

How to Deploy NodeJs-express app with a static front-end in a pug on Elestio

So you made a static full stack frontend pug with nodejs backend application and want to deploy it to the cloud ☑☑

You probably heard about Kubernetes (and all its complexity) or various options to deploy your apps like Heroku, Render Fly, or Railways. They all have something in common, those products are building your own source code on every commit from your GIT repository.

Elestio is doing the same ... **but different!** Instead of deploying your app to a shared cluster, we deploy to dedicated VMs.

In this tutorial, we will deploy a simple NodeJs-express app with a static front-end in a pug that the backend in nodejs was created using the npm init command. You can use any existing application, create a new one with npm init, or simply fork and use our example by following the [link](#).

To learn more about the elestio CI-CD, go [here](#).

If you're new, sign up for [Elestio](#), otherwise, login to your existing account.

Deploy a NodeJs-express app with a static front-end in a pug with CI/CD to the cloud

Step 1:

Go to CI/CD from the left sidebar.

Step 2:

Now, select the deployment source.

Create CI/CD pipeline

1 Source

2 Target

3 Configuration

1. Deployment Method



Select the deployment method of your Service between Github, Gitlab and Docker.

When using the Github/Gitlab deployment method, each time a change is pushed to your repository, a new deployment of your service will occur.



Github

Git-Hub, Inc. is a provider of Internet hosting for software development and version control using Git. It offers the distributed version control and source code management...



Gitlab

GitLab Inc. is the open-core company that provides GitLab, the DevOps software that combines the ability to develop, secure, and operate software in a single application



Docker compose

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application's services

In this tutorial, I'm deploying using GITHUB, but you can also use GITLAB if you have a project there.

Step 3:

Select the repository.

If you have already authenticated your GITHUB or GITLAB account in ci-cd for repository access, you can choose the desired repository to deploy directly. Otherwise, you must first authenticate your GIT account with elestio ci-cd for repository access.

Import Git Repository

Git Account

amitshuklabag

Git Scope

elestio-examples

nodejs

nodejs-express-pug

4 months ago

Selected

Import Third-Party Git Repository →

Step 4:

Choose Deployment Targets

1. Choose Deployment Targets



Where do you want this service to be deployed?

Deploy on a new VM

Deploy on existing VM

Deployment mode

Single Node

Cluster

Elestio offers two types of deployment targets "**Deploy on a new VM**" and "**Deploy on an existing VM**".

You are allowed to set up n pipelines on each elestio Ci-CD target/VM. According to the project configuration you select and the project you're deploying, the number of pipelines varies.

If you want to deploy these projects as a pipeline on a new Target/VM or don't have any installed targets, choose "Deploy on a new VM." If you already have any installed or previously configured ci-cd targets/VMs, choose "Deploy on an existing VM," and then choose the existing target from the targets dropdown.

Follow the steps below only if you select "**Deploy on a new VM**," otherwise click the next button to proceed.

CI/CD Pipelines by Elestio are available with our 5 cloud partners (AWS Lightsail, Digital Ocean, Vultr, Linode & Hetzner) in 85 locations over 27 countries but also on any cloud (AWS, Azure, Google, Oracle, ...) and on-premise with [BYOVM](#).


- Select Service Cloud Provider



- Select Service Cloud Region

Europe North America


fsn1

 Germany - Falkenstein

hel1





 Finlande - Helsinki

nbg1





 Germany - Nuremberg

- Select Service Plan





SMALL-1C-2G

 1 CPU  2 GB RAM  20 GB Storage  20 TB Bandwidth included





SMALL-2C-2G

 2 CPU  2 GB RAM  40 GB Storage  20 TB Bandwidth included

MEDIUM-2C-4G


 2 CPU  4 GB RAM  40 GB Storage  20 TB Bandwidth included

MEDIUM-3C-4G

 3 CPU  4 GB RAM  80 GB Storage  20 TB Bandwidth included

- Now Customize the target name and project (where the CICD Target will be created).

5. Provide a name for your new CICD Target instance

 The CICD Target name cannot be changed afterwards. This is the name of the instance that will contain one or more pipelines.

Name*

6. Select the target project where the CICD Target will be created

elestio-services

“ If you want to deploy it with a different name and a different project, you can customize it. By default, we configure it with a dynamic target name and the current project.

Step 5:

Configure your Project

Configure your nodejs-express-pug Project



Define your project build and run

Project Name

nodejs-express-pug

Branch

main

Runtime

Node.js

Version

latest (27-10-2022)

Framework

No Framework

Root Directory

/

You can configure the project details by filling up the project name, branch, run time, version, framework, and root directory.

Build and Output Setting

Install command

npm install

Run command

npm start

Build command

Build Output dir

/

You can configure your project install, run, and build commands in the Build and output setting.

Life cycle scripts (optional)

Pre install command

./scripts/preInstall.sh

Post install command

./scripts/postInstall.sh

Pre Backup command

./scripts/preBackup.sh

Post Backup command

./scripts/postBackup.sh

Pre Restore command

./scripts/preRestore.sh

Post Restore command

./scripts/postRestore.sh

The configuration of life cycle scripts is always optional; they should only be used if you want to execute a specific command before and after building your project. Otherwise, leave them empty.

Environment variables



Use Environment variables to store configuration values. API keys and secrets. You can access them in your service like regular environment variables.

.env

```
1 ENV=production
2 PORT=3000
3
```

You can list all of your project's API keys and secrets here if they were saved in ENV

Exposed Ports



Configure the ports which should be exposed. When the ports are not exposed publicly, they are only accessible to other Services of the same App via the internal network.

Interface

172.17.0.1

Host Protocol

HTTP



Host Port

3000

Container Port

3000



+ Add Another

Reverse proxy configuration



Configure the ports which should be exposed. When the ports are not exposed publicly, they are only accessible to other Services of the same App via the internal network.

Listen			Target			
Protocol	Port		Protocol	IP	Port	Path
HTTPS	443	→	HTTP	172.17.0.1	3000	/
<input type="checkbox"/> Require Basic Auth						
+ Add Another						

The final step is to configure the exposed port and reverse proxy settings. You can specify the port on which your project will run here.

“ If your project includes **elestio.yml**, Elestio will auto-fill all of these fields. As in this tutorial, we're using our NodeJs elestio example, so you can see in the above images that all of our fields are auto-filled.

Refer to these links to learn how to create our own [elestio.yml](#) for the project.

A sample elestio.yml for NodeJs is shown below. check it out on [github](#)

```
config:
  runTime: 'NodeJs'
  version: ''
  framework: 'NoFramework'
  buildCommand: ''
  buildDir: ''
  runCommand: 'npm start'
  installCommand: 'npm install'
  icon: "public/images/nodejs.svg"
  screenshot: "public/images/screenshot.jpg"
ports:
  - protocol: "HTTPS"
```

```
targetProtocol: "HTTP"
listeningPort: "443"
targetPort: "3000"
targetIP: "172.17.0.1"
public: true
path: "/"
isAuth: false
login: ""
password: ""
environments:
  - key: 'ENV'
    value: 'production'
  - key: 'PORT'
    value: '3000'
webUI:
  - url: "https://[CI_CD_DOMAIN]"
    label: "Website"
```

Step 6:

Click the **Create CI/CD pipeline** button to deploy your pipeline.



Provider

Hetzner Cloud

Region

Europe, Germany
Falkenstein

Estimated Monthly Price*

\$10

*Estimated monthly price is based on
730 hours of usage.

Create CI/CD pipeline

In a couple of moments, your application was successfully deployed on elestio .



Details

Tools

Backups

Build & Deploy

History

Domain Management

Termination protection

Disabled. Pipeline can be terminated.

Protection deactivated



Pipeline Details

CNAME: nodejs-express-pug-u76.vm.appdrag.net
Deployment method: Github
Runtime: NodeJs (16.15)
Framework: No Framework
Branch name: main
Repo url: <https://github.com/elestio-examples/nodejs-express-pug>

Website

Show

View Instructions

View Instructions from README

View Instructions

Manage Stack

View running logs

Restart Stack

Stop Stack



Welcome to Elestio

Deploy your apps quickly with the elestio CI/CD system

This Host nodejs-express-pug-u76.vm.appdrag.net

Your IP 117.207.219.235

Your Location [IN, Nanded](https://IN,Nanded)

Latency to server [144 ms](https://144ms)

Deploy on Elestio

You can now view your deployed URL and access your application by going to desired application pipeline details.

Please let us know by contacting our support [email](#) or [ticketing](#) system if you give it a shot and encounter any problems or if anything goes wrong.

Join us on [**discord**](#) to know more.

Revision #6

Created 3 November 2022 12:16:08 by Amit Shukla

Updated 7 November 2022 12:39:54 by Amit Shukla