## Back-End Web API Setup

The following instructions are copied word-for-word from the README.md file for the teams-api sample (available Under [“Code Examples”](https://github.com/sictweb/web422))

## Teams API

### Detailed Instructions:

Once you have cloned the repository, open the folder in Visual Studio code and issue the command “npm install” from the integrated terminal. This will look at the package.json file and add the required dependant modules.

If you open the “server.js” file, you will notice that the top 2 lines define the constants “mongoDBConnectionString” and “HTTP\_PORT”. The HTTP\_PORT is fine the way it is (unless you have a conflict on your local machine with port 8081), however the mongoDBConnectionString value will need to change…

### MongoDB Database:

For this course we will be continuing to use MLab:

1. Log into your MLab Account here: <https://mlab.com/login/>
2. Create a new “MongoDB Deployment” by clicking on the “Create new” button on the top right of the table of existing deployments (you should see a few from your WEB322 assignments)
3. Be sure you select a “Sandbox” Plan Type and click the blue “Continue” button
4. Select “US East” and click the blue “Continue” button
5. On the next screen, name your database something that you will remember, i.e. “teams-api-db” and click the blue “Continue” button
6. Finally click the “Submit Order” button - your “Total Price” should read “FREE”

### Adding a New User:

As you will recall from WEB322, we aren’t done yet - we still need to add a user to the new deployment:

1. Click on your newly created deployment in the list. This will take you to a separate screen with the “Collections” tab selected.
2. Click on the “Users” tab and click the button “Add database user”
3. This will prompt you to enter a username and password - pick something you will remember, as you will need it for the connection string and mongorestore. (Leave the “Make read-only” checkbox unchecked)
4. Click “Create”. This will close the modal and your new user will be listed at the bottom under “Database Users”.
5. Next - make note of the string beneath the text “To connect using the mongo shell:” at the top of the screen… it should look something like:

% mongo ds######.mlab.com:#####/<dbname> -u <dbuser> -p <dbpassword>

where

* ds######.mlab.com:##### is the address of the newly created deployment
* <dbname> is the name of the database (should be something like “teams-api-db”)
* <dbuser> will be the user name that you just created to access the new database
* <dbpassword> will be the password that you created for the above user

1. Once you have recorded all of that information, go back to the integrated terminal in Visual Studio code and Change the working directory (“cd”) in the integrated terminal to “API-data-restore”
2. Once this is done and your working directory is “API-data-restore”, execute the command:

mongorestore -h ds######.mlab.com:##### -d <dbname> -u <dbuser> -p <dbpassword>"

using the information identified above - it should look something like:

mongorestore -h ds123456.mlab.com:23456 -d teams-api-db -u user -p pass

1. You should see a number of lines output to the terminal indicating the progress, and then finally a “done” message

### UPDATING The mongoDBConnectionString:

Now that we have our back-end MongoDB all set up and populated with data, we just have to make that all-important update to the mongoDBConnectionString constant.

1. First, switch back to the previous directory using the integrated terminal (cd ..) so that our working directory has the “server.js” file in it.
2. Next, open the server.js file and using the credentials identified above, modify the mongoDBConnectionString value to use the following format:

mongodb://<dbuser>:<dbpassword>@ds######.mlab.com:#####/<dbname>

using the information identified above - it should look something like:

mongodb://user:password@ds123456.mlab.com:23456/teams-api-db

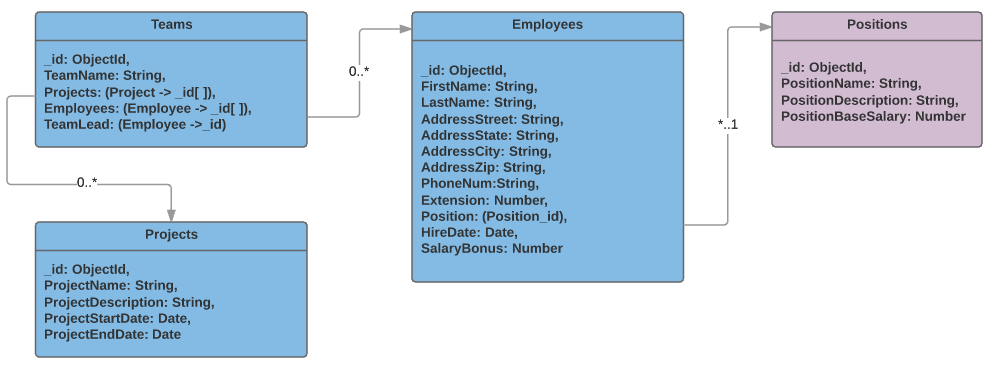
1. Save the changes and run the server from the integrated terminal using the familiar command “node server.js”

## USING THE API

The TeamsAPI project is a simple Web Api consisting of 4 collections:

* Employees
* Positions
* Projects
* Teams

These collections are related in the following way:



Essentially, this will allow us to manage a number of teams by making AJAX requests to the API server running in the background. Each team will work on 0 or more projects and will have 0 or more employees and one single team lead.

You can access Create, Read and Update (no Delete) operations on each of the collections (using JSON) with the following routes:

#### Employees

* GET /employees (returns all employees)
* GET /employee/:employeeId (returns a single employee)
* PUT /employee/:employeeId (updates a single employee)
* POST /employees (adds an employee to the system)

#### Positions

* GET /positions (returns all positions)
* GET /position/:positionId (returns a single position)
* PUT /position/:positionId (updates a single position)
* POST /positions (adds a position to the system)

#### Projects

* GET /projects (returns all projects)
* GET /project/:projectId (returns a single project)
* PUT /project/:projectId (updates a single project)
* POST /projects (adds a project to the system)

#### Teams

* GET /teams (returns all teams)
* GET /team/:teamId (returns a single team)
* PUT /team/:teamId (updates a single team)
* POST /teams (adds a team to the system)

© 2017 - Seneca School of ICT