**Plan de test – Fonction postBadge**

**1 - Identification du test**

**Nom** : Test de création de badge  
**Numéro** : T8

**2 - Référence du module testé**

routes/pgs.py – Fonction postBadge

**3 - Objectif du test**

Vérifier que la création d’un badge fonctionne correctement :

* Ajout d’un badge si les infos sont valides
* Refus de création si l’UID est déjà pris
* Refus de création si l’utilisateur associé n’existe pas

**4 - Procédure du test**

* **Initialisation** : Ajouter ou non des badges existants et utilisateurs en base
* **Lancement** : Requête POST sur /badge/ avec un JSON de type :

{

"uid": "A1B2C3D4",

"actif": true,

"id\_utilisateur": 5

}

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

**5 - Résultats attendus**

| **N° Test** | **Condition** | **BDD Préparée** | **Résultat attendu** | **Statut attendu** |
| --- | --- | --- | --- | --- |
| 1 | UID déjà existant | Un badge avec le même uid existe | 400 Badge déjà enregistré | Erreur |
| 2 | Utilisateur inexistant | Aucun utilisateur avec id\_utilisateur donné | 400 Utilisateur introuvable | Erreur |
| 3 | Création réussie | UID nouveau + utilisateur valide | 200 Retour du badge avec uid, actif, date | Succès |

**6 - Moyens à mettre en œuvre**

* **Logiciels** : FastAPI, SQLAlchemy, Pytest
* **Matériel** : Poste de développement avec base locale ou conteneurisée
* **Préconditions** :
  + Table Badge bien initialisée
  + Structure BadgeCreate définie dans schemas.py
  + Vérification de l’utilisateur possible dans Utilisateur