TP4 – Mock LocalStorage et Mock API avec Playwright

Objectifs

- Manipuler le **localStorage** pour simuler des données persistées dans le navigateur
- Intercepter et **mock** des requêtes API HTTP avec Playwright
- Comprendre l'intérêt du mocking pour des tests E2E stables et reproductibles

1. Introduction

Le **mocking** est une technique utilisée pour isoler une application de ses dépendances externes (API, base de données, stockage local, etc.).
Dans ce TP, nous aborderons :

1. Le **mock du localStorage** avec l'application TodoMVC
2. Le **mock d'une API HTTP** sur `https://reqres.in`

2. Préparation du projet

```
mkdir tp-mock
cd tp-mock
npm init playwright@latest
```

3. Mock du LocalStorage – TodoMVC

3.1 Objectif

Simuler des **tâches existantes** dans l'application TodoMVC avant le chargement de la page.
Le site ne communique pas avec une API : toutes les données sont stockées dans le **localStorage**.

3.2 Injection de données

Créer le fichier `tests/mock.todomvc.spec.ts` :

```
import { test, expect } from '@playwright/test';

test('mock localStorage avec tâches existantes', async ({ page }) => {
  // Injecter un jeu de données avant que la page ne se charge
  await page.addInitScript(() => {
    const mockedTodos = [
      { title: 'Acheter du pain', completed: false },
      { title: 'Préparer le repas', completed: true },
      { title: 'Lire la documentation Playwright', completed: false }
    ];
    localStorage.setItem('react-todos', JSON.stringify(mockedTodos));
  });

  await page.goto('https://demo.playwright.dev/todomvc');

  // Vérification des tâches visibles dans l'interface
  await expect(page.getByText('Acheter du pain')).toBeVisible();
  await expect(page.getByText('Préparer le repas')).toBeVisible();
  await expect(page.getByText('Lire la documentation Playwright')).toBeVisible();
});
```

Ce test injecte directement des données dans le localStorage **avant** que la page ne se charge, simulant ainsi un état préexistant.

```
npx playwright test tests/mock.todomvc.spec.ts --headed --project=chromium
```

3.4 Exercice – LocalStorage

1. Injecter 4 tâches avant chargement de la page TodoMVC
2. Marquer la 3e tâche comme complétée dans le localStorage
3. Supprimer la 1re tâche
4. Vérifier que la liste affichée correspond bien au contenu simulé

4. Mock d'API HTTP – ReqRes

4.1 Objectif

Intercepter les appels API et retourner des **réponses simulées** sans dépendre du backend réel.

4.2 Création du test

Créer `tests/reqres.users.spec.ts` :

```
import { test, expect } from '@playwright/test';

test('mock de la liste des utilisateurs', async ({ page }) => {
  await page.route('**/api/users?page=2', async route => {
    await route.fulfill({
      status: 200,
      contentType: 'application/json',
      body: JSON.stringify({
        data: [
          { id: 1, first_name: 'Jean', last_name: 'Dupont', email: 'jean.dupont@example.com' },
          { id: 2, first_name: 'Claire', last_name: 'Martin', email: 'claire.martin@example.com' }
        ]
      })
    });
  });

  await page.goto('https://reqres.in/');
  await page.click('text=List Users');
  await expect(page.getByText('Jean')).toBeVisible();
  await expect(page.getByText('Claire')).toBeVisible();
  await page.pause();
  await page.unroute('**/api/users?page=2');
  await page.pause();
});
```

Ce test intercepte la requête `GET /api/users?page=2` et renvoie une réponse simulée avec deux utilisateurs fictifs.

4.3 Exercice – Mock API

Intercepter les differentes routes du swagger sur https://reqres.in/api-docs/

5. Bonnes pratiques

- Toujours isoler les mocks par test (`page.unroute()` après usage)
- Nommer les fichiers et fonctions de test de manière explicite
- Vérifier la cohérence des données simulées (clé du localStorage, format JSON, etc.)

6. Conclusion

À la fin de ce TP, vous devez être capables de :

- Injecter et manipuler des données dans le `localStorage`
- Intercepter et simuler des requêtes API avec `page.route()`
- Tester une application sans dépendre d'un backend réel
- Concevoir des tests plus rapides et stables grâce au **mocking**