**Plan de test – Fonction associerBadge**

**1 - Identification du test**

**Nom** : Test d'association d’un badge à un utilisateur  
**Numéro** : T11

**2 - Référence du module testé**

routes/pgs.py – Fonction associerBadge

**3 - Objectif du test**

Valider que la route permet d’associer un badge à un utilisateur uniquement si :

* Les deux existent
* Le badge n’est pas déjà attribué
* L’utilisateur ne possède pas déjà un autre badge

Et gérer toutes les erreurs logiques possibles.

**4 - Procédure du test**

* **Initialisation** : Créer ou non un utilisateur et un badge dans la base
* **Lancement** : Requête PUT sur /pgs/associer/utilisateur/{id\_utilisateur}/badge/{uid\_badge}  
  avec un JSON de type :

{

"uid": "123ABC45",

"id\_utilisateur": 2

}

* **Observation** : Vérifier le code de réponse et le retour JSON ou l’erreur

**5 - Résultats attendus**

| **N° Test** | **Condition** | **BDD Préparée** | **Résultat attendu** | **Statut attendu** |
| --- | --- | --- | --- | --- |
| 1 | Utilisateur inexistant | Aucun utilisateur avec id\_utilisateur donné | 404 Utilisateur non trouvé | Erreur |
| 2 | Badge inexistant | Aucun badge avec cet uid | 404 Badge non trouvé | Erreur |
| 3 | Badge déjà attribué | Badge avec id\_utilisateur déjà renseigné | 400 Ce badge est déjà attribué à un utilisateur | Erreur |
| 4 | Utilisateur a déjà un badge | Autre badge déjà lié à l’utilisateur | 400 Cet utilisateur possède déjà un badge | Erreur |
| 5 | Association réussie | UID libre + utilisateur existant sans badge | 200 Badge retourné avec uid et id\_utilisateur | Succès |

**6 - Moyens à mettre en œuvre**

* **Logiciels** : FastAPI, SQLAlchemy, Pytest
* **Matériel** : Poste de développement, base locale
* **Préconditions** :
  + Table Badge opérationnelle
  + Table Utilisateur remplie avec ID valides
  + Modèle AssoRequest bien défini dans schemas