# MEAN.JS Notes

**Important Links**

[Github repository](https://github.com/brocktubre/MeanMenu)

[Google sheet with requirements](https://docs.google.com/spreadsheets/d/18b5qBCqbLVjCMRJGWkDqSE2Hl3ZbS59YIkxxjxe11LQ/edit?usp=sharing)

[Link to working site](http://brocktubre.com:3000)

**PART 1**

November 14, 2014 — November 15, 2014

Yeoman Generator –

* I’m using [this](https://www.youtube.com/watch?v=XHpMH_5n2fQ&list=PLL7fkBpvAzS740UzH_ClVTDFeqQ76Fs1h&index=5) tutorial for my first look at the yeoman generator. Here is a [link](http://yeoman.io/) to the website. Here is also a [link](http://yeoman.io/generators/) to yo generators even though I’m not sure why or when I would want to use them.

Tools –

* Yo scaffolds out a new application, writing your Grunt configuration and pulling in relevant Grunt tasks and Bower dependencies that you might need for you build.
* The **Build System** is used to build, preview and test your project. [Grunt](http://gruntjs.com) and [Gulp](http://gulpjs.com) are two popular options.
* The **Package Manager** is used for dependency management, so that you no longer have to manually download and manage your scripts. [Bower](http://bower.io) and [npm](npmjs.org) are two popular options.

Getting Started--

* To install use: $ sudo npm install –g generator-meanjs
* To start the generator use: $ yo meanjs
* Next there will be a series of questions asking what you would like the yeoman to generate for you.
* To start the application simply run: $ grunt
* When grunt was ran it didn’t have the mongodb downloaded so I had to download it and install it.

MongoDB-

* Visit this [link](http://mongodb.org) to download and learn more about MongoDB.
* After it was installed, use: $ monod to start the database
* This will allow the user to interact with the db: $ mongo
* I later installed [MongoHub](http://mongohub.todayclose.com/) as a GUI for the local database

Running application-

* You can run the application using the web browser and address: http://localhost:3000

**PART 2**

Next we wanted to add the CRUD features (Create, Read, Update, Delete).

To do this we can use the yo generator to generate some basic views, controllers, and models for the basic CRUD operations.

Use:$ yo meanjs:crud-module <module name>

• menuitems

This will create some files and folders in the /public/module/<module name>/views directory.

• create-<module name>.client.view.html

• edit-<module name>.client.view.html

• list-<module name>.client.view.html

• view-<module name>.client.view.html

And many more, these are just the HTML files. This part is somewhat confusing but I will try to understand it better later.

**PART 3**

Next we want to add an angular controller using the yo generator.

Use:$ yo meanjs:anular-controller <controller name>

• menuitem

It will ask “Which module does this controller belongs to?”

It will store the below file into the public/modules/<module name>/controllers

• <controller name>.client.controller.js

It will store the below files into the public/modules/<module name>/tests

• <controller name>.client.controller.test.js

Next we want to add an angular view using the yo generator.

Use:$ yo meanjs:angular-view <view name>

It will ask “Which module does this view belong to?”

It will ask “What is the name for the controller this view will use?”

It will ask “Would you like to add a route for this view?” (Y)

It will ask “What is your view route path?” <controller name ? view name>

After this is done, it creates the file <view name>.client.view.html in the public/modules/<controller name>/views

• What my understanding is of Part 3. We want to create some controllers and views to do some fun stuff with. We can use to yo genterator to set them up for us instead of doing it all by hand. It creates files in the public/…../controllers and public/…../views

**PART 4**

January 8, 2015

I initially wanted to build everything from scratch and learn each piece and why certain things were generated. I seemed to hit a brick wall and struggled to complete some of the applications basic requirements. This is when I decided to use the yeoman generator to generate a meanjs application. If you look at [Comment 1](#_top), this explains exactly what the yeoman generator is. I started to watch [these](https://www.youtube.com/watch?v=LxqVlYHRooU&index=2&list=PL6rhBJX0L3TWYrwrQIi1_MzQDvVhkUVPI) tutorials and I will continue until further notice. The steps below are actually documented above too. I realize this process is the best way to kick start my application and get the ball rolling.

Before getting started, the “server” (local or remote) running a mean stack application will need a few things installed onto it before the application will be useable. [This](http://www.bossable.com/303/install-mean-stack/) website explains everything pretty great and will make a good reference for later use. All of the installs below are needed for the development environment, but only some are needed for the hosting environment. I will be installing them on the developing environment and also on the hosting environment just to be safe.

1. Download and install node.js. Click and download [here](http://nodejs.org/download/).
2. Download and install mongoDB. Click and download [here](http://www.mongodb.org/downloads).
3. Installing bower: $ npm install –g bower
4. Install grunt: $ npm install –g grunt-cli
5. Install yo: $ npm install –g yo
6. To install node\_modules use: $ npm install

Now this is where the fun begins. We are ready to start generating all of the files and folder to begin development on our application. Using the steps below explains what must be done to get an app ready for development.

1. Using npm install you want to install the meanjs generator globally use this command: $ sudo npm install –g generator-meanjs
2. Next thing will be to create a directory for the project. Since I am on version 3 I will use: $ mkdir MeanMenu\_v3 && cd MeanMenu\_v3
3. Now we are ready to generate an app. We will use the yeoman generate is going to copy over the meanjs templates into our working directory. Use: $ yo meanjs
4. Once the prompted questions have been answered we can simply run the application using: $ grunt

Below is the file structure the generator generated for us. I am still trying to learn each piece but the progress so far has been coming along fine.

-- File Structure

public – this will host all of the angular specific code. All about what happen on the client, on the computer itself. Guts of the app.

config – connections to mongoDB, running any API. This will be the server side configuration. Connections strings.

app – server side related code, split into MVC model. Views will be 404, 500 error views. They live outside and surround/package your angular app.

To add the first CRUD module for the menuitems I used Part 2 steps. This will basically allow a client to create, read, update, and delete a menu item. For example in this application it is necessary for the client to be able to add their menu items to their online food-ordering menu.

* $ yo meanjs:crud-module menuitems

Model

When a crud module is added it will store the schema information in the app/models/<module name>.server.model.js

*Menuitem*

Category – the category of the menu item (e.g. Sandwiches, Burgers, …) [String]

Name –the name of the menu item (e.g. Turkey melt, Buster burger, …) [String]

Description – a description of the menu item (e.g. Best burger in town! … ) [String]

Price – the price of the menu item (e.g. $5.00, $6.75) [Number]

Created – the date the menu item was created (Date) [Date]

User – the user who added the menu item [ObjectId]

View

The views for the crud model are added to the directory public/modules/<module name>/views. The files below will be created and added into the views folder to make a working crud module.

• create-<module name>.client.view.html

• edit-<module name>.client.view.html

• list-<module name>.client.view.html

• view-<module name>.client.view.html

Controllers

The controllers for the crud model added to the directory public/modules/<module name>/controllers. The file below will be created and added to the controllers folder.

* <module name>.client.controller.js

The next CRUD module that I added was the events model. This will basically allow a client to create, read, update, and delete an upcoming events item. For example in this application it is necessary for the client to be able to add the new and upcoming events for their restaurant like bands and other shows happening.

* $ yo meanjs:crud-module upcoming-events

*Events*

Name –the name of the menu item [String]

Date – the date the event will be held on [Date]

startTime – the time the event will start [String]

endTime – the time the event will end [String]

Details – details about the event(e.g. Come listen to … ) [String]

Cover – the price of the menu item (e.g. $5.00, $6.75, blank is no cover) [Number]

Created – the date the menu item was created (Date) [Date]

User – the user who added the menu item [ObjectId]

**PART 5**

February 8, 2015

Adding new packages and scripts into the application. To add a new script we can do this in the:

• config/env/all.js

To add new dependencies we must first add it to the public/config.js and then inject that module into the controller.

# Other Services

[These](https://www.youtube.com/watch?v=yB-kRo28xzM):

gulp.js

broccoli

fez

Backbone

Ember

Handlebar

[These](https://www.youtube.com/watch?v=FqMIyTH9wSg):

Jade templates

[Hogan](http://twitter.github.io/hogan.js/) (basically mustache)

[Less](http://lesscss.org/)

Stylus

Sass