

Andrea Christians
B00412494

Tools used and justification:

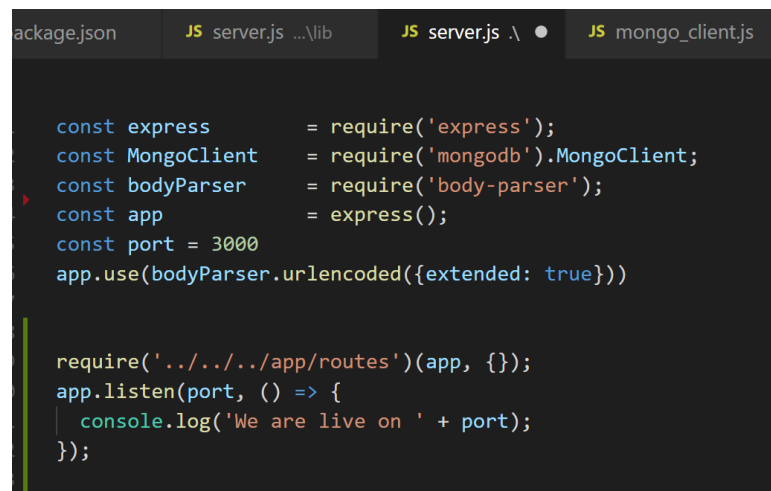
I followed two tutorials to create this application using Node.js (see references). I have limited knowledge of Node.js so I felt this would be a good learning opportunity. Specifically, in order to configure my server, I used express for my get/post requests. In order to create a dynamic web-page, I used EJS. It rendered the data for both requests, specifically it was used to display the 'get' dynamically. MongoDB is an open source backend tool that I used to store data in under the collection 'cats'. I used the 'mLab' tool to easily access the database and the database that was deployed through AWS. Finally, I deployed the application through Heroku which is a simple tool for a web application such as Node.js.

Tools:

- Express
- Nodemon
- EJS
- MongoDB with mLab through AWS
- Heroku for deployment

Methodology:

1. Launch Node.js App and install add-on's, i.e. express, mongoDB
2. Configure server for localhost i.e. create server.js file and routes, parse data to json through bodyParser add-on



```
package.json  JS server.js ...lib  JS server.js  JS mongo_client.js

const express      = require('express');
const MongoClient   = require('mongodb').MongoClient;
const bodyParser    = require('body-parser');
const app           = express();
const port = 3000
app.use(bodyParser.urlencoded({extended: true}))

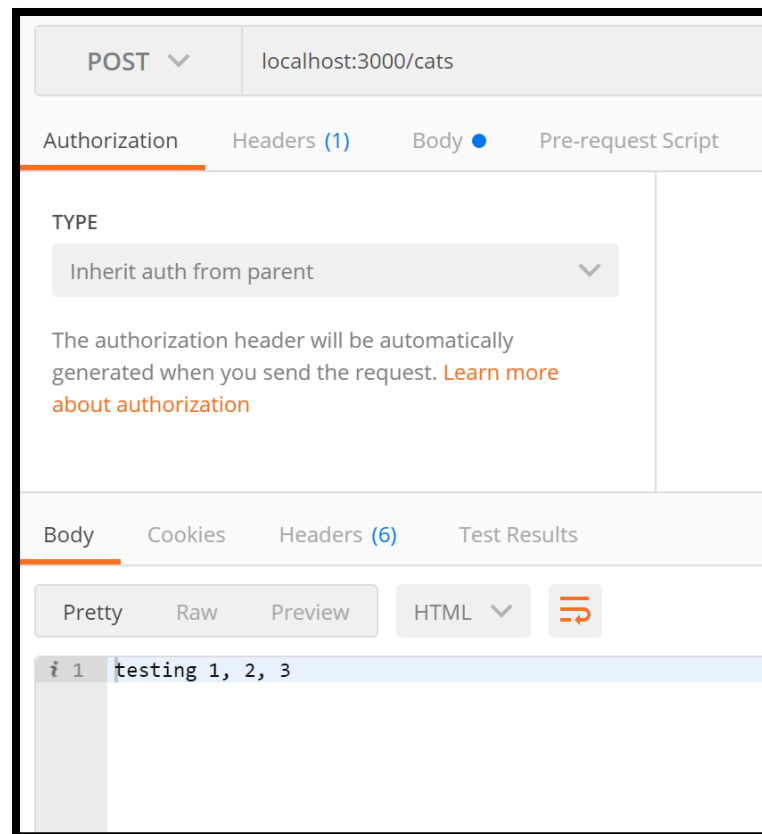
require('../../../app/routes')(app, {});
app.listen(port, () => {
  console.log('We are live on ' + port);
});
```

3. Run a simple 'post' request on localhost

Create a post route in routes :

```
app.post('/cats', (req, res) => {  
  res.send(["testing 1, 2, 3"])  
})
```

Test in Postman:



4. Configure MongoDB on mLab in AWS – with collection ‘cats’

Database: lab1

Delete database

To connect using the mongo shell:
% mongo ds113855.mlab.com:13855/lab1 -u <dbuser> -p <dbpassword>

To connect using a driver via the standard MongoDB URI ([what's this?](#)):
`mongodb://<dbuser>:<dbpassword>@ds113855.mlab.com:13855/lab1`

mongod version: 3.6.9 (MMAPv1)

⚠ Sandbox databases do not have redundancy and therefore [are not suitable for production](#). Read our documentation on [how to upgrade](#).

Collections

Users

Stats

Backups

Tools

Collections

Delete all collections

+ Add collection

NAME	DOCUMENTS	CAPPED?	SIZE ⓘ
cats	2	false	8.20 KB

I configured a user called ‘mittens’ to access the database.

Database Users

+ Add database user

NAME	READ ONLY?
mittens	false

5. Configure MongoDB to the server.js file, update get and post commands to send data to database.

In server.js file:

```
MongoClient.connect(dblk, (err, database) => {
  if (err) return console.log(err)
  db = database.db("lab1")
  require('./app/routes')(app, database);

  app.listen(process.env.PORT || 3000, () => {
    console.log('We are live on ' + port);
  });
})
```

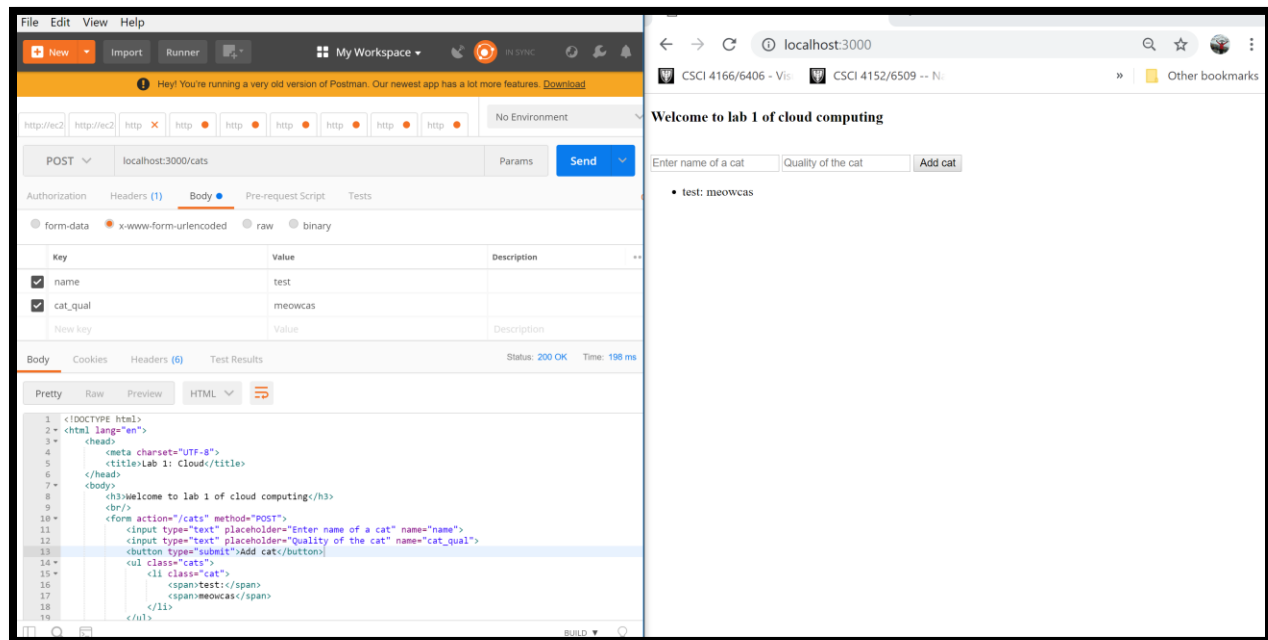
In routes:

```
app.post('/cats', (req, res) => {
  db.collection('cats').save(req.body, (err, result) => {
    if (err) {
      return console.log(err)
    } else {
      console.log('saving the cats!')
      res.redirect('/')
    }
  })
})

// app.post('/cats', (req, res) => {
//   res.send("testing 1, 2, 3")
// })

app.get('/', (req, res) => {
  db.collection('cats').find().toArray((err, result) => {
    if (err) {
      return console.log(err)
    } else {
      console.log('rendering')
      res.render('index.ejs', {cats: result})
    }
  })
})
```

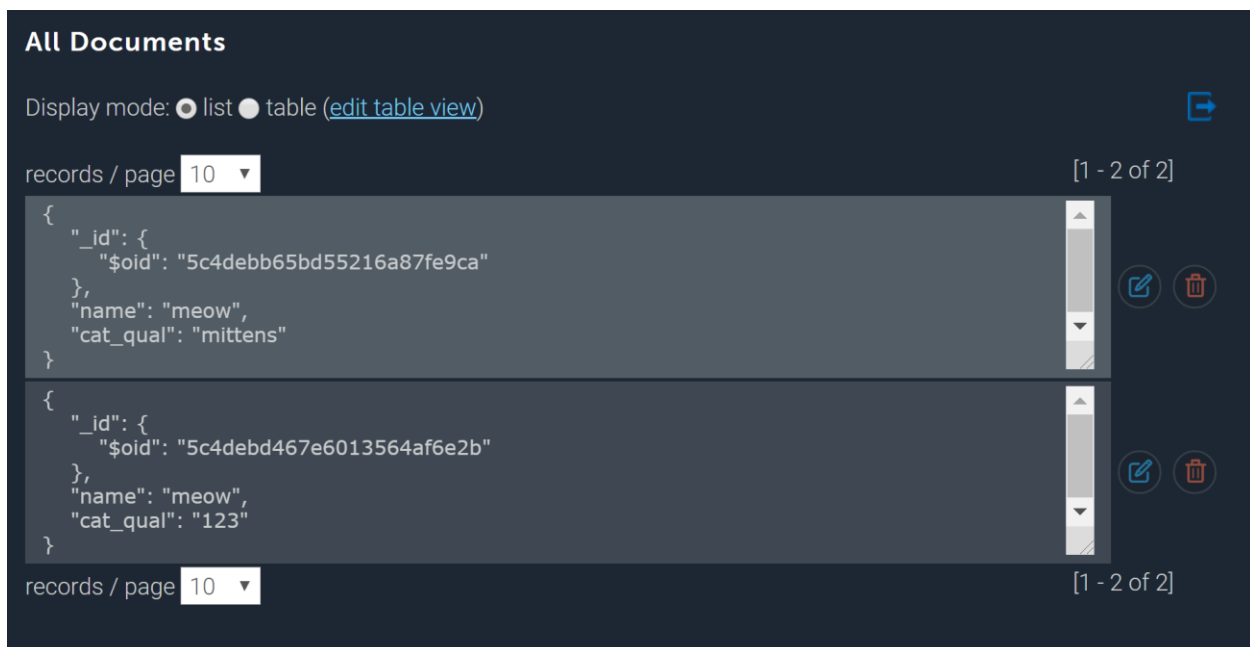
6. Run the server 'post' with MongoDB



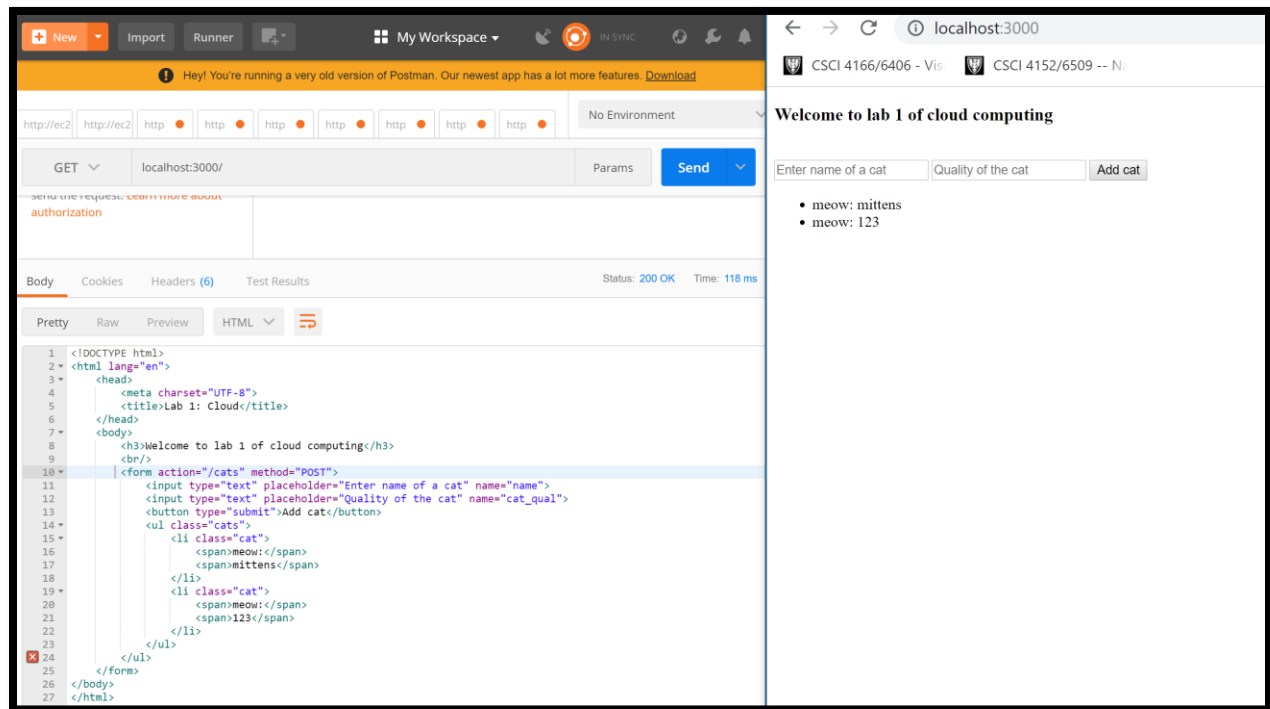
```
C:\Users\AChristians\Desktop\Cloud\lab1api>npm start
```

```
> lab1api@1.0.0 start C:\Users\AChristians\Desktop\Cloud\lab1api
> node server.js
```

We are live on 3000
saving the cats!



7. Run 'get' request with MongoDB



```
C:\Users\AChristians\Desktop\Cloud\lab1api>npm start
```

```
> lab1api@1.0.0 start C:\Users\AChristians\Desktop\Cloud\lab1api
> node server.js
```

We are live on 3000
rendering



8. Deploy on Heroku.

In gitbash:

```
AChristians@DESKTOP-4QTBUE MINGW64 ~/Desktop/cloud/lab1api (master)
$ 2019-01-27T02:54:00.000000+00:00 app[api]: Build succeeded
2019-01-27T02:54:00.000000+00:00 app[api]: Build succeeded
2019-01-27T02:54:01.582107+00:00 heroku[web.1]: Starting process with command `npm s
2019-01-27T02:54:01.582107+00:00 heroku[web.1]: Starting process with command `npm s
2019-01-27T02:54:03.615859+00:00 app[web.1]:
2019-01-27T02:54:03.615876+00:00 app[web.1]: > lab1api@1.0.0 start /app
2019-01-27T02:54:03.615878+00:00 app[web.1]: > node server.js
2019-01-27T02:54:03.615879+00:00 app[web.1]:
2019-01-27T02:54:03.615859+00:00 app[web.1]:
2019-01-27T02:54:03.615876+00:00 app[web.1]: > lab1api@1.0.0 start /app
2019-01-27T02:54:03.615878+00:00 app[web.1]: > node server.js
2019-01-27T02:54:03.615879+00:00 app[web.1]:
2019-01-27T02:54:04.108966+00:00 app[web.1]: We are live on 3000
2019-01-27T02:54:04.108966+00:00 app[web.1]: We are live on 3000
2019-01-27T02:54:05.210998+00:00 heroku[web.1]: State changed from starting to up
2019-01-27T02:54:05.210998+00:00 heroku[web.1]: State changed from starting to up
```

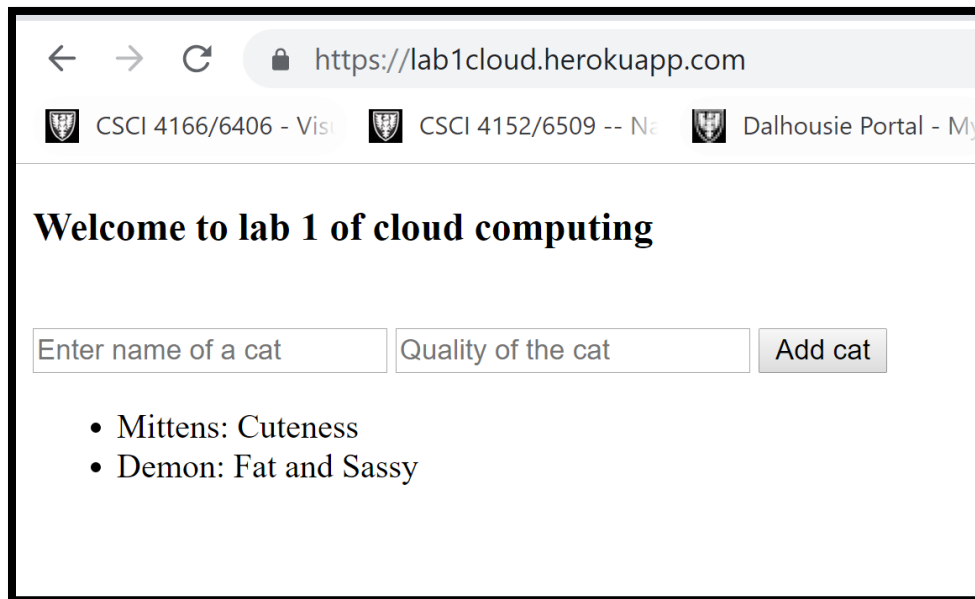
Code for page:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Lab 1: Cloud</title>
</head>
<body>
<h3>Welcome to lab 1 of cloud computing</h3>
<br/>
<form action="/cats" method="POST">
  <input type="text" placeholder="Enter name of a cat" name="name">
  <input type="text" placeholder="Quality of the cat" name="cat_qual">
  <button type="submit">Add cat</button>

  <ul class="cats">
    <% for(var i=0; i<cats.length; i++) {%>
      <li class="cat">
        <span><%= cats[i].name %></span>
        <span><%= cats[i].cat_qual %></span>
      </li>
    <% } %>
  </ul>

</ul>
</form>
</body>
</html>
```

Final deployment of application



The screenshot shows a web browser window with the address bar displaying `https://lab1cloud.herokuapp.com`. The browser's tab bar shows three tabs: "CSCI 4166/6406 - Visi", "CSCI 4152/6509 -- Na", and "Dalhousie Portal - My". The main content area of the browser displays the following:

Welcome to lab 1 of cloud computing

Below the heading, there is a form with two input fields and one button:

- Input field: "Enter name of a cat"
- Input field: "Quality of the cat"
- Button: "Add cat"

Below the form, there is a bulleted list:

- Mittens: Cuteness
- Demon: Fat and Sassy

References:

<https://medium.freecodecamp.org/building-a-simple-node-js-api-in-under-30-minutes-a07ea9e390d2>

<https://zellwk.com/blog/crud-express-mongodb/>