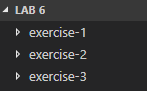
**COMP 3123 – Full Stack Development – Lab 6**

* Express + Postman
* MongoDb + Mongoose

**Developer Note:**When working on your exercises, please create separate folder for your work. This way you won’t putting all your code in the same file, which can pollute the global name space. In short, it will prevent you from overwriting your own work and causing your code to compile incorrectly.

Organize your folder structure in this way.

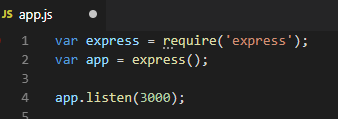


**Exercise #1 – Download and Configure Express.js**

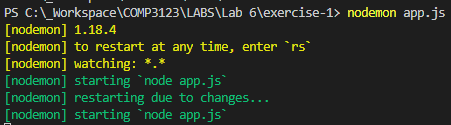
1. Create a folder named Lab 6 and run the following command to install express js.



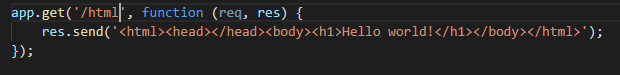
1. Open a command prompt create a directory for **exercise-1**
2. Open Visual Studio Code and open the folder **exercise-1**
3. Add a file named **app.js** and write the following code.



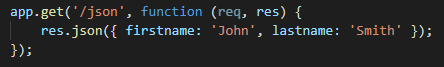
1. Run **nodemon app.js** at the command line to start the web server



1. Configure the following route using express to return HTML content when **‘/html’** route is requested



1. Configure the following route using express to return JSON content when the **‘/json’** route is requested



1. Use the browser to request the routes **/html** and **/json** to view the results.
2. Configure the following route and use express for matching the route paths.

<http://expressjs.com/en/guide/routing.html>

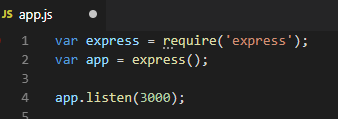


1. Create a pattern to match the following routes.   
   There should be a wild card operator in some or one of the characters **\*, +, ?, to** allow any text in between the ‘toronto’ and ‘team’ route pattern.

/toronto**raptors**team  
/toronto**marlies**team  
/toronto**abc**team  
/torontoteam

**Exercise #2 – Postman and Middleware**

1. Open a command prompt create a directory for **exercise-2**
2. Open Visual Studio Code and open the folder **exercise-2**
3. Add a file named **app.js** and write the following code.

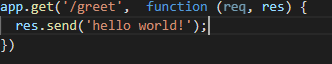


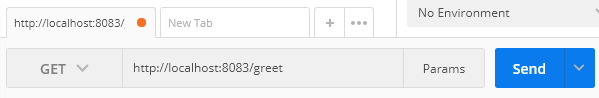
1. Run **nodemon app.js** at the command line to start the web server

1. Install **Postman** extension for Chrome as a REST client

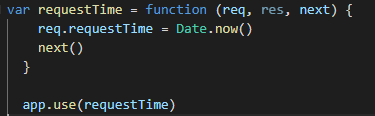
<https://developers.sap.com/cis/tutorials/api-tools-postman-install.html>

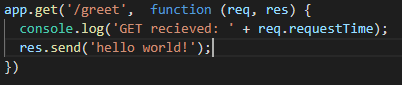
1. Write the following code in **app.js** and use Postman to trigger the routes and inspect the results.





1. Add the following custom middleware function to log the request time. Update the GET route to use the requestTime. Use POSTMAN to test the results.





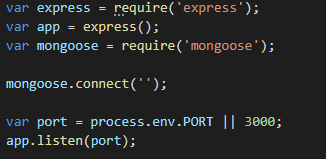
1. Write a route for POST, PUT, DELETE routes and test with POSTMAN

**Exercise #3 – MongoDb and Mongoose**

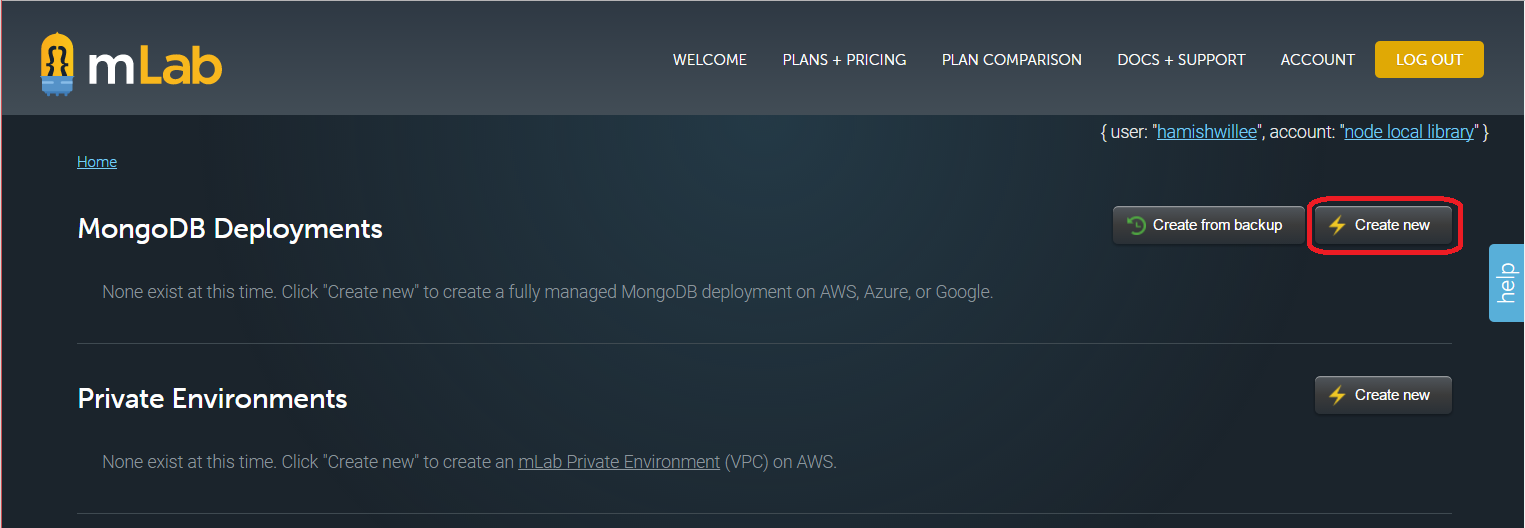
1. Open a command prompt create a directory for **exercise-3**
2. Open Visual Studio Code and open the folder **exercise-3**
3. Using npm install Mongoose ORM to connect to MongoDb

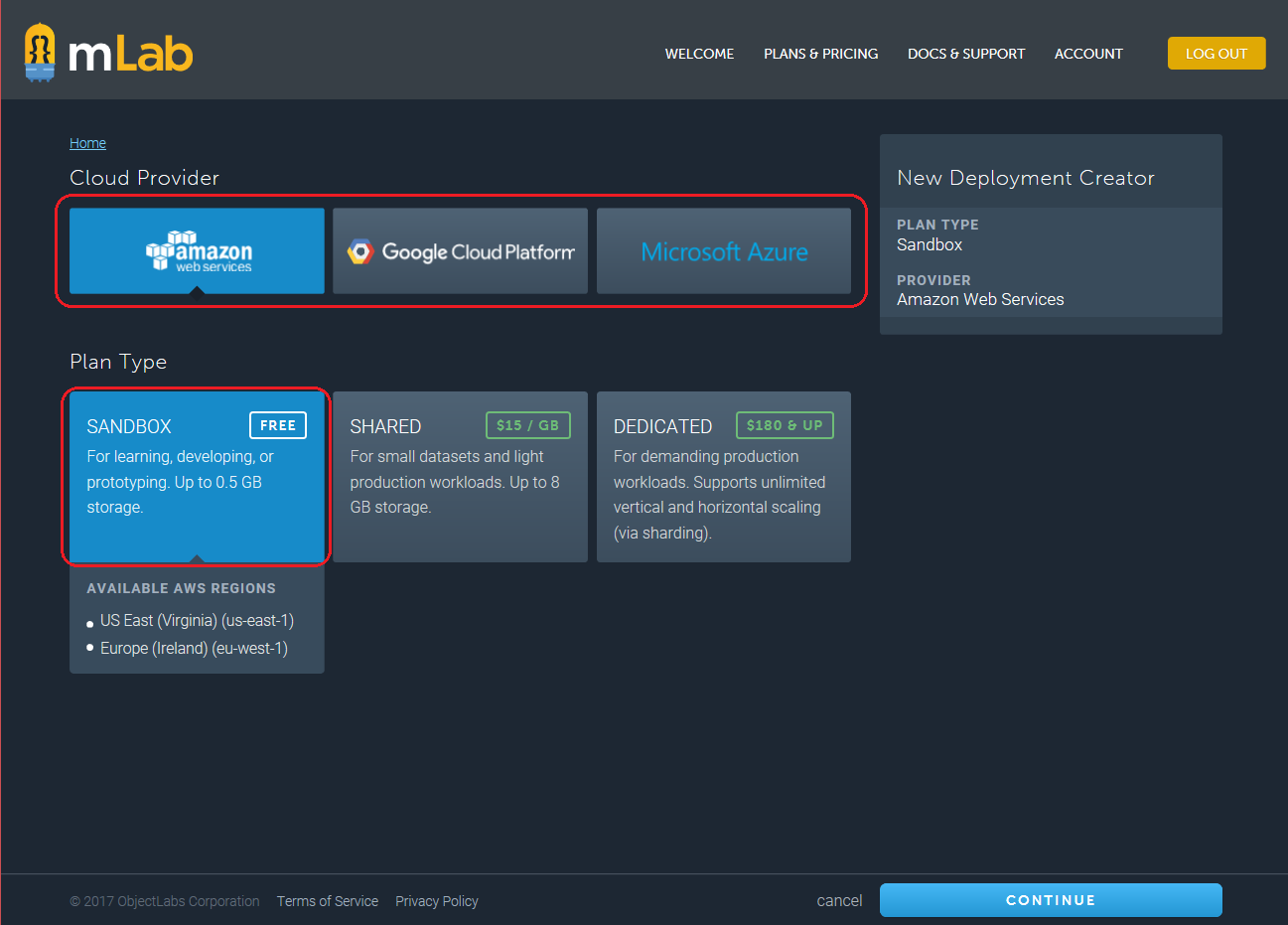


1. Create an app.js file and write the following code:

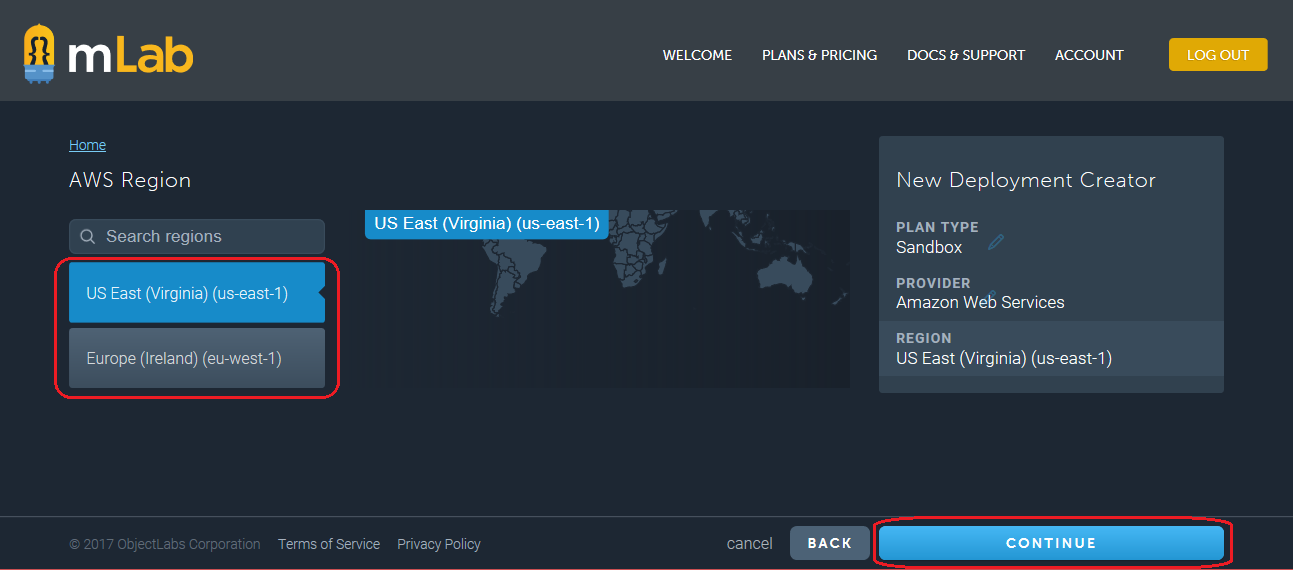


1. Sign up and create a new MongoDb deployment at <https://mlab.com>.
2. Click **Create New** in the MongoDB Deployments section.



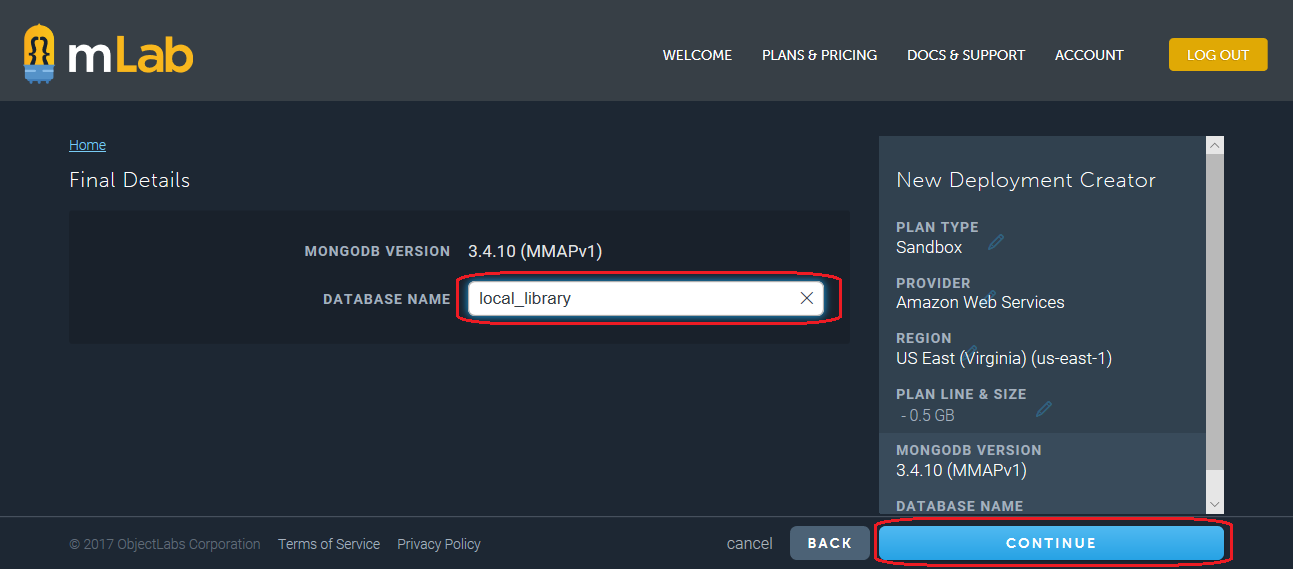
1. This will open the Cloud Provider Selection screen  
   

* Select the SANDBOX (Free) plan from the Plan Type section.
* Select any provider from the *Cloud Provider*section. Different providers offer different regions (displayed below the selected plan type).
* Click the **Continue** button.

1. This will open the Select Region screen.  
   

* Select the region closest to you and then **Continue**.

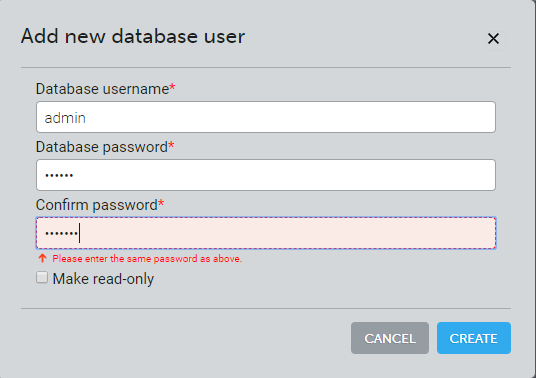
1. This will open the Final Details screen.



* Enter the name for the new database as TestDb and then select **Continue**.

1. You will be returned to the home screen. Click the Users tab and create a new Database user

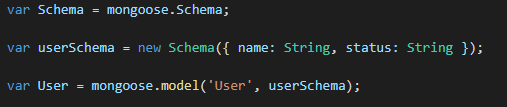
1. Enter a username and password (twice), and then press **Create**. Do not select Make read only.

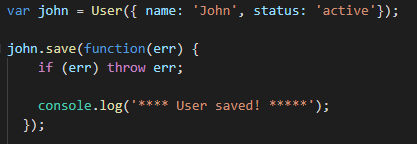


1. You have now created the database, and have an URL (with username and password) that can be used to access it.   
   
2. Copy the MongoDB URI and paste it in the app.js Change the **dbuser** and **dbpassword** to the values you entered in step 8.



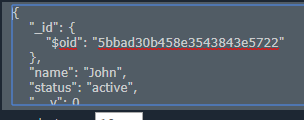
1. In the app.js file create the MongoDB schema and model for User



1. Create a new User model and use save to write the data to the MongoDB database.   
     
   
2. Run node or nodemon app.js at the command line. Go to the mLabs sandbox and refresh the page. In the collections tab you will see a collection named ‘users’ with one document.



1. Click the name ‘users’ to view the following newly saved User document



1. Create a new Mongoose User Schema and save a new User model for named ‘Jane’