

A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is light green. They are positioned diagonally, with the blue one partially covering the green one.

Rattrapages MSC - I

Tio Gobin

0.3.B06 - Adapt Communication Scheme to an objective

Label issues and pull requests for new contributors [Dismiss](#)

Now, GitHub will help potential first-time contributors [discover issues](#) labeled with **good first issue**

Filters Labels 9 Milestones 0 [New issue](#)

3 Open ✓ 4 Closed

<input type="checkbox"/>	Author	Label	Projects	Milestones	Assignee	Sort
<input type="checkbox"/>	Stream logs to front-end	help wanted question wontfix				1
	#7 opened on May 28 by 2O4 5 tasks					
<input type="checkbox"/>	connect to AWS bucket	enhancement				
	#5 opened on May 24 by 2O4					
<input type="checkbox"/>	async endpoints	bug enhancement wontfix				
	#4 opened on May 24 by 2O4					

ProTip! Type **g** **i** on any issue or pull request to go back to the issue listing page.

-Utilisation des issues sur la plupart des projets pour organiser la communication au sein des projets (ex: ici projet ESP - DHOST pour le repo build-microservice)

0.3.B07 - Define Procedures

-Process du projet ESP - DHOST

Méthode de travail et process

TDD: Test-Driven Development, ou Développements Pilotés par les Tests en français, est une méthode de développement de logiciel, qui consiste à concevoir un logiciel de façon itérative et incrémentale, en **écrivant chaque test avant d'écrire le code source et en remaniant le code continuellement**.

Gantt: Pour la planification des tâches sur le temps.

Kanban: Pour l'affinage, le suivi et la distribution des tâches.

Processus d'assurance qualité



TDD: Test-Driven Development, ou Développements Pilotés par les Tests en français, est une méthode de développement de logiciel, qui consiste à concevoir un logiciel de façon itérative et incrémentale, en **écrivant chaque test avant d'écrire le code source et en remaniant le code continuellement**.

Code reviews (à chaque **pull requests**) : La revue de code est un examen systématique du code source d'un logiciel. L'objectif étant de trouver des bugs ou des vulnérabilités potentielles ou de corriger des erreurs de conception afin d'améliorer la qualité, la maintenabilité et la sécurité du logiciel.

Test #1 Github

User Story	En tant qu'utilisateur, je souhaite envoyer les fichiers source de mon site sur la plateforme à l'aide de github .
Acceptance Criteria	L'utilisateur se connecte avec son compte Github. L'utilisateur peut choisir un repository à utiliser comme source pour le site qu'il veut héberger. L'utilisateur peut choisir les paramètres de compilation (bundle). L'utilisateur peut faire compiler ses fichiers source. L'utilisateur peut uploader son site (bundle) sur l'IPFS. L'utilisateur peut accéder à son site avec son navigateur web sans add-ons via la gateway DHost ou une autre.
Critical	Oui
Test Result	En attente
Comments	La plateforme ne sera pas publiée sans cette fonctionnalité

Test #2 Bundle

User Story	En tant qu'utilisateur, je souhaite envoyer le bundle de mon site sur la plateforme directement (par drag & drop).
Acceptance Criteria	L'utilisateur peut choisir un fichier à utiliser comme bundle pour le site qu'il veut héberger. L'utilisateur peut uploader son site (bundle) sur l'IPFS. L'utilisateur peut accéder à son site avec son navigateur web sans add-ons via la gateway DHost ou une autre.
Critical	Oui
Test Result	En attente
Comments	La plateforme ne sera pas publiée sans cette fonctionnalité

Test #3 CLI

User Story	En tant qu'utilisateur, je souhaite envoyer les fichiers source de mon site sur la plateforme à l'aide du CLI .
Acceptance Criteria	L'utilisateur se connecte avec son compte DHost. L'utilisateur peut choisir un repository à utiliser comme source pour le site qu'il veut héberger s'il son compte est liée à Github. L'utilisateur peut choisir un dossier à utiliser comme source.

0.3.B09 - Build a functional test sequences

-Tests
fonctionnels,
projet
EpicRoadTrip

```
import unittest
import requests
import subprocess
import sqlite3

LOCAL_URI = "http://127.0.0.1:5000/"

class TestAuthMethods(unittest.TestCase):

    def test_password_does_not_match_on_create(self):
        response = requests.post("{}signup".format(LOCAL_URI), data={
            "username": "test1",
            "pwd1": "test",
            "pwd2": "test2",
        })
        self.assertEqual(response.status_code, 400)

    def test_password_does_match_on_create(self):
        response = requests.post("{}signup".format(LOCAL_URI), data={
            "username": "test2",
            "pwd1": "test",
            "pwd2": "test",
        })
        self.assertEqual(response.status_code, 201)

    def test_failed_authentication(self):
        response = requests.post("{}signin".format(LOCAL_URI), data={
            "username": "test2",
            "password": "tes",
        })
        self.assertEqual(response.status_code, 401)

class TestResourcesMethods(unittest.TestCase):

    def test_base_coordinate_not_defined_on_eat(self):
        response = requests.post("{}eat".format(LOCAL_URI), json={
            "range": 10
        })
        self.assertEqual(response.status_code, 400)

    def test_base_coordinate_defined_on_eat(self):
        response = requests.post("{}eat".format(LOCAL_URI), json={
            "base_coordinate": ["48.88764803067599", "2.3933235937058703"],
            "range": 10
        })
        self.assertEqual(response.status_code, 200)
```

```
def test_base_coordinate_defined_on_eat(self):
    response = requests.post("{}eat".format(LOCAL_URI), json={
        "base_coordinate": ["48.88764803067599", "2.3933235937058703"],
        "range": 10
    })
    self.assertEqual(response.status_code, 200)

def test_base_coordinate_not_defined_on_enjoy(self):
    response = requests.post("{}enjoy".format(LOCAL_URI), json={
        "range": 10
    })
    self.assertEqual(response.status_code, 400)

def test_base_coordinate_defined_on_enjoy(self):
    response = requests.post("{}enjoy".format(LOCAL_URI), json={
        "base_coordinate": ["48.88764803067599", "2.3933235937058703"],
        "range": 10
    })
    self.assertEqual(response.status_code, 200)

def test_base_coordinate_not_defined_on_drink(self):
    response = requests.post("{}drink".format(LOCAL_URI), json={
        "range": 10
    })
    self.assertEqual(response.status_code, 400)

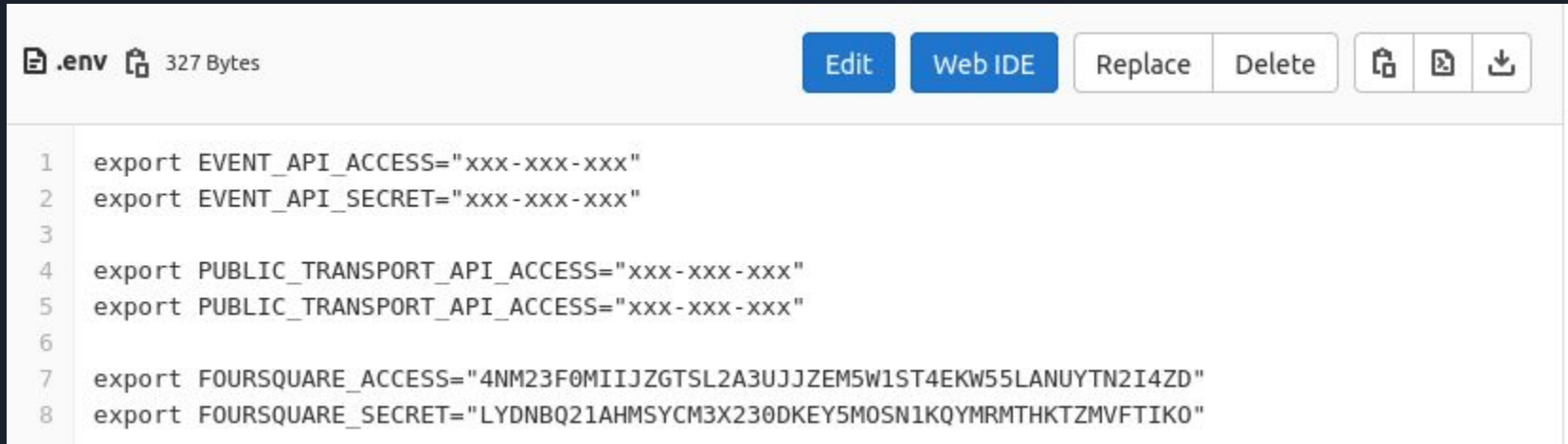
def test_base_coordinate_defined_on_drink(self):
    response = requests.post("{}drink".format(LOCAL_URI), json={
        "base_coordinate": ["48.88764803067599", "2.3933235937058703"],
        "range": 10
    })
    self.assertEqual(response.status_code, 200)

def test_base_coordinate_not_defined_on_travel(self):
    response = requests.post("{}travel".format(LOCAL_URI), json={
        "range": 10
    })
    self.assertEqual(response.status_code, 400)

def test_base_coordinate_defined_on_travel(self):
    response = requests.post("{}travel".format(LOCAL_URI), json={
        "base_coordinate": ["48.88764803067599", "2.3933235937058703"],
        "range": 10
    })
    self.assertEqual(response.status_code, 200)

if __name__ == '__main__':
    unittest.main()
```

4.2.B01 - Keep environnement variables coherent

A screenshot of a code editor interface. At the top left, there is a file icon followed by ".env" and a folder icon with "327 Bytes". To the right of this are four buttons: "Edit" (blue), "Web IDE" (blue), "Replace" (light gray), and "Delete" (light gray). Further right are three icons: a folder, a document, and a download arrow. The main area of the editor shows a list of environment variables, each preceded by a line number from 1 to 8. The variables are: EVENT_API_ACCESS, EVENT_API_SECRET, PUBLIC_TRANSPORT_API_ACCESS (listed twice), and FOURSQUARE_ACCESS and FOURSQUARE_SECRET.

```
1 export EVENT_API_ACCESS="xxx-xxx-xxx"
2 export EVENT_API_SECRET="xxx-xxx-xxx"
3
4 export PUBLIC_TRANSPORT_API_ACCESS="xxx-xxx-xxx"
5 export PUBLIC_TRANSPORT_API_ACCESS="xxx-xxx-xxx"
6
7 export FOURSQUARE_ACCESS="4NM23F0MIIJZGTSL2A3UJJZEM5W1ST4EKW55LANUYTN2I4ZD"
8 export FOURSQUARE_SECRET="LYDNBQ21AHMSYCM3X230DKEY5M0SN1KQYMRMTHKTZMVFTIKO"
```

-Utilisation de variable d'environnement
cohérentes, dans le cadre du projet
EpicRoadTrip

4.2.B03 - Strengthen security using password policy

-Utilisation de token
JWT dans le cadre du
projet Voltron, groupe
Data

user.routes.js 665 Bytes

```
1 const { authJwt } = require("../middleware");
2 const controller = require("../controllers/user.controller");
3
4 module.exports = function(app) {
5   app.use(function(req, res, next) {
6     res.header(
7       "Access-Control-Allow-Headers",
8       "x-access-token, Origin, Content-Type, Accept"
9     );
10    next();
11  });
12
13  app.get("/api/test/all", controller.allAccess);
14
15  app.get("/api/test/user", [authJwt.verifyToken], controller.userBoard);
16
17  app.get(
18    "/api/test/mod",
19    [authJwt.verifyToken, authJwt.isModerator],
20    controller.moderatorBoard
21  );
22
23  app.get(
24    "/api/test/admin",
25    [authJwt.verifyToken, authJwt.isAdmin],
26    controller.adminBoard
27  );
28  };
```

4.2.B05 -Manage Access

-Gestion des accès utilisateurs a certaines routes à travers l'utilisation d'un token modifiable (uuid) stocké en id d'un utilisateur en base, projet EpicRoadTrip

```
def account_exist(user_id):  
    with sql.connect("database.db") as con:  
        con.row_factory = sql.Row  
        cur = con.cursor()  
        query = "SELECT * FROM users WHERE id = '{}'.format(user_id)"  
        cur.execute(query)  
        users = cur.fetchall()  
        if len(users) > 0:  
            return True  
        return False
```

```
if is_auth and resource_id:  
    with sql.connect("database.db") as con:  
        cur = con.cursor()  
        cur.execute(  
            "INSERT INTO histories (user_id, lat, lng, range, category, resource_id) VALUES ('{}', '{}', '{}', '{}', '{}', '{}').format(  
                user, base_coordinate[0], base_coordinate[1], range, action, resource_id  
            )  
        )  
        con.commit()  
  
return jsonify(parse_resources(resources)), 200
```

5.1.B05 - Define Public Access

```
client = foursquare.Foursquare(client_id=FOURSQUARE_ACCESS, client_secret=FOURSQUARE_SECRET)
if resource_id:
    resources = client.venues(resource_id)
else:
    resources = client.venues.search(
        params={
            "categoryId": RESOURCE_CATEGORY_IDS[action],
            "ll": ", ".join(data["base_coordonate"]),
            "radius": range * 1000,
        }
    )

if is_auth and resource_id:
    with sql.connect("database.db") as con:
        cur = con.cursor()
        cur.execute(
            "INSERT INTO histories (user_id, lat, lng, range, category, resource_id) VALUES ('{}', '{}', '{}', '{}', '{}', '{}')".format(
                user, base_coordonate[0], base_coordonate[1], range, action, resource_id
            )
        )
    con.commit()

return jsonify(parse_resources(resources)), 200
```

-Si aucun utilisateur n'est connecté, la requête faite à l'API effectue tout de même l'action sans conserver l'historique de recherche, projet EpicRoadTrip

5.1.B06 - Define Administrator access

-Définition de plusieurs rôles dans le cadre du projet Voltron pour la gestion des droits, notamment le rôle Administrateur

```
function initial() {
  Role.estimatedDocumentCount((err, count) => {
    if (!err && count === 0) {
      new Role({
        name: "user"
      }).save(err => {
        if (err) {
          console.log("error", err);
        }

        console.log("added 'user' to roles collection");
      });

      new Role({
        name: "moderator"
      }).save(err => {
        if (err) {
          console.log("error", err);
        }

        console.log("added 'moderator' to roles collection");
      });

      new Role({
        name: "admin"
      }).save(err => {
        if (err) {
          console.log("error", err);
        }

        console.log("added 'admin' to roles collection");
      });

      new Role({
        name: "test"
      }).save(err => {
        if (err) {
          console.log("error", err);
        }

        console.log("added 'test' to roles collection");
      });
    }
  });
}
```



5.1.B08 - Make IP automatically available

-Projet de Pre-MSC
Pro, You Shall Not
Pass, le DHCP (pour
attribuer
automatiquement
une adresse IP)
fonctionnait.

Services (4 / 7) :

Manque les users, DNS et MySQL