# **Missions :**

# test\_app.py

##### `test\_home\_without\_login` : pour s'assurer qu’on ne peut pas être rooter vers la page Home (`conversion`) sans *a priori* être connecté ?

**Pour vous aider :**

* C’est un test où on montre qu’une Qry Get ne devrait jamais rooter sur la page Home si le User ne s’est pas loggé auparavant
* Voir dans app.py, la fonction index(), la condition if de la ligne 20

##### `test\_register\_user\_success` : pour s'assurer que créer un User fonctionne bien et que nous allons être redirigé ?

##### `test\_register\_user\_bad\_input` : pour s'assurer qu'on ne peut pas être rooter vers la page Home si dans la page Register on valide sans avoir saisi les infos de création d'un compte : *name*, *email* et *password*?

**Pour vous aider :**

* Voir app.py, la fonction register(), la condition if de la ligne 143

##### `test\_register\_user\_email\_exists`?

**Pour vous aider :**

* `Email already exists`, s'assurer qu'on ne peut pas créer un compte avec le même email ?
* Vous procédez de cette façon ds le test : créer un user, le déconnecter, recréer un compte avec le même mail et voir si le status\_code renvoi 403

##### `test\_login\_user\_success`?

**Pour vous aider :**

* Pour s'assurer que le process de création d'un compte, déconnexion et reconnexion fonctionne très bien ?

##### `test\_login\_user\_invalid\_credentials`?

**Pour vous aider :**

* `Invalid email or password` : Pour s'assurer que se connecter avec un email et un pwd erronés renvoi un status\_code 403 ?

##### `test\_login\_user\_bad\_data`?

**Pour vous aider :**

* `Veuillez renseigner ce champ` : Pour s'assurer que se connecter sans avoir saisi un email et/ou un pwd renvoi un status\_code 400 ?

##### `test\_calculation\_hex`?

**Pour vous aider :**

* `Veuillez renseigner ce champ` : Pour s'assurer que se connecter sans avoir saisi un email et/ou un pwd renvoi un status\_code 400 ?

##### `test\_calculation\_hex` :

**Pour vous aider :**

* # S'assurer que suite à la création d'un new User, ce dernier lorsqu'il cherche à convertir `A` (Hexadecimal base), la Qry Post renvoi un status\_code 200 et il obtient les bons résultats ?

##### `test\_calculation\_dec` :

**Pour vous aider :**

* S'assurer que suite à la création d'un new User, ce dernier lorsqu'il cherche à convertir `10` (Decimal base), la Qry Post renvoi un status\_code 200 et il obtient les bons résultats ?

##### `test\_calculation\_bin` :

**Pour vous aider :**

* Pour s'assurer que suite à la création d'un new User, ce dernier lorsqu'il cherche à convertir `1010` (Binary base), la Qry Post renvoi un status\_code 200 et il obtient les bons résultats ?

##### `test\_calculation\_bad\_data` :

**Pour vous aider :**

* `Please enter valid input` : pour s'assurer que suite à la création d'un new User, ce dernier lorsqu'il cherche à convertir `200` (Binary base) [donnée erronée], la Qry Post renvoi un status\_code 403 ?

##### `test\_calculation\_bad\_data` :

**Pour vous aider :**

* Pour s'assurer que suite à la création d'un new User, ce dernier lorsqu'il cherche à convertir `10` (Hexadecima base) [base erronée : `10A`, au-delà de la liste déroulante], la Qry Post renvoi un status\_code 400 ?

##### `test\_history:

**Pour vous aider :**

* Pour s'assurer que suite à la création d'un new User, ce dernier après avoir réalisé des opé de conversion, on retrouve celles-ci bien présentes dans le dashboard ?

# test\_user.py

##### `test\_registration\_with\_unique\_email` : pour s'assurer que l’inscription est possible pour tout nouvel User (avec un email unique) ?

**Pour vous aider :**

* Vous pouvez, par exemple, créer une classe `TestUser` qui permet d’instancier un new user avec `self.name`, `self.email`, `self.password` et de réaliser dedans le test `test\_registration\_with\_unique\_email`
* Vous devez utiliser la méthode `user.register`

##### `test\_registration\_with\_non\_unique\_email` : pour s'assurer que l’inscription n’est plus possible pour un User déjà enregistré (email n’est pas unique) ?

**Pour vous aider :**

* Toujours à l’aide de la même classe `TestUser` et du même user (donc à l’aide de son email), vous rajoutez dedans le test `test\_registration\_with\_non\_unique\_email`
* Vous tester si `user.register` retourne None

##### `test\_is\_email\_exists\_with\_unique\_email` : pour s'assurer du bon fonctionnement de la méthode is\_email\_exists (d'une façon unitaire) ?

**Pour vous aider :**

* Toujours au sein de la même classe `TestUser` et du même user (donc à l’aide de son email), vous rajoutez le test `test\_is\_email\_exists\_with\_unique\_email`
* Vous tester si la Qry SELECT fonctionne très bien ?

##### `test\_is\_email\_exists\_with\_non\_unique\_email` : test réciproque du dernier test pour s'assurer si un email existe déjà ?

**Pour vous aider :**

* Toujours au sein de la classe `TestUser, vous tester si la Qry retourne None pour un nouvel email ; `test\_is\_email\_exists\_with\_non\_unique\_email`

##### `test\_login\_correct\_email\_correct\_password` : pour s'assurer que la connexion pour un User dont l’email et le pwd sont valides fonctionnement très bien ?

**Pour vous aider :**

* Toujours au sein de la même classe `TestUser`, vous rajoutez le test `test\_login\_correct\_email\_correct\_password` qui fait appel à la méthode `user\_login` ?

##### `test\_login\_correct\_email\_incorrect\_password` : (complément du dernier test) pour s'assurer que la connexion pour un User dont l’email est correcte et le pwd est incorrecte retourne None ?

##### `test\_login\_incorrect\_email\_correct\_password` : (complément du test) pour s'assurer que la connexion pour un User dont l’email est incorrecte et le pwd est correcte retourne None ?

##### `test\_login\_incorrect\_email\_correct\_password` : (complément du test) pour s'assurer que la connexion pour un User dont l’email est incorrecte et le pwd est incorrecte retourne None ?

# test\_calculation.py

##### `test\_get\_user\_history\_empty\_state` : pour s'assurer que le dashboard DB est vide pour tout new User enregistré sur le site ?

**Pour vous aider :**

* Pour rappel, la DB contient l'historique des opé de conversions réalisées par tt User.
* Il faut alors créer un new User et vérifier si sa data est initialement vide (historique de conversion).
* Vous pouvez, par exemple, créer une classe `TestCalculation` qui permet d’instancier un new user, l’enregistrer, le connecter au site et réaliser dedans le test `test\_get\_user\_history\_empty\_state`
* Vous devez utiliser les méthodes suivantes : `user.register`, `user.login`, `calculation.get\_user\_history`, …

##### `test\_get\_user\_history\_non\_empty\_state` : pour s'assurer que suite à une opération de conversion (par un new User), le dashboard n’est pas vide et l’historique est affichable ?

**Pour vous aider :**

* Vous pouvez augmenter la classe `TestCalculation` par ce new test : `test\_get\_user\_history\_non\_empty\_state` ; ce qui permet d’instancier un new user, l’enregistrer et le connecter au site.
* Vous devez utiliser les méthodes suivantes : `calculation.add\_calculation`, `calculation.get\_user\_history`

##### `test\_get\_user\_history\_non\_empty\_state\_bad\_id` : pour s'assurer que si on demande d'afficher l’historique des opérations d'un User dont l'Id est incorrecte, le dashboard est vide ?

**Pour vous aider :**

* Il suffit de passer par `calculation.get\_user\_history(wrong\_id)` en indiquant un mauvais id.

##### `test\_add\_calculation` : pour s'assurer que si un User réalise une opé de conversion, celle-ci est bien stockée dans la DB ?

**Pour vous aider :**

* Il suffit de vérifier que la méthode `calculation.add\_calculation` retourne bien 1.
* Remarquer que cette méthode réalise un `rollback` si une exception est levée ; vous pouvez réaliser un test dans ce sens.