# Lecture 3: API Copilot

Olivier Liechti
TWEB

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

# What's it for?

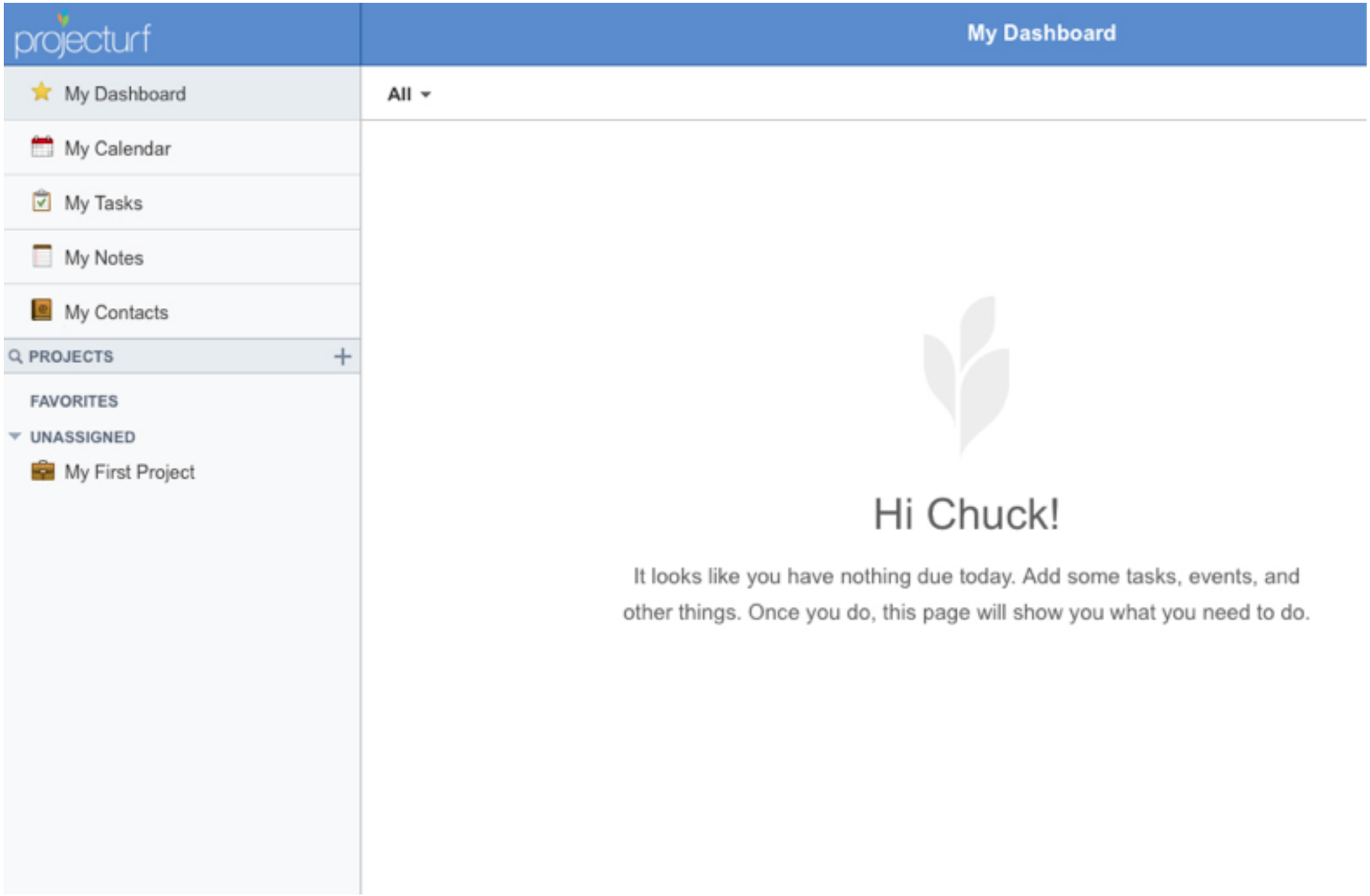
heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud


I have an API (or a website, or an app) to test...

I want to write
automated tests...

# But I have no data!

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud



# API Copilot

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

The purpose of API Copilot is to
help you create some **test data**.

# Based on Node.js

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Cool scripting language
Can make HTTP requests

# But...

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

## Why not just use Node.js since it can make HTTP requests?

```
var http = require('http');

//The url we want is: 'www.random.org/integers/?num=1&min=1&max=10&col=1&base=10&format=plain&rnd=new'
var options = {
  host: 'www.random.org',
  path: '/integers/?num=1&min=1&max=10&col=1&base=10&format=plain&rnd=new'
};

callback = function(response) {
  var responseBody = '';

  //another chunk of data has been received, so append it to `str`
  response.on('data', function (chunk) {
    responseBody += chunk;
  });

  //the whole response has been received, so we just print it out here
  response.on('end', function () {
    console.log(responseBody);
  });
}

http.request(options, callback).end();
```

Ouch...

# API Copilot: Scenarios

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- A **scenario** is a series of steps that are executed in order.
- Steps are executed **asynchronously**, but API Copilot attempts to **hide** this complexity.

```
var copilot = require('api-copilot');

var scenario = new copilot.Scenario({
  name: 'My Demo Sample Data'
});

scenario.step('create some data', function() {
  return 'some data';
});

scenario.step('log the data', function(data) {
  console.log(data);
});
```

# API Copilot: HTTP requests

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- API Copilot includes another NPM package, **request**, to make HTTP requests simpler.
- Simply return **`this.get`**, **`this.post`**, **`this.put`**, **`this.delete`**, etc.
- You will receive the HTTP response as an argument in the next step.

```
scenario.step('POST a user', function() {

  // make a POST HTTP request
  return this.post({
    url: '/users',
    body: {
      name: 'bob',
      password: 'changeme'
    }
  });
});

scenario.step('log the user', function(response) {
  console.log(response.body);
});
```

# API Copilot: Multiple HTTP requests

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

- You can make **multiple HTTP requests** in one step by passing them to **`this.all`**.
- The requests will be executed **asynchronously** and **in parallel**.

```
// Data to populate into the API.
var people = [
  { firstName: "John", lastName: "Doe" },
  { firstName: "Jane", lastName: "Doe" },
  { firstName: "Bob", lastName: "Smith" }
];

scenario.step('create people', function() {

  // Array of requests.
  var requests = [];

  // Convert person objects into HTTP requests.
  var n = people.length;
  for (var i = 0; i < n; i++) {
    requests.push(this.post({
      url: '/people',
      body: people[i]
    }));
  }

  // Pass the requests to `this.all`.
  return this.all(requests);
});
```

# API Copilot: MVC example project

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

https://github.com/SoftEng-HEIGVD/Teaching-HEIGVD-TWEB-Example-MVC

## Current API

```
GET /beers
POST /beers
```

# Exercise

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- **Clone** the repository. (If you have already cloned it, **pull** the latest changes.)
- **Add** the following REST resources:

```
GET /beers/{id}
DELETE /beers/{id}
```

- You will find useful methods in **`app/services/datastore.js`**.
- **Write an API Copilot scenario** to populate test data using your API. The scenario should have the following steps:
  - Get the list of all beers
  - Delete each beer
  - Create a pre-defined list of beers
- Since your scenario deletes and recreates all beers, running it two or three times should produce **the same result**.

# MVC example project: Usage

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- Clone the repository:
  - `git clone https://github.com/SoftEng-HEIGVD/Teaching-HEIGVD-TWEB-Example-MVC.git`
- *In the directory of the MVC example project,* install its dependencies:
  - `npm install`
- Start the Node.js application:
  - `npm start`
- Make JSON HTTP requests to test the API:
  - `curl -H "Content-Type: application/json" -X POST -d '{"name":"Heineken","country":"Netherlands"}' http://localhost:3000/beers`

# API Copilot: Installation

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- Install the API Copilot command line tool globally (you might need `sudo`):
  - `npm install -g api-copilot-cli`
- Install API Copilot *in the directory of the MVC example project*:
  - `npm install --save-dev api-copilot`
  - `mkdir api`
- Create your scenario file:
  - `api/testData.scenario.js`
- Use the API Copilot documentation to write your scenario.

# Resources

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

**API Copilot (scenarios & steps)**

https://github.com/AlphaHydrae/api-copilot

**Request library (HTTP request options)**

https://github.com/request/request#requestoptions-callback