13
Node and MongoDB In the Cloud

Host your MongoDB database on mLab

- mLab - A cloud database hosting service for MongoDB.
- Free to sign up and use the sandbox databases, which is fine for our purposes and the term project Foodie! app.
- https://mlab.com
- Under “MongoDB Deployments” click “create new” and give the new database a name.
- Add a user to the database and remember the password because you will need it later for your node app to connect to the database.
- Add a collection to your database and make sure it matches the collection name you are using in your app’s code.

Additional Testing Considerations

- After setting up your MongoDB database on mLab's website, you should run your local MongoDB shell and make sure you can connect to the remote database and run commands like find and insert. If you have trouble connecting to the database from the shell, your app will also have trouble.

- You can run your app locally (instead of on Heroku) and connect to your remote database instead of your local MongoDB on your machine by changing the connection uri in your app.js. Your app should run just like before except now you are using a remote database instead of a local one. Do this to test that the database works before deploying your app to Heroku.

Host your app on Heroku and your code on GitHub

- Make sure to create a new git repository all by itself for your node apps!

- Having apps in their own repos makes it much easier to deploy to Heroku. Don’t try to deploy from a directory in our class repo as that seems to confuse Heroku’s deployment service.

- For the Foodie! final project you can push your code to a public repo on GitHub - but make sure you do not commit your database password to this repo! More on this later.

Hosting your application on Heroku

- Make sure you have the Heroku CLI installed.
- Run “heroku login” in your app directory and enter your Heroku account email and password. If you setup 2-factor authentication then you will also be prompted to enter your auth token.
- Then run “heroku create” to create a new app on Heroku. The Heroku CLI should add a new remote to your git repo named heroku.
- Finally run “git push heroku master” to deploy your app on Heroku.

How to keep your database password a secret

- Git keeps a history of all your commits so you really don’t want to accidentally commit some type of sensitive information like passwords to a repository because it's a major pain to erase those commits from the history. But since your connection string is in your code, how do you keep it a secret?

- The solution is to place your database password (and really you should place the entire connection string uri) into a config variable on Heroku.

- Run the command “heroku config:set MONGODB_URI=<uri_from_mlab>” to create a config variable on Heroku that holds your database connection info.

How to keep your database password a secret

- You can then run “heroku config:get MONGODB_URI” to verify the data is correctly stored in Heroku.

- Then to access this environment variable from your node app, replace the connection string uri parameter with the following code: process.env.MONGODB_URI
- When the app runs on Heroku it will then use the connection data stored in this variable to connect to your remote database on mLab.
- Now it is ok to commit your changes to your app.js code since it does not contain any sensitive information, only the name of the environment variable.

Creating a Heroku config variable for database uri

```
➜  node-express-mongo-app git:(master) ✗ heroku config:set MONGODB_URI=mongodb://derick:derick@ds161630.mlab.com:61630/students
Setting MONGODB_URI and restarting ⬢ agile-taiga-48517... done, v4
MONGODB_URI: mongodb://derick:derick@ds161630.mlab.com:61630/students
➜  node-express-mongo-app git:(master) ✗ heroku config:get MONGODB_URI
mongodb://derick:derick@ds161630.mlab.com:61630/students
➜  node-express-mongo-app git:(master) ✗ _
```

app.js

```
1  var express = require('express')
2  var bodyParser = require('body-parser')
3  var path = require('path');
4  var expressValidator = require('express-validator')
5  var mongojs = require('mongojs')
6  var db = mongojs(process.env.MONGODB_URI, ['students'])
7  var ObjectId = mongojs.ObjectId
8
```

Some Useful Resources

- mLab - https://en.wikipedia.org/wiki/MLab
- Deploying node apps on Heroku - https://devcenter.heroku.com/articles/deploying-nodejs
- Setup a database connection from Heroku to mLab - https://devcenter.heroku.com/articles/mongolab
- Tutorial that goes over setting up a db connection from Heroku to mLab - https://scotch.io/tutorials/use-mongodb-with-a-node-application-on-heroku