МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

**Лабораторная работа №1**

**По дисциплине «Web-программирование»**

**Хостинг веб-приложения на сервисе Heroku**

**Выполнили студент группы М33081**

**Аль Даббагх Харит Хуссейн**

**Проверил**

**Приискалов Роман Андреевич**

САНКТ-ПЕТЕРБУРГ

2022

Содержание

[Installation 2](#_Toc96607983)

[Installing NodeJS 2](#_Toc96607984)

[Installing NestJS CLI 2](#_Toc96607985)

[Registration on Heroku 3](#_Toc96607986)

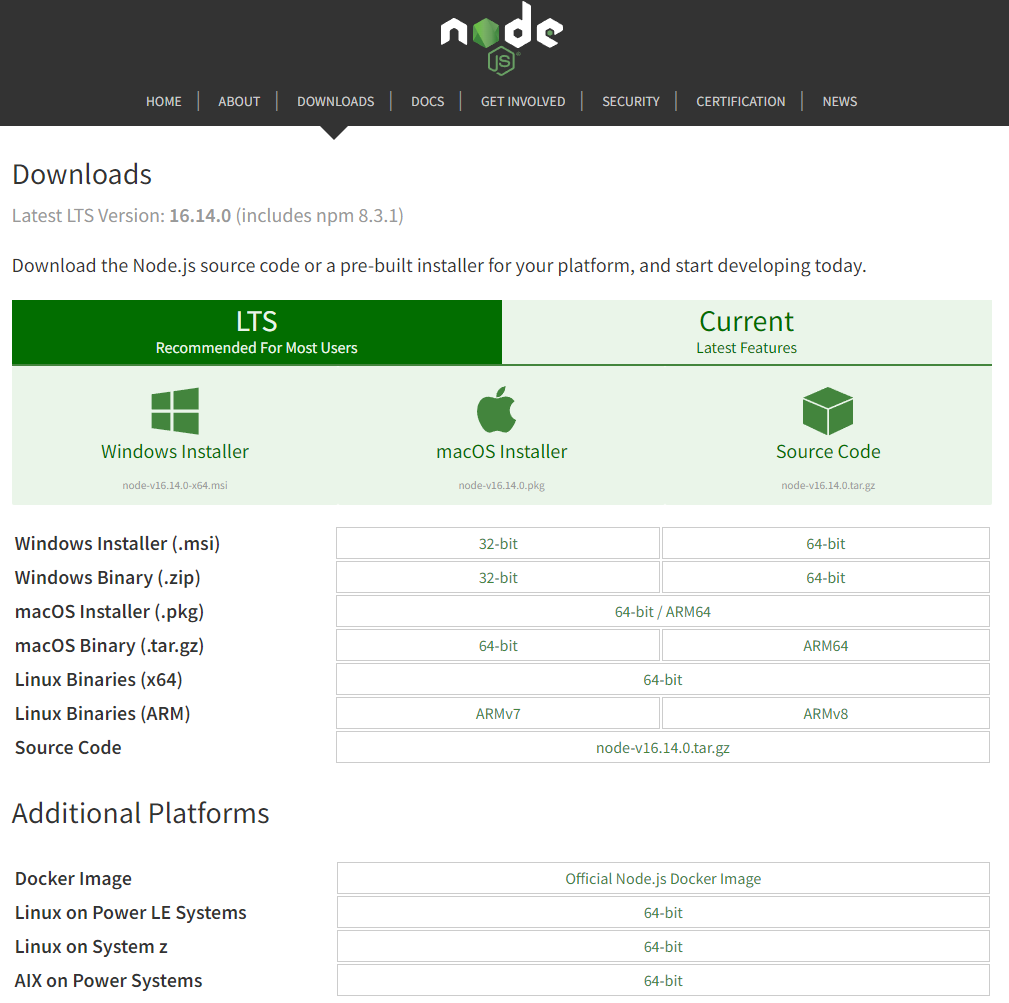
[Installing Heroku CLI 4](#_Toc96607987)

[Project Creation 5](#_Toc96607988)

# Installation

## Installing NodeJS

To install node we can easily do that from their [website download page](https://nodejs.org/en/download/):



## Installing NestJS CLI

After installing NodeJS we’re gonna get the option to install other packages inside our app and/or globally.

For example to install NestJS we can use:

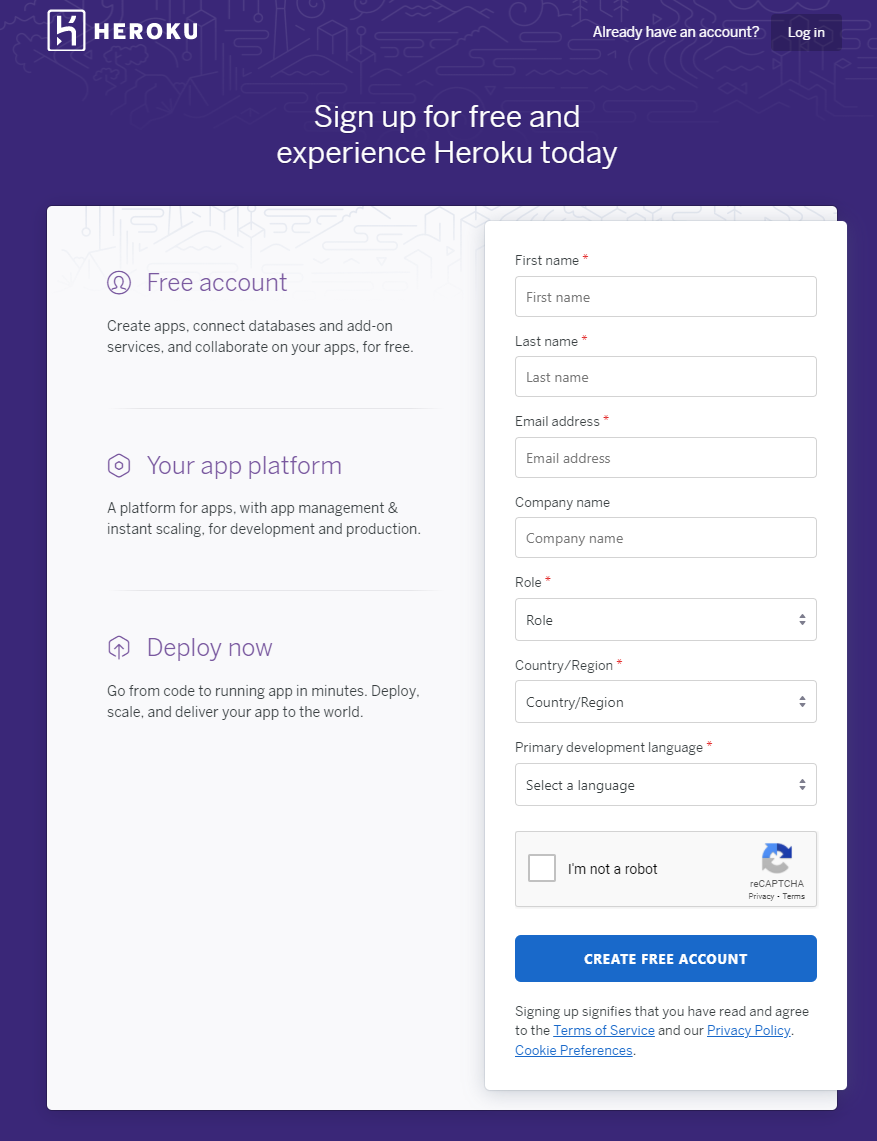
npm install -g @nestjs/cli

npm is the default package manager for node, there are also other package managers that can be installed and used like yarn and pnpm.

-g flag means that we want to install globally.

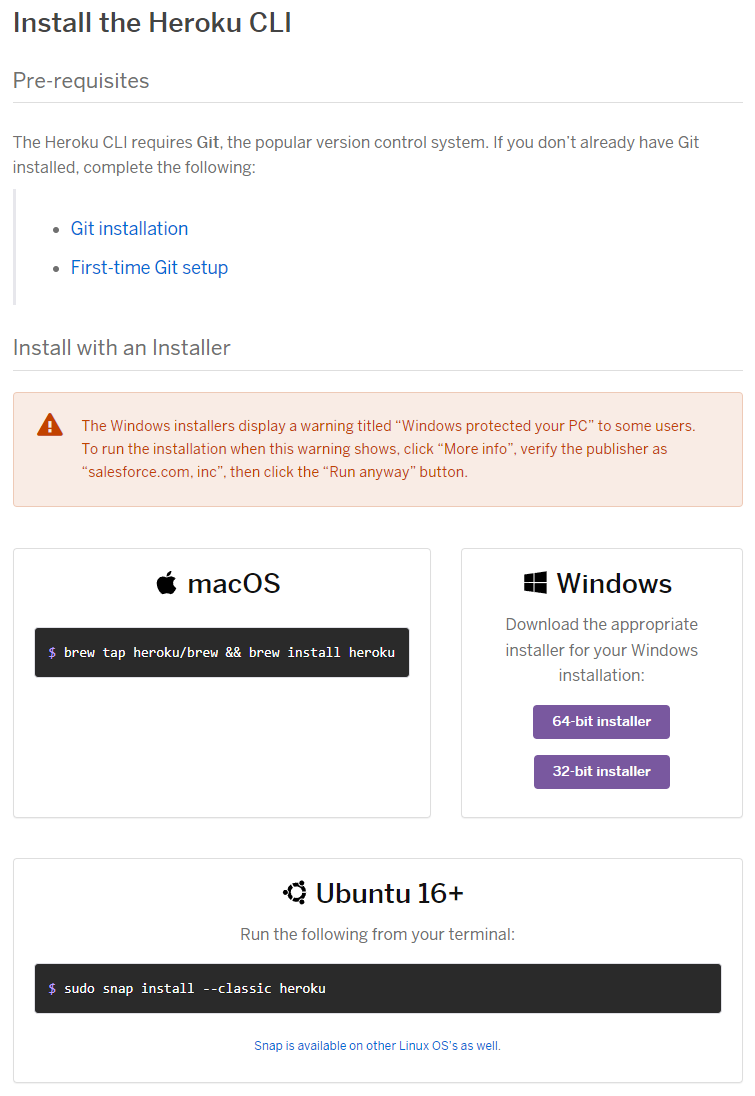
## Registration on Heroku

You can register to heroku easily using their form:



## Installing Heroku CLI

Heroku CLI can be installed depending on our system using their page:



In my case using windows 64-bit installer.

# Project Creation

To create a NestJS app with boilerplate we use the following command:

nest new web-6th-sem

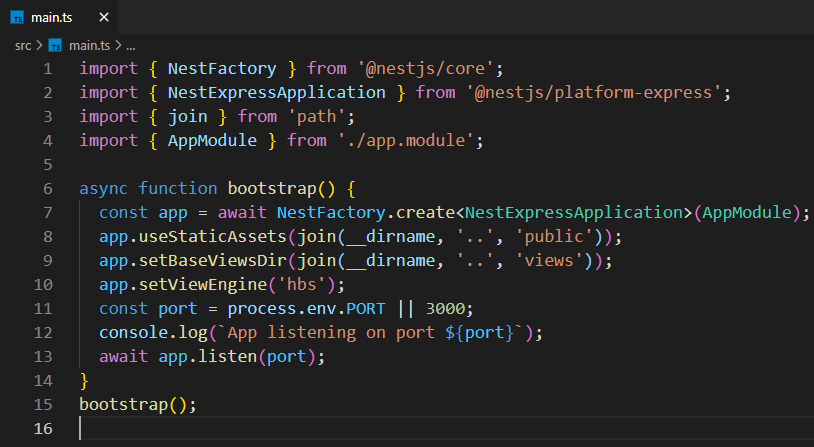
Within the boilerplate there is a package.json file. We changed some of it’s contents according to the lab requirements:



During the lab work we are required to use environment variables for our app. To do this we installed the [configuration module](https://docs.nestjs.com/techniques/configuration) for nest.

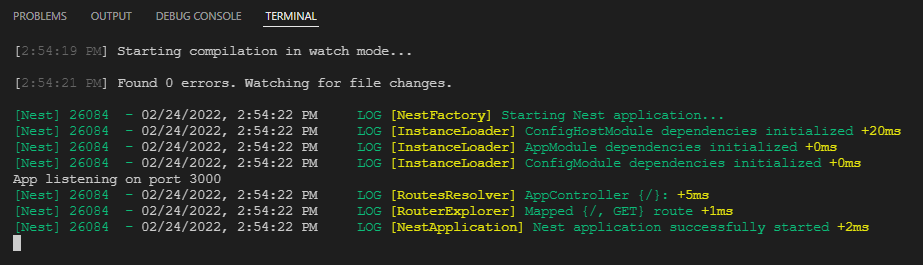
The variable we’re going to define is the port number.

Made some changes to the already created src/main.ts file to check if the variable is being used or undefined. If it’s undefined then it’s gonna use port 3000.



We check the app if it’s working locally using the command:

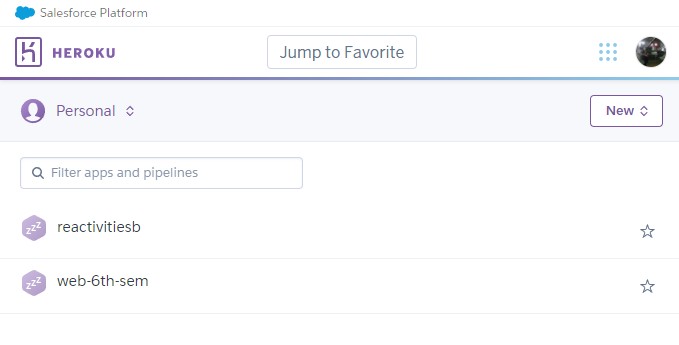
npm run start:dev



Next we can create an App on Heroku using it’s installed CLI:

heroku apps:create web-6th-sem

And check on the website if the app is created:



For Heroku to be able to start the app, we need to specify the command a file called Procfile, the contents of which is this single line:

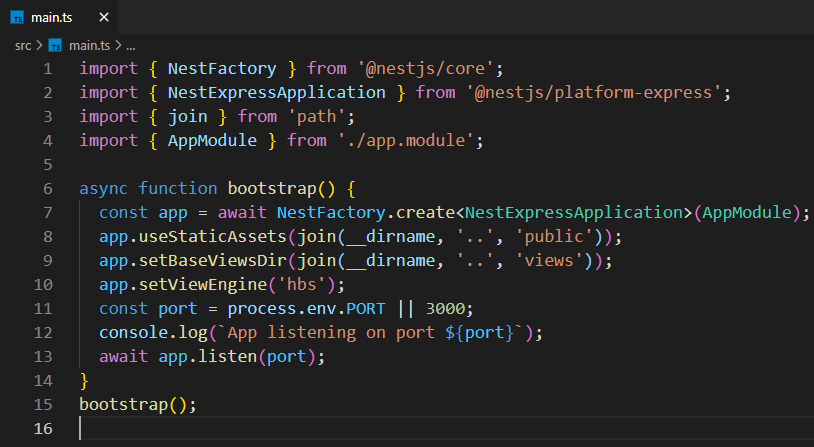
web: npm run start:prod

NOW, this has all been on the backend side of the application. Now we need to bring our frontend that we created in the last semester.

To do that we created a MVC app inside our project, but we also need a template engine render our HTML views, for example Handlebars. To install Handlebars:

npm install --save hbs

And as before. We need to add the following to our main.ts file:



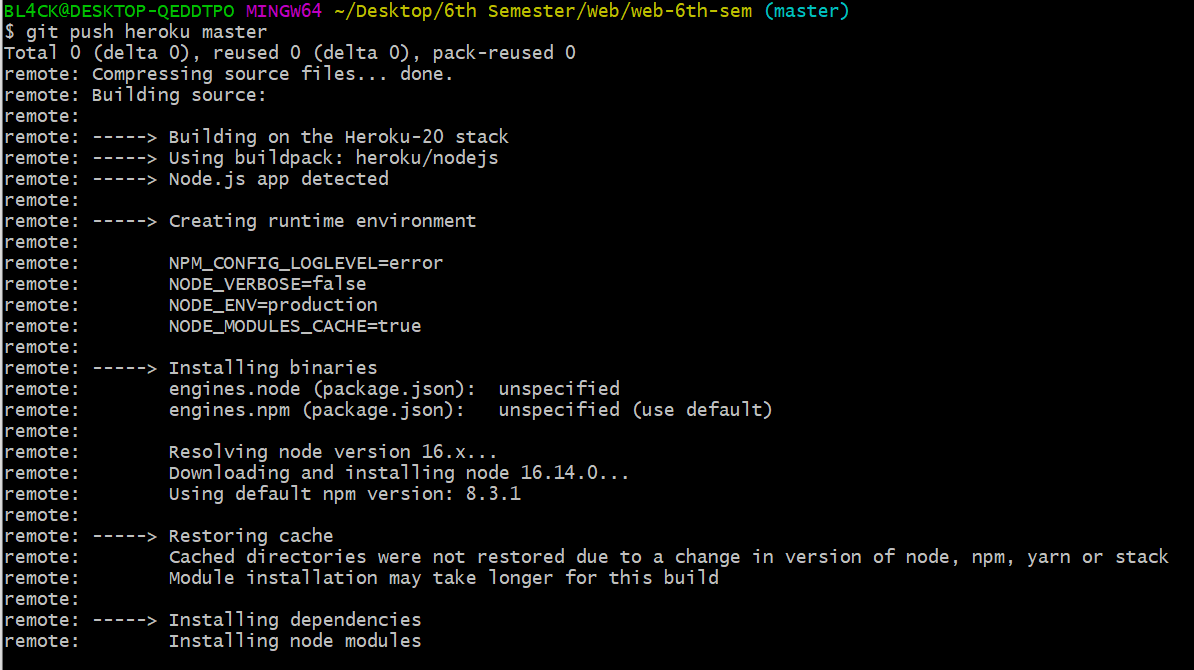
This means that the public directory will be used for storing static assets. Now we just need to copy our assets from the last semester to the public folder and restart the app.

Now we can commit and push to Heroku using the commands:

git add .

git commit -m "MESSAGE"

git push heroku master



Heroku will build and deploy the app and host it on:

<https://web-6th-sem.herokuapp.com/>