**Proiect**

**Testare Automată**

# **Introducere**

Proiectul de testare automată a fost inițiat pentru a asigura funcționalitatea corectă a librăriei virtuale Carturești. Testele automate au fost concepute pentru a valida diverse caracteristici și funcționalități ale aplicației și pentru a identifica eventuale probleme sau erori.

Pentru realizarea proiectului a fost utilizată metodologia de dezvoltare software numită BDD (Behavior-Driven Development) care se concentrează pe colaborarea membrilor echipei și pe descrierea comportamentului aplicației într-un limbaj simplu, precum Gherkin și Cucumber. A fost alesă metodologia BDD pentru a facilita comunicarea între dezvoltatori, testeri și alte părți interesate și pentru a crea teste automate care reflectă comportamentul specificat în mod clar de părțile interesate.

Beneficiile BDD includ, comunicare clară, teste ușor de înțeles și actualizate, alinierea între cerințe și implementare, încurajează munca în echipă și garantează că dezvoltarea se concentrează pe crearea unor funcționalități utile care să răspundă așteptărilor utilizatorilor.

Pentru aplicarea proiectului de testare a fost utilizată atât unealta de testare Selenium WebDriver (împreună cu limbajele Python si Gherkin, în cadrul mediului de dezvoltare integrat Pycharm), cât și unealta Cypress (împreună cu limbajele JavaScript și Cucumber, în cadrul mediului de dezvoltare integrat Visiual Studio Code).

De asemnea, proiectul a fost organizat folosind structura POM (Page Object Model). Reutilizarea, încapsularea, ușurința întreținerii, lizibilitatea și rezistența la schimbare, îmbunătățirea dezvoltării/întreținerii testelor automate și a structurii codului, sunt câteva dintre beneficiile POM.

# **Descrierea Caracteristicilor**

Aplicația Carturești prezintă o serie de caracteristici și funcționalități importante, printre care:

* Pagina acasă: funcționalități de căutare, filtre, și redirecționare.
* Pagina de autentificare: verificarea corectitudinii datelor introduse în formularul de autentificare, autentificarea cu succes, remiterea mesajelor reușite/de eroare și validarea corectitudinii textului mesajului, verificarea erorii de autentificare cu date invalide;
* Pagina de înregistrare: validarea datelor introduse în formularul de înregistrare, înregistrarea cu succes, remiterea mesajelor reușite/de eroare și validarea corectitudinii textului mesajului, verificarea erorii de înregistrare cu date invalide;
* Pagina de coș de cumpărături: verificarea adăugării produselor în coș și calculul prețului total.
* Alte funcționalități: verificarea mesajelor de eroare, a traducerilor și a interfeței.

# **Descrierea Testelor**

Testele automate au acoperit următoarele scenarii și funcționalități:

* Testarea funcționalităților paginii de start, inclusiv căutare folosind cuvinte cheie, filtrele de căutare avansată pe diferite criterii (precum preț, marcă, caracteristici, disponibilitate), afișare corectă/ relevantă pentru căutarea produselor specificate, navigare/redirecționare, traducere și asistență.
* Testarea procesului de autentificare folosind date valide și invalide, prezența mesajelor de succes/eroare asociate, respectiv comportamentul platformei în situații de succes/eșec.
* Testarea procesului de înregistrare folosind date valide și invalide, prezența mesajelor de succes/eroare asociate, respectiv comportamentul platformei în situații de succes/eșec.
* Testarea funcționalităților coșului de cumpărături, a listei pentru produse favorite, inclusiv adăugarea/afișarea corectă a produselor, respectiv verificarea corectitudinii prețurilor.
* Verificarea mesajelor de eroare și a traducerilor.

Aceste scenarii acoperă o varietate de situații pentru a se asigura că funcționalitățile cheie ale platformei sunt testate exhaustiv și că erorile pot fi identificate și gestionate în consecință.

# **Configurarea mediului de testare și rularea testelor automate**

## **Configurarea Mediului de Testare**

* Descărcarea și Instalarea Oracle VM VirtualBox.
* accesați site-ul oficial Oracle VM VirtualBox (https://www.virtualbox.org);
* descărcați ultima versiune disponibilă pentru sistemul de operare utilizat;
* urmați pașii de instalare oferiți de asistentul de instalare.
* Descărcarea imaginii ISO a sistemului de operare.
* identificați și descărcați o imagine ISO a sistemului de operare potrivit pentru configurarea mașinii virtuale.
* Crearea unei mașini virtuale în VirtualBox.
* lansați VirtualBox după instalare.
* accesați "Machine" din meniul superior și selectați "New" pentru a crea o mașină virtuală nouă;
* alegeți un nume pentru mașina virtuală și selectați tipul și versiunea sistemului de operare pe care doriți să-l instalați;
* selectați imagina ISO descărcată anterior;
* atribuiți o cantitate adecvată de memorie RAM pentru mașina virtuală;
* alegeți opțiunea "Create a virtual hard disk now" și continuați. Recomandat este să alocați suficient spațiu pentru sistemul de operare și aplicațiile de testare;
* finalizați crearea mașinii virtuale;
* Pornirea Mașinii Virtuale
* deschideți Oracle VM VirtualBox;
* selectați mașina virtuală pe care ați creat-o anterior;
* apăsați butonul "Start" pentru a porni mașina virtuală;
* Instalarea/descărcarea uneltelor necesare:

1. Browserul Google Chrome:

* accesați site-ul oficial Google Chrome (https://www.google.com/chrome/what-you-make-of-it/);
* descărcați și instalați browserul conform instrucțiunilor oferite.

1. PyCharm - Community Edition:

* accesați site-ul oficial JetBrains PyCharm (https://www.jetbrains.com/pycharm/download/?section=windows);
* descărcați și instalați versiunea Community Edition a PyCharm;
* în timpul procesului de instalare, bifați opțiunile "Update PATH Variable" și "Create Association .py" pentru o integrare mai bună.

1. Visual Studio Code:

* accesați site-ul oficial Visual Studio Code (https://code.visualstudio.com/);
* descărcați și instalați Visual Studio Code conform instrucțiunilor oferite.

1. Python:

* accesați site-ul oficial Python (https://www.python.org/downloads/);
* descărcați și instalați versiunea Python potrivită pentru sistemul de operare;
* după instalare, adăugați Python la variabila de mediu PATH folosind pașii menționați anterior.

1. Node.js:

* accesați site-ul oficial Node.js (https://nodejs.org/en/download) ;
* descărcați și instalați Node.js, asigurându-vă că opțiunea "Automatically install the necessary tools" este bifată.

1. Proiectul de Testare Selenium WebDriver:

* accesați repository-ul GitHub al proiectului de testare Selenium WebDriver (https://github.com/SergiuGF/Automation\_BDD\_Selenium\_Carturesti) ;
* clonați sau descărcați proiectul și urmați instrucțiunile de instalare și configurare.

1. Proiectul de Testare Cypress:

* accesați repository-ul GitHub al proiectului de testare Cypress (https://github.com/SergiuGF/Automation\_BDD\_Cypress\_Carturesti);
* clonați sau descărcați proiectul și urmați instrucțiunile de instalare și configurare.
* Configurarea IDE-ului PyCharm:
* deschideți PyCharm și navigați la meniul "File" > "Settings";
* în secțiunea "Plugins", căutați și instalați plugin-ul pentru limbajul Gherkin;
* deschideți terminalul din PyCharm și executați următoarele comenzi pentru a instala dependențele necesare:
* pip install selenium
* pip install webdriver\_manager
* pip install behave-html-formatter
* Configurarea IDE-ului Visual Studio Code:
* deschideți Visual Studio Code și accesați tab-ul "Extensions" (shortcut: Ctrl+Shift+X);
* căutați și instalați extensia pentru limbajul Cucumber (Gherkin) Full Support;
* deschideți terminalul din Visual Studio Code și executați următoarele comenzi pentru a instala pachetele necesare:
* npm install cypress
* npm install --save-dev cypress-cucumber-preprocessor
* npm install @cucumber/cucumber
* npm i --save-dev cypress-mochawesome-reporter

## **Rularea Testelor Automate**

* **Selenium WebDriver:**

Deschideți terminalul în IDE-ul PyCharm și tastati următoarele comenzi pentru a rula testele:

* behave -f html -o report-all-tests.html - pentru a rula toate testele.
* behave --tags=scenario\_tag\_name -f html -o scenario\_tag\_name.html -pentru a rula un singur scenariu, selectat utilizând tag-ul scenariului (de exemplu, @Login1).
* **Cypress:**

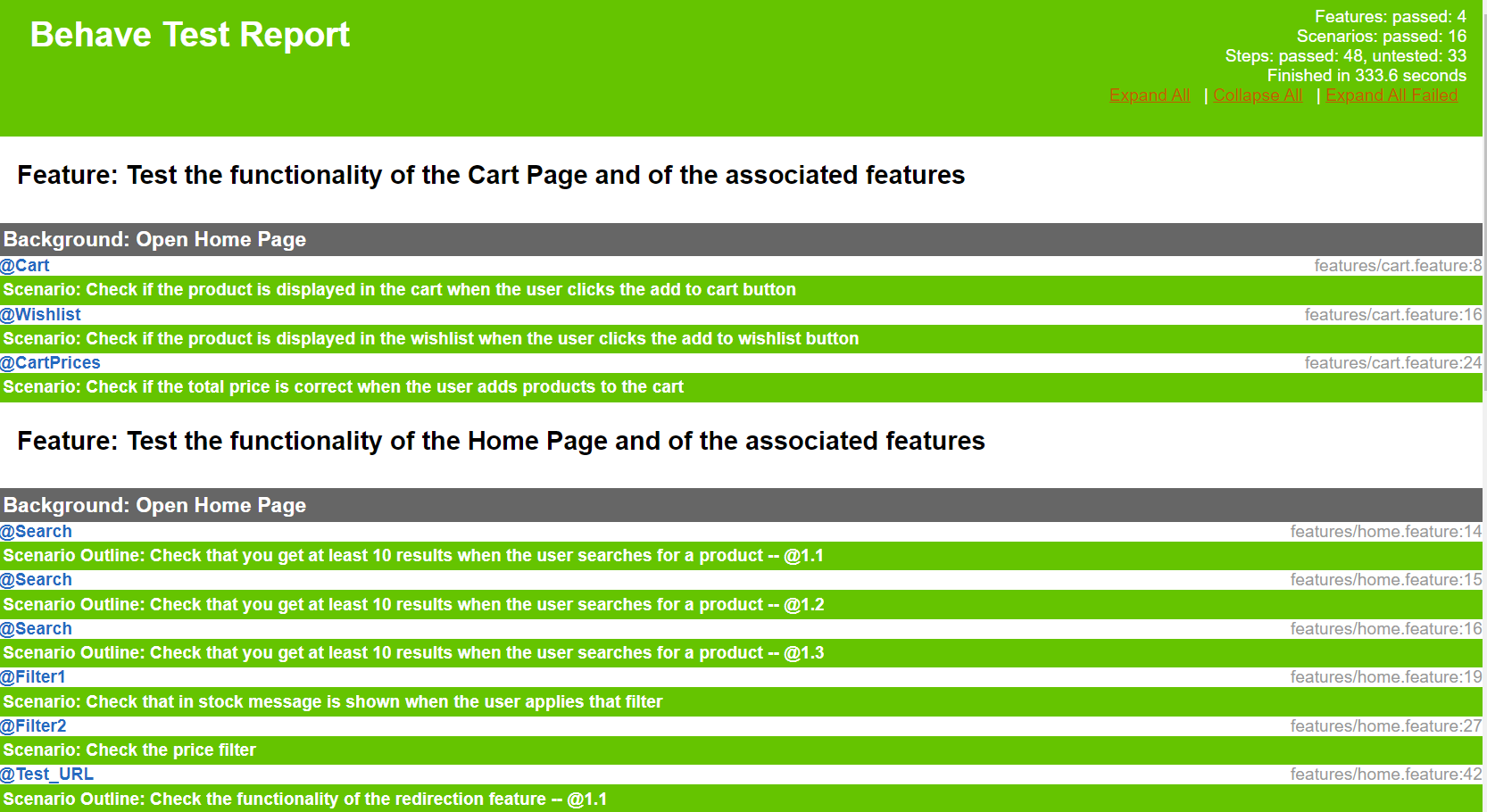
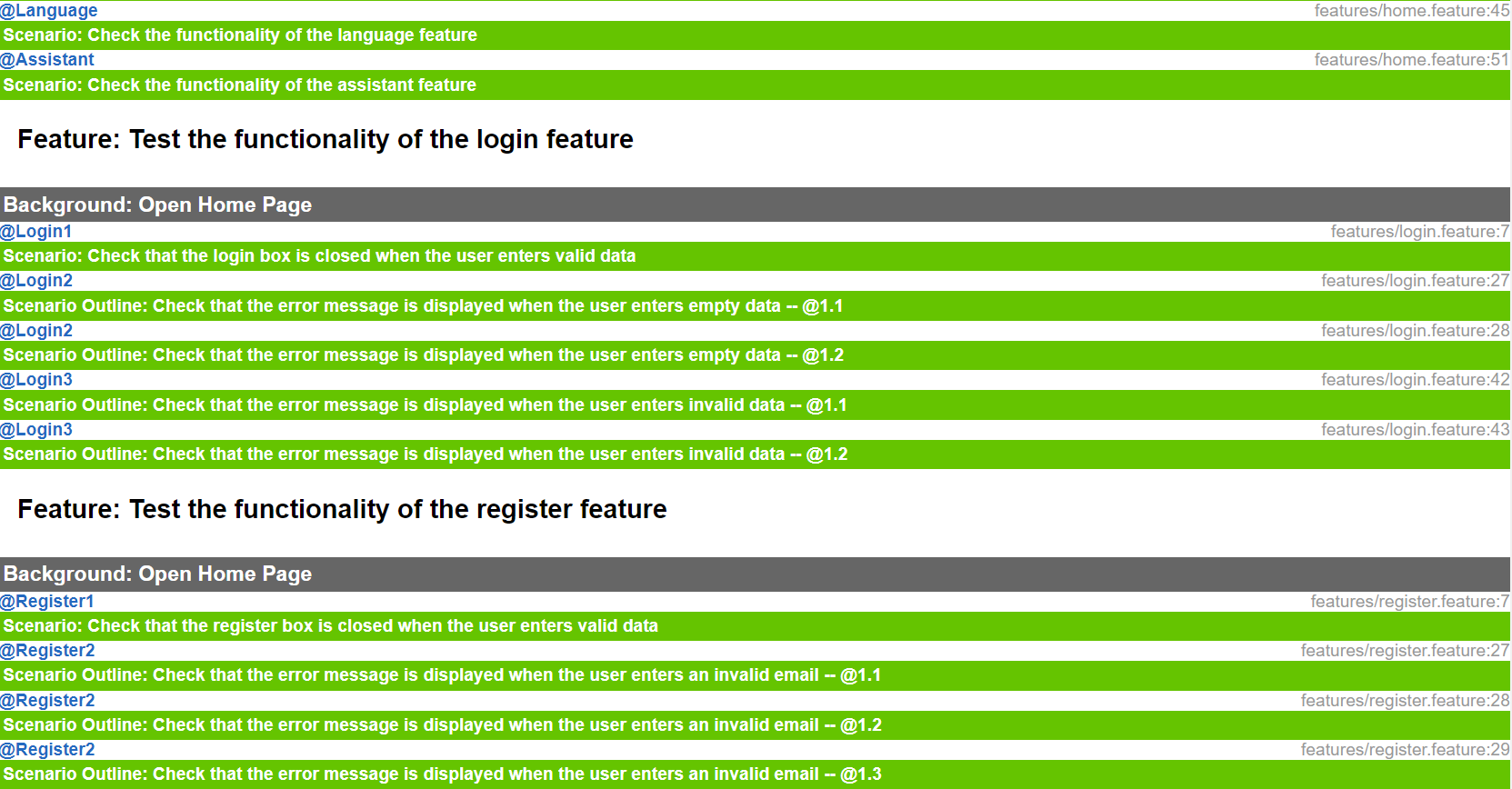
Deschideți terminalul în IDE-ul Visual Studio Code și tastati următoarele comenzi pentru a rula testele:

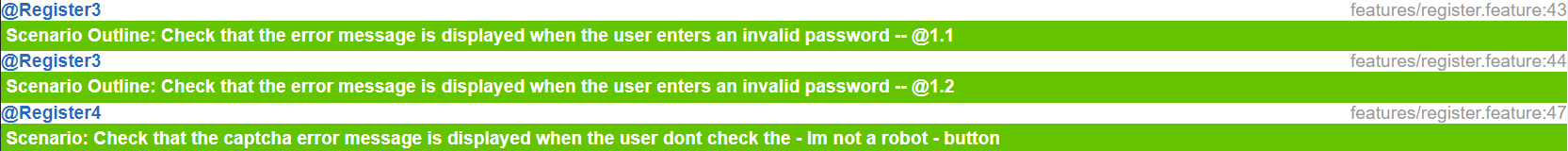
* npx cypress run - pentru a rula toate testele.
* npx cypress run --browser chrome - pentru a rula toate testele în Google Chrome.
* npx cypress run --headed --browser chrome - pentru a rula toate testele în Google Chrome în modul headed.
* npx cypress run --spec test\_path - pentru a rula un singur fișier de testare.
* npx cypress open - pentru a rula testele în modul headed.

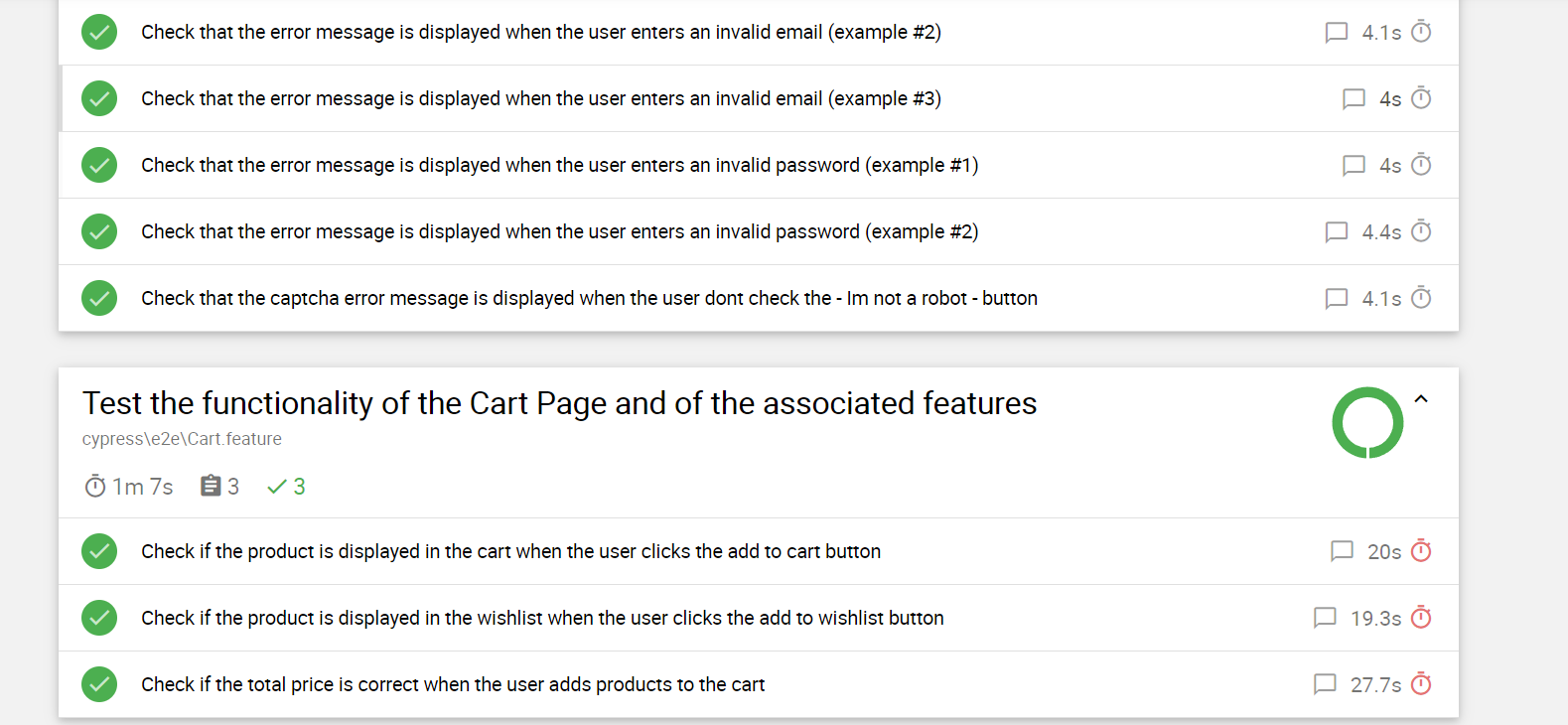
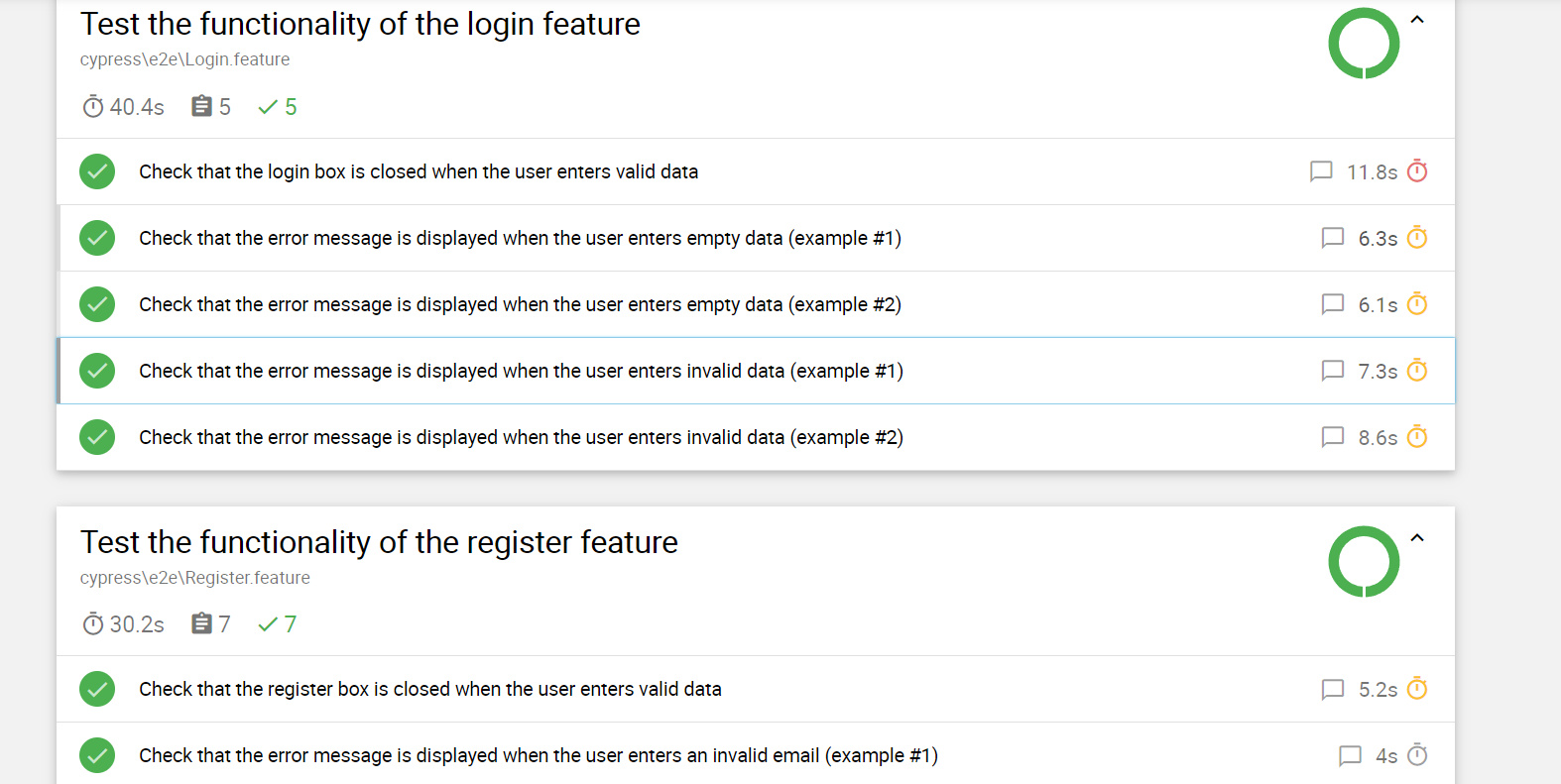
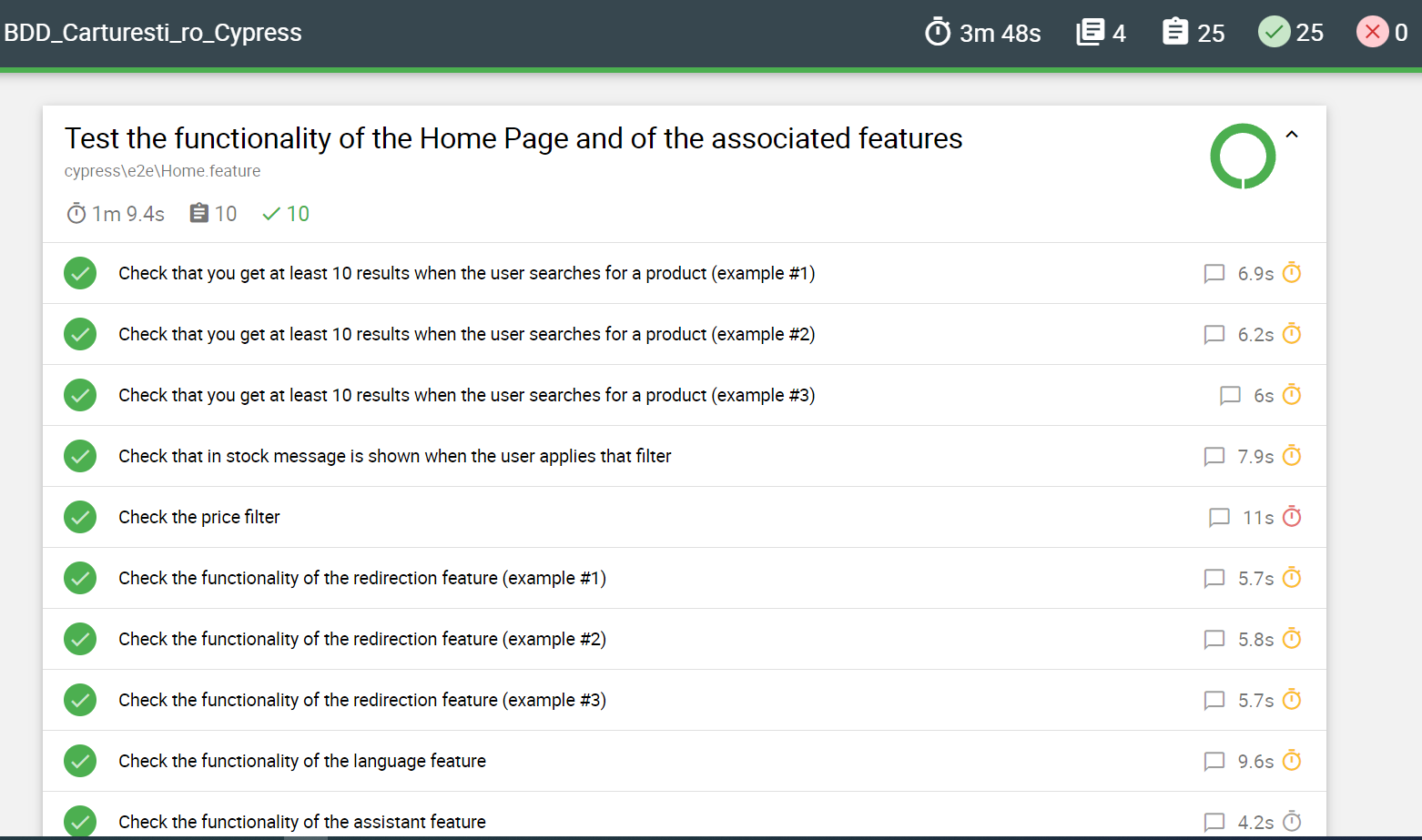
# **Rezultatele Testelor Automate**

Testele automate au fost efectuate cu succes și au furnizat următoarele rezultate:

* Validarea corectitudinii funcționalităților testate.
* Identificarea și remedierea unor erori și probleme.
* Generarea rapoartelor detaliate pentru fiecare test efectuat
* Viziualizarea rapoartelor prin deschiderea fișierului generat (ex. report.html) în browser-ul dorit. În cazul Cypress există și posibilitatea vizualizării rezultatelor în interfața Cypress (în situația în care testele au fost rulate în modul headed).







# **Indicatorii de Performanță pentru Selenium și Cypress**

## **Acuratețea**

**Selenium**:

Selenium oferă o acuratețe ridicată în validarea funcționalităților aplicației web. Capacitatea de a interacționa cu elementele interfeței utilizatorului (UI) și de a simula comportamentul utilizatorului contribuie la asigurarea unei acuratețe ridicate în testare. Testele automate dezvoltate cu Selenium sunt capabile să detecteze cu precizie problemele și să valideze corectitudinea funcționalităților.

**Cypress**:

Cypress este cunoscut pentru acuratețea sa în validarea interfeței utilizatorului (UI). Capacitatea sa de a face aserțiuni directe asupra elementelor UI contribuie la asigurarea unei acuratețe ridicate în testare.

Testele automate scrise cu Cypress sunt robuste și capabile să detecteze rapid eventualele probleme în aplicație. Dispune de un mecanismul automat de așteptare și reîncercare, care ajută la asigurarea unor teste stabile și fiabile chiar și în condiții de rețea imprevizibile.

## **Ușurința în implementare**

**Selenium:**

Implementarea testelor automate cu Selenium poate fi realizată relativ ușor, datorită disponibilității unui set bogat de metode și funcționalități.

Selenium oferă suport pentru multiple limbaje de programare (în comparație cu Cypress care suportă doar JavaScript) și integrare cu diferite framework-uri de testare, ceea ce face procesul de dezvoltare mai flexibil. Acceptă testarea pe diferite browsere și sisteme de operare, inclusiv pentru dispozitive mobile.

**Cypress:**

Cypress vine cu un set integrat de instrumente și biblioteci care facilitează dezvoltarea și rularea testelor automate. Dpre deosebire de Selenium, dispune de un test runner integrat care facilitează rularea testelor și vizualizarea rezultatelor

Structura clară și intuitivă a Cypress permite dezvoltatorilor să scrie teste eficiente și să ruleze rapid testele pe diferite configurații.

## **Eficiența**

**Selenium**:

Selenium poate fi eficient în rularea testelor automate, cu condiția să fie configurat și gestionat corespunzător.

Posibilitatea de a paraleliza teste și de a executa teste pe mai multe platforme și browsere simultan poate contribui la îmbunătățirea eficienței procesului de testare.

**Cypress**:

Cypress este recunoscut pentru viteza sa în rularea testelor automate și detectarea rapidă a problemelor (testele au fost rulate în decurs de 3,48 minute în cumparație cu 5,33 minute în Selenium). Cypress rulează teste direct în browser, ceea ce elimină suprasolicitarea de comunicare în rețea și face testele mai rapide și mai fiabile. Dispune de un tablou de bord încorporat care oferă rezultate ale testelor, inclusiv videoclipuri.

Arhitectura modernă și orientată pe performanță a Cypress contribuie la îmbunătățirea eficienței procesului de testare.

## **Evaluarea performanței între Selenium și Cypress**

Atât Selenium, cât și Cypress sunt unelte puternice pentru testarea automată a aplicațiilor web.

Alegerea între Selenium și Cypress depinde de nevoile specifice ale proiectului și de preferințele echipei de dezvoltare.

Indiferent de uneltele utilizate, asigurarea unei acuratețe ridicate, a ușurinței în implementare și a eficienței în procesul de testare este esențială pentru obținerea unor rezultate de calitate în dezvoltarea aplicațiilor web.

# **Concluzii**

Eficiența dezvoltării software depinde de testarea automată. Viteza, reproductibilitatea, acoperirea extinsă, reutilizarea, ușurința de integrare cu practicile de dezvoltare agile și detectarea timpurie a erorilor sunt principalele beneficii. Acest lucru ajută în mod constant la asigurarea calității software-ului.

Proiectul de testare automată a fost un succes și a adus asigurarea calității necesare pentru biblioteca virtuală Carturești. Testele automate au demonstrat funcționalitatea corectă a aplicației, rezultatele returnare au fost conform așteptărilor, fără identificarea unor potențiale probleme, nefiind astfel nevoie de implicare unor resurse suplimentare, precum echipa de dezvoltare. În cadrul proiectului au fost implementate testele necesare pentru acoperirea cerințelor și testarea principalelor funcționalități ale platofrmei.