

# Topic 9: MEAN Stack

CITS3403 Agile Web Development

---

Getting MEAN with Mongo,  
Express, Angular and Node,  
Chapter 1.

---

Semester 1, 2018

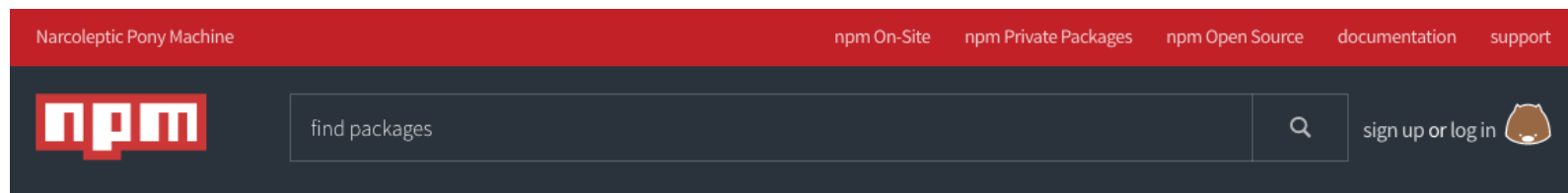
# Full Stack Development?

- Full stack development refers to developing all parts of a web application: database, web server, application logic and front end.
- There are various “Full stacks” people use to develop:
  - LAMP (Linux, Apache, MySQL and PHP)
  - Ruby on Rails
  - Django (Python)
- We’re going to use the MEAN stack in this unit:
  - **M**ongo: a document database
  - **E**xpress: the web framework
  - **A**ngularJS: the front end framework
  - **N**ode: the web server.
- Homework: read “What is Code”, by Paul Ford:  
<http://www.bloomberg.com/graphics/2015-paul-ford-what-is-code/>



# Why MEAN?

- At the core of MEAN is Node.js. It uses the Google V8 JavaScript engine and allows the easy creation of web servers.
- Node has an excellent package manager, npm, maintained by a strong community. There are packages for email, authentication, web frameworks, database connections, testing, image processing, ....



mongoose public  
mongoose MongoDB ODM

Mongoose is a **MongoDB** object modeling tool designed to work in an asynchronous environment.



#### Private npm Registry

Host your own private, on-premises npm registry with npm On-Site. [Learn more...](#)

# Why MEAN?

- Mongo, Express, Angular and Node are written in JavaScript, so we can write the whole application in one language.
- MEAN is lightweight, self contained and does not require a lot of configuration.
- It is very configurable, not opinionated, and you can see the moving parts (it not *auto-magical*).



# What's in the stack



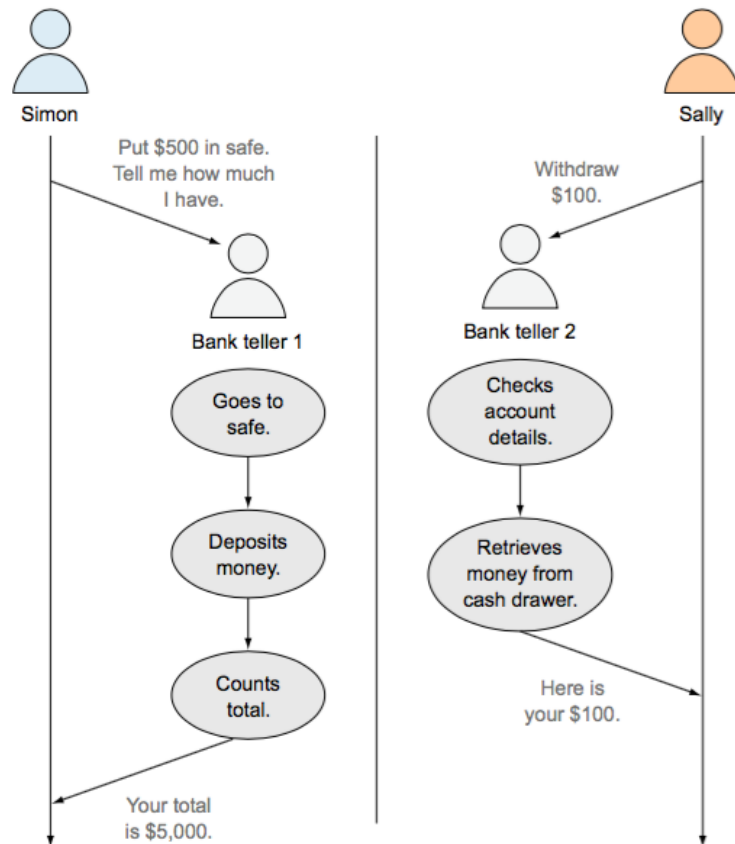
- NodeJS: A server side runtime environment. It supports an event driven architecture and asynchronous I/O.
- Express: A NodeJS server framework. Does routing, session support, templating,...
- Mongo: A document database, fast, scalable, and everything's JSON (or BSON)
- Angular: Front end framework for 2-way data binding.

# Node

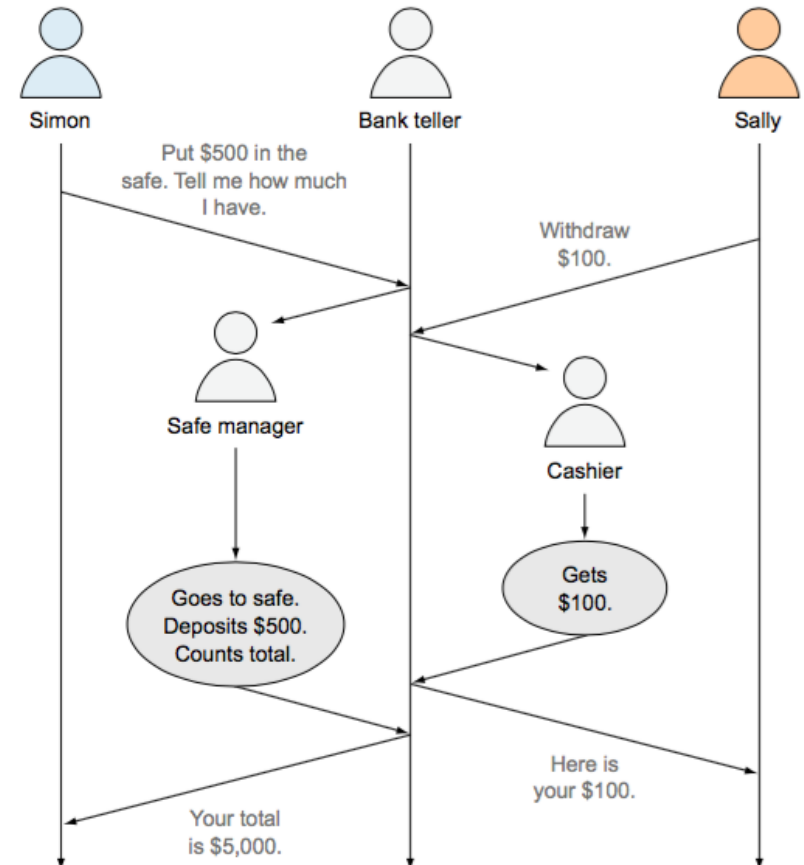
- Node is a runtime environment. It is self contained, so you can create and configure a webserver for each project.
- It is single threaded, rather than multi-threaded, so it uses less resources.
- NPM is the *Node Package Manager*. There are thousands of packages that are available to incorporate into a Node project.

# Single Threaded Servers

## Multi-threaded



## Single-threaded



From: Getting MEAN with Mongo, Express Angular and Node, by Simon Holmes



# Non-blocking Code in Node

- To make Node run efficiently you have to write *non-blocking* code. This typically involves callbacks:

```
var fs = require("fs");  
  
var data = fs.readFileSync('input.txt');  
  
console.log(data.toString());  
console.log("Program Ended");
```

The program waits for each line to complete before calling the next.

```
var fs = require("fs");  
  
fs.readFile('input.txt', function (err, data) {  
    if (err) return console.error(err);  
    console.log(data.toString());  
});  
  
console.log("Program Ended");
```

An anonymous function is passed to the readfile as a parameter. The function executes when the read is done, but the program continues in the meantime.



# Express

- Express is a web framework that makes it easy to build a simple webserver for your application.
- Express-Generator will build a generic directory structure for your project:

```
[bash-3.2$ express todo -e  
  
create : todo  
create : todo/package.json  
create : todo/app.js  
create : todo/public  
create : todo/public/javascripts  
create : todo/public/images  
create : todo/public/stylesheets  
create : todo/public/stylesheets/style.css  
create : todo/routes  
create : todo/routes/index.js  
create : todo/routes/users.js  
create : todo/views  
create : todo/views/index.ejs  
create : todo/views/error.ejs  
create : todo/bin  
create : todo/bin/www
```

# Express

- Express provides a simple template for expressing *routes* and uses a templating engine (jade) for populating *views*
- Session management is built in and authentication is done using modules
- Express supports, but does not enforce the Model-View-Controller architecture.

```
36 // Routes
37 app.use( routes.current_user );
38 app.get( '/', routes.index );
39 app.post( '/create', routes.create );
40 app.get( '/destroy/:id', routes.destroy );
41 app.get( '/edit/:id', routes.edit );
42 app.post( '/update/:id', routes.update );
43
44 app.use( static( path.join( __dirname, 'public' ) ) );
45
46 |
47 http.createServer( app ).listen( app.get( 'port' ), function () {
48     console.log( 'Express server listening on port ' + app.get( 'port' ) );
49 });
```

# Mongo

- Almost all web apps require a server side data store, and in the MEAN stack we use Mongo.db, a NoSQL database
- Mongo is really just an indexed collection of javascript objects. There are no tables or enforced schemas.
- You can interact directly with the database via the Mongo shell, to query, sort, insert etc.

```
> db.todos.find().pretty()
{
  "_id" : ObjectId("570dcfe3d0c1cdcb6633310f"),
  "user_id" : "mARbKt9yfuNvwcfHlTgAT8r62MuEmRgf",
  "content" : "Write Lecture 11",
  "updated_at" : ISODate("2016-04-13T04:50:20.281Z"),
  "__v" : 0
}
{
  "_id" : ObjectId("570dd016d0c1cdcb66333110"),
  "user_id" : "mARbKt9yfuNvwcfHlTgAT8r62MuEmRgf",
  "content" : "Sort out labs",
  "updated_at" : ISODate("2016-04-13T04:50:30.704Z"),
  "__v" : 0
}
{
  "_id" : ObjectId("570dd1448c82abf9666f36ec"),
  "user_id" : "mARbKt9yfuNvwcfHlTgAT8r62MuEmRgf",
  "content" : "mark projects",
  "updated_at" : ISODate("2016-04-13T04:55:32.671Z"),
  "__v" : 0
}
```

# Mongoose

- Mongoose is an ORM (object relational modelling) tool that is available in Node.
- It makes it simple to define model schemas and synchronise with the Mongo database.

- ...but the mongo syntax is not the nicest in the world.

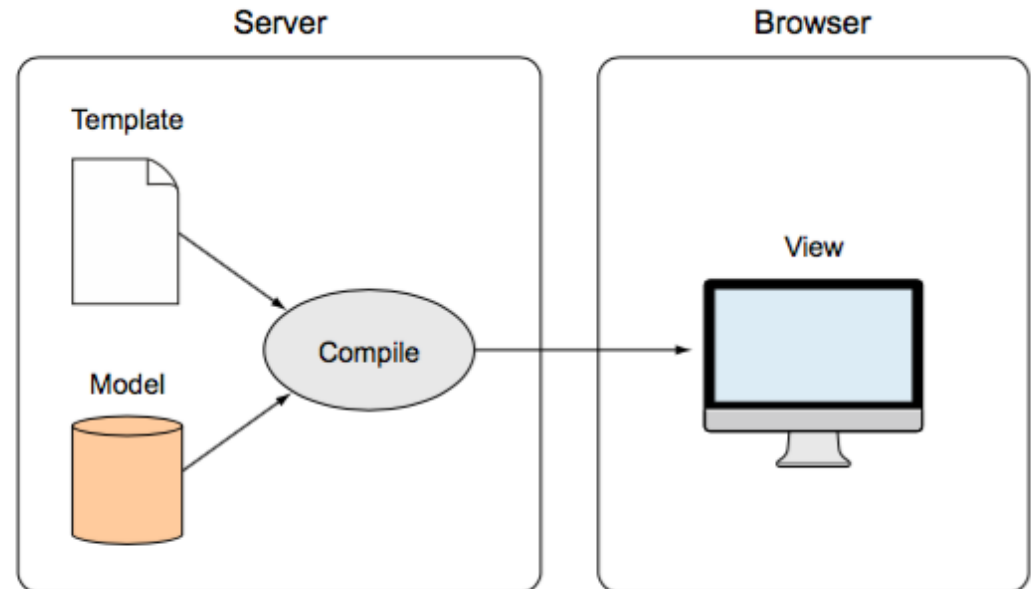
```
db.inventory.find(  
  {  
    $or: [ { qty: { $gt: 100 } }, { price: { $lt: 9.95 } } ]  
  }  
)
```

```
var mongoose = require( 'mongoose' );  
var Schema   = mongoose.Schema;  
  
var Todo = new Schema({  
  user_id    : String,  
  content    : String,  
  updated_at : Date  
});  
  
mongoose.model( 'Todo', Todo );  
mongoose.connect( 'mongodb://localhost/express-todo' );
```

# Angular

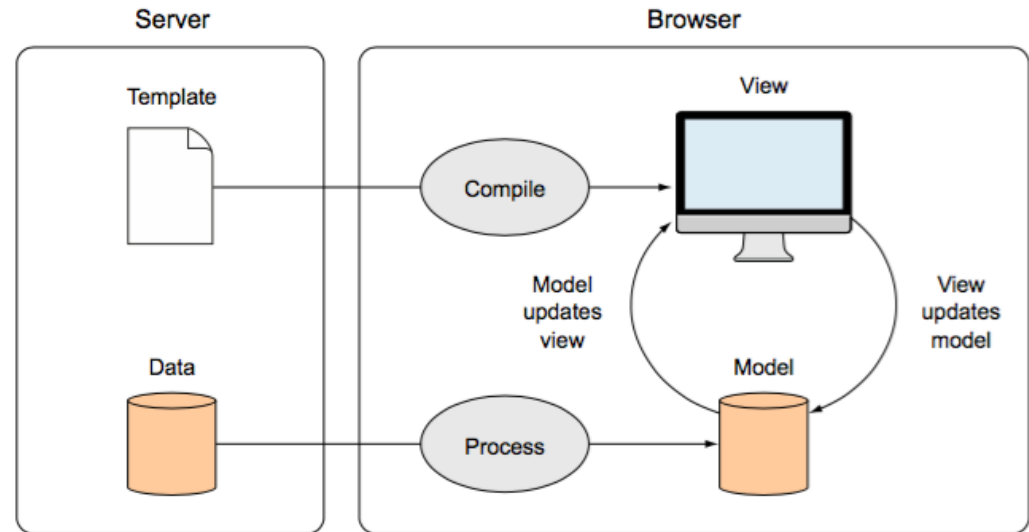
- Angular is a web framework from Google that makes 2 way data binding easy.
- It uses declarative programming, and supports Single page applications
- We won't cover this in detail in this unit...

- One way data binding accepts requests from the client, builds views and returns them.



# Model View Controller with Angular

- Two way data binding links the view with a client side model.



- This means more work can be done on the client side and pages are more responsive.

```
<body>
<div ng-app="">
  <p>Name : <input type="text" ng-model="name"></p>
  <h1>Hello {{name}}</h1>
</div>
</body>
```

# Supporting Cast

- Editors: Atom, Brackets, Eclipse, Sublime (\$), WebStorm (\$), VSCode, **vim**
- **GIT** is a version control system that supports collaboration.
- Web Hosts: **Heroku**, **mLab**, **Azure**, **AWS**
- Test Frameworks (**Mocha**)
- Scaffolders (Yeoman)
- Task runners (grunt)
- Design frameworks (Bootstrap, **Jade/Pug**)



# Sample App.


- A very simple app can be built along the lines of:


<http://dreamerslab.com/blog/en/write-a-todo-list-with-express-and-mongodb/>


(note it is a bit out of date, so the GITHub code is required to work with the latest versions).

## Express Todo Example

mark projects 

Sort out labs 

Write Lecture 11 

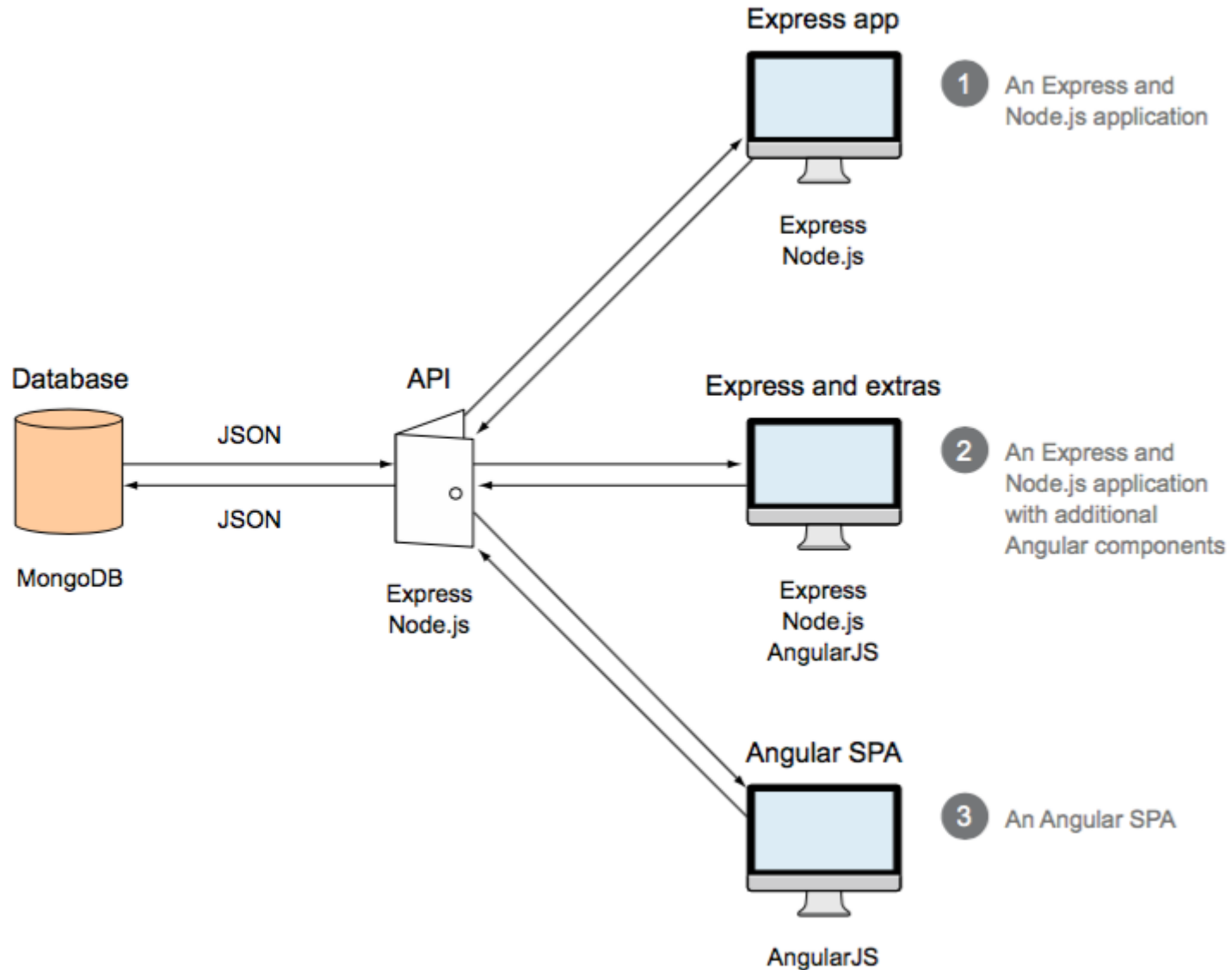
# MEAN Project Architecture



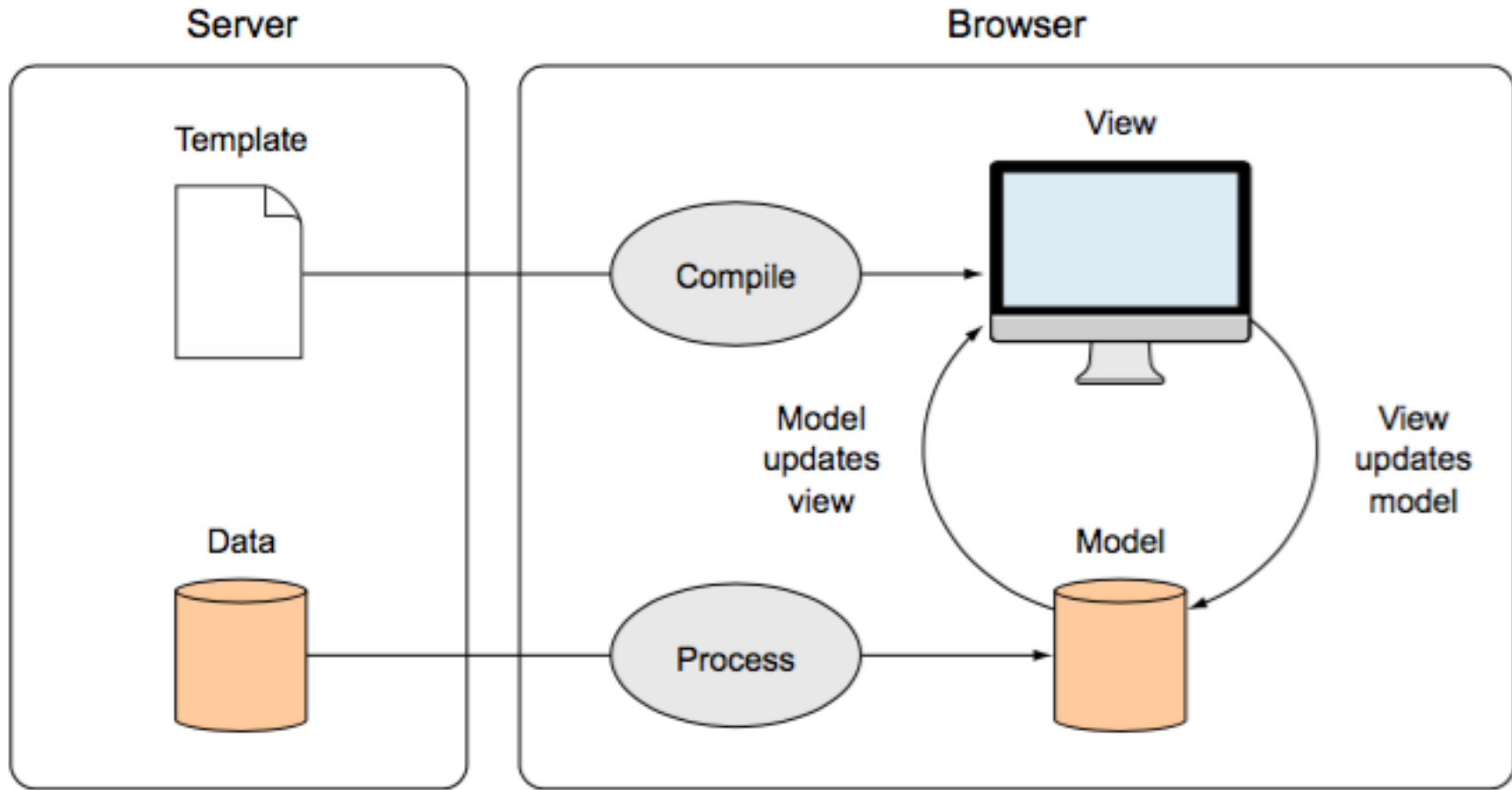
A fullstack MEAN project consists of

- A Node runtime environment
- Express running a webserver handling requests from the client end
- Angular presenting data on the client side
- Mongo DB serving the data to Node
- Still a lot of gaps: MEAN is not opinionated...

# A Restful API Architecture



# A Single Page Application Architecture



# Setting Everything Up

Setting everything up on your local machine is relatively easy:

- Install Node - <https://nodejs.org/en/>
- Install Mongo - <https://www.mongodb.org/>
- NPM Express - <https://www.npmjs.com/package/express>
- Link Angular – It's just a JS file.

This gives you a development environment, but you wouldn't want your local machine to be an application server.

Solution.... GIT (next lecture)

# MEAN working environments

- Localhost: Most development machines aren't servers, but content can be viewed at local host.
- Cloud servers: AWS, Google Cloud, Azure all offer cloud hosting of virtual machines. These virtual machines can be preconfigured with a software stack.
- Containers: Heroku and mLab are light weight containers that sync with Git and run your code.

# Continuous Integration

- Continuous integration is the idea of having constant patches and releases, rather than staged versions.
- Software like Drone.io and Travis use Git and automated testing to keep deployed software up to date.
- Git can also be used for ensuring that files on a server aren't tampered with.



# Agile Web Development

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Agile web development needs responsive, collaborative tools to adapt and manage software production.

# Suggested Reading

Read “What is Code” by Paul Ford:

<http://www.bloomberg.com/graphics/2015-paul-ford-what-is-code/>



There are bugs in your code! Click the line of code that looks like it's bug-free. But be careful: Any time you don't fix a bug, a new one is born.



```
var salesPlusFour = 4 + sales;
```

```
var salesPlusFour = "4" + sales;
```



```
for (var i = 0; i < 10; i++)
```

```
for (var i = 0; i < 10 i++)
```