heig-vd

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

www.heig-vd.ch

07 - AngularJS live coding

Hands-on intro to AngularJS concepts

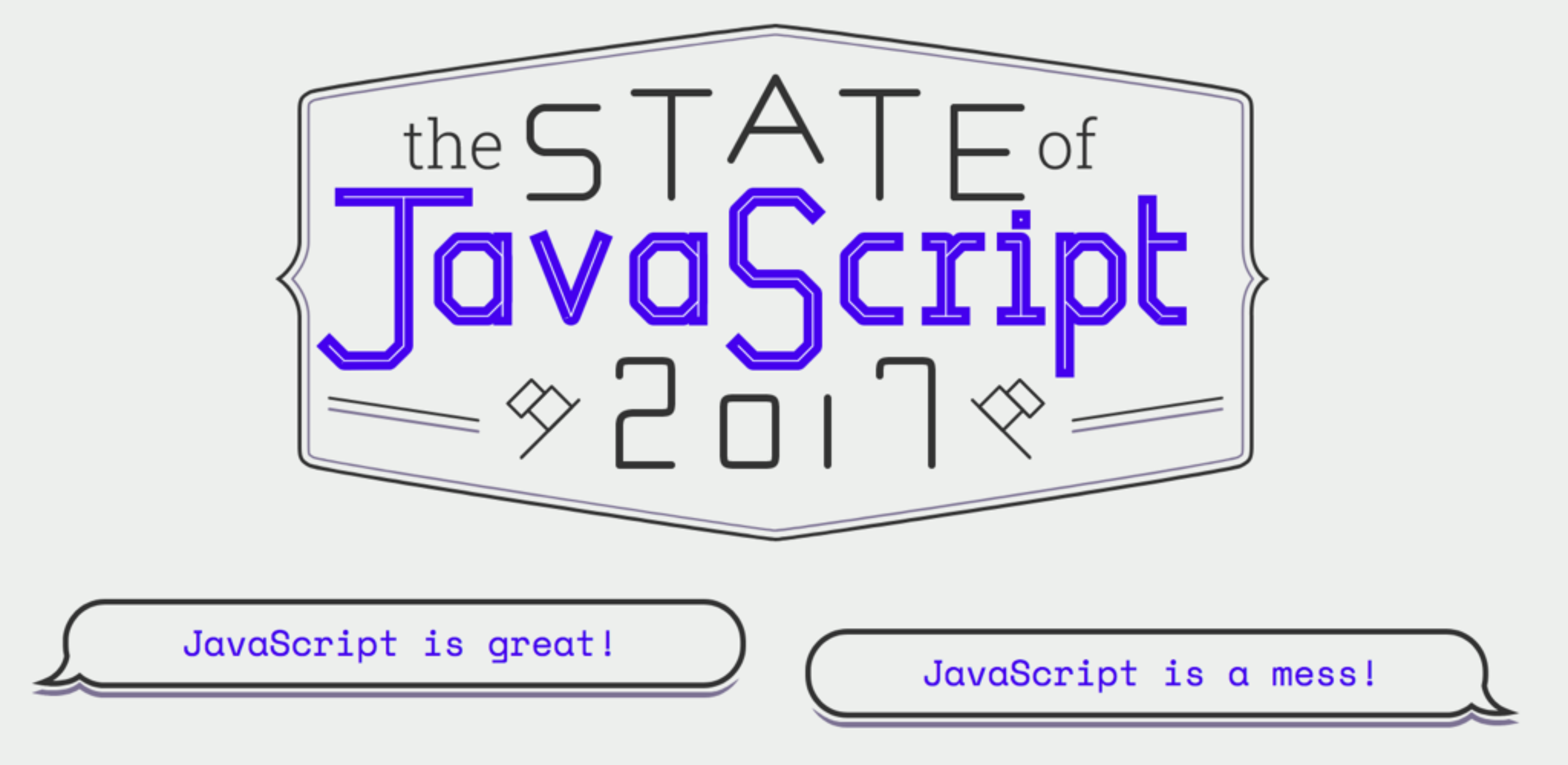
TWEB 2017

Olivier Liechti

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

https://t.me/joinchat/CPPWmAsLLgWdXQhoXTaNHw

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



2017 results just published

Ingredients

Yo

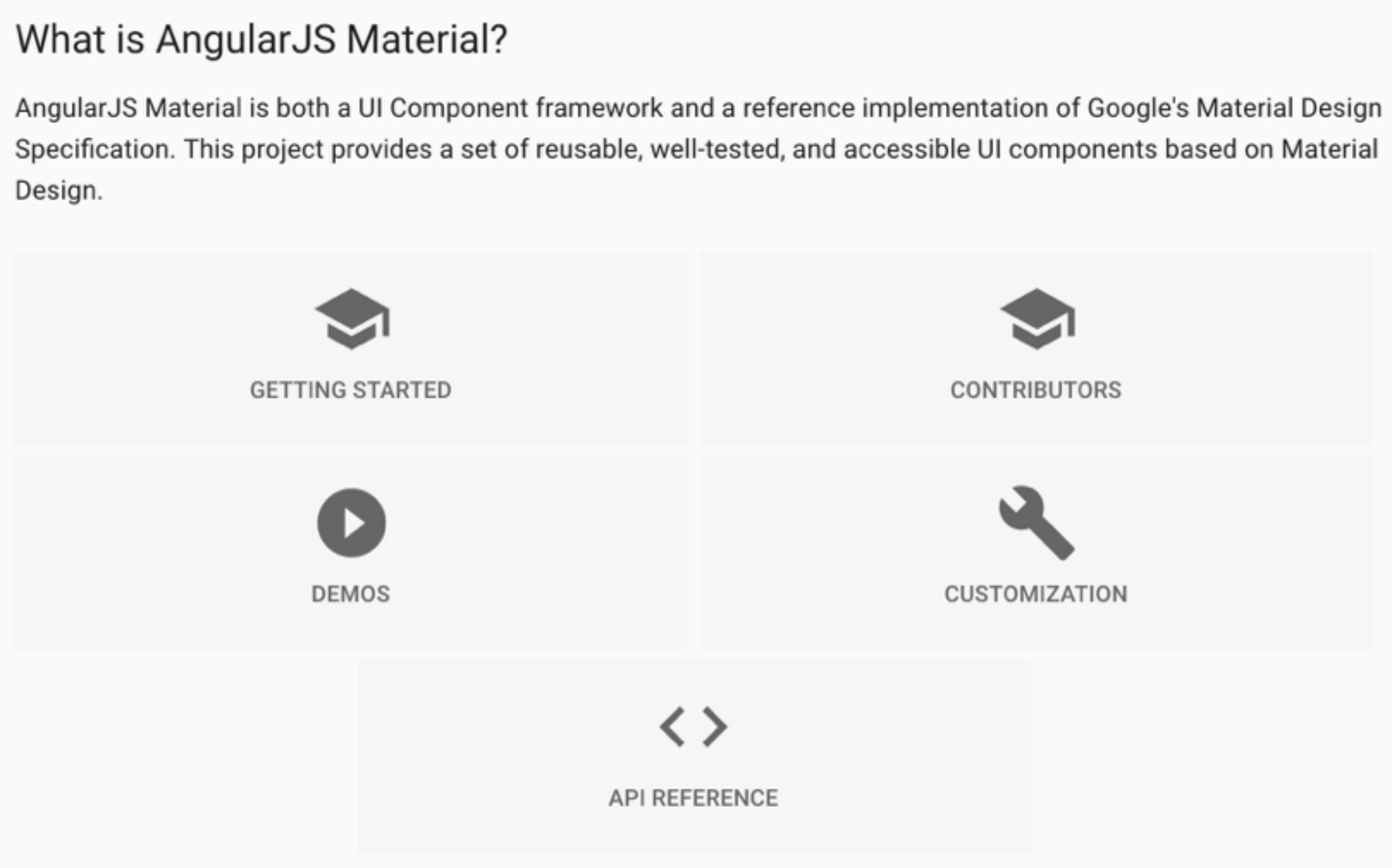
johnpapa / angular-styleguide

AngularJS Patterns: Clean Code

Learn how to write AngularJS code using battle tested patterns to produce code that is easier to maintain, debug, and scale.

ANGULARJS
Material

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



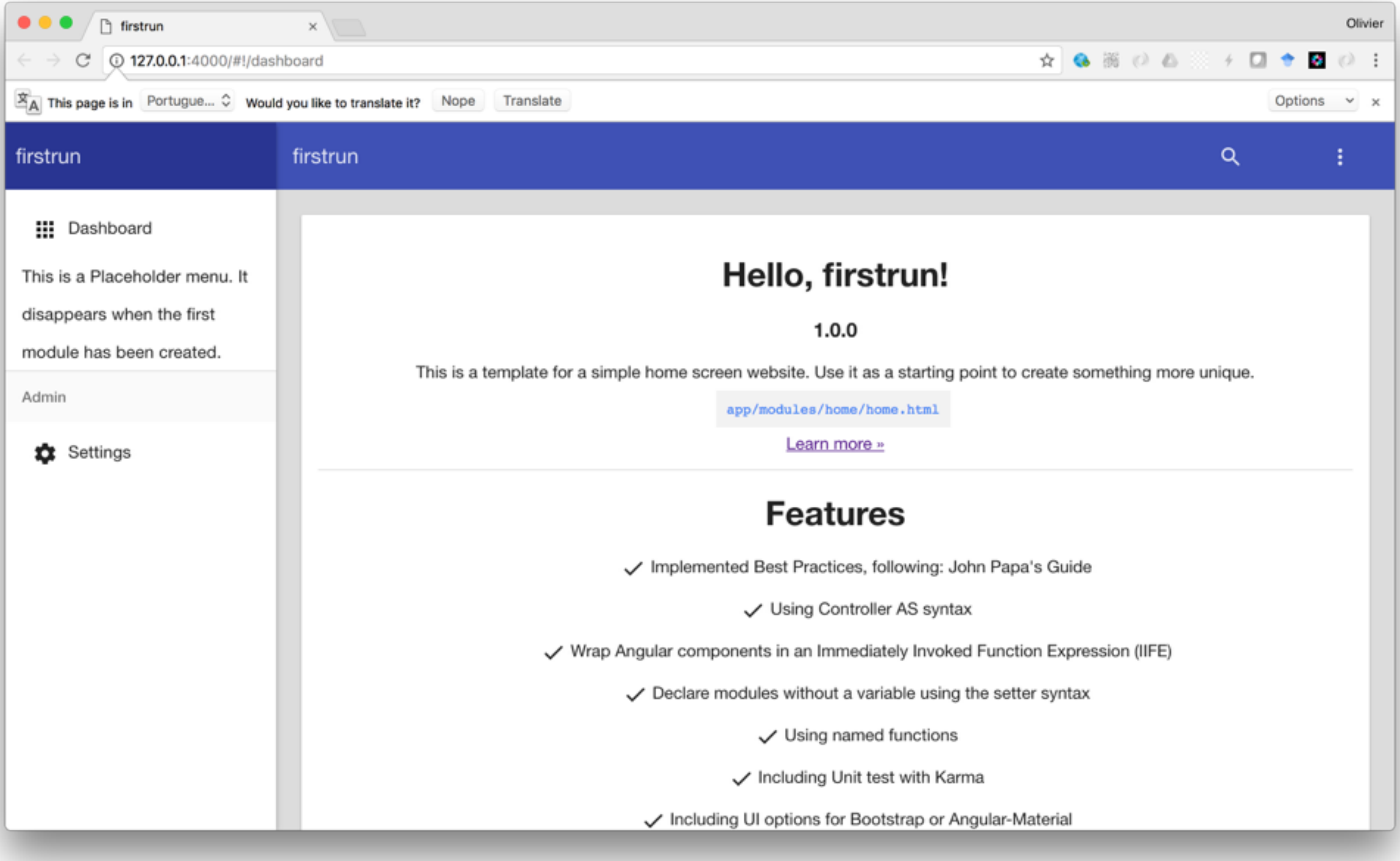
What is AngularJS Material?

AngularJS Material is both a UI Component framework and a reference implementation of Google's Material Design Specification. This project provides a set of reusable, well-tested, and accessible UI components based on Material Design.

Demo

```
$ npm install -g yo
$ npm install -g generator-angm
$ npm install -g grunt-cli
$ npm install -g bower-installer
```

```
$ yo angm
$ grunt dev
```



This is a Placeholder menu. It disappears when the first module has been created.

Hello, firstrun!

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

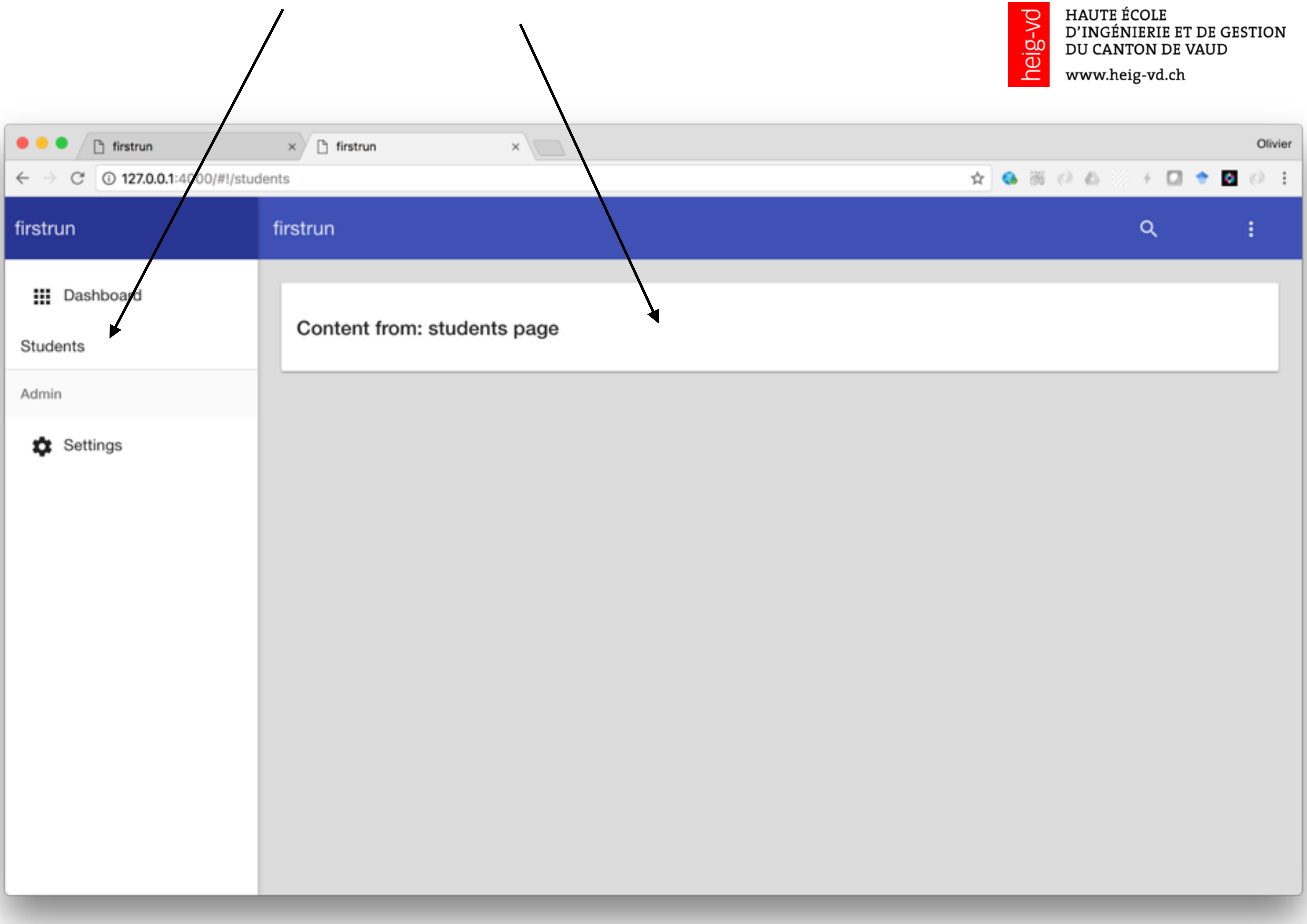
✓ Including Unit test with Karma

✓ Including UI options for Bootstrap or Angular-Material

Sub-generator

```
$ yo angm:angm-module
```

The yo generator helps us with the organization of our code in folders and files.



127.0.0.1:4000/#!/students

Content from: students page

Source files

```
app/
├── app.config.js
├── app.js
├── assets
|    ├── css
|    ├── fonts
|    └── images
└── modules
     ├── home
     |    ├── dashboard.html
     |    ├── home-test.js
     |    ├── home.html
     |    ├── homeCtrl.js
     |    ├── homeModule.js
     |    ├── homeRoute.js
     |    └── homeService.js
     ├── layouts
     |    ├── main-page
     |    |    ├── main-page.html   ?
     |    |    └── mainPageCtrl.js
     |    ├── nav-bar
     |    |    └── navBarService.js
     |    └── side-nav
     |         ├── sidenav.html
     |         ├── sidenavCtrl.js
     |         └── sidenavService.js
     ├── shared
     └── students
          ├── students-test.js
          ├── students.html
          ├── studentsCtrl.js
          ├── studentsModule.js
          ├── studentsRoute.js
          └── studentsService.js
```

Bootstrapping

```
<!DOCTYPE html>
<html lang="pt" ng-app="firstrun">
<head>
      <base href="/">
      <!-- injector:css -->
      <link rel="stylesheet" href="/src/bower_components/angular-material-icons/angular-material-icons.css">
      <link rel="stylesheet" href="/src/bower_components/angular-material/angular-material.css">
      <!-- endinjector -->
</head>

<body ng-cloak>

<!--Beginning-->
<div ui-view layout="row" layout-fill></div>
<!--End-->

<!--Templates. Remove then to a better location -->
<script type="text/ng-template" id="tabDialog.tmpl.html">
    <md-dialog
</script>

<!-- injector:js -->
<script src="/src/bower_components/angular/angular.js"></script>
<script src="/src/bower_components/angular-material/angular-material.js"></script>
<script src="/src/bower_components/angular-ui-router/release/angular-ui-router.js"></script>
<script src="/app/app.js"></script>
<script src="/app/app.config.js"></script>
<script src="/app/modules/home/homeModule.js"></script>
<script src=“/app/modules/students/studentsModule.js"></script>
...
<script src="/app/modules/layouts/nav-bar/navBarService.js"></script>
<script src="/app/modules/layouts/side-nav/sidenavService.js"></script>
<script src="/app/modules/students/studentsService.js"></script>
<!-- endinjector -->
```

Bootstrapping

```
injector: {
        options: {},
        dev: {
                files: {
                        'index.html': [
                                'bower.json',
                                'app/app.js',
                                'app/app.config.js',
                                'app/**/*Module.js',
                                'app/**/*Route.js',
                                'app/**/*Ctrl.js',
                                'app/**/*Service.js',
                                'app/**/*Directive.js'
                        ]
                }
        },
        production: {
                files: {
                        'index.html': [
                                'app/assets/css/**/*.css',
                                'app/assets/js/*.js'
                        ]

                }
        }
},
```

Modules

```
(function () {
   'use strict';

   /**
    * @ngdoc function
    * @name app.module:studentsModule
    * @description
    * # studentsModule
    * Module of the app
    */

   angular.module('students', []);

})();
```

declare a new module,
which does not have dependencies

Routing

lookup the module and call its configuration function

```
angular.module('students')
   .config(['$stateProvider', function ($stateProvider) {

      $stateProvider
         .state('home.students', {
            url:'/students',
            templateUrl: 'app/modules/students/students.html',
            controller: 'StudentsCtrl',
            controllerAs: 'vm'
         });


   }]);
```

inject a dependency on the $stateProvider service into the config function

In the template, we will have an instance of StudentsCtrl and refer to it as 'vm'

Templates

```
<md-sidenav layout="column" class=“md-sidenav-left>
</md-sidenav>
<div layout=“column" ng-controller="LayoutCtrl as layout">
    <md-content layout="column" flex md-scroll-y>
        <div ui-view></div>
    </md-content>
</div>
```

The students.html view is injected in the home/home.html template (in the ui-view div)

```
<div class="md-padding" flex layout-sm="column">
    <md-card>
        <md-card-content>
            <h2 class="md-title">Content from: students page</h2>
        </md-card-content>
    </md-card>
</div>
```

MVC

```
(function() {
    'use strict';

    angular
        .module('students')
        .controller('StudentsCtrl', Students);

        Students.$inject = []; X

        /*
        * recommend
        * Using function declarations
        * and bindable members up top.
        */

        function Students() {
            /*jshint validthis: true */
            var vm = this;
        }

})();
```

```
(function() {
    'use strict';

    /**
     * @ngdoc function
     * @name app.service:studentsService
     * @description
     * # studentsService
     * Service of the app
     */

    angular
        .module('students')
        .factory('StudentsService', Students);
        // Inject your dependencies as .$inject = X
['$http', 'someSevide'];
        // function Name ($http, someSevide) {...}

        Students.$inject = ['$http'];

        function Students ($http) {

        }

})();
```

```
<div class="md-padding" flex layout-sm="column">
    <md-card>
        <md-card-content>
            <h2 class="md-title">Content from: animals page</h2>
            {{ vm.displayMessage() }}
        </md-card-content>
    </md-card>
</div>
```

view

```
function Animals(AnimalsService) {
    var vm = this;
    vm.displayMessage = function() {
        return 'hello from controller, ' +
        AnimalsService.getMessageFromRestApi();
    }
}
```

controller

```
angular
    .module('animals')
    .factory('AnimalsService', Animals);
    // Inject your dependencies as .$inject = ['$http', 'someSevide'];
    // function Name ($http, someSevide) {...}

    Animals.$inject = ['$http'];

    function Animals ($http) {
        return {
            getMessageFromRestApi: function() {
                return "hello from REST API";
            }
        }
    }
```

service