Extra Notes - Node.js Todos API - Heroku and Postman

- Dr Nick Hayward

Contents

app - deploy app to Heroku with MongoDB publication

We can now modify our app for test publication to Heroku with a live instance of MongoDB.

Config modifications for current app code include,

`server.js`

- add environment port variable - Heroku can dynamically set port number of app or app will default to local 3030
- `app.listen()` - updated to use variable `port` for dynamic port setting

`package.json`

- add `start` to scripts - tells Heroku how to run our app
  - `"start": 'node server/server.js'`
- specify node version to use for hosted app
  - add `engines` with current version used for developing app on local system
  - e.g. `"engines": {"node": "8.2.1"}`

Then, we need to create a remote instance of MongoDB for our app's data. On Heroku, we install an add-on for *mLab MongoDB*, which is a hosted service for MongoDB.

Further details on Heroku add-ons,

- https://elements.heroku.com/addons

To install this add-on we can run the following command in our project's CWD,

```
heroku create //create a new heroku app if necessary
heroku addons:create mongolab:sandbox //install add-on for mLab MongoDB with free
sandbox plan
```

mLab offers a free sandbox option, which is installed for our app.

Then, for connections to MongoDB, we can modify `mongoose-config.js` to add a dynamic URL for the db. This will either include the return URL for mLab or the current default for the local machine, e.g.

```
//connect to MongoDB using Mongoose - use mLab or local uri
mongoose.connect(process.env.MONGODB_URI || 'mongodb://localhost:27017/NodeTodoApp');
// process environment returns mLab uri
```

For some apps and Git repos, it may also be necessary to ensure that the local project is linked to the remote app repo on Heroku, e.g.

```
heroku git:remote -a app-name
```

Then, we can push our project repo to Heroku, and deploy the app, e.g.

```
git push heroku master
```

or for a subtree directory in a git repo, e.g.

```
git subtree push --prefix node/apps/node-todos-api/v0.8 heroku master
```

We can then open the heroku app with the following command, e.g.

```
heroku open
```

As a useful extra, we can also check the logs for this type of install, e.g.

```
heroku logs
```

app - use Postman with Heroku app

We can now test our new Heroku app with Postman, both GET and POST requests for the new remote app.

We might test sending a POST request to the app, e.g.

- https://your-app-url.herokuapp.com/todos

which will create a test todo item that is set in the app. The return object for this POST request will be as follows,

```
{
    "__v": 0,
    "text": "postman test todo item - another one...",
    "_id": "597cd962d828090011f2b9ce",
    "completedAt": null,
    "completed": false
}
```

If we then submit a GET request to the app's API,

- https://your-app-url.herokuapp.com/todos

we'll get the expected object containing an array of todo items, e.g.

```
{
    "todos": [
        {
            "_id": "597cd962d828090011f2b9ce",
            "text": "postman test todo item - another one...",
            "__v": 0,
            "completedAt": null,
            "completed": false
        }
    ]
}
```

We can test retrieving a single todo item by ID, e.g.

- https://your-app-url/todos/597cd962d828090011f2b9ce

which will return an object with the single todo item,


```
{
    "todo": {
        "_id": "597cd962d828090011f2b9ce",
        "text": "postman test todo item - another one...",
        "__v": 0,
        "completedAt": null,
        "completed": false
    }
}
```


To ease switching test environments in Postman, we can create environments for local, Heroku &c. and then save them for easy recall.

e.g. in the top right corner of Postman is a drop down menu for environemnt.

So, we can now create an environment for the local dev and remote dev projects.

In *Manage Environments*, we can add an environment, e.g. `Todo App Local`, and then set values for the following

- url = localhost:3030

Then, we can do the same for Heroku, and set the URL value to the Heroku app url, e.g.

- https://your-app-url.herokuapp.com

We can also abstract routes and params as required for testing with defined environments.

e.g. for the GET request in the *Todo App* collection we can modify the URL as follows,

- {{url}}/todos

If we switch to either the local or Heroku environment, this single request is now abstracted to either environment.

We can also do the same for the POST request in the collection.