

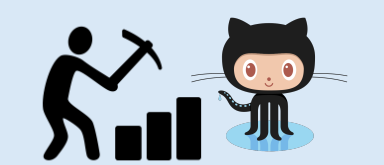
Lecture 5: authentication & testing

Olivier Liechti
TWEB

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Pour ce projet, vous travaillez en équipes de 2 étudiant(e)s

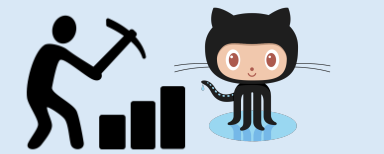


19.09.2016	Bootcamp	Introduction Javascript 101, squelette e2e, heroku	As a user, I can visit the product landing page, transition to the web app and back to the landing page.
26.09.2016		Introduction à AngularJS Coding guidelines, controllers, services, directives	As a user, I can navigate between several pages. As a user, I can see a graph (pie chart)
03.10.2016		Programmation asynchrone Callbacks, promesses, async.js	As a user, I can give provide a GitHub repo, user or organization and see "some" analysis (TBD)
10.10.2016		Persistence NoSQL MongoDB, mongoose	As a user, I can see the history of my previous queries (towards GitHub) As a user, I can select one query in the history and see the results again (data comes from DB)
17.10.2016			As a user, I can see the history of my previous queries (towards GitHub)

Vendredi 28.10.2016 (23h59)
Remise projet "Mining data in GitHub"

Mercredi 09.11.2016
Travail écrit 1

31.10.2016	Itération 1	Sondages interactifs simples Implémentation de la fonctionnalité de base	As a presenter, I can ask a question to the audience As an auditor, I can answer the questions As a participant, I can see the answers of the audience
07.11.2016		Applications interactives Socket.IO	
14.11.2016	Itération 2	Gestion des comptes Authentification et autorisation	As a guest, I can register and create an account As a registered user, I can log in and out of my account As a registered user, I have access to specific features (TBD)
21.11.2016			



28.11.2016	"Excursion"	Analyse de données en JavaScript Introduction à Elasticsearch, D3.js et dockerode	As an app, I can send a stream of user events via the /events endpoint As an app owner, I can define a rule to award a badge when some conditions are met (via the /rules endpoint)
05.12.2016			

Vendredi 09.12.2016 (23h59)
Remise "Interactive polls" 1

12.12.2016	Itération 4	Fonctionnalités "avant le cours" Que peut-on faire avant une session?	TBD
19.12.2016			
09.01.2016	Itération 5	Fonctionnalités "après le cours" Que peut-on faire après une session?	TBD
16.01.2016			

Lundi 16.01.2017
Travail écrit 2
Vendredi 20.01 (23h59)
Remise finale "Interactive polls"

Lundi 23.01 et mercredi 25.01
Présentations & démos (2 projets!)

23.01.2016	Wrap-up
------------	---------

Pour ce projet, vous travaillez en équipes de 2 étudiant(e)s



19.09.2016

Bootcamp

Introduction

Javascript 101, squelette e2e, heroku

As a user, I can visit the product landing page, transition to the web app and back to the landing page.

26.09.2016

Introduction à AngularJS

Coding guidelines, controllers, services, directives

As a user, I can navigate between several pages.

As a user, I can see a graph (pie chart)

03.10.2016

Programmation asynchrone

Callbacks, promesses, async.js

As a user, I can give provide a GitHub repo, user or organization and see "some" analysis (TBD)

10.10.2016

Persistence NoSQL

MongoDB, mongoose

As a user, I can see the history of my previous queries (towards GitHub)

As a user, I can select one query in the history and see the results again (data comes from DB)

17.10.2016

As a user, I can see the history of my previous queries (towards GitHub)

Vendredi 28.10.2016 (23h59)

Remise projet "Mining data in GitHub"

31.10.2016

Itération 1

Sondages interactifs simples

Implémentation de la fonctionnalité de base

As a presenter, I can ask a question to the audience

As an auditor, I can answer the questions

As a participant, I can see the answers of the audience

07.11.2016

Applications interactives

Socket.IO

14.11.2016

Itération 2

Gestion des comptes

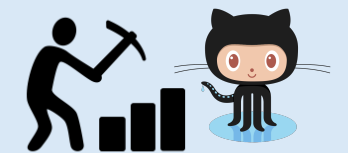
Authentification et autorisation

As a guest, I can register and create an account

As a registered user, I can log in and out of my account

As a registered user, I have access to specific features (TBD)

21.11.2016



28.11.2016

"Excursion"

Analyse de données en JavaScript

Introduction à Elasticsearch, D3.js et dockerode

As an app, I can send a stream of user events via the /events endpoint

As an app owner, I can define a rule to award a badge when some conditions are met (via the /rules endpoint)

05.12.2016

Vendredi 09.12.2016 (23h59)

Remise "Interactive polls" 1

12.12.2016

Itération 4

Fonctionnalités "avant le cours"

Que peut-on faire avant une session?

TBD

19.12.2016

09.01.2016

Itération 5

Fonctionnalités "après le cours"

Que peut-on faire après une session?

TBD

16.01.2016

Lundi 16.01.2017

Travail écrit 2

Vendredi 20.01 (23h59)

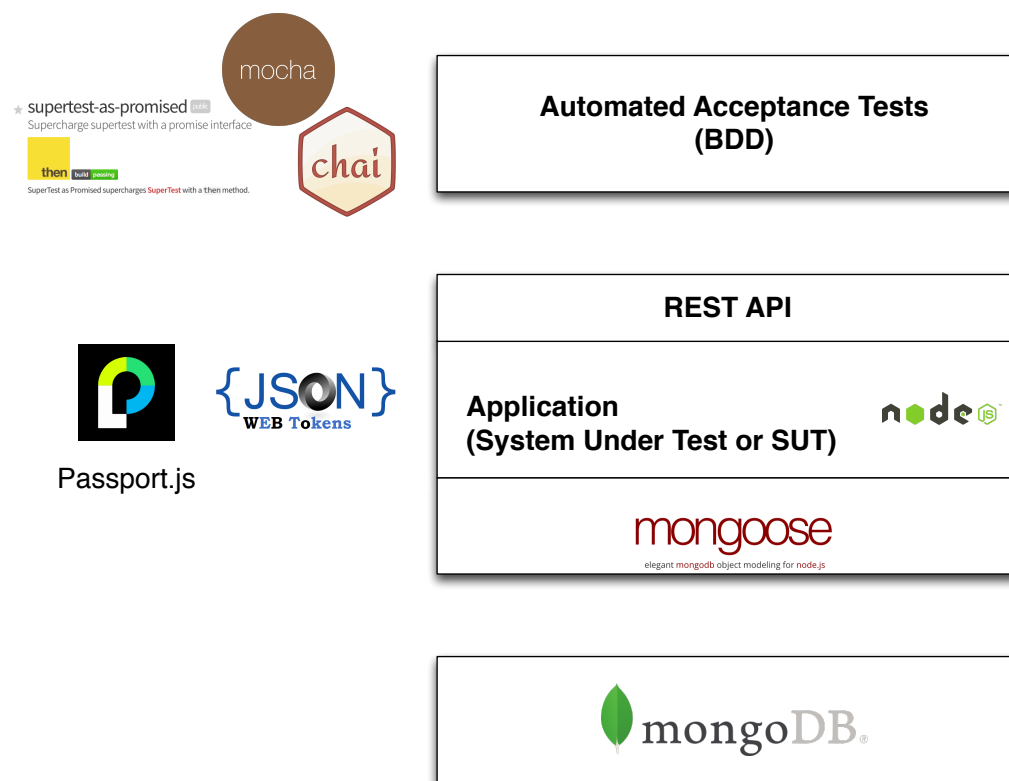
Remise finale "Interactive polls"

Lundi 23.01 et mercredi 25.01

Présentations & démos (2 projets!)

23.01.2016

Wrap-up



Tasks

1. Setup the environment

- 1.1. Start a MongoDB server in a Docker container
- 1.2. Create an Express.js project
- 1.3. Add dependencies

2. Prepare the automated tests

- 2.1. Install mocha
- 2.2. Add dependencies on mocha, chai and supertest-as-promised

3. Implement the /register endpoint

- 3.1. Specify behavior in an automated test
- 3.2. Implement the Mongoose schema and model
- 3.3. Implement the Express.js router
- 3.4. Validate

4. Implement the /auth endpoint

- 4.1. Specify behavior
- 4.2. Implement a route and return a JSON Web Token upon success
- 4.3. Validate

5. Implement and protect the GET operation on /users

- 5.1. Specify behavior (only authenticated users can invoke the operation)
- 5.2. Update the Express.js router
- 5.3. Validate

What is Mongoose?

What is the relationship between mocha,
chai and supertest-as-promised?

What is Passport.js?

What are JSON Web Tokens?

<https://blog.hyphe.me/token-based-authentication-with-node/>
<https://scotch.io/tutorials/authenticate-a-node-js-api-with-json-web-tokens>
<http://beletsky.net/2013/10/securing-express-dot-js-http-endpoints.html>
<https://www.toptal.com/web/cookie-free-authentication-with-json-web-tokens-an-example-in-laravel-and-angularjs>

POST /register HTTP/1.1

Content-type: application/json

```
{
  "username" : "",
  "firstName" : "",
  "lastName" : "",
  "password" : ""
}
```

HTTP/1.1 201 Created

Location: /users/2982938923

Constraints:

- all fields are mandatory
- username must be unique in the DB
- password length must be at least 8

Server creates a new user in the DB

Server returns 201 and Location header

Server returns 422 otherwise

POST /auth HTTP/1.1

Content-type: application/json

```
{
  "username" : "",
  "password" : ""
}
```

HTTP/1.1 200 OK

Content-type: application/json

eyJ0eXAiOiJKV1QiLCJhb...

Constraints:

- all fields are mandatory
- username must exist in the DB
- password must match with

Server responds 200 and sends back a JSON Web Token

Server responds 401 otherwise

GET /users HTTP/1.1

Accept: application/json

Authorization: Bearer eyJ0eXAi...

HTTP/1.1 200 OK






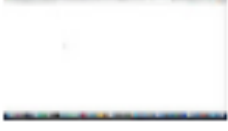
Content-type: application/json

[{... }, { ... }]

Constraints:

- A valid JSON Web Token must be passed in the Auth header

Server sends back the list of all registered users

17		Secure REST API with automated tests (1) by oliechti	9:59
18		Secure REST API with automated tests (2) by oliechti	5:24
19		Secure REST API with automated tests (3) by oliechti	15:18
20		Secure REST API with automated tests (4) by oliechti	19:16
21		Secure REST API with automated tests (5) by oliechti	12:18
22		Secure REST API with automated tests (6) by oliechti	10:55