

# Getting started with Node.js, Mongoose and MongoDB Part 2



by [Micha Hernandez van Leuffen](#) - Jun 21 2013

[wercker](#)[nodejs](#)[mongodb](#)[mongoose](#)[tdd](#)[javascript](#)

This article is outdated. Please visit our [Dev Center](#).

This is a followup from our [previous post](#) where we created an application with [node.js](#), powered by [10gen's MongoDB](#). Not that we've set up our build pipeline with wercker, we are going to deploy our application to [Heroku](#) and use [MongoLab](#), a MongoDB-as-a-Service cloud platform.

[Archive »](#)

## Tags

[Wercker](#) (28)[Weekinreview](#) (19)[Golang](#) (17)[Boxes](#) (16)[Opendelivery](#) (15)[Ui](#) (12)[Deployment](#) (11)[Docker](#) (10)[Design](#) (9)[Python](#) (9)



## Introduction

In our [previous post](#) we created a [node.js](#) application powered by [express](#) and [mongoose](#), a object document mapper for [MongoDB](#). We created this application using test-driven development using [Mocha](#) and [SuperTest](#). This article is also available on our [dev center](#)

You can visit the finished application on wercker [here](#).

The code to this tutorial app is [open source](#) on GitHub so feel free to fork and clone it.

Now, let's deploy our app!

## Deploying our app

We will deploy our application to Heroku, so make sure you have a (verified) [Heroku](#) account and have installed the [Heroku Toolbelt](#). Let's first create an app:

[Nodejs](#) (8)

[Steps](#) (8)

[Heroku](#) (6)

[Announcement](#) (5)

[Github](#) (5)

[Django](#) (5)

[Git](#) (5)

[Android](#) (5)

[Ruby](#) (5)

[Containers](#) (5)

[S3](#) (4)

[Javascript](#) (4)

[Structure](#) (4)

[Jekyll](#) (3)

[Rethinkdb](#) (3)

[Funding](#) (3)

[API](#) (3)

[Ewok](#) (3)

[Cli](#) (3)

[Aws](#) (3)

[Pipeline](#) (3)

[PostGIS](#) (3)

[Gigaom](#) (2)

[Linuxcontainers](#) (2)

[Paas](#) (2)

[Bitbucket](#) (2)

```
heroku create
```

```
Creating guarded-atoll-9149... done, stack is cedar  
http://guarded-atoll-9149.herokuapp.com/ | git@heroku.com:guarded-atoll-9149  
Git remote heroku added
```

Now we need a MongoDB instance in the cloud! Fortunately, Heroku has a Marketplace (wercker is [on it](#) by the way) filled with addons. One of which are our friends at [Mongolab](#), so let's use their [add-on](#) to [provision](#) a MongoDB database:

```
heroku addons:add mongolab
```

```
Adding mongolab on guarded-atoll-9149... done, v3 (free)  
Welcome to MongoLab. Your new subscription is ready for use.  
Use `heroku addons:docs mongolab` to view documentation.
```

For our `test` environment we used the

`WERCKER_MONGODB_HOST` environment variable that was

provided by wercker and was defined through the

`wercker.yml`. We now need a similar environment variable

for our production setting which is powered by Heroku and Mongolab. We can retrieve this environment variable, again via the Heroku command line interface:

```
heroku config | grep MONGOLAB_URI
```

```
MONGOLAB_URI: mongodb://heroku_app3489u7438034:ofs0gfj:
```

[Api](#) (2)

[Maintenance](#) (2)

[Couchdb](#) (2)

[Caching](#) (2)

[Contributions](#) (2)

[Internals](#) (2)

[Mongodb](#) (2)

[Mongoose](#) (2)

[Tdd](#) (2)

[Devcenter](#) (2)

[Hires](#) (2)

[Flask](#) (2)

[Redis](#) (2)

[Gocov](#) (2)

[Codecoverage](#) (2)

[Compass](#) (2)

[Sass](#) (2)

[Buildsteps](#) (2)

[Security](#) (2)

[Badges](#) (2)

[Notifications](#) (2)

[Evangelism](#) (2)

[Geodjango](#) (2)

[Capistrano](#) (2)

[Building](#) (2)

[Php](#) (2)

We now need to use this environment variable,

`MONGODB_URI`, in our actual application. Modify the **configure** section in your `app.js` file in the following way:

```
// Configure express
app.configure('development', function() {
  mongoose.connect('mongodb://localhost/todos');
});

app.configure('test', function() {
  mongoose.connect('mongodb://' + process.env.WERCKER_MONGODB_URI);
});

app.configure('production', function() {
  mongoose.connect('mongodb://' + process.env.MONGODB_URI);
});
```

Similarly to adding the `NODE_ENV=test` environment variable to wercker in the previous post, we need to do the same for Heroku, but now of course `NODE_ENV=production` as Heroku is our production environment.

```
heroku config:set NODE_ENV=production

Setting config vars and restarting guarded-atoll-9149.
NODE_ENV: production
```

We now have successfully set up our production environment consisting of Heroku and MongoLab.

[Dart](#) (2)

[Meteor](#) (1)

[Collaboration](#) (1)

[Addon](#) (1)

[Cloud](#) (1)

[Sidebar](#) (1)

[Seedround](#) (1)

[Digitalocean](#) (1)

[Walter](#) (1)

[Deploykeys](#) (1)

[Dockerfile](#) (1)

[Selenium](#) (1)

[Hipchat](#) (1)

[Provisioning](#) (1)

[Awards](#) (1)

[Xmas](#) (1)

[Riak](#) (1)

[Ricon 2013](#) (1)

[Box](#) (1)

[Service](#) (1)

[Database](#) (1)

[Open Source](#) (1)

[Pullrequests](#) (1)

[Tutorial](#) (1)

[Go](#) (1)

[Golang](#) (1)

# Creating a Heroku Procfile

Heroku needs to know which process to run on their cloud platform to actually launch your application. This is done through the Heroku [Procfile](#).

Create a file called `Procfile` in your project directory with the following line of code:

```
web: node app.js
```

Let's add this file to our repository and push it to our version control system:

```
git add Procfile
git commit -am 'added Procfile'
git push origin master
```

This will trigger a new build on wercker (which should pass as we didn't change anything dramatic) and we're now ready to deploy our application.

## Add deploy target

First, we add a deploy target. Go to your settings tab for your application on wercker and look for the **deploy targets** section. Although wercker has an [add-on] on the Heroku Marketplace, making deployment even easier, we are going

[Chef](#) (1)

[Werckeryml](#) (1)

[Mesosphere](#) (1)

[Marathon](#) (1)

[Mesos](#) (1)

[Store](#) (1)

[Osx](#) (1)

[Mac](#) (1)

[Node-Webkit](#) (1)

[Private Repositories](#) (1)

[RVM](#) (1)

[Registrations](#) (1)

[Bundle](#) (1)

[Npm](#) (1)

[Pip](#) (1)

[Docs](#) (1)

[Rails4](#) (1)

[Rails](#) (1)

[Ux Design Foundation](#) (1)

[Continuous Deployment](#) (1)

[Videos](#) (1)

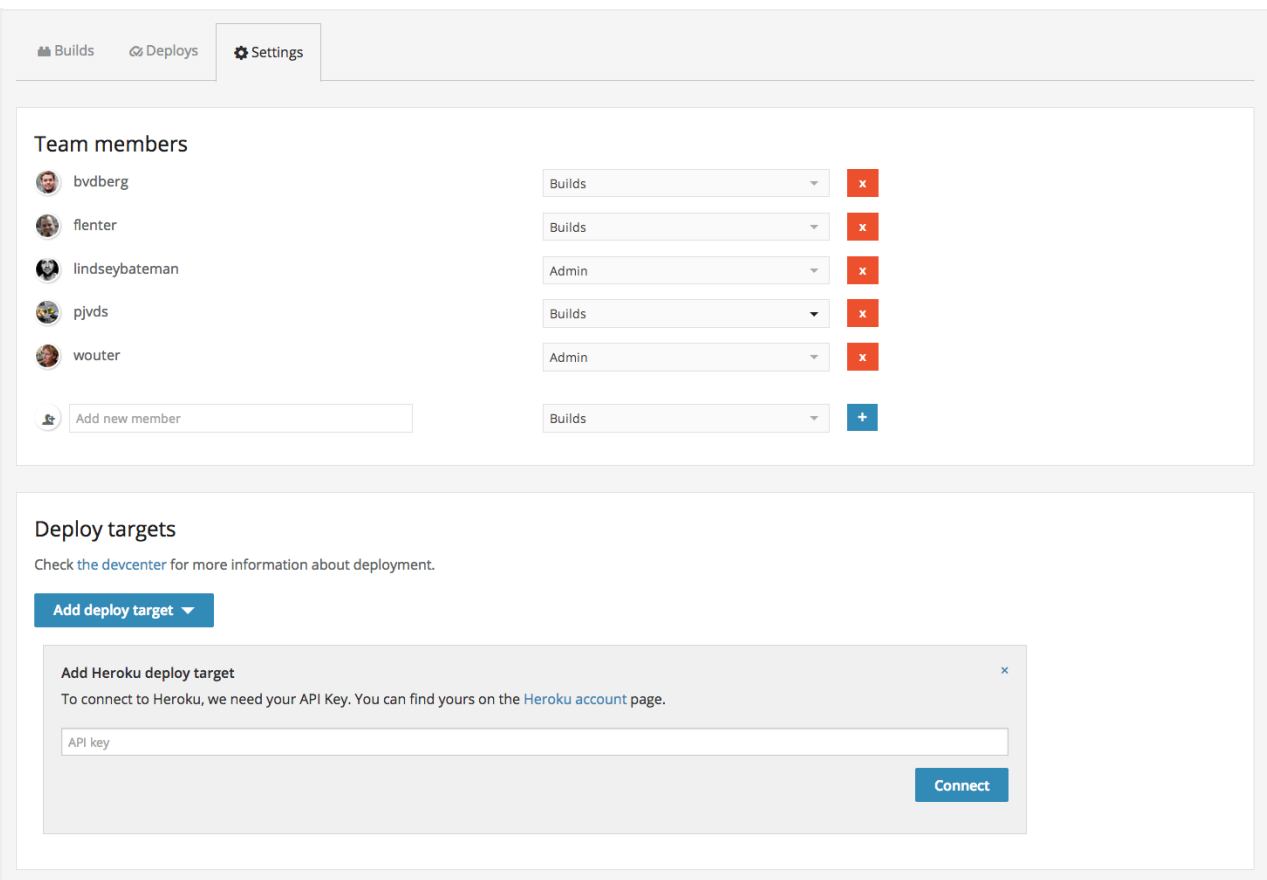
[Chrome](#) (1)

[Css](#) (1)

[Animation](#) (1)

[Fabric](#) (1)

to add Heroku manually as a deploy target.



The screenshot shows the Heroku Dev Center interface. At the top, there are tabs for 'Builds', 'Deploys', and 'Settings'. The 'Settings' tab is active. Below the tabs, there's a 'Team members' section with a list of users: bvdberg, flenter, lindseybateman, pjvds, and wouter. Each user has a dropdown menu for their role (Builds, Admin) and a red 'x' icon. Below the list is an 'Add new member' button. Below the team members section is a 'Deploy targets' section with a link to 'Check the devcenter for more information about deployment.' and an 'Add deploy target' button. A modal is open for 'Add Heroku deploy target', showing a text input for 'API key' and a 'Connect' button. The modal text says: 'To connect to Heroku, we need your API Key. You can find yours on the Heroku account page.'

As the page indicates, retrieve your Heroku API key from your [Heroku Dashboard](#) and paste it in the form. Next, you are presented with a form where you can *name* your deploy target (let's go with *production*) and are able to auto deploy specific branches. We've previously written a post on auto-deployment [here](#). We are also able to either select an existing Heroku app that we want to deploy to, or create one. I'm going to pick the application I've previously created using the `heroku create` command and to which I've also added the [MongoLab add-on](#)

[Ec2](#) (1)

[Openshift](#) (1)

[Redhat](#) (1)

[Gems](#) (1)

[Development](#) (1)

[Kubernetes](#) (1)

[Minimal](#) (1)

[Coreos](#) (1)

[Quay](#) (1)

[Scratch](#) (1)

[Testing](#) (1)

[Node](#) (1)

[Browser](#) (1)

[Pypi](#) (1)

[Ssh](#) (1)

[Local](#) (1)

[Devmode](#) (1)

[Suchdevwow](#) (1)

[Fortrabbbit](#) (1)

[Status](#) (1)

[Svg](#) (1)

[Retina](#) (1)

[Campfire](#) (1)

[Amazon](#) (1)

[ECS](#) (1)

[Deployment](#) (1)

Deploy targets

Check the [devcenter](#) for more information about deployment.

Add deploy target ▼

production (*heroku*) Edit Remove

✔ Currently running: **deployed build** by mies, on *June 20 2013, 1:17:55 pm*

Edit Heroku deploy target

Deploy target name  
production

Auto deploy  
☐ auto deploy successful builds to branch(es):  
branch names (seperated with spaces)  
*note: builds from all teammembers on the specified branch(es) will deploy.*

Heroku app name  
guarded-atoll-9149  
[Create new Heroku app](#)

Environment variables  
[+ Add new variable](#)

Save

Let’s deploy our application!

Deploying our application

Go to the latest green build on wercker and hit the *deploy* *this build* button. A pull down menu will appear and you can select the target (‘production’) that you’ve just created,

Applications / mies / getting-started-nodejs-mongoose / 7a95b7b

✓ PASSED BY: MIES

MASTER

Deploy this build ▼

12 MINUTES AGO 41 SEC

Deploy to: production

7a95b7b mongolab uri

STATUS	STEP	
✔	get code	0 sec
✔	setup environment wercker/nodejs defined in wercker.yml. wercker/mongodb defined.	21 sec
✔	environment variables	0 sec
✔	npm-install	16 sec
✔	npm-test	0 sec
✔	package	1 sec

If you go to the **deploys** tab you can see your deploy in action:

[Automation](#) (1)

[Documentation](#) (1)

[Organization](#) (1)

[Usecase](#) (1)

[Pivotal](#) (1)

[Organizations](#) (1)

[Flow](#) (1)

[Blog](#) (1)

[Modules](#) (1)

[Node.js](#) (1)

[Dokku](#) (1)

[Digital Ocean](#) (1)

[Developers](#) (1)

[Logo](#) (1)

[Lookandfeel](#) (1)

[Middleman](#) (1)

[Heartbleed](#) (1)

[Endpoint](#) (1)

[Build](#) (1)

[Ansible](#) (1)

[Vagrant](#) (1)

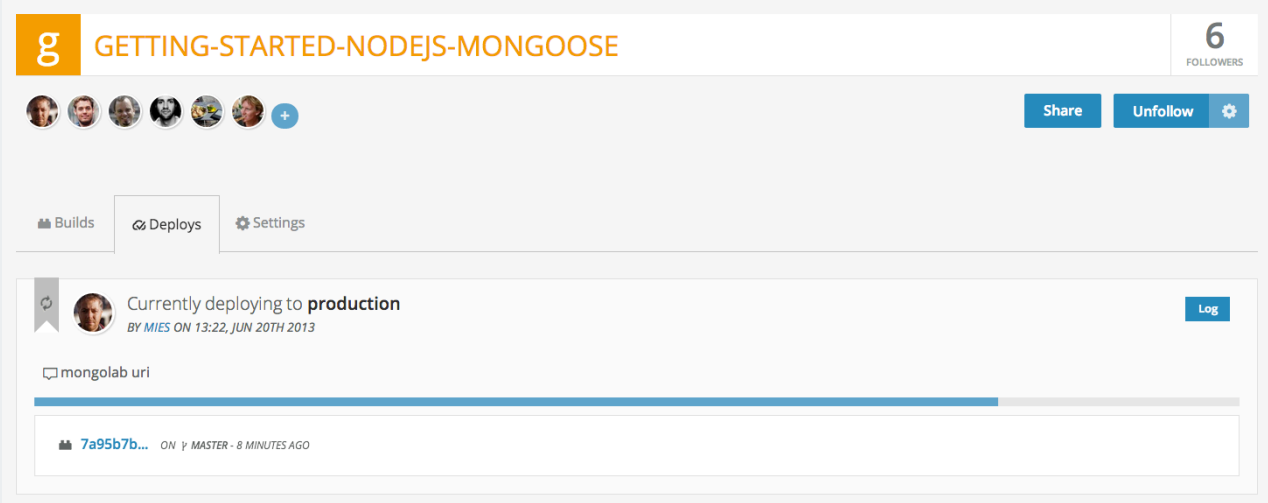
[Windows](#) (1)

[.Net](#) (1)

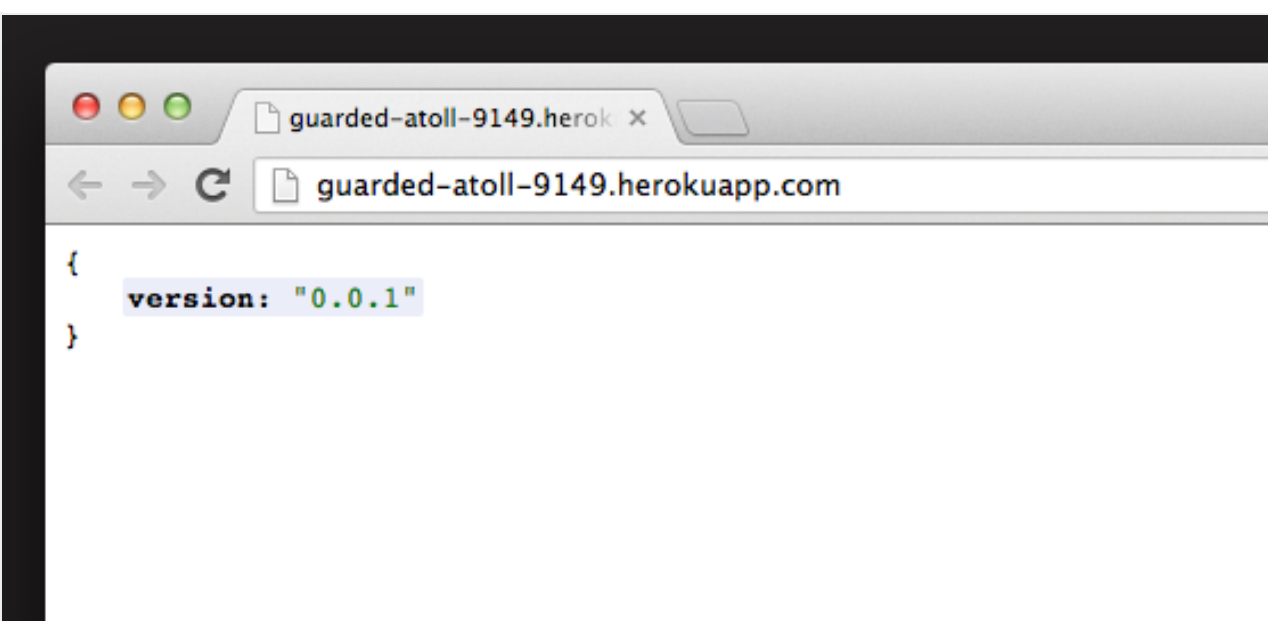
[Dotnet](#) (1)

[C#](#) (1)

[Azure](#) (1)



If we visit the root of our application on Heroku we can see it in action:



Congratulations! You’ve succesfully built a REST API using test-driven development and leveraged wercker for continuous delivery. Subscribe to the [feed](#) of our blog and follow us on [twitter](#) for more updates.

## Earn some stickers!

Let us know about the applications you build with wercker. Don’t forget to tweet out a screenshot of your first green build with **#wercker** and we’ll send you some [@wercker](#) stickers.

[Capgemini](#) (1)

[Apollo](#) (1)

[Continuous Integration](#) (1)

[Puppet](#) (1)

[Bundler](#) (1)

[Slack](#) (1)

[Services](#) (1)

[Docker](#) (1)

[Termie](#) (1)

[Logging](#) (1)

[Logs](#) (1)

[Guestpost](#) (1)

[Services](#) (1)

[AppEngine](#) (1)

[Statistics](#) (1)

[Marketplace](#) (1)

[Builds](#) (1)

[Filter](#) (1)

[Devopsdays](#) (1)

[Conference](#) (1)

[Amsterdam](#) (1)

[Ux](#) (1)

[Menu Bar](#) (1)

[Css3](#) (1)

[Css-Animation](#) (1)

[Product Design](#) (1)



Don't forget to sign up for wercker for free [here](#).

[Autodeployment](#) (1)

[Feature](#) (1)

[Releases](#) (1)

[Assets](#) (1)

[Drupal](#) (1)

[Email](#) (1)

[Dartlang](#) (1)

[Hapi](#) (1)

[Phusion Passenger](#) (1)

Share this post on:



Automation-driven development

[Sign up and get started >](#)

Join the wercker community on    

[Join the developer newsletter >](#)

Copyright ©2016 wercker   [Contact & support](#)   [About](#)   [Downloads](#)   [Terms](#)   [Security](#)