**CHAPITRE 3 : REALISATION**

**Mise en place du système de test automatisés**

**Développement des Cas de Test avec Cypress : Exemple de scripts de test, structure du code, et organisation des fichiers de test**

**Projet :** Test d’automatisation des fonctionnalités de ‘Landing Page – Bootstrap’.

**Objectif :** Assurer la qualité en validant les fonctionnalités critiques et détecter les anomalies si présentes. De cette base adapter le système pour tout autre type de projet.

**Outil utilisé :** Cypress (version 13.15.0)

Dans ce guide qualitativement structuré, nous verrons étape par étape les preuves de notre réalisation. L’action primordiale ici serait d’abord de configurer notre fichier *cypress.config.js* dans lequel on pointera l’url de l’application avec laquelle on souhaite travailler.

const { defineConfig } = require("cypress");

const fs = require('fs');

const path = require('path');

module.exports = defineConfig({

  projectId: '6nf6ro',

  projectID : 'cypadw ',

  e2e: {

    setupNodeEvents(on, config) {

baseUrl: 'http://127.0.0.1:3000/dist/index.html',

  },

}) ;

Pour des besoins un peu plus esthétiques, on va se rendre dans le répertoire *C:\Users\...\cypress\support\e2e.js* et taper la commande :

beforeEach(() => {

  cy.log('Test Start');

});

Elle ne sert qu’à afficher le message ‘Test Start’ au début de chaque test.

À présent nous pouvons nous diriger activement dans notre répertoire de tests *C:\Users\...\cypress\e2e\test.cy.js.*

/// <reference types="cypress" />

Définissons tout d’abord un plan de tests à effectuer.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Nom du test | Description | Type de test | Priorité |
| TC1 | Accès et chargement de Landing Page | Accéder à l’URL et vérifier que tous les éléments de la page sont chargés et s’affichent correctement | Interface utilisateur | Haute |
| TC2 | Vérification de la barre de navigation | Vérifier l’affichage et l’état de la barre de navigation | Interface utilisateur |  |
| TC3 | Vérification de la validation de l’email | Teste le champ d’entrée de l’adresse e-mail et vérifie les contraintes liées à la soumission du formulaire | Unitaire, interface utilisateur, fonctionnel, intégration | Haute |
| TC4 | Vérification de l’affichage des fonctionnalités de l’application. | Vérifie que les icones des fonctionnalités s’affichent correctement ainsi que leur contenu | Interface utilisateur |  |
| TC5 | Vérification de la section de témoignages | Vérifie l’affichage, l’authenticité des témoins et de leur parole | Interface utilisateur |  |
| TC6 | Vérifications du footer | Vérifie la présence effective de tous les éléments du footer ainsi que la redirection des différents liens | Interface utilisateur, unitaire, fonctionnel, acceptation utilisateur |  |

Définissons notre premier test que nous allons nommer ‘Landing Page Test’ qui servira à ouvrir l’application dans notre environnement de tests.

describe('Landing Page Tests', () => {

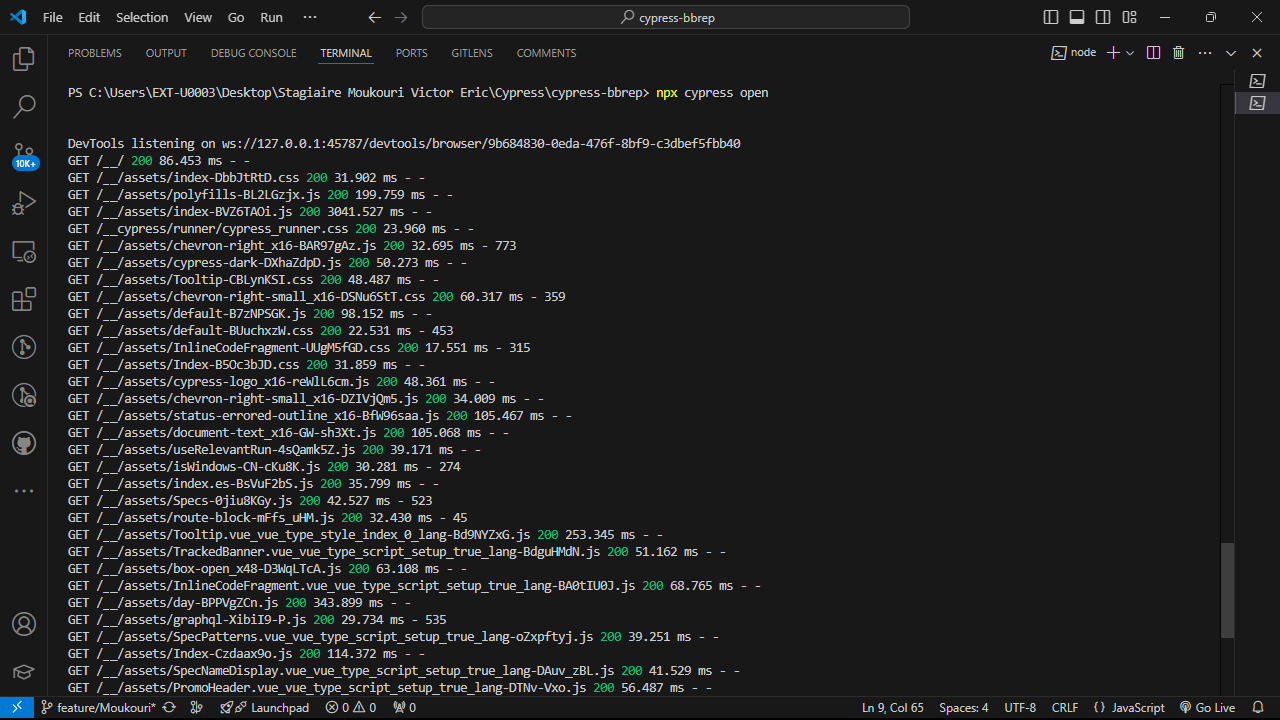
    beforeEach(() => {

        // Aller sur la page de test avant chaque cas de test

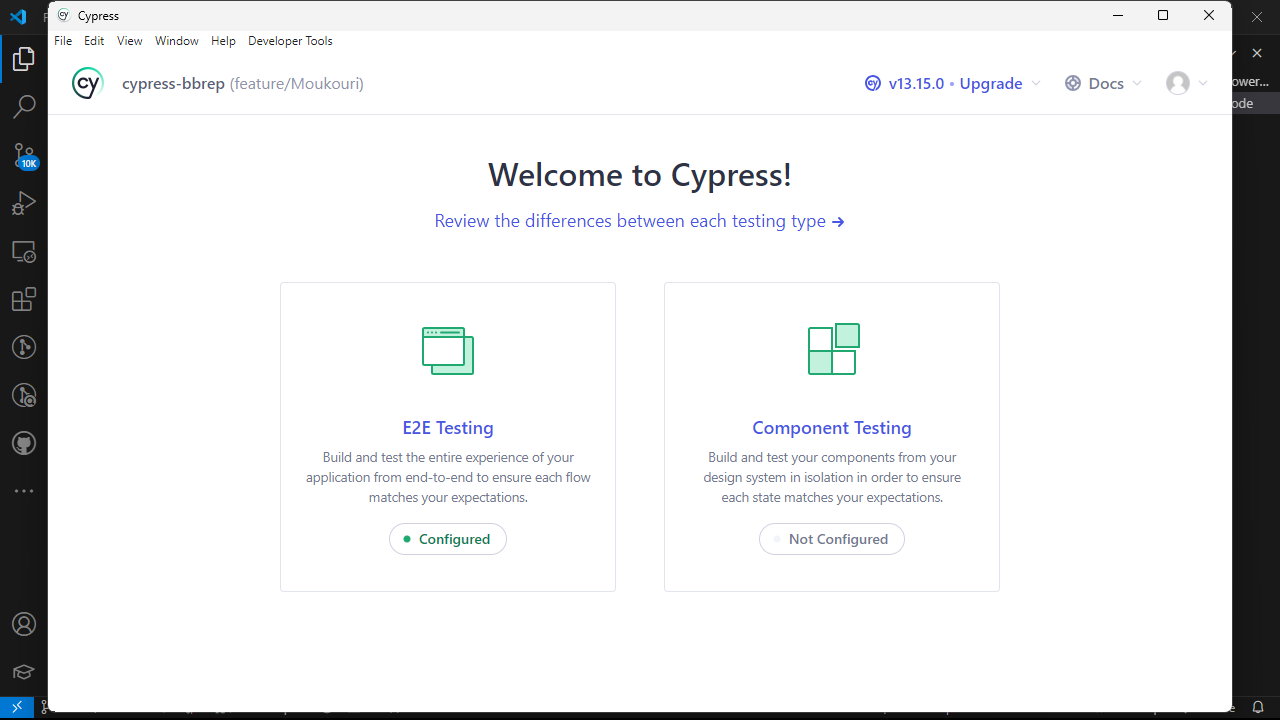
        cy.visit('http://127.0.0.1:3000/Bootstrap/startbootstrap-landing-page/dist/index.html') // Permet d'accéder au projet à partir de son url.

    })

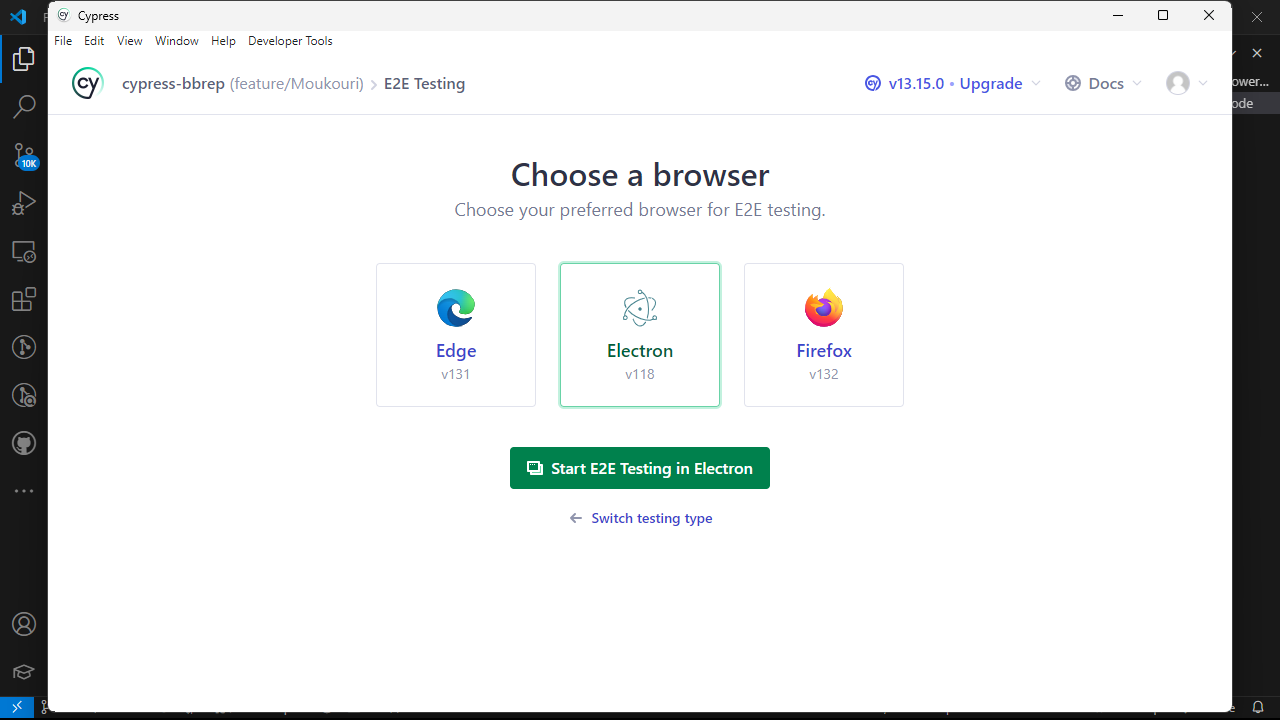
Ouvrons ensuite cypress avec la commande npx cypress open.



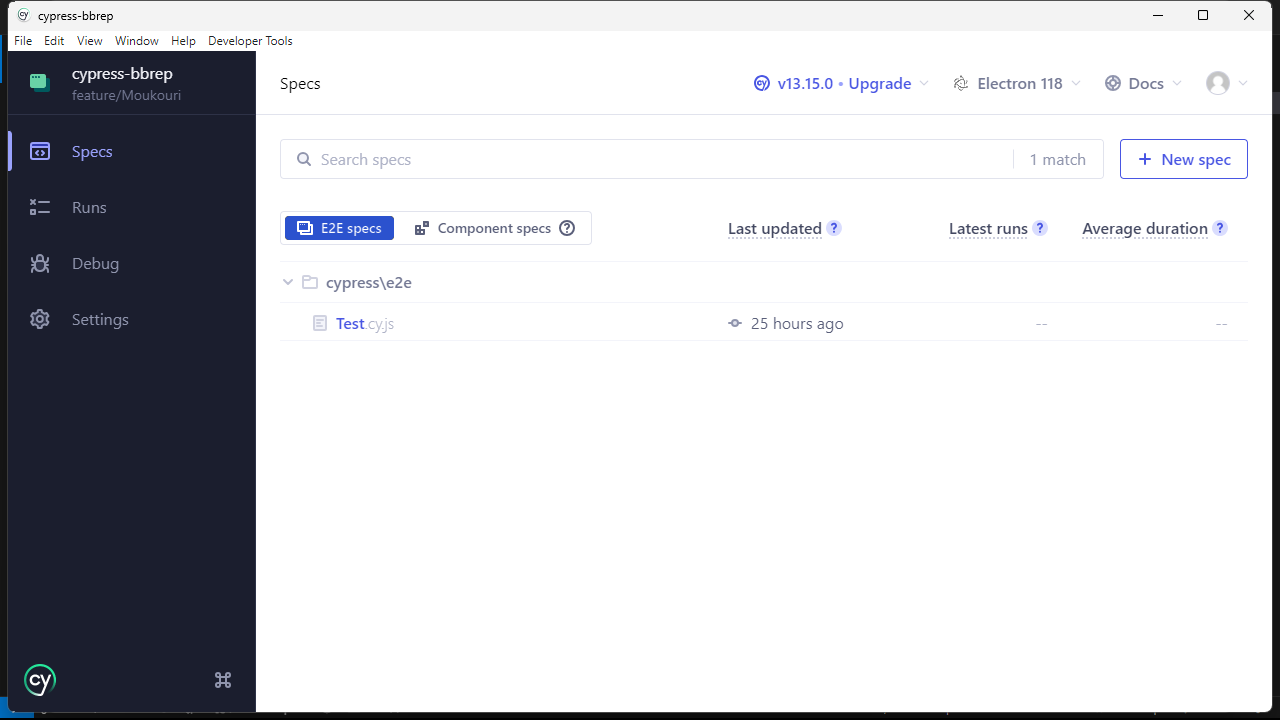
Nous sommes désormais sur le GUI de Cypress et nous pouvons avancer. Nous nous concentrerons premièrement dans les tests e2e et sans plus tarder passons à la configuration…

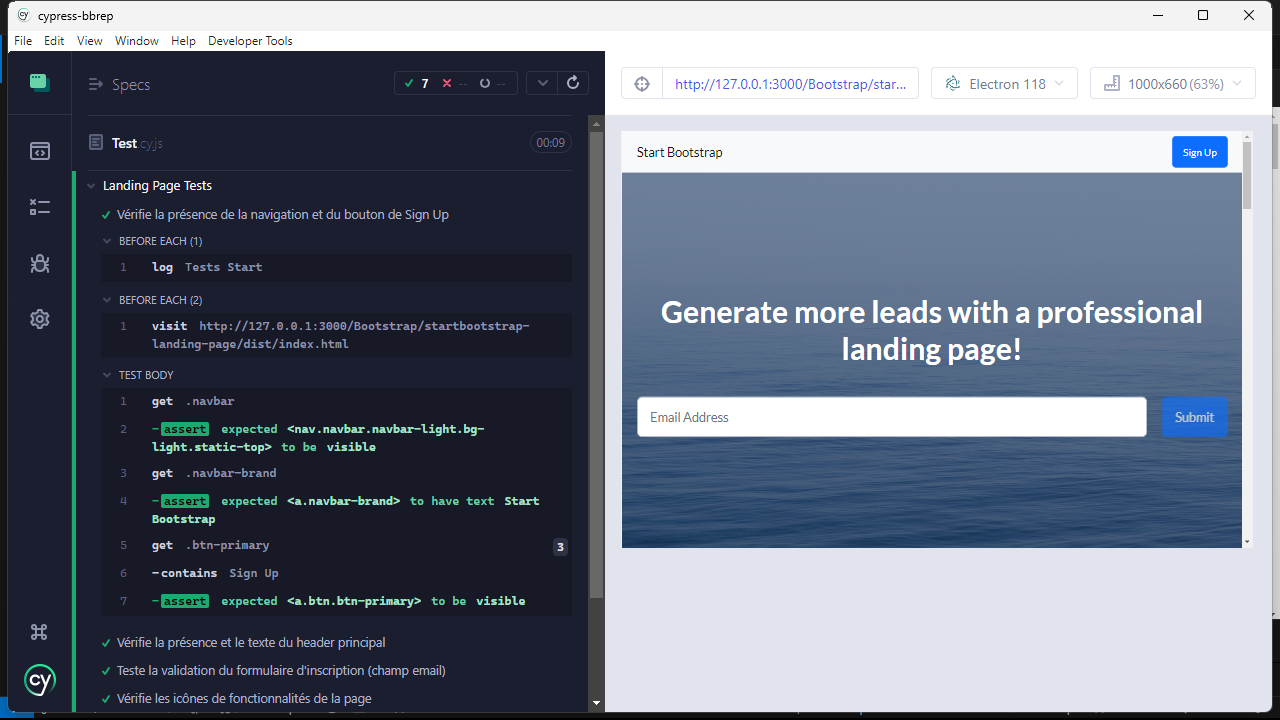


Cypress propose d’effectuer des tests avec les navigateurs que vous possédez. Electron est de base intégré à l’outil. Nous le choisissons donc.



Voici notre répertoire de tests e2e. Pour lancer le test il n’y a qu’à cliquer sur le fichier de test.





Notre premier test a été réalisé avec succès et on peut l’apercevoir avec la console à notre gauche. Si un test n’est pas valide ils s’afficheront en rouge et nous aurons un message informatif sur l’erreur.