Cahier de Tests - Lab 2 : Tests d'Interface Web avec Robot Framework

Informations générales

- **Projet** : Tests automatisés d'interface web avec Robot Framework et SeleniumLibrary
- **Application testée** : CRM Automation Playground
- **URL** : https://automationplayground.com/crm/
- **Framework de test** : Robot Framework
- **Bibliothèque** : SeleniumLibrary
- **Navigateur** : Chrome/Firefox (configuré via variables)

Entités testées

- **Page d'accueil** : Vérification du chargement et contenu
- **Authentification** : Login/Logout avec différents scénarios
- **Gestion des clients** : Navigation et ajout de nouveaux clients

---

1. TESTS DE LA PAGE D'ACCUEIL

1.1 Chargement de la page d'accueil

TC_HOME_001 - Chargement de la page d'accueil

- **Objectif** : Vérifier que la page d'accueil se charge correctement
- **Prérequis** : Navigateur web disponible et connexion internet
- **Type** : Test de fumée (Smoke Test)
- **Keyword Robot Framework** : `User examines home page contents`

Données de test :

- URL : ${URL} (configuré dans variables.py)
- Navigateur : ${BROWSER} (configuré dans variables.py)
- Titre attendu : ${HomeTitle}

Étapes d'exécution :

1. Ouvrir le navigateur web
2. Naviguer vers l'URL de l'application
3. Maximiser la fenêtre du navigateur

4. Vérifier la présence du titre de la page d'accueil
5. Examiner le contenu de la page

Résultats attendus :

- La page d'accueil se charge sans erreur
- Le titre ${HomeTitle} est présent sur la page
- Les éléments de navigation sont visibles
- Aucune erreur JavaScript dans la console

---

2. TESTS D'AUTHENTIFICATION

2.1 Authentification réussie

TC_AUTH_001 - Connexion avec des identifiants valides

- **Objectif** : Vérifier qu'un utilisateur peut se connecter avec des identifiants valides
- **Prérequis** : Page d'accueil chargée, identifiants valides disponibles
- **Type** : Test de fumée (Smoke Test)
- **Keywords Robot Framework** : `Click Login Link`, `Type Valide Credentials`

Données de test :

- Login : ${LOGIN} (configuré dans variables.py)
- Mot de passe : ${PWD} (configuré dans variables.py)
- Lien de connexion : "Sign In"
- Champs de saisie : id=email-id, id=password
- Bouton de soumission : id=submit-id

Étapes d'exécution :

1. Cliquer sur le lien "Sign In"
2. Vérifier que la page de connexion se charge (contient "Login")
3. Saisir l'email valide dans le champ id=email-id
4. Saisir le mot de passe valide dans le champ id=password
5. Cliquer sur le bouton de soumission (id=submit-id)
6. Attendre le chargement de la page suivante

Résultats attendus :

- La page de connexion se charge correctement

- Les identifiants sont acceptés
- Redirection vers la page "Customers"
- Le lien "Sign Out" devient visible
- Message de bienvenue ou indication de connexion réussie

2.2 Authentification échouée

TC_AUTH_002 - Connexion avec des identifiants vides

- **Objectif** : Vérifier que la connexion échoue avec des identifiants manquants
- **Prérequis** : Page d'accueil chargée
- **Type** : Test fonctionnel
- **Keywords Robot Framework** : `Click Login Link`, `Type Empty Credentials`

Données de test :

- Email vide : ${EMPTY_EMAIL} (chaîne vide)
- Mot de passe vide : ${EMPTY_PASSWORD} (chaîne vide)
- Page attendue après échec : contient "Login"

Étapes d'exécution :

1. Cliquer sur le lien "Sign In"
2. Vérifier que la page contient "Customer Service"
3. Saisir des identifiants vides (force True pour champs vides)
4. Cliquer sur le bouton de soumission
5. Vérifier que l'utilisateur reste sur la page de connexion

Résultats attendus :

- L'utilisateur reste sur la page de connexion
- Message d'erreur affiché (si applicable)
- Pas de redirection vers la page clients
- La page contient toujours "Login"

2.3 Fonctionnalité "Remember Me"

TC_AUTH_003 - Persistance de l'email avec "Remember Me"

- **Objectif** : Vérifier que la case "Remember me" sauvegarde l'adresse email
- **Prérequis** : Page d'accueil chargée, identifiants valides
- **Type** : Test fonctionnel

- **Keywords Robot Framework** : `Type Valid Credentials`, `Click Remember Me Checkbox`, `Check Email Prepopulated`

Données de test :

- Case à cocher : id=remember
- Champ email : id=email-id
- Lien de déconnexion : "Sign Out"

Étapes d'exécution :

1. Se connecter avec des identifiants valides
2. Cocher la case "Remember Me" (id=remember)
3. Cliquer sur le bouton de soumission
4. Attendre le chargement de la page "Customers"
5. Cliquer sur le lien "Sign Out"
6. Attendre la page "Signed Out"
7. Retourner à la page de connexion
8. Vérifier que l'email est pré-rempli

Résultats attendus :

- Connexion réussie avec case cochée
- Déconnexion réussie
- Retour à la page de connexion
- Le champ email contient la valeur ${LOGIN}
- L'attribut "value" du champ id=email-id correspond à ${LOGIN}

---

3. TESTS DE GESTION DES CLIENTS

3.1 Navigation vers la page Customers

TC_CUSTOMERS_001 - Navigation vers la page des clients après connexion

- **Objectif** : Vérifier l'accès à la page de gestion des clients
- **Prérequis** : Utilisateur authentifié
- **Type** : Test de fumée (Smoke Test)
- **Keywords Robot Framework** : `Navigate to Customers Page`

Données de test :

- Localisateur page clients : locator=customers
- Page attendue : contient "Customers"

Étapes d'exécution :

1. Se connecter avec des identifiants valides
2. Vérifier la redirection automatique vers la page clients
3. Confirmer la présence de l'élément locator=customers

Résultats attendus :

- Redirection automatique après connexion
- Page "Customers" chargée correctement
- Élément identifiant la page clients présent
- Interface de gestion des clients accessible

3.2 Ajout d'un nouveau client

TC_CUSTOMERS_002 - Ajout d'un nouveau client complet

- **Objectif** : Vérifier la création complète d'un nouveau client
- **Prérequis** : Utilisateur connecté sur la page clients
- **Type** : Test de fumée (Smoke Test)
- **Keywords Robot Framework** : `Add New Customer`, `Type Customer Email`, etc.

Données de test :

- Bouton "Add Customer" : id=${AddCustomerBtnId}
- Email : ${EmailAddress}
- Prénom : ${FirstName}
- Nom : ${LastName}
- Ville : ${City}
- État : ${State}
- Genre : ${Gender}
- Case promotion : ${Promotion_Checkbox}
- Bouton soumission : ${Submit_Button}

Étapes d'exécution :

1. Cliquer sur le lien "Add Customer" (id=${AddCustomerBtnId})
2. Vérifier que la page contient ${AddCustomerTitle}

3. Remplir le champ email (id=${Email_Input})
4. Remplir le prénom (id=${First_Name_Input})
5. Remplir le nom (id=${Last_Name_Input})
6. Remplir la ville (id=${City_Input})
7. Sélectionner l'état dans la liste (id=${State_Input})
8. Sélectionner le genre via bouton radio (${Gender_Input})
9. Optionnellement cocher la case promotion
10. Cliquer sur le bouton de soumission

Résultats attendus :

- Page d'ajout de client chargée (${AddCustomerTitle})
- Tous les champs acceptent les données saisies
- Soumission réussie du formulaire
- Page de succès affichée avec message "Success!"
- Nouveau client enregistré dans le système

3.3 Annulation d'ajout de client

TC_CUSTOMERS_003 - Annulation de l'ajout d'un nouveau client

- **Objectif** : Vérifier la fonctionnalité d'annulation lors de l'ajout d'un client
- **Prérequis** : Utilisateur connecté, page d'ajout de client ouverte
- **Type** : Test fonctionnel
- **Keywords Robot Framework** : `Click New Customer Button`, `Click Cancel Button`

Données de test :

- Bouton "New Customer" : id=${new_customer_link_id}
- Bouton "Cancel" : texte "Cancel"
- Page de retour attendue : contient "Our Happy Customers"

Étapes d'exécution :

1. Se connecter avec des identifiants valides
2. Cliquer sur le bouton de connexion
3. Cliquer sur le bouton "New Customer"
4. Attendre 5 secondes pour le chargement
5. Vérifier que la page contient "Add Customer"
6. Cliquer sur le bouton "Cancel"

7. Vérifier le retour à la page précédente
8. Attendre 10 secondes pour confirmation

Résultats attendus :

- Page d'ajout de client s'ouvre correctement
- Bouton "Cancel" fonctionnel
- Retour à la page des clients existants
- Page contient "Our Happy Customers"
- Aucune donnée de client créée

---

4. CONFIGURATION DES TESTS

4.1 Prérequis techniques

Environnement :

- Robot Framework installé
- SeleniumLibrary installée
- ChromeDriver ou GeckoDriver selon le navigateur
- Python 3.x
- Navigateur web (Chrome/Firefox)

Variables de configuration :

- **URL** : https://automationplayground.com/crm/
- **BROWSER** : chrome/firefox (configuré dans variables.py)
- **LOGIN** : identifiants valides
- **PWD** : mot de passe valide
- **HomeTitle** : titre de la page d'accueil
- Localisateurs d'éléments dans locator.py

4.2 Structure des fichiers

Fichiers de test :

- `test_case1.robot` : Test de chargement de la page d'accueil
- `test_case2.robot` : Test de connexion réussie
- `test_case3.robot` : Tests d'authentification (échec, Remember Me, navigation, ajout client)
- `test_case7.robot` : Test d'annulation d'ajout de client

Fichiers de ressources :

- `keywords1.robot` : Keywords pour page d'accueil
- `keywords2.robot` : Keywords pour authentification réussie
- `keywords3.robot` : Keywords pour tests d'authentification avancés
- `keyword7.robot` : Keywords pour annulation d'ajout client
- `../ressource/variables.py` : Variables de configuration Python
- `../ressource/locator.py` : Localisateurs d'éléments

4.3 Setup et Teardown

Suite Setup :

- Ouverture du navigateur web (`Open web browser`)
- Navigation vers l'URL de l'application
- Maximisation de la fenêtre du navigateur

Suite Teardown :

- Fermeture du navigateur (`Close Web Browser`)
- Nettoyage des sessions en cours

Test Cleanup :

- Attentes explicites pour stabilité (Sleep)
- Vérifications post-conditions
- Gestion des états de session

---

5. DONNÉES DE TEST

5.1 Données valides

Identifiants de connexion :

- Email : ${LOGIN} (depuis variables.py)
- Mot de passe : ${PWD} (depuis variables.py)

Informations client :

- Email : ${EmailAddress}
- Prénom : ${FirstName}
- Nom : ${LastName}
- Ville : ${City}

- État : ${State}
- Genre : ${Gender}

5.2 Données invalides

Identifiants vides :

- Email : ${EMPTY_EMAIL} (chaîne vide)
- Mot de passe : ${EMPTY_PASSWORD} (chaîne vide)

5.3 Localisateurs d'éléments

- Boutons : id=${AddCustomerBtnId}, id=${login_button_id}
- Champs de saisie : id=email-id, id=password
- Liens : "Sign In", "Sign Out", "Cancel"
- Cases à cocher : id=remember, ${Promotion_Checkbox}

---

6. RÉSULTATS ATTENDUS

6.1 Métriques de succès

Tests passants (Smoke Tests) : 4 cas de test

- 1 test de chargement de page d'accueil
- 1 test de connexion réussie
- 1 test de navigation clients
- 1 test d'ajout de client complet

Tests fonctionnels : 3 cas de test

- 1 test d'échec de connexion
- 1 test "Remember Me"
- 1 test d'annulation d'ajout client

Taux de réussite attendu : 100% (7/7 tests)

6.2 Temps d'exécution estimé

- Tests unitaires : ~2-5 secondes par test
- Attentes explicites : 5-10 secondes selon les tests
- Suite complète : ~2-3 minutes

---

7. EXÉCUTION DES TESTS

7.1 Commandes d'exécution

Exécution complète :

```bash
robot tests/
```

Exécution par test :

```bash
robot test_case1.robot    # Page d'accueil
robot test_case2.robot    # Connexion réussie
robot test_case3.robot    # Tests d'authentification
robot test_case7.robot    # Annulation ajout client
```

Exécution avec tags :

```bash
robot --include smoke tests/      # Tests de fumée uniquement
robot --include functional tests/ # Tests fonctionnels uniquement
```

7.2 Rapports générés

Fichiers de sortie :

- `report.html` : Rapport détaillé des tests avec captures d'écran
- `log.html` : Log détaillé de l'exécution avec actions Selenium
- `output.xml` : Résultats au format XML pour intégration CI/CD

Captures d'écran :

- Automatiques en cas d'échec de test
- Screenshots des étapes critiques
- Preuves visuelles des résultats attendus

CONCLUSION

Ce cahier de tests couvre l'ensemble des fonctionnalités critiques de l'application CRM avec Robot Framework et SeleniumLibrary. Il respecte les exigences du Lab 2 en proposant :

- **Tests de fumée (Smoke Tests)** : Validation des fonctionnalités de base
- **Tests fonctionnels** : Validation des scénarios d'usage avancés

- **Gestion des erreurs** : Tests avec données invalides ou actions d'annulation
- **Couverture complète** : De l'authentification à la gestion des clients

Chaque test est conçu pour être stable, reproductible et maintenir l'isolation entre les cas de test grâce aux configurations de Setup et Teardown appropriées.