Install Express

1. Ensure you have completed previous exercises – especially installation of mongo
2. Create directory D:\Projects\hello-express
3. Open Visual Studio Code
4. Open

File->Open Folder..-> D:\Projects\hello-express Select Folder

1. In Visual Studio Code select View->Integrated Terminal
2. Run following to create a package for express

npm init

Accept all defaults. This creates a package.json file.

1. In Integrated Terminal run

npm install express --save

This downloads and installs external modules (including express) in node\_modules

Write First Express Application

1. Create a file called server.js
2. Enter following:

var express = require('express');

var app = express();

app.get('/', (req, res) => {

res.send('<h1>Home page TODO</h1>');

});

app.get('/login', (req, res) => {

res.send('<h1>Login page TODO</h1>');

});

app.listen(3000, () => {

console.log('listening on 3000');

});

1. In the Integrated Terminal start the node application server.js by typing:

node server

1. Open a browser and test each of these endpoints

Start mongodb

1. Start mongoDB in a command prompt.
2. In another command prompt open a mongodb shell and list all the contents of the LibraryApp;

Integrate mongodb with Express (and Node)

1. Still using the server.js app, in the Integrated Terminal install mongoDB through npm

npm install mongodb --save

(mongodb will be added to node\_module)

1. Modify file, by adding reference to mongoDB module and library collection

var mongodb = require('mongodb').MongoClient;

var url = 'mongodb://localhost:27017/libraryApp';

1. Add a route to get all books in the library. Using the code snippet below and modifying collection.find to retrieval the whole collection:

mongodb.connect(url, (err, db) => {

if ( err ) {

console.log('Error connecting to mongoDB', err);

return;

}

// Connected to DB

var collection = db.collection('books');

collection.find<PUT YOUR QUERY HERE>

.toArray((err, docs) => {

console.log(docs);

db.close();

// Response OK write to browser

res.writeHead(200, { 'Content-Type': 'text/html' });

var resText = '';

for (i=0; i <docs.length; i++) {

resText = resText + '<h1>' + docs[i].title + '</h1>';

}

res.end(resText);

});

});

1. Test your route (note you have to stop and start the previous node session).
2. Add a route that finds all books by genre e.g. on browser '/allGenre/History'.
3. Add route /allGenre/:genreName
4. Extract the generName using req.params.genreName
5. Use mongoDB’s reqex to look for similar matches

{ genre: { $regex:’History’} } // similar to like ‘%History%’

(note test out your mongo query from the mongo shell)

1. Test out your code with /allGenre/History & /allGenre/Hist