Documentation Tests

Introduction

Bienvenue dans la documentation utilisateur destinée à guider l'utilisation et la compréhension des tests en Python et en PHP développés pour le projet.

Table des matières

Equipe

Documentation réalisée par :

- MOINY Yanis
- MASSIP Romain
- BABEL Teddy

Tests en Python

Prérequis

Avant de commencer, assurez-vous d'avoir installé les outils et librairies nécessaires. Vous aurez besoin de :

- Python (version 3.x recommandée)
- Bibliothèques Python : paho-mqtt, PyYAML, behave via la commande :

```
pip install paho-mqtt PyYAML behave
```

et autres dépendances spécifiques au projet Lien vers la documentation générale python

Exécution des Tests

Afin d'exécuter les tests, placez-vous dans le dossier `/IOT/PYTHON` grâce à la commande :

```
cd IOT/PYTHON
```

Puis, lancez la commande suivante :

```
behave
```

WARNING	Il est obligatoire d'être connecté sur un réseau externe à l'IUT et exécuter le test sur un système d'exploitation autre que Windows.

Cela devrait donner le rendu suivant :

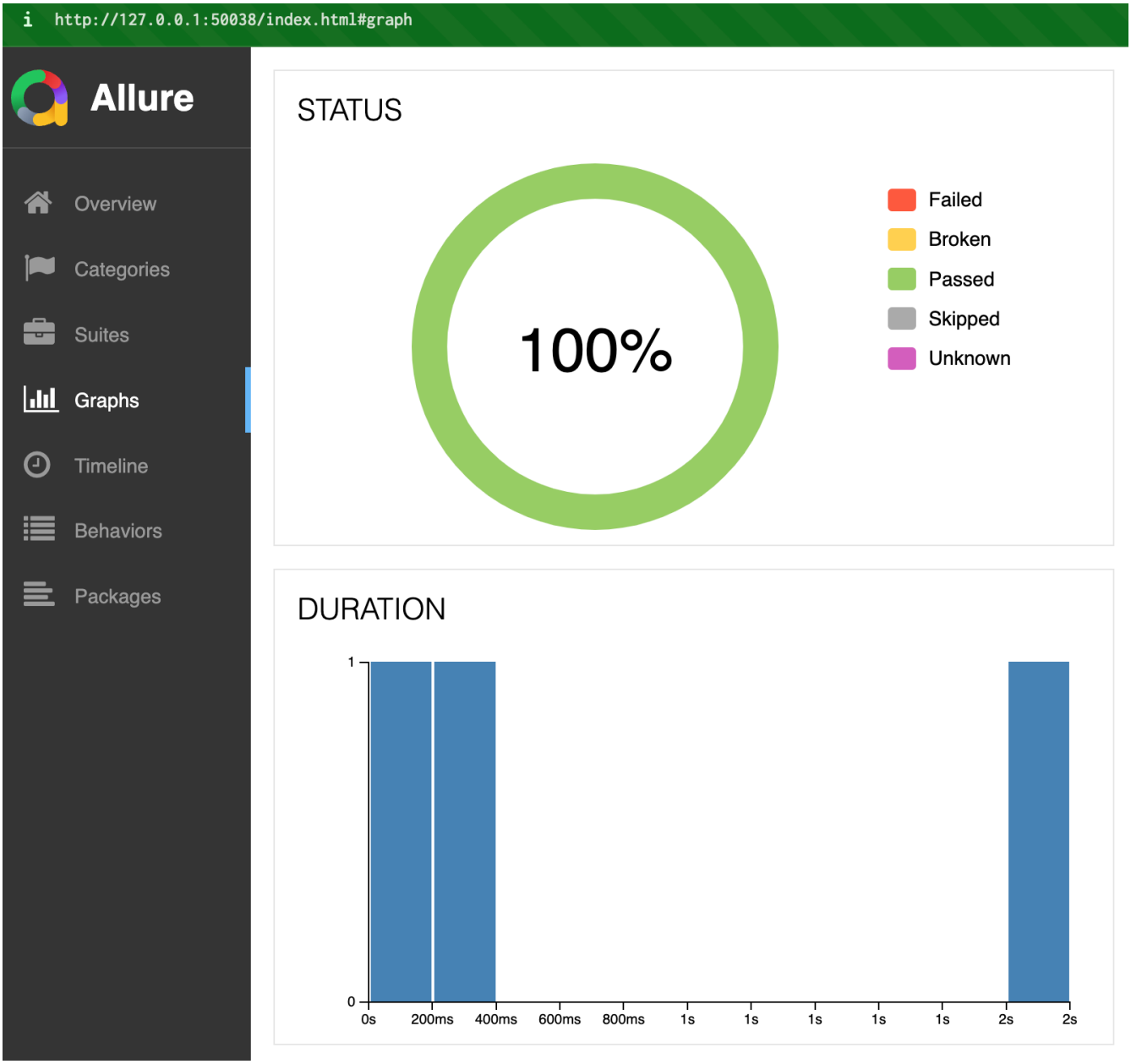
```
Feature: MQTT Server Connection # features/steps/test.feature:1
  As a user of the storage condition monitoring application
  I want to connect to the MQTT server
  So that I can receive data from the sensors
  Scenario: Successfully establishing a connection to the MQTT server  # features/steps/test.feature:6
    Given an available MQTT server                                     # features/steps/test.py:25 0.001s
    When the client attempts to connect                                # features/steps/test.py:59 0.210s
    Then a connection is successfully established                      # features/steps/test.py:68 0.098s

  Scenario: Successfully subscribing to MQTT topics  # features/steps/test.feature:11
    Given a connected MQTT client                    # features/steps/test.py:32 0.158s
    When the client subscribes to a topic            # features/steps/test.py:75 0.003s
    Then the subscription is successful              # features/steps/test.py:91 0.073s

  Scenario: Successfully receiving and processing MQTT messages  # features/steps/test.feature:16
    Given a connected MQTT client subscribed to topics           # features/steps/test.py:44 1.191s
    When a message is published to a subscribed topic            # features/steps/test.py:100 1.008s
    Then the message is received and processed correctly         # features/steps/test.py:131 0.009s

1 feature passed, 0 failed, 0 skipped
3 scenarios passed, 0 failed, 0 skipped
9 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m2.752s
```

Affichage des résultats

Il est possible d'afficher les résultats des tests dans un fichier HTML pour une meilleure lisibilité. Pour cela, il suffit d'exécuter la commande suivante (dans le dossier `/IOT/PYTHON`) :

```
allure serve reports
```

Cela ouvrira une page web dans votre navigateur par défaut avec les résultats des tests :



Structure des Tests

Le premier test consiste à vérifier la capacité du client MQTT à établir une connexion avec le serveur MQTT.

```
Scenario: Successfully establishing a connection to the MQTT server
  Given an available MQTT server
  When the client attempts to connect
  Then a connection is successfully established
```

```
@given('an available MQTT server')
def step_impl(context):
    context.server_url = "chirpstack.iut-blagnac.fr"
    context.server_port = 1883
    context.client = mqtt.Client()
    context.client.connected_flag = False
```

```
@when('the client attempts to connect')
def step_impl(context):
    context.client.on_connect = on_connect
    try:
        context.client.connect(context.server_url, context.server_port, 60)
        context.client.loop_start()
    except Exception as e:
        context.connection_exception = e
```

```
@then('a connection is successfully established')
def step_impl(context):
    context.client.loop_stop()
    if hasattr(context, 'connection_exception'):
        raise context.connection_exception
    assert context.client.connected_flag == True
```

Le deuxième test s'attache à valider que le client MQTT, une fois connecté, peut s'abonner à des topics spécifiques.

```
Scenario: Successfully subscribing to MQTT topics
  Given a connected MQTT client
  When the client subscribes to a topic
  Then the subscription is successful
```

```
@given('a connected MQTT client')
def step_impl(context):
    context.client = mqtt.Client()
    context.client.on_connect = on_connect
    context.client.connected_flag = False
    context.client.subscribed_flag = False
    context.client.connect("chirpstack.iut-blagnac.fr", 1883, 60)
    context.client.loop_start()
    while not context.client.connected_flag:
        pass  # Wait for connection
```

```
@when('the client subscribes to a topic')
def step_impl(context):

    with open("configuration.yaml", "r") as file:
        config = yaml.safe_load(file)

    context.client.on_subscribe = on_subscribe
    for topic in config["topics"]:
        try:
            context.client.subscribe(topic)
            print("~ Subscribed to " + topic)
            context.client.subscribed_flag = True
        except Exception as e:
            print("~ Failed to subscribe to {}: {}".format(topic, str(e)))
            context.client.subscribed_flag = False
```

```
@then('the subscription is successful')
def step_impl(context):
    assert context.client.subscribed_flag == True
    context.client.loop_stop()
```

Le troisième test vérifie la capacité du client MQTT à recevoir et à traiter des messages sur les topics auxquels il est abonné (donc que le message est bien récupérer sur le fichier CSV).

```
Scenario: Successfully receiving and processing MQTT messages
  Given a connected MQTT client subscribed to topics
  When a message is published to a subscribed topic
  Then the message is received and processed correctly
```

```
@given('a connected MQTT client subscribed to topics')
def step_impl(context):
    with open("configuration.yaml", "r") as file:
        config = yaml.safe_load(file)
    context.client = mqtt.Client()
    context.client.on_connect = on_connect
    context.client.connect(config["url"], config["port"], config["keepalive"])
    context.client.loop_start()
    while not hasattr(context.client, 'connected_flag') or not
context.client.connected_flag:
        time.sleep(0.1)  # Wait for connection
    for topic in config["topics"]:
        context.client.subscribe(topic)
    time.sleep(1)  # Wait for subscription
```

```
@when('a message is published to a subscribed topic')
def step_impl(context):
    with open("configuration.yaml", "r") as file:
        config = yaml.safe_load(file)

    test_topic = "AM107/by-room/E208/data"
    test_message_data = {
        "temperature": 21,
        "humidity": 59,
        "co2": 1371,
        "activity": 0,
        "tvoc": 391,
        "illumination": 2,
        "infrared": 2,
        "infrared_and_visible": 5,
        "pressure": 993.3
    }
    test_message_info = {
        "deviceName": "AM107-TestDevice",
        "devEUI": "00a1b2c3d4e5f678",
        "room": "B106",
        "floor": 2,
        "Building": "E"
    }
    test_message = [test_message_data, test_message_info]
    thread = threading.Thread(target=publish_test_message, args=(context.client,
test_topic, test_message))
    thread.start()
    thread.join()
    time.sleep(1)  # Permettre le traitement du message
```

```
@then('the message is received and processed correctly')
def step_impl(context):
    with open("configuration.yaml", "r") as file:
        config = yaml.safe_load(file)
    expected_data = {
        "temperature": 21,
        "humidity": 59,
        "co2": 1371,
        "activity": 0,
        "tvoc": 391,
        "illumination": 2,
        "infrared": 2,
        "infrared_and_visible": 5,
        "pressure": 993.3
    }
    with open(config["dataFile"], mode='r') as csvfile:
        csv_reader = csv.DictReader(csvfile)
        for row in csv_reader:
            if all(float(row[key]) == value for key, value in expected_data.items()):
                break
        else:
            assert False, "Les données attendues ne sont pas présentes dans le CSV"
```

Tests en PHP

Prérequis

Afin de pouvoir exécuter les tests en PHP, il est nécessaire d'avoir installé les outils suivants :

- PHP (8.* recommandé)
- Behat (Installation Behat)

Exécution des Tests

Afin d'exécuter les tests, placez-vous dans le dossier `/Site_eCommerce` grâce à la commande :

```
cd Site_eCommerce
```

Puis, lancez la commande suivante :

```
vendor/bin/behat
```

Cela devrait donner le rendu suivant :

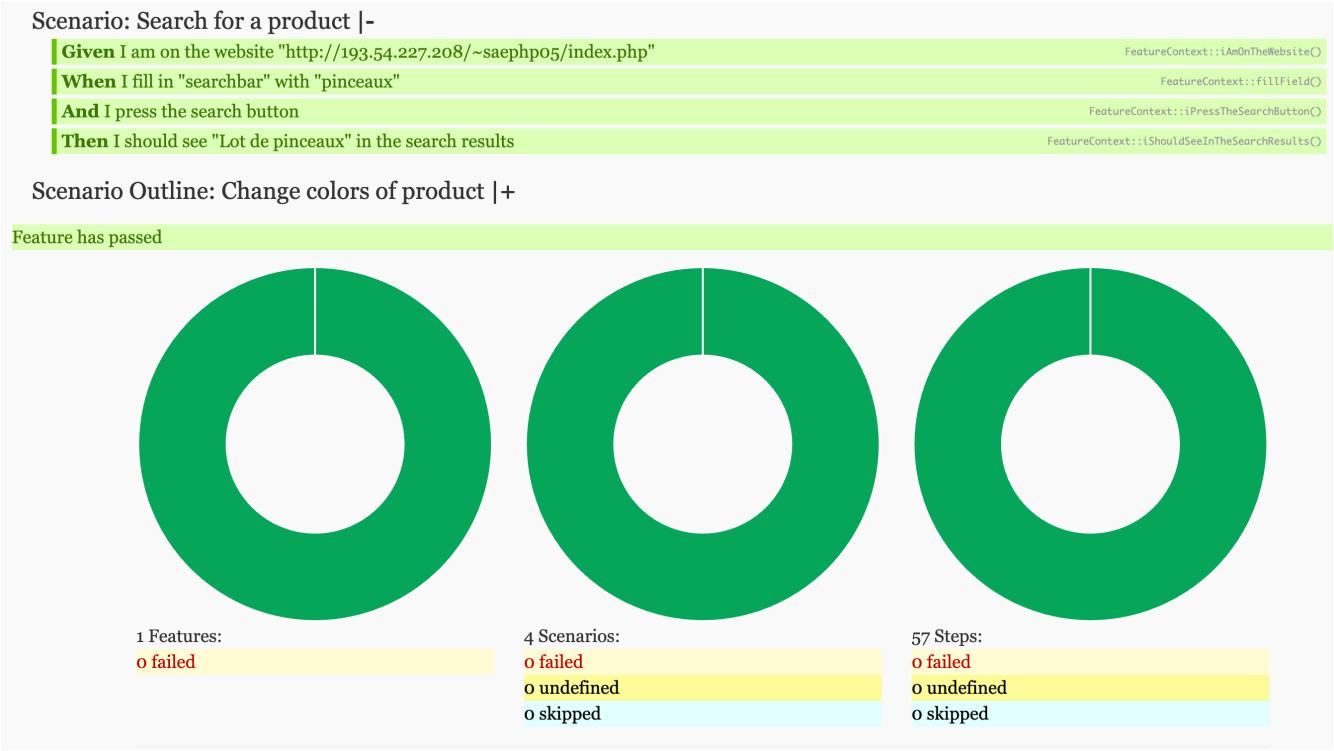
```
Feature: Website URL Check
  In order to verify that my website is accessible
  As a user
  I need to be able to access the website URL

  Scenario: Check if the website URL is accessible                    # features\test.feature:6
    Given I am on the website "http://193.54.227.208/~saephp05/index.php" # FeatureContext::iAmOnTheWebsite()
    Then I should see the website loaded successfully                 # FeatureContext::iShouldSeeTheWebsiteLoadedSuccessfully()

  Scenario Outline: Check if the page works     # features\test.feature:10
    Given I am on the website "<urlDepart>"     # FeatureContext::iAmOnTheWebsite()
    When I click on the "<Lien>" link           # FeatureContext::iClickOnTheLink()
    Then I should be on the page "<urlArriver>" # FeatureContext::iShouldBeOnThePage()

    Examples:
      | urlDepart                                                          | Lien            | urlArriver                                                    |
      | http://193.54.227.208/~saephp05/index.php                          | Les mieux notés | http://193.54.227.208/~saephp05/produits.php?categorie=MieuxNotés     |
      | http://193.54.227.208/~saephp05/index.php                          | Dessins         | http://193.54.227.208/~saephp05/produits.php?categorie=Dessins        |
      | http://193.54.227.208/~saephp05/index.php                          | Matériels d'art | http://193.54.227.208/~saephp05/produits.php?categorie=Materiel dart  |
      | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=black | Matériels d'art | http://193.54.227.208/~saephp05/produits.php?categorie=Materiel dart |

  Scenario: Search for a product                                      # features\test.feature:25
    Given I am on the website "http://193.54.227.208/~saephp05/index.php" # FeatureContext::iAmOnTheWebsite()
    When I fill in "searchbar" with "pinceaux"                        # FeatureContext::fillField()
    And I press the search button                                     # FeatureContext::iPressTheSearchButton()
    Then I should see "Lot de pinceaux" in the search results         # FeatureContext::iShouldSeeInTheSearchResults()

9 scenarios (9 passed)
27 steps (27 passed)
0m1.13s (12.28Mb)
```

WARNING Il se peut que la commande vendor/bin/behat ne retourne rien. Dans ce cas, il vous faudra supprimer le dossier vendor et exécuter la commande suivante :

```
composer install
```

Affichage des résultats

Il est possible d'afficher les résultats des tests dans un fichier HTML pour une meilleure lisibilité. Pour cela, il suffit d'ouvrir le fichier `/Site_eCommerce/reports/index.html` dans votre navigateur :



Afin de mettre à jour le fichier HTML, il suffit d'exécuter la commande suivante (dans le dossier `/Site_eCommerce`) :

```
vendor/bin/behat --format html --out ./reports
```

Structure des Tests

Le premier test consiste à vérifier que le site web est accessible et que la page a été chargée avec succès.

```
Scenario: Check if the website URL is accessible
  Given I am on the website "http://193.54.227.208/~saephp05/index.php"
  Then I should see the website loaded successfully
```

```
    /**
     * @Given I am on the website :url
     */
    public function iAmOnTheWebsite($url)
    {
        $this->visit($url);
    }
```

```
    /**
     * @Then I should see the website loaded successfully
     */
    public function iShouldSeeTheWebsiteLoadedSuccessfully()
    {
        $statusCode = $this->getSession()->getStatusCode();
        if ($statusCode != 200) {
            throw new Exception("Website did not load successfully. Status code:
$statusCode");
        }
    }
```

Le deuxième test consiste à vérifier que le fonctionnement de plusieurs liens en utilisant un ensemble d'exemples pour spécifier différentes URLs de départ, textes de liens, et URLs d'arrivée attendues après le clic

```
Scenario Outline: Check if the page works
  Given I am on the website "<urlDepart>"
  When I click on the "<Lien>" link
  Then I should be on the page "<urlArriver>"

Examples:
  | urlDepart | Lien | urlArriver |
  | http://193.54.227.208/~saephp05/index.php | Les mieux notés | http://193.54.227.208/~saephp05/produits.php?categorie=MieuxNotés |
  | http://193.54.227.208/~saephp05/index.php | Mentions légales | http://193.54.227.208/~saephp05/mentionsLegales.php |
  | http://193.54.227.208/~saephp05/index.php | Promotions | http://193.54.227.208/~saephp05/produits.php?categorie=Promotions |
  | http://193.54.227.208/~saephp05/index.php | Peintures | http://193.54.227.208/~saephp05/produits.php?categorie=Peintures |
  | http://193.54.227.208/~saephp05/index.php | Dessins | http://193.54.227.208/~saephp05/produits.php?categorie=Dessins |
  | http://193.54.227.208/~saephp05/index.php | Matériels d'art | http://193.54.227.208/~saephp05/produits.php?categorie=Materiel dart |
```

```
    /**
     * @When I click on the :linkText link
     */
    public function iClickOnTheLink($linkText)
    {
        $this->clickLink($linkText);
    }
```

```
    /**
     * @Then I should be on the page :url
     */
    public function iShouldBeOnThePage($url)
    {
        $currentUrl = $this->getSession()->getCurrentUrl();
        if ($currentUrl !== $url) {
            throw new Exception("Expected to be on page '$url' but found '$currentUrl'
instead.");
        }
    }
```

Le troisième test consiste à verifier si la barre de recherche fonctionne. Pour cela on va recherche l'élément "pinceaux".

```
Scenario: Search for a product
  Given I am on the website "http://193.54.227.208/~saephp05/index.php"
  When I fill in "searchbar" with "pinceaux"
  And I press the search button
  Then I should see "Lot de pinceaux" in the search results
```

```
    /**
     * @When I press the search button
     */
    public function iPressTheSearchButton()
    {
        $button = $this->getSession()->getPage()->find('css', '#searchbutton');
        if (null === $button) {
            throw new \Exception("Le bouton de recherche n'a pas été trouvé.");
        }
        $button->press();
    }
```

```
    /**
     * @Then I should see :text in the search results
     */
    public function iShouldSeeInTheSearchResults($text)
    {
        $page = $this->getSession()->getPage();
        $searchResults = $page->find('css', '#produits');
        if (null === $searchResults) {
            throw new \Exception("La zone de résultats de recherche n'a pas été
trouvée.");
        }

        if (strpos($searchResults->getText(), $text) === false) {
            throw new \Exception("Le texte '$text' n'a pas été trouvé dans les
résultats de recherche.");
        }
    }
```

Le quatrième test consiste à vérifier si les liens pour changer la couleur des produits marchent. Pour cela, on va utiliser un Scénario Outline pour parcourir tous les cas possibles rapidement et efficacement.

```
Scenario Outline: Change colors of product
  Given I am on the website "http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=black"
  When I click on couleur case "<color>"
  Then I should be on the page "<urlArriver>"

Examples:
  | color | urlArriver |
  | black | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=black |
  | blue | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=blue |
  | brown | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=brown |
  | green | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=green |
  | orange | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=orange |
  | pink | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=pink |
  | purple | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=purple |
  | red | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=red |
  | white | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=white |
  | yellow | http://193.54.227.208/~saephp05/produit.php?reference=ART1&couleur=yellow |
```

```
    /**
     * @When I click on couleur case :couleur
     */
    public function iClickOnCouleurCase($couleur)
    {
        $xpath = sprintf('//a[div[@class="square" and contains(@style, "background-
color:%s;")]]', $couleur);
        $element = $this->getSession()->getPage()->find('xpath', $xpath);

        if ($element) {
            $element->click();
        } else {
            throw new \RuntimeException(sprintf('Link containing square with color %s
not found', $couleur));
        }
    }
```