Lab 7
MongoDB via Mongoose

CSCI2720 Building Web Applications

Dr. Chuck-jee Chau
chuckjee@cse.cuhk.edu.hk

Agenda

- Setting up MongoDB
- Connecting via Mongoose
- Schema and Model
- Handling GET/POST for database access

CSCI2720 VM

- Remember the special URL for you

  http://csci2720-g?.cse.cuhk.edu.hk

  - *?* is from `0` (not `00`) to `130`, different for every user
  - *Note: This is only accessible via CUHK network (e.g. CUHK VPN)*
- Please check your @link email for the account details
  - Username: **s1155xxxxxx**
  - Password: **shown in email**
  - *Check spam/junk box if it's not there!*

SSH to the server

- You have to access the server via SSH with the assigned account (*s1155xxxxxx*)
- On campus/ CUHK VPN
  - Directly SSH to `csci2720.cse.cuhk.edu.hk`

Syntax here is: username@server

```
[chuckjee@pc89230 ~ % ssh chuckjee@csci2720.cse.cuhk.edu.hk
[chuckjee@csci2720.cse.cuhk.edu.hk's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-136-generic x86_64)
```

Your work from Lab 6

- In lab 6, you have already built a basic Express server
- It should be capable to understand **GET** and **POST** requests to one of this addresses with an **`eventId`**, a **`locId`** and a **`loginId`** (for **POST** only)
  - *http://csci2720…/**event**/…/**loc**/…*
- We will carry on from there

Using MongoDB

- At the VM, you can access the mongo shell by typing

  `mongo -u username userdb`

  - Type your proper username instead of *username*, and for *userdb* too
- There is a separate password for MongoDB, as in the email
  - Sorry you cannot change your password for MongoDB here (too much hassle!)
- In Mongo, you are only entitled to your assigned database (the one named as your username)
  - Switch to your assigned database:

    `use userdb`

Important!

Make sure you can do this step properly, since you'll also work on the course project in CSCI2720 VM

A new database item
a.k.a. ***document in collection***

- Now let's create a new *collection* **`new`** with a new *document*

```
db.new.insert({
  x: "world",
  y: 2720,
  z: [1, 2, 3],
  easy: true
});
```

- If you see **`WriteResult({"nInserted":1})`**, that means you succeeded
- Verify the insert using

```
db.new.find();
```

- You should see this new document with its **`_id`** property, and a *string*, an *integer*, an *array*, and a *Boolean*
- *This collection/document is for testing only!*

More on the mongo shell

- MongoDB allows easy manipulation using the **`mongo`** shell
- It is a handy tool for checking the database structure, and doing direct changes to the database (e.g. create, find, …)
- A lot more commands are available in the **`mongo`** shell:

  https://docs.mongodb.com/manual/reference/mongo-shell/

- We will use ***Mongoose*** for interfacing from Express

Connecting via Mongoose

- Back to the Express server created last time
  - On CSCI2720 VM in ***lab6***
- Install Mongoose support

```
npm install mongoose
```

- In your js file, add this

```
var mongoose = require('mongoose');
mongoose.connect('server URL');
```

- The ***server URL*** would be either one of these:
  - `mongodb://user:pass@localhost/userdb`
  - Use your *username*, *password* and *userdb* there
- You may also use the ***options*** parameter as in the lecture slides

Connecting via Mongoose

- Then you can show appropriate messages depending on whether the connection is successful
- This is our current js file…

```
var express = require('express');
var app = express();

var mongoose = require('mongoose');
mongoose.connect('your actual server URL');

var db = mongoose.connection;
// Upon connection failure
db.on('error', console.error.bind(console,
'Connection error:'));
// Upon opening the database successfully
db.once('open', function () {
  console.log("Connection is open...");
});

/* ... Lab 6 work on app.get() and app.post() ... */

var server = app.listen(/*assigned port*/);
```

- Run/reload your server and see if the database connection is successful

Schema and Model

- Our aim is to store some event information into the database
- Insert the following schema definition after the db connection is made successfully

```
var EventSchema = mongoose.Schema({
    eventId: { type: Number, required: true,
unique: true },
    name: { type: String, required: true },
    loc: { type: String },
    quota: { type: Number }
});
```

- And create a model based on this schema

```
var Event = mongoose.model('Event', EventSchema);
```

Handling GET requests

- To avoid confusion, you may comment out the GET request handler in Lab 6, since we will make a simpler one here first

```
app.get('/event/:eventId', function(req,res) {
  Event.findOne(
    {eventId: req.params['eventId']},
    'eventId name loc quota',
    function(err, e) {
      if (err)
        res.send(err);
      res.send("This is event "+e.eventId+":<br>\n" +
       "Event name: " + e.name + "<br>\n" +
       "Event location: " + e.loc + "<br>\n" +
       "Event quota: " + e.quota + "<br>\n" +
       "Ref: " + e);
  });
});
```

- ***Check your syntax very carefully!!***
- **Important**: You need to manually create a document in mongo to test, using `db.events.insert(…)`

Handling POST requests

- You may also comment out the POST request handler you made in Lab 6

```
app.post('/event', function(req,res) {
  var e = new Event({
    eventId: req.body['eventId'],
    name: req.body['name'],
    loc: req.body['loc'],
    quota: req.body['quota']
  });
  e.save(function(err) {
     if (err)
       res.send(err);
     res.send("Ref: " + e);
  });
});
```

- You can get this form for submitting the POST request

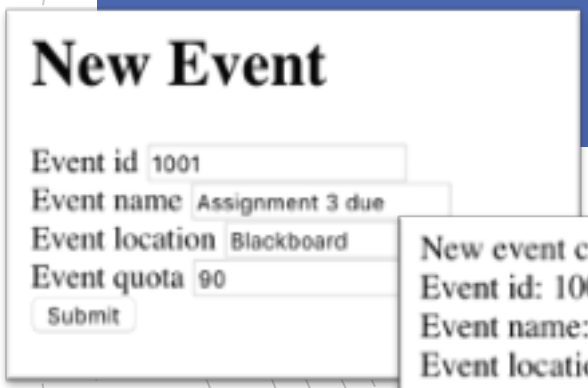
  https://www.cse.cuhk.edu.hk/~chuckjee/2720lab7/form.html

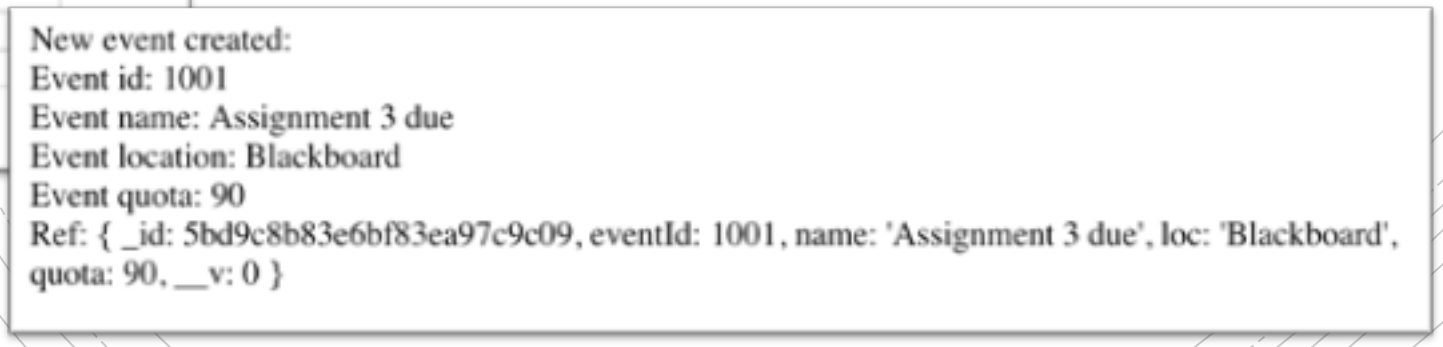
Handling POST requests

- For this POST handler to work, you need the ***body parser***

```
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({extended:false}
));
```

- You have to insert the correct URL for the ***form action*** in the HTML file
  - `http://csci2720.../event`
- Are you able to improve the result after posting, like this?





Quick Summary

- Try your work using the CSCI2720 VM environment
  - You could try on your own device later
- A number of techniques are involved in this lab:
  - Node.js + Express
  - MongoDB + Mongoose
  - HTML form, HTTP GET/POST
- It is very easy to get lost
- It is likely you get confused by syntax
- ***Thank you for your patience!***

Submission

- No submission is needed for labs
- What you have done could be useful for your further exploration or the upcoming assignment
- **Please keep your own code safely**