to the Console Output
- ☐ Inspect build log for published build scans
- ☒ Provide Node & npm bin/ folder to PATH

NodeJS Installation

Specify needed nodejs installation where npm installed packages will be provided to the PATH

NodeJS 19.0.1

npmrc file

- use system default -







Select the **Execute shell** option from **Build steps > Add build step** and paste or type the following into the box:

```
cd result
```

```
npm install
```

```
npm ls
```

Configure

-  General
-  Source Code Management
-  Build Triggers
-  Build Environment
-  **Build Steps**
-  Post-build Actions

Build Steps

Execute shell ?

Command

See [the list of available environment variables](#)

```
cd result
npm install
npm ls
```







Advanced...

Save the project and build it.

Create another project with the name **result-test**, this time copying from **result-build**.

Change the build trigger to run after **result-build** is run.

Configure

-  General
-  Source Code Management
-  **Build Triggers**
-  Build Environment
-  Build Steps
-  Post-build Actions

Build Triggers

☐ Trigger builds remotely (e.g., from scripts) ?

☒ **Build after other projects are built** ?

Projects to watch

result-build,

 No such project 'res'. Did you mean 'result-test'?

☒ Trigger only if build is stable

☐ Trigger even if the build is unstable

☐ Trigger even if the build fails

☐ Always trigger, even if the build is aborted

Update the command to run **npm test** instead of **npm ls**.

Build Steps

≡ Execute shell ?

Command

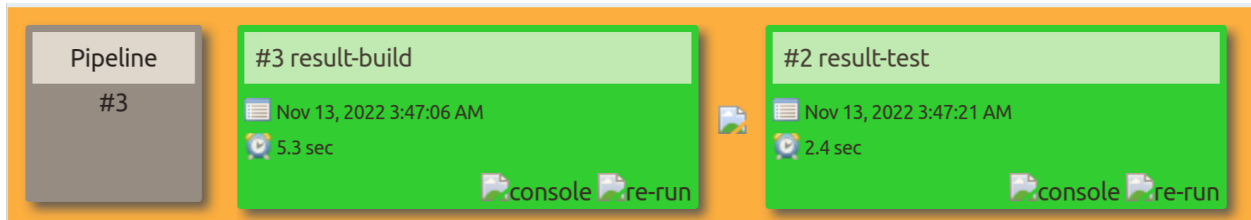
See [the list of available environment variables](#)

```
cd result
npm install
npm test
```

Advanced...

Add build step ▾

Create a new pipeline view, this time by choosing **result-build** as the first job to run.
After **result-build** runs, you should see both jobs turn green in the pipeline as follows:



This completes the CI pipeline for the **result** app created with NodeJS.