L’installazione di Jenkins è stata una delle parti nuove per tutti membri del team, essa ha impiegato parecchio tempo perché non avendo nessuna nozione di base è servita una buona parte di ricerca di informazioni al riguardo. Dopo una parte di informazioni si è proceduti con l’installazione che, dopo svariati tentativi, si è rivelata piuttosto semplice. Essa è riportata interamente nella guida allegata.

La scrittura di Selenium era in parte una novità perché tutti i membri del gruppo conoscevano già il linguaggio Java ma nessuno avava mai utilizzato un WebDriver. A causa di questa mancanza è stata necessaria una parte di raccolta di informazioni che ha prolungato leggermente questa parte del progetto. I test sono abbastanza intuitivi e facili da creare, bisogna innanzitutto creare una classe di che conterrà solamente il metodo main e che richiamerà i test.

public class SeleniumTest {

public static void main(String[] args) {

SeleniumTestTest st = new SeleniumTestTest();

st.test();

}

}

Fatto ciò si potrà, premendo la combinazione di tasti **ctrl+shift+t,** creare una classe di test. Essa conterrà tutto il codice necessario a Selenium per controllare il corretto funzionamento delle pagine web.

Come prima cosa all interno della classe bisognerà dichiarare quale è l URL del sito di cui si vogliono eseguire i test e il WebDriver che per il momento resterà di valore nullo.

String URL = "http://cashyland.tk/";

WebDriver driver = null;

Dopodiché si dovrà nel metodo test, generato automaticamente da IntellIJ, specificare la posizione del WebDriver, generare le opzioni per il WebDriver ed istanziare lo stesso con le impostazioni appena create. Le impostazioni utilizzate da noi servono per eseguire i test senza un interfaccia grafica.

System.setProperty("webdriver.chrome.driver","/usr/bin/chromedriver");

ChromeOptions options = new ChromeOptions();

options.addArguments("--headless", "--disable-gpu", "--ignore-certificate-errors");

driver = new ChromeDriver(options);

Fatto ciò si potrà usare qualsiasi elemento della pagina semplicemente creando un WebElement ed assegnandogli l elemento corrispondente tramite uno degli svariati metodi che consentono di trovare un tag tramite uno dei suoi attributi, in base al testo che contiene, alle sue dimensioni e così via per innumerevoli possibilità. Una volta trovato il tag cercato potremo eseguire molte operazioni con esso come cliccarlo, eseguirne l hover, scriverci del testo e via dicendo.

WebElement home = null;

home = driver.findElement(By.linkText("Home"));

home.click();

waitMillis(1000);

System.out.println(driver.getTitle());

assertEquals("CashyLand - Home",driver.getTitle());

Con il WebElement potremo inoltre prendere le informazioni dell elemento come il titolo, la dimensione,… e controllare che siano quelle che desideriamo con degli assert.

I test possono essere eseguiti sia con l`interfaccia grafica che mostra nel browser cosa sta facendo il test oppure senza di essa se ad esempio si sta utilizzando un sistema operativo senza GUI.

Per far funzionare i test con l`interfaccia basterà farli partire dal IDE che si sta utilizzando.

Per eseguire i test senza interfaccia grafica bisognerà utilizzare il driver web di Xvfb che è di base un sistema headless. Utilizzando esso basterà poi seguire questa procedura:

Innanzi tutto andare nella cartella dove sono contenuti i tests e fare partire il server con il comando:

*Xvfb :0 &*

Fatto ciò bisognerà salvare il numero dello schermo in una variabile di sistema, questo verrà fatto con il seguente comando:

*export DISPLAY=:0*

Dopodiché si potrà eseguire il file che contiene il metodo main e i test dovrebbero funzionare, se questo non è vero bisognerà fermare i processi già in esecuzione di firefox, selenium e del geckodriver per poi riprovare la procedura. Per bloccare i processi si possono usare i seguenti tre comandi:

*pkill selenium*

*pkill geckodriver*

*pkill firefox*