Our very first MEAN application

In the previous period you created a simple REST-API for a joke-application using a simple JavaScript array as the data store. In this exercise we are going to change this into a real MEAN application.

The API is going to be expanded to support all four CRUD operations

`GET`	`api/joke/random`	Fetch an random joke
`GET`	`api/jokes`	Get a List of all Jokes
`POST`	`api/joke`	Create a new Joke
`PUT`	`api/joke`	Edit an existing Joke
`DELETE`	`api/joke/:_id`	Delete an existing Joke

The representation of a single joke should be implemented as outlined in the example below:

```
{
  "_id" : "a MongoDB generated objected"
  "joke" : "Reality is an illusion created by a lack of alcohol",
  "type" : ["short", "alcohol", "quote"]
  "reference": { "author" : "Name of author", "link" : "url…"}
  "lastEdited" : "date for last edit"
}
```

Today we will use the basic *mongodb* driver to get a feeling about the true nature of mongoDB. Later we will repeat the exercise using Mongoose (something similar to a ORM for a document db).

Getting started

This exercise assumes that you have installed MongoDB locally, or alternatively have created a free account and a database on http://mlab.com (will make it easier to deploy your Express app to Digital Ocean).
To get some initial data to play around with, you can download the file: jokeSetup.js and execute it like this:

- **Local database**: `mongo jokeSetup.js`
- **http://mlab**: mongo xxxxxxxxx.mlab.com:PORT/YOUR_DATABASE -u <dbuser> -p <dbpassword>

If you would like to have a GUI-client, try robomongo.org (but, there are many others).

1)

Create a new Express application as the starting point for this exercise
In the terminal install the `mongodb-driver:` `npm install mongodb --save`

2)
Create a folder called *model*, and add a JavaScript-file `jokes.js` to the folder
This will be our façade to the database. Implement the following methods, one by one and test.

```
exports.allJokes =      function(callback){..};
exports.findJoke =      function(id,callback){..};
exports.addJoke =       function(jokeToAdd,callback) { .. };
exports.editJoke =      function(jokeToEdit,callback) { .. };
exports.deleteJoke =    function(id,callback){ .. };
exports.randomJoke =    function randomJoke(callback){..};
```

All callbacks must be on the usual node-form.

```
        function(err,data){..}
```

data is the joke or jokes in question:

I suggest you create a folder db with a file db.js to act as a singleton container for the connection as explained here: http://js-plaul.rhcloud.com/mongoDB/mongo.html#24

3)
Implement the REST-API described in the start, using the factory implemented in part 2

4)
Test the REST-API, inspired by last weeks exercises (for this version it's OK to use the development database)

5)
Upload the project to Digital Ocean and verify that the API is accessible (if you have setup a database on mlab.com, this should be straight forward since you don't have to change the connection parameters.

6) Create a simple Angular application that should use most of the API implemented above