heig-vd

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD

www.heig-vd.ch

06 - SPA Frameworks

Build “rich” UIs with AngularJS & Co.

TWEB 2017

Olivier Liechti

https://softeng-heigvd.github.io/Teaching-HEIGVD-TWEB-2017-Main/

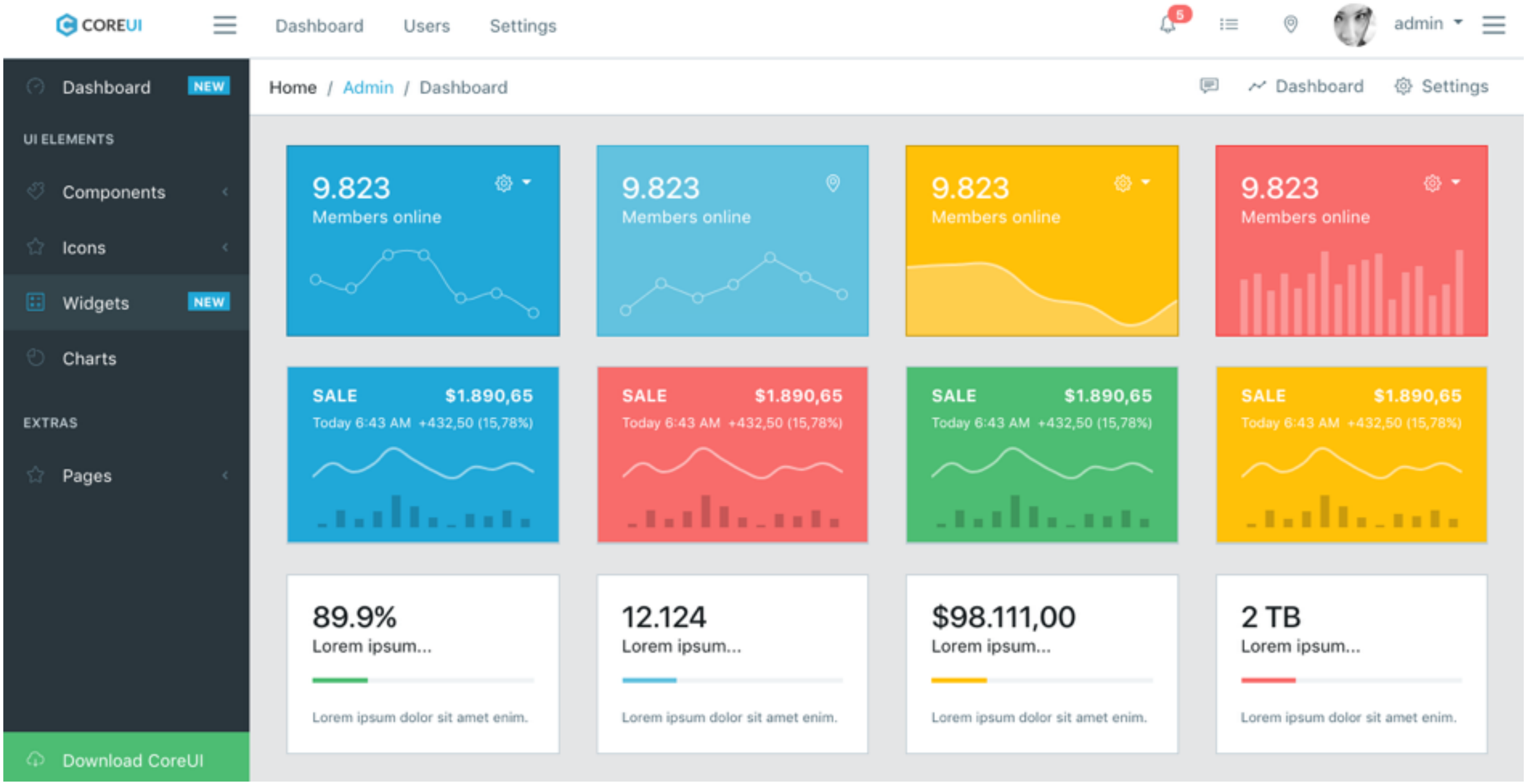
https://t.me/joinchat/CPPWmAsLLgWdXQhoXTaNHw
https://t.me/joinchat/AAAAAEE3IWzr-jZRRMq3qg

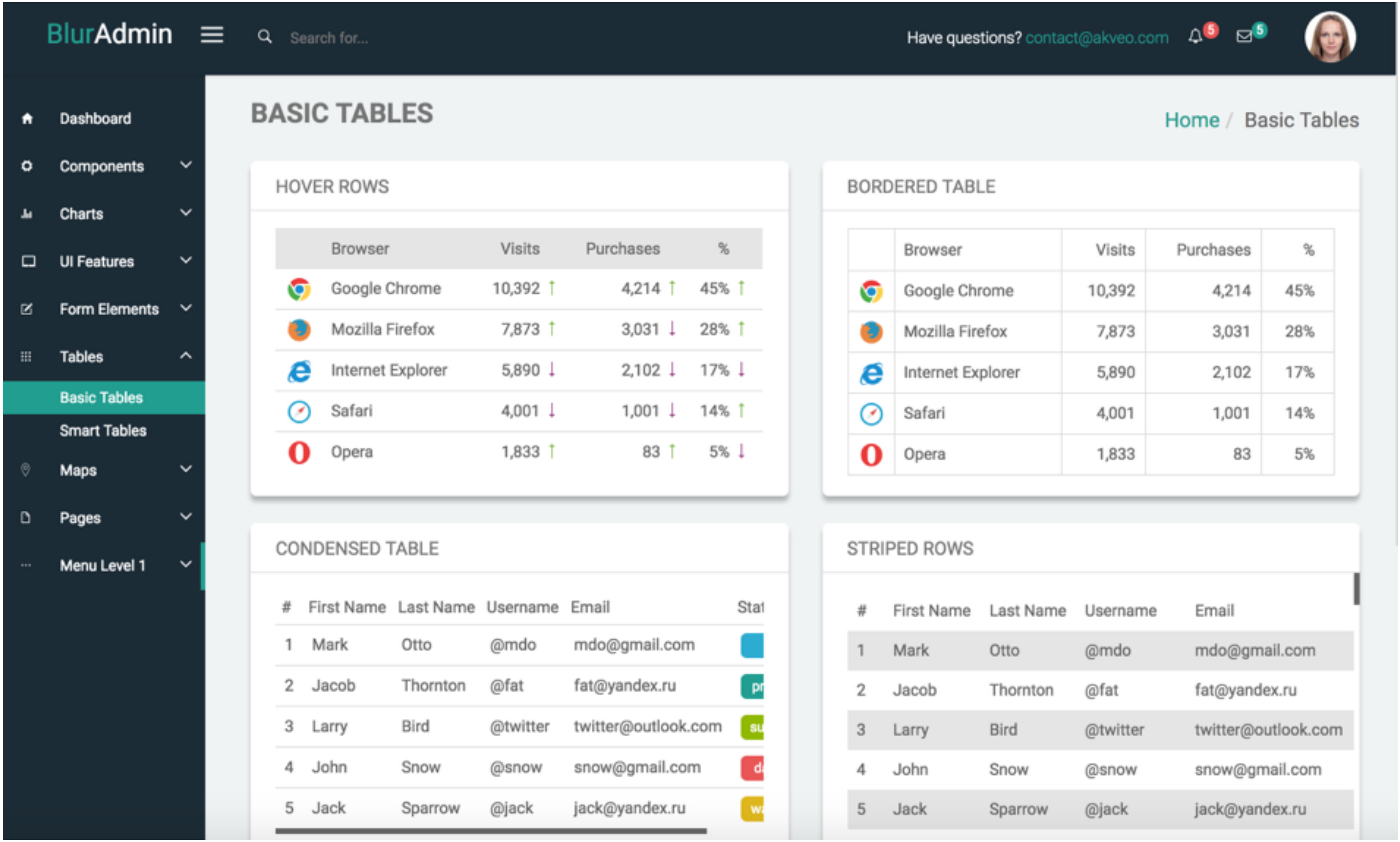
What’s next?

Now that we are finishing our stories, what are we going to learn until the end of the semester?

Frameworks

- You certainly have heard about **Angular** and **React**. Maybe of Meteor, Ember, Vue or Aurelia.
- Why do we need to these **frameworks**, if we already have JQuery?
- **Which one** should we learn and use?


COREUI
Dashboard
Users
Settings
5
admin
Dashboard NEW
UI ELEMENTS
Components
Icons
Widgets NEW
Charts
EXTRAS
Pages
Download CoreUI
Home / Admin / Dashboard
Dashboard
Settings
9.823
Members online
9.823
Members online
9.823
Members online
9.823
Members online
SALE $1.890,65
Today 6:43 AM +432,50 (15,78%)
SALE $1.890,65
Today 6:43 AM +432,50 (15,78%)
SALE $1.890,65
Today 6:43 AM +432,50 (15,78%)
SALE $1.890,65
Today 6:43 AM +432,50 (15,78%)
89.9%
Lorem ipsum...
Lorem ipsum dolor sit amet enim.
12.124
Lorem ipsum...
Lorem ipsum dolor sit amet enim.
$98.111,00
Lorem ipsum...
Lorem ipsum dolor sit amet enim.
2 TB
Lorem ipsum...
Lorem ipsum dolor sit amet enim.



BASIC TABLES

HOVER ROWS

	Browser	Visits	Purchases	%
	Google Chrome	10,392 ↑	4,214 ↑	45% ↑
	Mozilla Firefox	7,873 ↑	3,031 ↓	28% ↑
	Internet Explorer	5,890 ↓	2,102 ↓	17% ↓
	Safari	4,001 ↓	1,001 ↓	14% ↑
	Opera	1,833 ↑	83 ↑	5% ↓

BORDERED TABLE

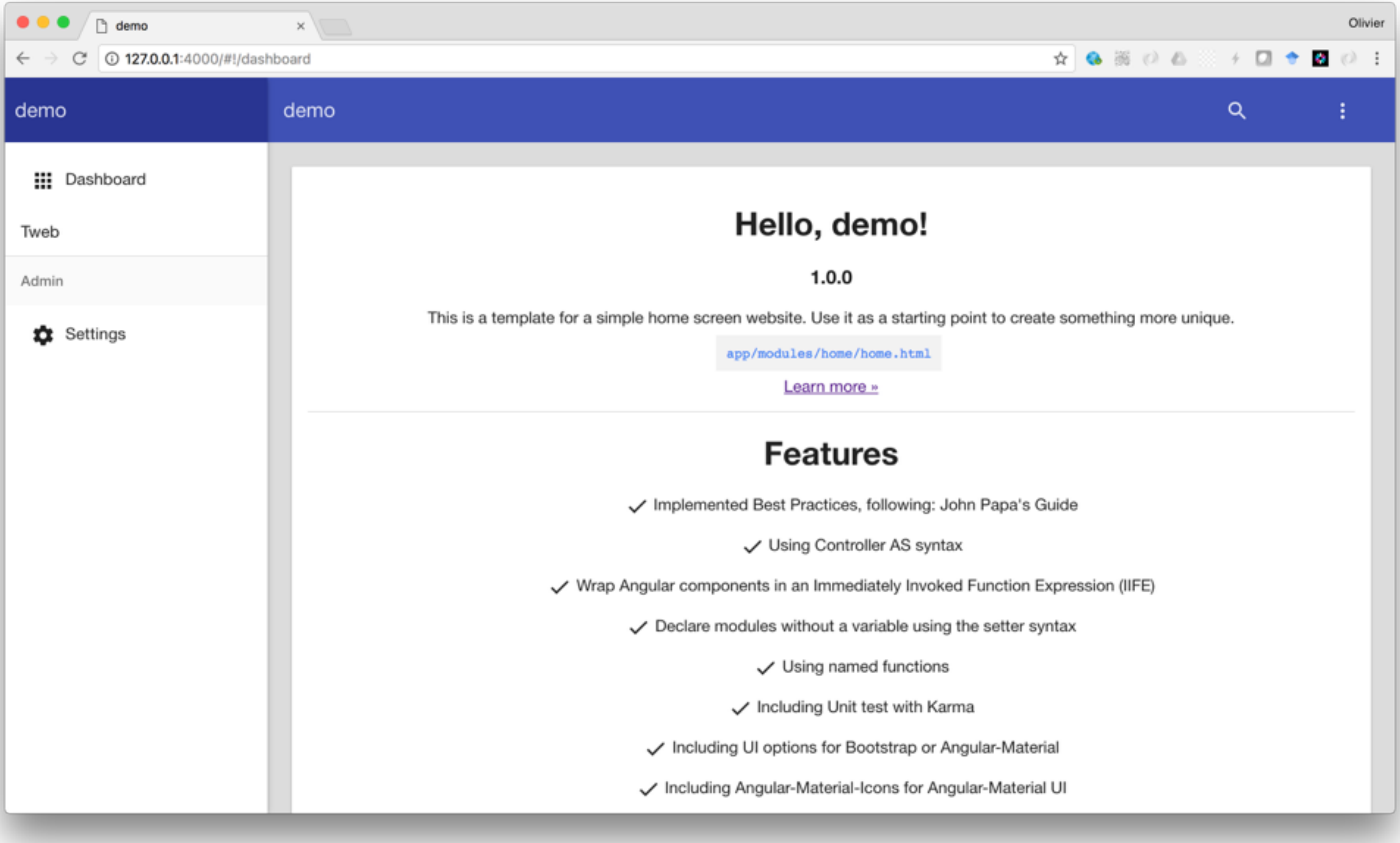
	Browser	Visits	Purchases	%
	Google Chrome	10,392	4,214	45%
	Mozilla Firefox	7,873	3,031	28%
	Internet Explorer	5,890	2,102	17%
	Safari	4,001	1,001	14%
	Opera	1,833	83	5%

CONDENSED TABLE

#	First Name	Last Name	Username	Email	Stat
1	Mark	Otto	@mdo	mdo@gmail.com	
2	Jacob	Thornton	@fat	fat@yandex.ru	pr
3	Larry	Bird	@twitter	twitter@outlook.com	su
4	John	Snow	@snow	snow@gmail.com	d
5	Jack	Sparrow	@jack	jack@yandex.ru	w

STRIPED ROWS

#	First Name	Last Name	Username	Email
1	Mark	Otto	@mdo	mdo@gmail.com
2	Jacob	Thornton	@fat	fat@yandex.ru
3	Larry	Bird	@twitter	twitter@outlook.com
4	John	Snow	@snow	snow@gmail.com
5	Jack	Sparrow	@jack	jack@yandex.ru



Hello, demo!

1.0.0

This is a template for a simple home screen website. Use it as a starting point to create something more unique.

`app/modules/home/home.html`

Learn more »

Features

✓ Implemented Best Practices, following: John Papa's Guide

✓ Using Controller AS syntax

✓ Wrap Angular components in an Immediately Invoked Function Expression (IIFE)

✓ Declare modules without a variable using the setter syntax

✓ Using named functions

✓ Including Unit test with Karma

✓ Including UI options for Bootstrap or Angular-Material

✓ Including Angular-Material-Icons for Angular-Material UI

Concepts

- **Asynchronous** programming
  - We have already seen **callbacks** and **promises**.
  - There are other abstractions: **reactive extensions (rxjs)**, **async/await** and others.

Concepts

- **State management**
  - The **Flux** pattern and **Redux**
  - How they work in combination with web frameworks such as **React**

The Big Lie

- The **Death of REST** APIs (and beloved DTOs)
- Next-generation Web APIs with **GraphQL**

Project

- **Initial plan**: design a Gamification service
- **Alternative plan**: build an admin console (e.g. for your AMT micro-services).
- **Your choice** (but we will come back to that next week).

This week?

- Today, we do a **first introduction** to web frameworks and AngularJS.
- You are **not expected to write code yet**, as you are still finishing the interactive stories.
- In the coming weeks, we will **revisit** theses concepts and then you will write code.

Problem

What is a SPA Framework and why should I care?

What's the difference with a library such as JQuery?

Forces

- We have used various **libraries** to implement Web based user interfaces.

- **JQuery** allows us to fetch data (AJAX), to select nodes in the DOM and to update them with the data. We call JQuery functions.

- For **large** applications (think ERP) and **complex** UIs (think Facebook), our code could become messy and repetitive. We need structure.

Definitions

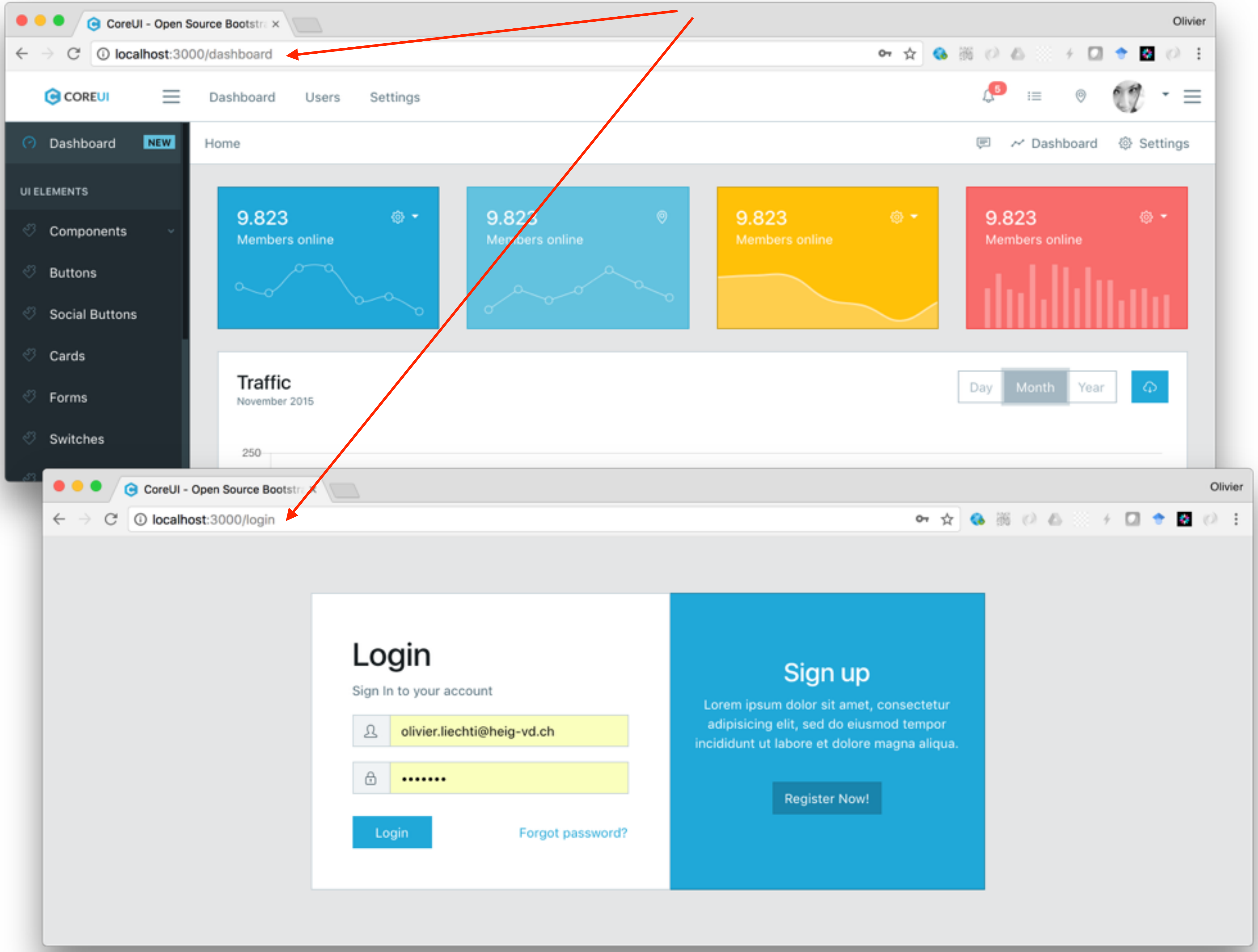
- The difference between a **library** and a **framework**:
  - our code calls the library (e.g. function calls)
  - the framework calls our code (e.g. callbacks)
- A framework is a set of components that implements a generic **behaviour**, which can be extended.

Definitions

- We know the **server-side MVC pattern** (e.g. implemented with servlets + JSPs).

- In this architecture, it is possible to **serve multiple views** and for these views to make AJAX calls to the server (usually for simple stuff).

- This is different from the **Single Page App (SPA) pattern**, where a single “skeleton” HTML page is loaded at the beginning of the session. Page navigation then happens entirely on the client. Pages can look very different from one another (login page, dashboard, etc.)



localhost:3000/dashboard

localhost:3000/login

Login

Sign In to your account

olivier.liechti@heig-vd.ch

•••••••

Sign up

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Problem

How do I pick “the best” framework?

There are so many choices…

Forces

- **AngularJS** was not the first front-end framework, but it gained **massive adoption** a couple of years ago. You will encounter "legacy" apps for a long time.
- **Angular 2 / 4** are new versions that are not directly compatible. Is it worth rewriting an AngularJS app to the new version? Or is it better to switch to something completely different?
- **React** has gained traction from quite some time now. It is not only developed, but also used by Facebook.
- There are other contenders: vue, ember, meteor, aurelia, ...

Forces

- Your **productivity** depends on 2 factors:
  - the intrinsic quality of the tools you use
  - your experience and knowledge of these tools
- I claim that the 2 factors are equally important.
- Therefore, do not change tools all the time; only change the tool when it is ~5-10 times “better”.

Trends

- https://stateofjs.com/2016/frontend/
- https://hackernoon.com/5-best-javascript-frameworks-in-2017-7a63b3870282
- https://medium.com/unicorn-supplies/angular-vs-react-vs-vue-a-2017-comparison-c5c52d620176
- http://sotagtrends.com/?tags=%5Bangularjs,angular,reactjs,vue.js,meteor,aurelia,ember.js%5D&relative=true

My view

- I am now relatively productive with AngularJS. I also believe that the new patterns allow us to write simple and clean code.

- I don't see a compelling reason to move to Angular 4:

  - Typescript was much better than "old-style" Javascript, but with ES6 and a linter...

  - Every time I check Angular 4, I find the experience heavier than with AngularJS (dependencies, build time, etc.)

  - AngularJS has a rich(er?) ecosystem: plugins, templates, examples, etc.

- Therefore, if I change to a new framework, it will probably be for React. But I also keep an eye on Vue.js. Ask me in 6 months...

Problem

How do I get started with AngularJS?

Can I find inspiration and examples?

Caveats (1)

- AngularJS has changed a lot over the years (not even talking about Angular2/4).
- The **coding patterns** are very different from what they used to be (e.g. the "controllerAs" pattern).
- The code that you will find in "**legacy**" applications, examples and StackOverflow questions is different from latest best practices.
- Therefore, **be aware of the dates** when you check for resources.

Caveats (2)

- The **official documentation** is a reference, but I don't think it is the best to get started. It does not really explain you how to structure your code.

- There are often different ways to achieve the same result. The documentation does not tell you which one is the best.

- One of the most useful resources is this style guide: **johnpapa/angular-styleguide**.

Solutions

- For the **code structure and style**, I recommend the **angm** generator available on yeoman (yo angm).
- It is not perfect, but it implements the **John Papa's Angular Style Guide**. This is a must read.
- For the look&feel, there are free admin consoles including **mrholek/CoreUI-AngularJS** and **akveo/blur-admin**.

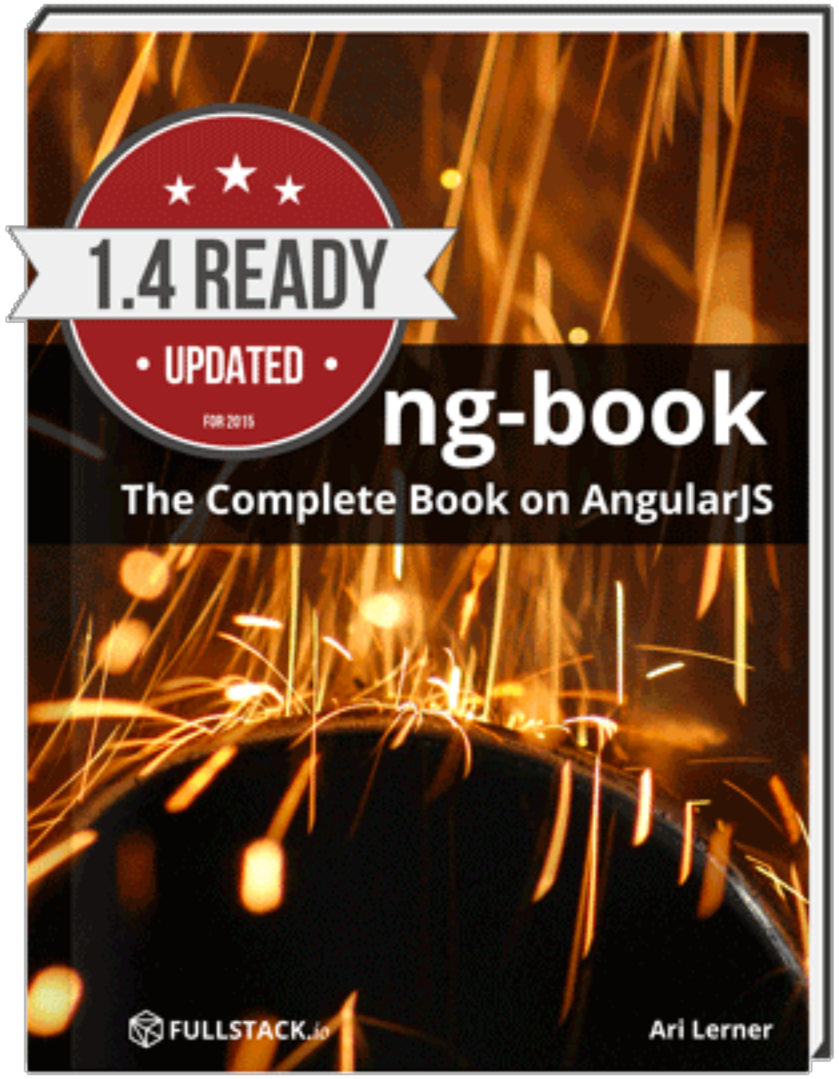
ANGULARJS
by Google

What is AngularJS?

- **Client-side** JavaScript framework
- Designed to create **Single-Page Applications** (SPAs), as opposed to "server-side MVC applications".
- Initially released in **2009**. Has become one of the most popular frameworks (see job offers!).
- **Large community** (many third-party modules, learning resources, etc.). Open source project with major contributions from **Google**.
- Current version (v1): 1.6.7

How to learn AngularJS?

- There are **many books**. I have found this one to be particularly helpful:
  - https://www.ng-book.com/
- There are many sites with **tutorials and webcasts**. Here is a good example:
  - https://egghead.io/technologies/angularjs
- One of the most efficient ways is to study and play with **existing code**:
  - Browse through GitHub repos.
  - Use a yeoman generator
- There are often different ways to do one thing. It is important to adopt coding conventions. See:
  - **https://github.com/johnpapa/angular-styleguide**



https://www.ng-book.com/

Core concepts

- **Bootstrapping**
  - What do I need to do so that AngularJS starts doing its magic with my app?
- **Modules**
  - How do I split my app into multiple libraries and how do I use libraries provided by other developers?
- **Directives**
  - How can I write views/templates for AngularJS? What are those extra HTML elements (elements and attributes)?
- **Controllers**
  - How do I define the data that is then rendered in the views/templates?
- **Services**
  - Where do I put the business logic and how do I
- ***Scopes***
  - *For the time being, we will ignore this concept that used to be a cornerstone of the AngularJS architecture. In "modern" patterns, it is hidden by an alternative syntax (aka controller as).*

How do I bootstrap AngularJS?

- To get started with AngularJS, you first need to **load the core framework** script. You can either use a **CDN**, download the file yourself, or use **bower**.
- You write **your code** in several scripts, which must also be loaded from index.html. In this example, all the code is in one script.

```
<html ng-app="tweb-demo-app">
   <body>
      <h1>Bootstrapping AngularJS</h1>

      <!-- load core AngularJS before any other AngularJS module -->
      <script src="js/angular.js"></script>

      <!-- load my AngularJS module -->
      <script src="js/tweb-demo-app.js"></script>
   </body>
<html>
```

Your script defines a **module** named "tweb-demo-app"

What is a Module?

- When you develop an AngularJS application, you create **controllers**, **services**, **directives**, etc.
- At the minimum, you need to put your components in an application "**module**", which is loaded during the application **bootstrap**.
- If you have a large application, or if you want to share/reuse some of your components, it is a good idea to create **several modules**.
- You can think of modules as "**containers of components**".
- Modules can have **dependencies** on other modules.

This creates a new module, named 'tweb.users'. AngularJS will add it to its registry. The empty brackets mean that the module has no dependency on other modules.

```
angular.module('tweb.users', []);
```

This looks up the module named 'tweb.users' in the AngularJS registry.

```
angular.module('tweb.users');
```

We **declare a new module** and give it a **name** (‘twebApp’). Later, we will be able to lookup this module with `angular.module(‘twebApp’)`, in other words by calling the module function without the second argument.

HAUTE ÉCOLE
D’INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

```
angular.module('twebApp', [
  'ngCookies',
  'ngResource',
  'ngSanitize',
  'btford.socket-io',
  'ui.router',
  'ui.bootstrap'
])
```

This will **lookup** the twebApp module.

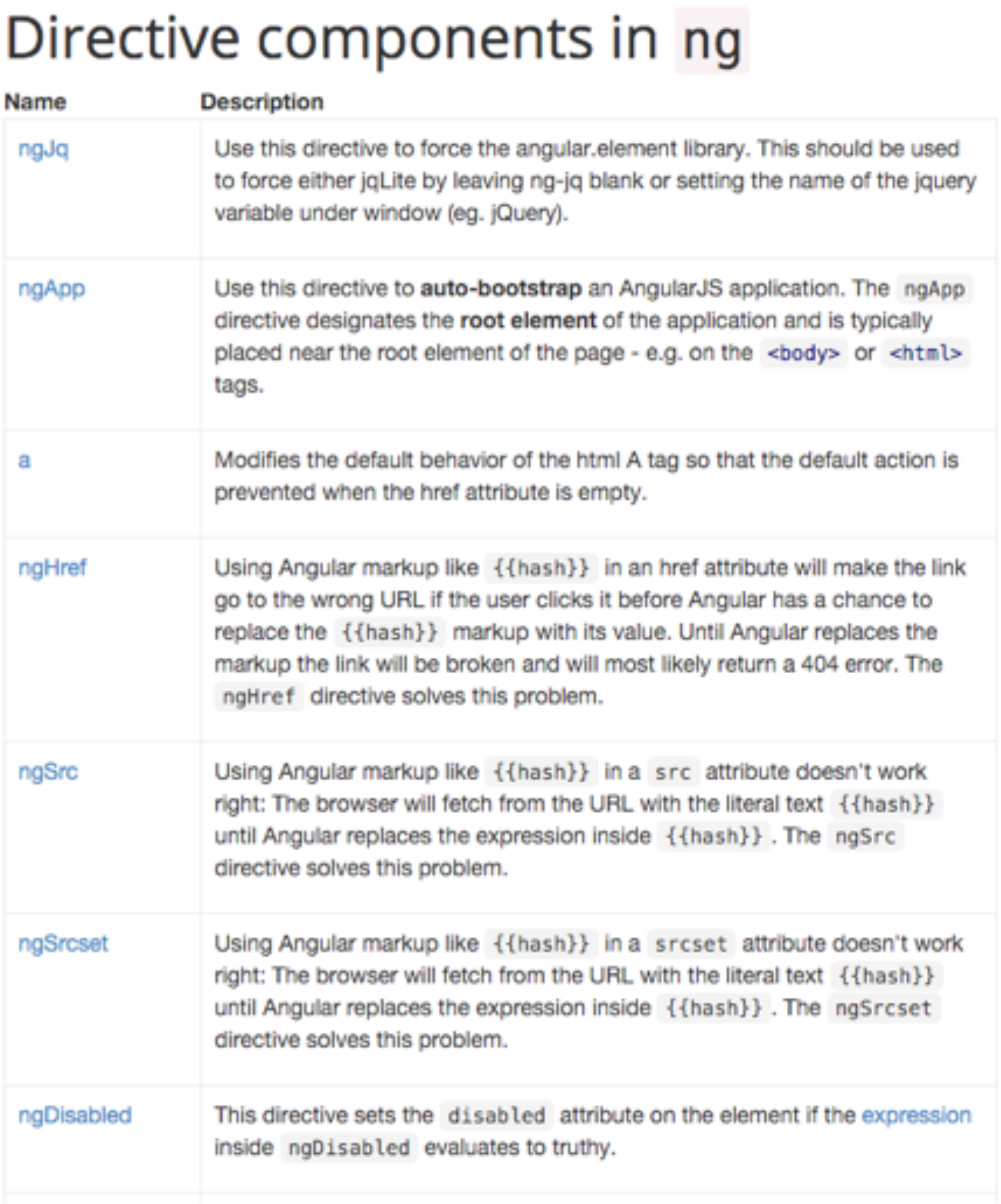
```
<body ng-app="twebApp">

  <!-- build:js({client,node_modules}) app/vendor.js -->
    <!-- bower:js -->
    <script src="bower_components/jquery/dist/jquery.js"></script>
    <script src="bower_components/angular/angular.js"></script>
    <script src="bower_components/angular-resource/angular-resource.js"></script>
    <script src="bower_components/angular-cookies/angular-cookies.js"></script>
    <script src="bower_components/angular-sanitize/angular-sanitize.js"></script>
    <script src="bower_components/angular-bootstrap/ui-bootstrap-tpls.js"></script>
    <script src="bower_components/lodash/dist/lodash.compat.js"></script>
    <script src="bower_components/angular-socket-io/socket.js"></script>
    <script src="bower_components/angular-ui-router/release/angular-ui-router.js"></script>
    <!-- endbower -->
    <script src="socket.io-client/socket.io.js"></script>
  <!-- endbuild -->
```

We **declare** that our module depends on 6 other modules (in this case, they are AngularJS and third-party modules). The corresponding *.js files must be **loaded in index.html**.

What is a Directive?

- An AngularJS directive is an **HTML extension** (e.g. a custom element, a custom attribute, which you include in your markup to **trigger some behavior**.
- AngularJS comes with a collection of **built-in directives**.
- **Third-party developers** have created additional directives.
- **You** can write your own directives (but we will not do that immediately)



Directive components in `ng`

Name	Description
ngJq	Use this directive to force the angular.element library. This should be used to force either jqLite by leaving ng-jq blank or setting the name of the jquery variable under window (eg. jQuery).
ngApp	Use this directive to **auto-bootstrap** an AngularJS application. The `ngApp` directive designates the **root element** of the application and is typically placed near the root element of the page - e.g. on the `<body>` or `<html>` tags.
a	Modifies the default behavior of the html A tag so that the default action is prevented when the href attribute is empty.
ngHref	Using Angular markup like `{{hash}}` in an href attribute will make the link go to the wrong URL if the user clicks it before Angular has a chance to replace the `{{hash}}` markup with its value. Until Angular replaces the markup the link will be broken and will most likely return a 404 error. The `ngHref` directive solves this problem.
ngSrc	Using Angular markup like `{{hash}}` in a `src` attribute doesn't work right: The browser will fetch from the URL with the literal text `{{hash}}` until Angular replaces the expression inside `{{hash}}`. The `ngSrc` directive solves this problem.
ngSrcset	Using Angular markup like `{{hash}}` in a `srcset` attribute doesn't work right: The browser will fetch from the URL with the literal text `{{hash}}` until Angular replaces the expression inside `{{hash}}`. The `ngSrcset` directive solves this problem.
ngDisabled	This directive sets the `disabled` attribute on the element if the expression inside `ngDisabled` evaluates to truthy.

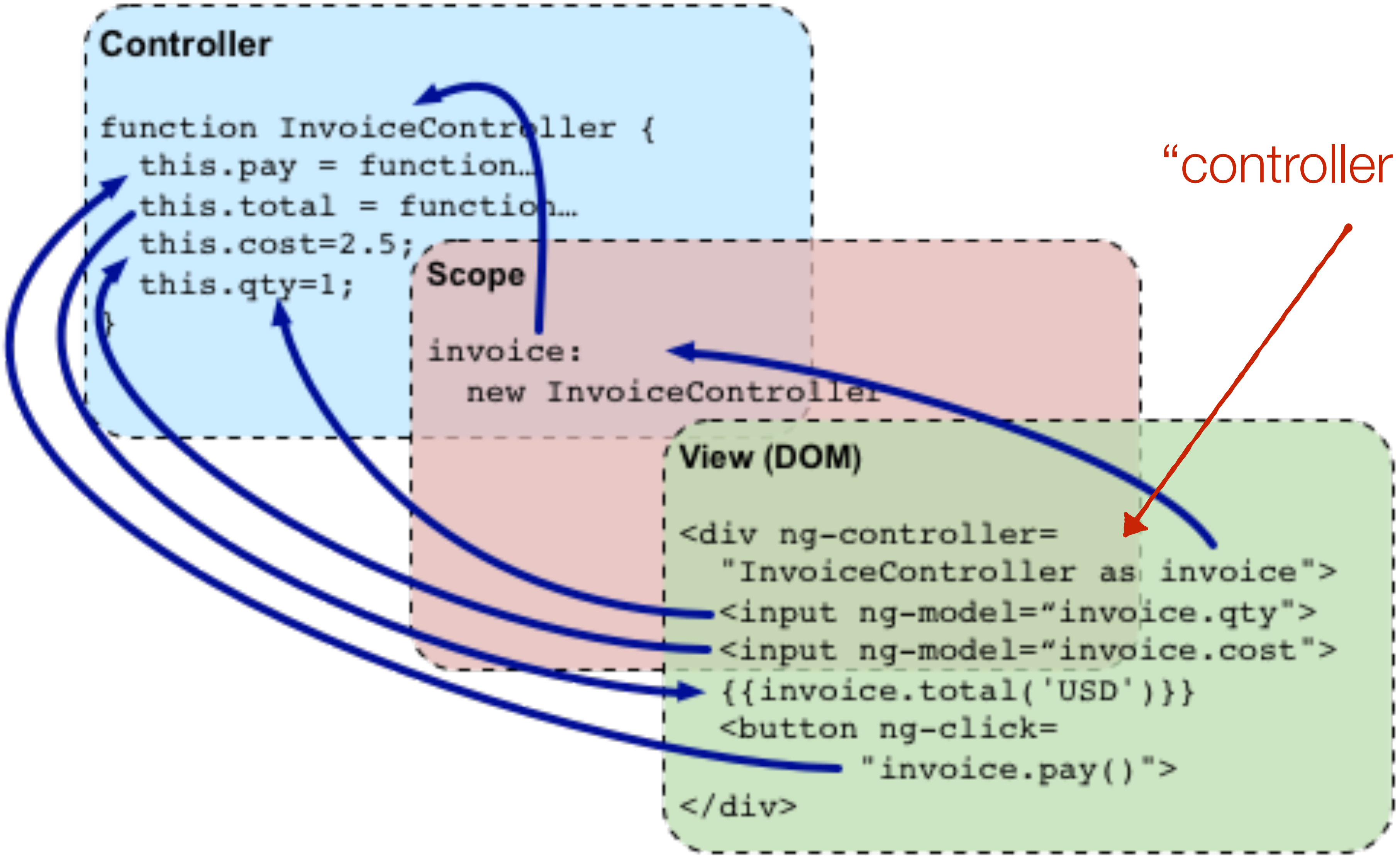
https://docs.angularjs.org/api/ng/directive

Which directives do we use often?

ngApp	Use this directive to **auto-bootstrap** an AngularJS application. The ngApp directive designates the root element of the application and is typically placed near the root element of the page - e.g. on the **<body>** or **<html>** tags.
ngController	The ngController directive **attaches a controller class to the view**. This is a key aspect of how angular supports the principles behind the **Model-View-Controller** design pattern.
ngModel	The ngModel directive **binds an input,select, textarea** (or custom form control) to a **property on the scope** using NgModelController, which is created and exposed by this directive.
ngRepeat	The ngRepeat directive **instantiates a template once per item from a collection**. Each template instance **gets its own scope**, where the given loop variable is set to the current collection item, and $index is set to the item index or key.
ngClick	The ngClick directive allows you to specify **custom behavior when an element is clicked**.
ngInclude	Fetches, compiles and includes an **external HTML fragment**.
ngClass	The ngClass directive allows you to **dynamically set CSS classes** on an HTML element by databinding an expression that represents all classes to be added.

What is a Controller?

- An AngularJS controller is used to **provide data** and **behavior** (functions) to them views/templates.



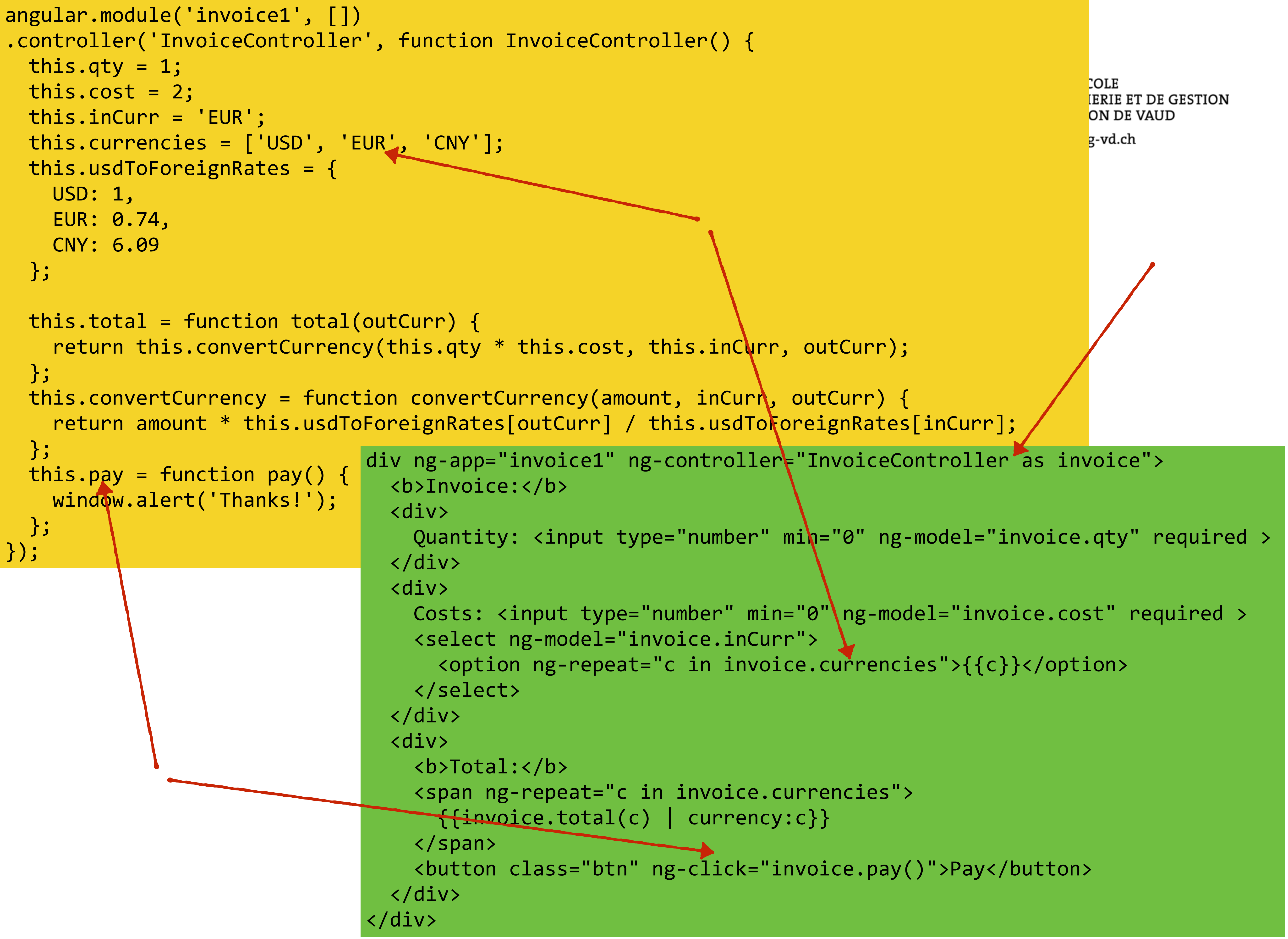
```
function InvoiceController {
  this.pay = function…
  this.total = function…
  this.cost=2.5;
  this.qty=1;
}
```

```
invoice:
  new InvoiceController
```

```
<div ng-controller=
  "InvoiceController as invoice">
  <input ng-model=“invoice.qty">
  <input ng-model=“invoice.cost">
  {{invoice.total('USD')}}
  <button ng-click=
       "invoice.pay()">
</div>
```

“controller as” syntax

https://docs.angularjs.org/guide/concepts#controller



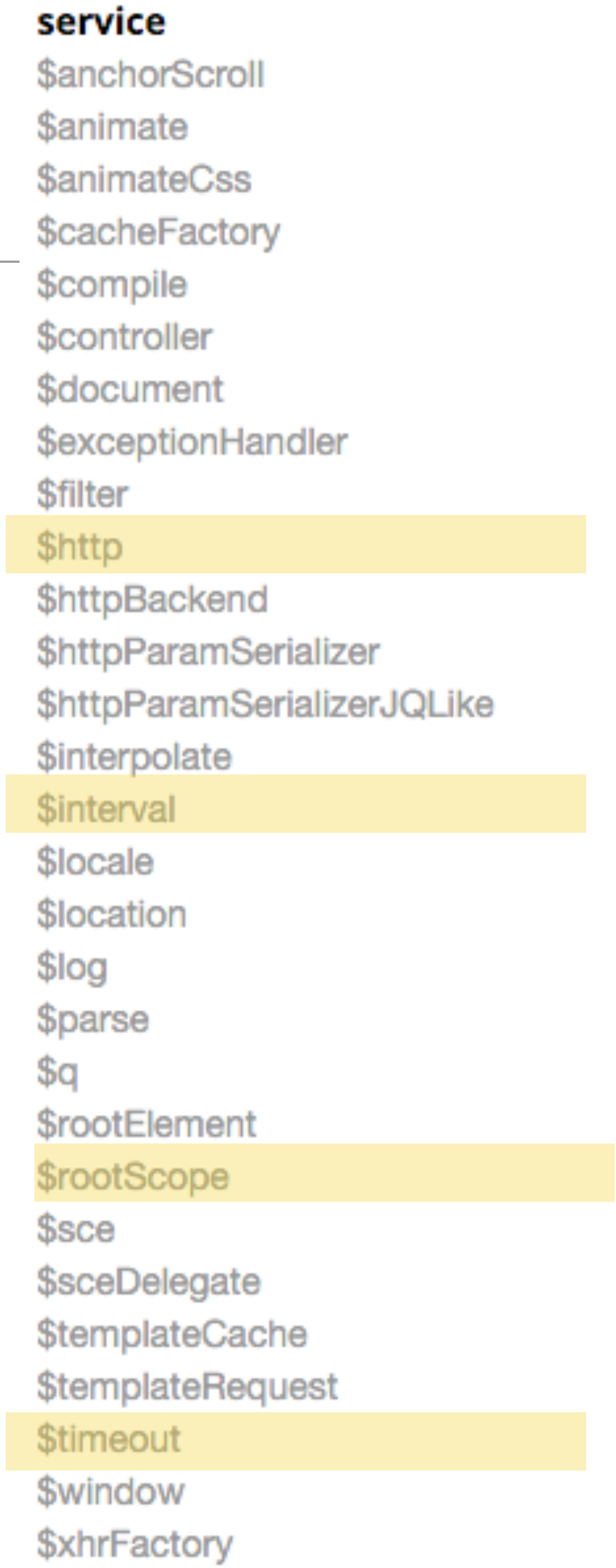
```
angular.module('invoice1', [])
.controller('InvoiceController', function InvoiceController() {
  this.qty = 1;
  this.cost = 2;
  this.inCurr = 'EUR';
  this.currencies = ['USD', 'EUR', 'CNY'];
  this.usdToForeignRates = {
    USD: 1,
    EUR: 0.74,
    CNY: 6.09
  };

  this.total = function total(outCurr) {
    return this.convertCurrency(this.qty * this.cost, this.inCurr, outCurr);
  };
  this.convertCurrency = function convertCurrency(amount, inCurr, outCurr) {
    return amount * this.usdToForeignRates[outCurr] / this.usdToForeignRates[inCurr];
  };
  this.pay = function pay() {
    window.alert('Thanks!');
  };
});
```

```
div ng-app="invoice1" ng-controller="InvoiceController as invoice">
  <b>Invoice:</b>
  <div>
    Quantity: <input type="number" min="0" ng-model="invoice.qty" required >
  </div>
  <div>
    Costs: <input type="number" min="0" ng-model="invoice.cost" required >
    <select ng-model="invoice.inCurr">
      <option ng-repeat="c in invoice.currencies">{{c}}</option>
    </select>
  </div>
  <div>
    <b>Total:</b>
    <span ng-repeat="c in invoice.currencies">
      {{invoice.total(c) | currency:c}}
    </span>
    <button class="btn" ng-click="invoice.pay()">Pay</button>
  </div>
</div>
```

What is a Service?

- AngularJS services are **singleton objects** that can be injected in controllers and that provide some functionality.
- It is a good practice to keep **controllers small**. For this reason, most of the complex behavior should be delegated to a service.
- A good example is the code that deals with **AJAX** requests.
- AngularJS provides a list of **built-in services**.
- You can implement **your own services**.



service
$anchorScroll
$animate
$animateCss
$cacheFactory
$compile
$controller
$document
$exceptionHandler
$filter
$http
$httpBackend
$httpParamSerializer
$httpParamSerializerJQLike
$interpolate
$interval
$locale
$location
$log
$parse
$q
$rootElement
$rootScope
$sce
$sceDelegate
$templateCache
$templateRequest
$timeout
$window
$xhrFactory

angular-ui / ui-router

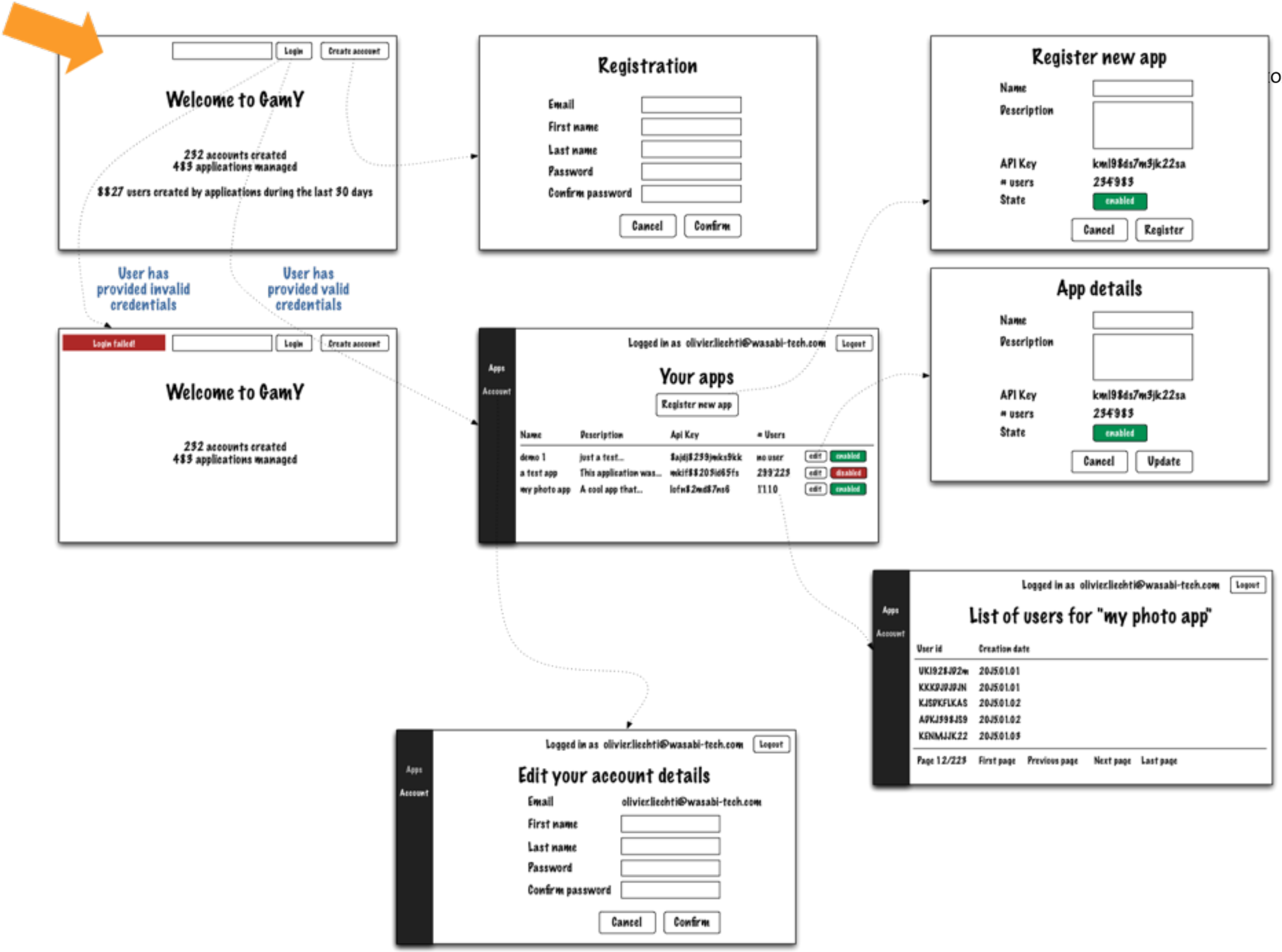
ui-router is a very popular alternative to the ngRoute service provided by AngularJS

AngularUI Router is a **routing framework** for AngularJS, which allows you to organize the parts of your interface into a state machine. Unlike the $route service in the Angular ngRoute module, which is organized around URL routes, **UI-Router is organized around states, which may optionally have routes, as well as other behavior, attached**.

States are bound to named, nested and parallel views, allowing you to powerfully manage your application's interface.

https://github.com/angular-ui/ui-router

https://github.com/angular-ui/ui-router/wiki/Quick-Reference



Login | Create account

Welcome to GamY

232 accounts created
483 applications managed

3327 users created by applications during the last 30 days

User has provided invalid credentials

User has provided valid credentials

Login failed! | Login | Create account

Welcome to GamY

232 accounts created
483 applications managed

Registration

Email	
First name	
Last name	
Password	
Confirm password	

Cancel | Confirm

Register new app

Name	
Description	
API Key	kml93ds7m3jk22sa
# users	234983
State	enabled

Cancel | Register

App details

Name	
Description	
API Key	kml93ds7m3jk22sa
# users	234983
State	enabled

Cancel | Update

Apps
Account

Logged in as olivier.liechti@wasabi-tech.com | Logout

Your apps

Register new app

Name	Description	Api Key	# Users		
demo 1	just a test...	3ajdj3239jmks9kk	no user	edit	enabled
a test app	This application was...	mkif33203id65fs	293'223	edit	disabled
my photo app	A cool app that...	lofw32md37ms6	1'110	edit	enabled

Apps
Account

Logged in as olivier.liechti@wasabi-tech.com | Logout

List of users for "my photo app"

User id	Creation date
UKI923J92m	2015.01.01
KKKPJPJPJN	2015.01.01
KJSPKFLKAS	2015.01.02
APKJ393JS9	2015.01.02
KENMJJK22	2015.01.03

Page 12/223 | First page | Previous page | Next page | Last page

Apps
Account

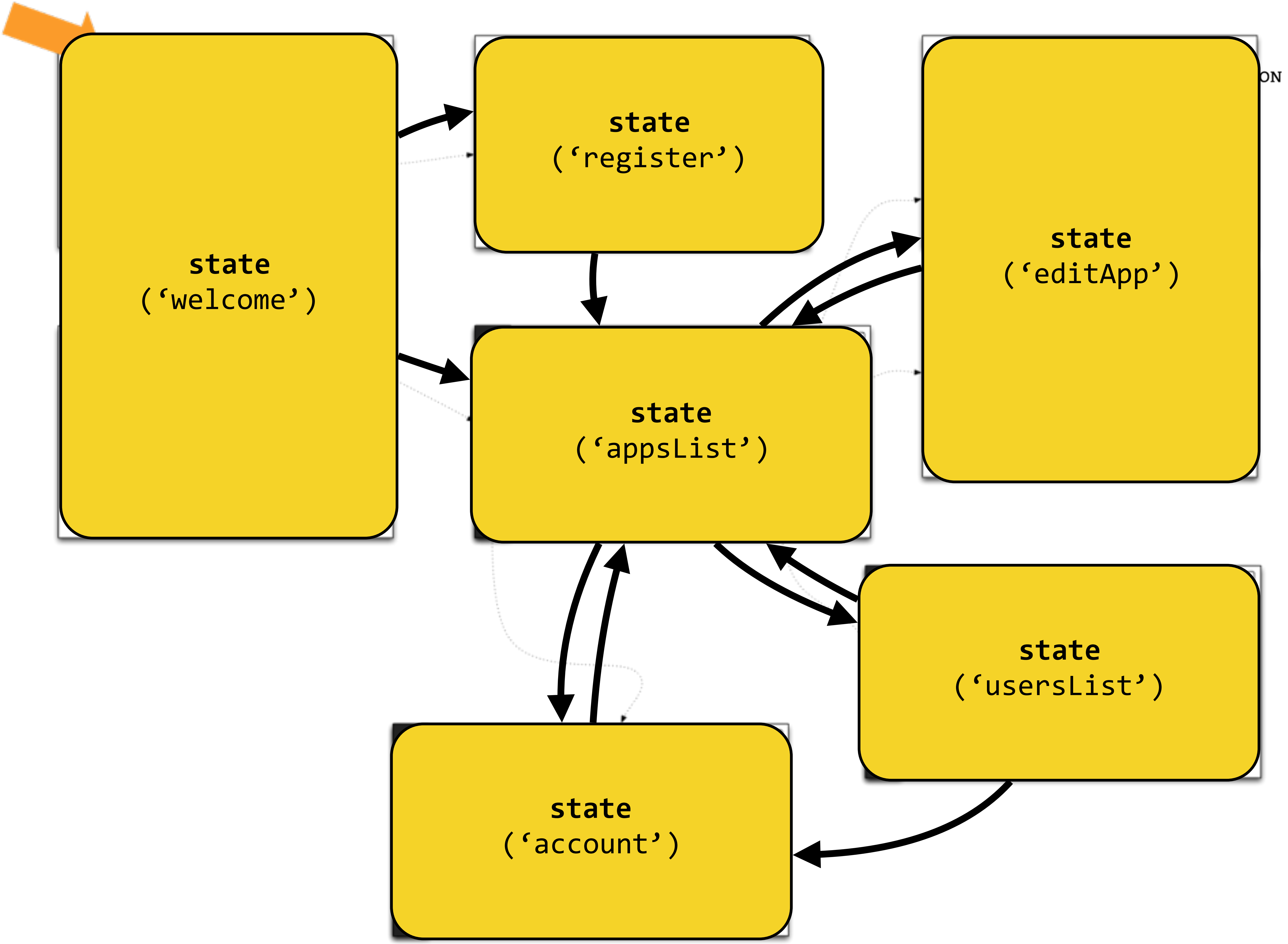
Logged in as olivier.liechti@wasabi-tech.com | Logout

Edit your account details

Email	olivier.liechti@wasabi-tech.com
First name	
Last name	
Password	
Confirm password	

Cancel | Confirm

ON



state
('welcome')

state
('register')

state
('editApp')

ON

state
('appsList')

state
('usersList')

state
('account')

Basic example

```
angular.module("myApp", ["ui.router"])

.config(function( $stateProvider ) {

    $stateProvider.state('welcome', {
      templateUrl: 'partials/welcome.html',
      url: '/welcome'
    });

    $stateProvider.state('about', {
      templateUrl: 'partials/about.html',
      url: '/about'
    });

});
```

This function is executed when the myApp module is **loaded**. We can configure the $stateProvider service (provided by ui.router).

A state has a **name** (about) and a **config object** with quite a few properties. Here, we only define the **page fragment** that will be injected in the **ui-view** element and the url that will be displayed in the **navigation bar** when the state is active.

```
<body ng-controller="MainCtrl">
   <a ui-sref="welcome">Home</a> | <a ui-sref="about">About</a>
  <section ui-view></section>
</body>
```

Learning how to use ui-router

- **Information is provided by the authors of the module**:
    - In a **short tutorial**: http://angular-ui.github.io/ui-router/
    - On the **GitHub wiki**: https://github.com/angular-ui/ui-router/wiki
    - In the **API reference**: http://angular-ui.github.io/ui-router/site/#/api/ui.router
    - In a **sample application**: http://angular-ui.github.io/ui-router/sample/#/ (source: https://github.com/angular-ui/ui-router/tree/master/sample)
- The **angular-fullstack generator** uses ui-router (well, it gives you the choice when you generate your skeleton).