**MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică și Microelectronică**

**Departamentul Ingineria Software și Automatică**

**RAPORT**

**la lucrarea de laborator Nr.2**

**Tema: „Selenium IDE”**

**Disciplina: „Testarea Software”**

**Efectuat:** st. gr. SI-211: Chihai Adrian

**Verificat:** Andrian Prisăcaru

**Chișinău 2024**

## INTRODUCERE

**Sarcina lucrării:** De creat un caz de testare folosind aplicația Selenium IDE (Integrated Development Environment).

**Testarea software** reprezintă o investigație empirică realizată cu scopul de a oferi părților interesate informații referitoare la calitatea produsului sau serviciului supus testării[1], luând în considerație contextul operațional în care acesta din urma va fi folosit. Testarea software pune la dispoziție o viziune obiectivă și independentă asupra produsului în dezvoltare, oferind astfel businessului posibilitatea de a înțelege și evalua riscurile asociate cu implementarea produsului soft. Tehnicile de testare includ, dar nu sunt limitate la, procesul de execuție a programului sau aplicației în scopul identificării defectelor/erorilor de software.

**Caz de testare** - este cea mai mică unitate a planului de testare - care include o descriere a acțiunilor și parametrilor necesari pentru a realiza și verifica comportamentul așteptat al unei anumite funcții sau a părții software-ului testat.

**Selenium IDE** este un instrument pentru înregistrarea interacțiunilor browser-ului pentru cazurile de testare. Acest instrument de testare open-source este ușor de instalat și de pornit: puteți descărca instrumentul ca plugin în browserul dvs. web, cum ar fi Chrome și Firefox. Folosind instrumentul, puteți crea scenarii de testare simple sau complicate. Ca și alte IDE-uri pentru dezvoltatori, puteți seta puncte de întrerupere, întrerupeți și verificați variabilele atunci când apare o eroare. În plus, puteți utiliza din nou cazul dvs. de testare într-un alt caz de testare, ceea ce face testarea repetitivă mai ușoară și mai rapidă. Selenium IDE înțelege declarații condiționate precum if și while. Această manipulare logică vă permite să extindeți și să acoperiți mai multe scenarii de testare diverse.

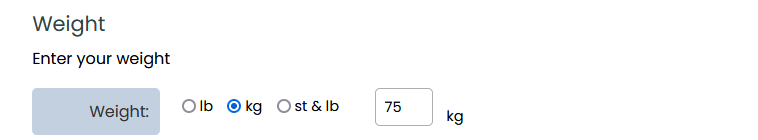
Cele mai folosite comenzi în Selenium IDE sunt:

* **Open** deschide o pagină folosind un URL
* **click/clickAndWait** execută o acțiune de apăsare și opțional, așteaptă încărcarea unei noi pagini
* **verifyTitle/assertTitle** verifică titlul unei pagini
* **verifyTextPresent** verifică prezența unui text undeva în pagină
* **verifyElementPresent** verifică prezența unui element al interfeței grafice în pagină, definit prin tag HTML
* **verifyText** verifică prezența unui text și a unui tag HTML în pagină
* **verifyTable** verifică conținutul așteptat al unui tabel
* **waitForPageToLoad** amână execuția până când o pagină așteptată se încarcă. Este chemată automat când se folosește clickAndWait
* **waitForElementPresent** amână execuția până când un element al interfeței grafice, definit prin tag HTML, este prezent în pagină

Am creat un test cu ajutorul aplicației Selenium IDE, astfel am înregistrat pașii asemănători cu cei din îndrumar pentru pagina web https://www.curs.md, și am urmărit scriptul înregistrat.

De asemenea am utilizat pluggin-ul Selenium IDE pentru Chrome, astfel am tastat butonul Selenium-ului și am creat un nou proiect, unde am introdus link-ul paginii web alese.

Instrumentul mi-a deschis o nouă pagină cu adresa https://runbundle.com/tools/cycling/cycling-calorie-calculator, unde erau afișate câmpuri pentru care am introdus valori aleatorii, aplicația înregistrând procesul. Valorile introduse sunt reprezentate în figura 2.1.



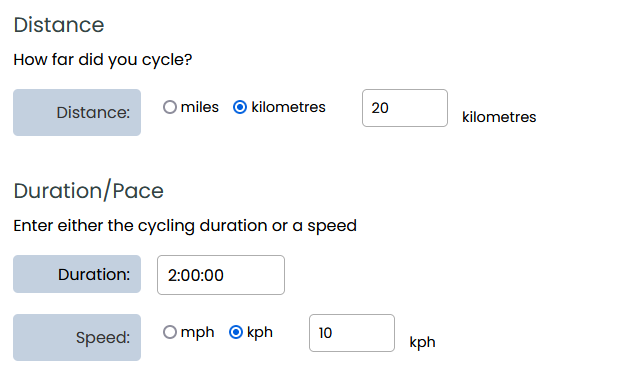


Figura 2.1 – Valorile introduse pe pagina https://runbundle.com/tools/cycling/cycling-calorie-calculator [[2]](#_BIBLIOGRAFIE)

După ce am introdus valorile, acestea s-au afișat pașii executați în Selenium IDE. Aceștia sunt reprezentate în trei coloane: Command, Target și Value, figura 2.2 .

În interfața grafică a Selenium-ului, în partea stânga sus, tastând butonul Play Current Test, am observat că testul rulează fiecare pas înregistrat, astfel pasul fiind executa corect culoare lui a devenit verde (figura 2.2).

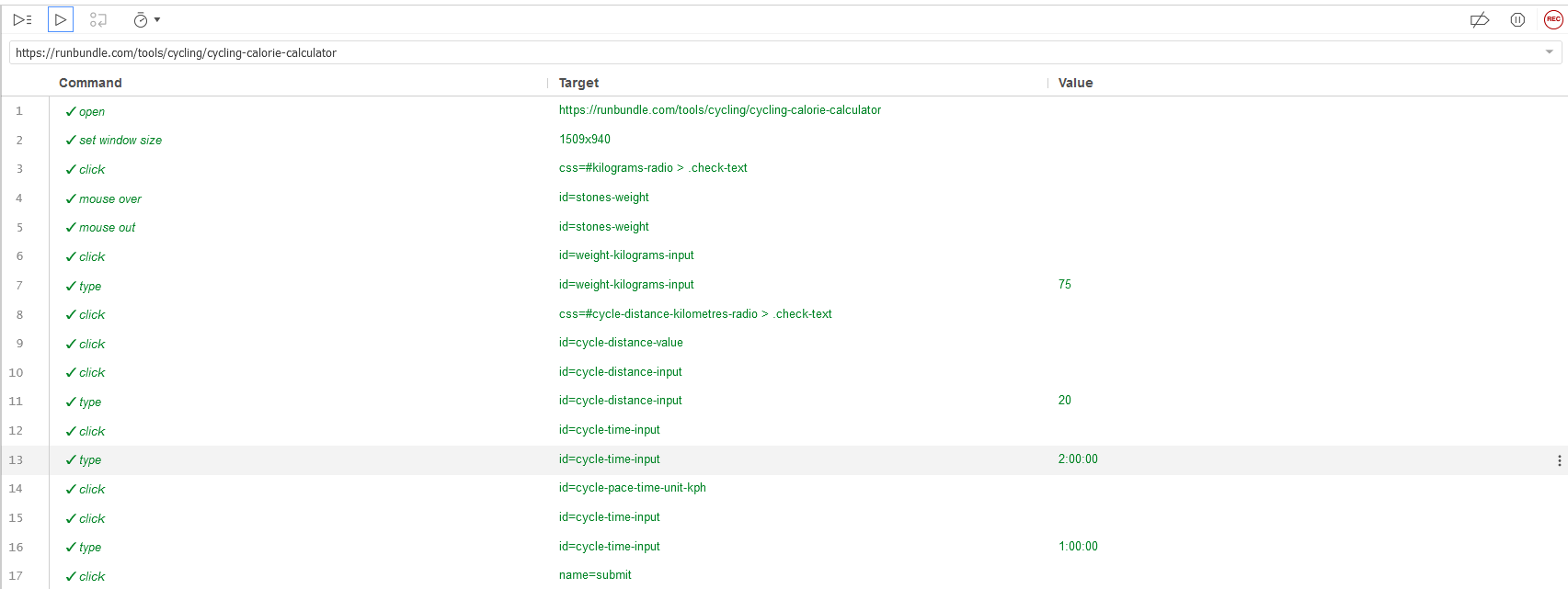


Figura 2.2 – Afișarea pașilor executați în pagina web https://runbundle.com/tools/cycling/cycling-calorie-calculator [[2]](#_BIBLIOGRAFIE) în instrumentul Selenium IDE

Următorul pas a fost de a modifica coloana Value din interfața grafică a Selenium-ului pentru diferiți pași înregistrați, figura 2.3.

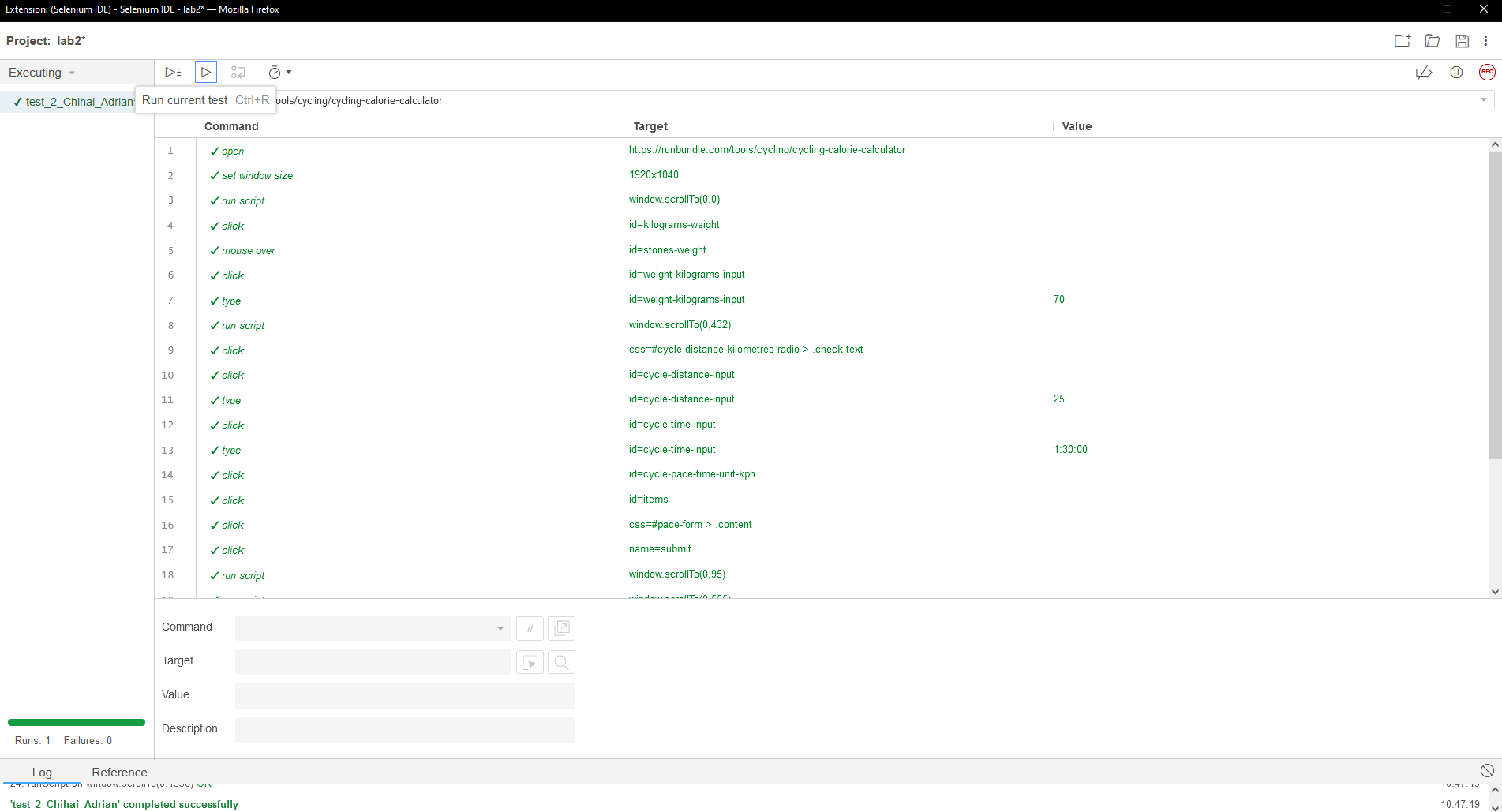


Figura 2.3 – Modificarea câmpului Value în interfața grafică a Selenium-ului și implementarea scriptului scrollTo.

Am implementat comanda execute script care ne permite sa facem scroll in pagina, astfel putem vizualiza raportul de la final

Pasul următor a fost de a selecta unul dintre rezultatele obținute și de a face click dreapta pe textul respectiv, s-a afișat un meniu de operații posibile, am ales operația „Mouse Over”, figura 2.5.

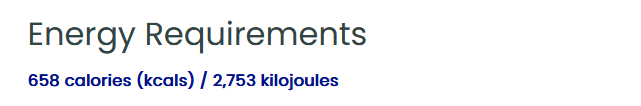




Figura 2.6 – Rezultatul final este rezumatul și detaliile despre timpul și distnața parcursa, atât cât și caloriile consumate

## CONCLUZIE

Laboratorul a efectuat un studiu asupra Selenium IDE, un instrument folosit pentru testarea automată a aplicațiilor web. Permite utilizatorilor să înregistreze și să reproducă interacțiunile aplicațiilor web, cum ar fi clicurile, inserarea, trimiterea formulelor și navigarea în pagină. Interfața intuitivă a lui Selenium IDE face înregistrarea și reproducerea testelor simple.