GESTIONE CASINÒ | Diario di lavoro - 03.04.2019

Matan Davidi, Thor Düblin, Matteo Forni, Carlo Pezzotti, Mattia Toscanelli

Trevano, 3 aprile 2019

Lavori svolti

Oggi Matan è finalmente riuscito a far funzionare le asserzioni nei test di Selenium in modo che essi producano errori in caso di test non passati. Ha infatti dovuto creare, con l'aiuto di IntelliJ (usando la scorciatoia CTRL + SHIFT + T), una classe di test "SeleniumTestTest" che contiene il seguente metodo, che asserisce che due stringhe sono uguali:

```
class SeleniumTestTest {

    @Test
    void testTitle(String expected, String title) {

        assertEquals(expected, title);

    }
}
```

Ha poi dovuto aggiungere alla classe principale il seguente codice:

```
SeleniumTestTest st = new SeleniumTestTest();
st.testTitle("CashyLand - Home", titleText);
```

Inoltre ha dovuto istanziare l'oggetto di tipo ChromeDriver utilizzato per navigare tra le pagine con un'istanza di ChromeOptions a cui è stato aggiunto il parametro "headless":

```
ChromeOptions options = new ChromeOptions();
options.addArguments("headless");
WebDriver driver = new ChromeDriver(options);
```

Oggi Matteo ha controllato che Jenkins funzionasse correttamente controllando che tutti i push della giornata generassero il risultato voluto sulla repo GestioneCasinoProduction. Fatto ciò ha provato a riprendere il lavoro svolto da Thor ma non è stato possibile dato che le sue prove non si trovavano sul git. Ha dunque chiesto a Matan il suo codice per capirne il funzionamento ed ha iniziato a cercare come eseguirli senza un'interfaccia grafica. Per fare ciò ha trovato il seguente sito: https://w w w.blazemeter.com/blog/headless-execution-selenium-tests-jenkins che spiega come eseguire i test di Selenium tramite quelli di Jenkins.

Mattia e Carlo oggi si sono dedicati alla pagina del dopo login. Più precisamente Mattia si è occupato di creare la pagina front-end per la gestione delle sale, giochi, utenti e promozioni. Questa pagina è composta da un input di tipo testo per il titolo, un input di tipo file per aggiungere un'immagine e una textarea per la descrizione. Per l'input di tipo file abbiamo cercato un template che permettesse di aggiungere l'immagine ,oltre che al pulsante, attraverso un drag and drop: https://bootsnipp.com/snippets/DOXy4. Il codice aggiunto per questo input è il seguente:
HTML:

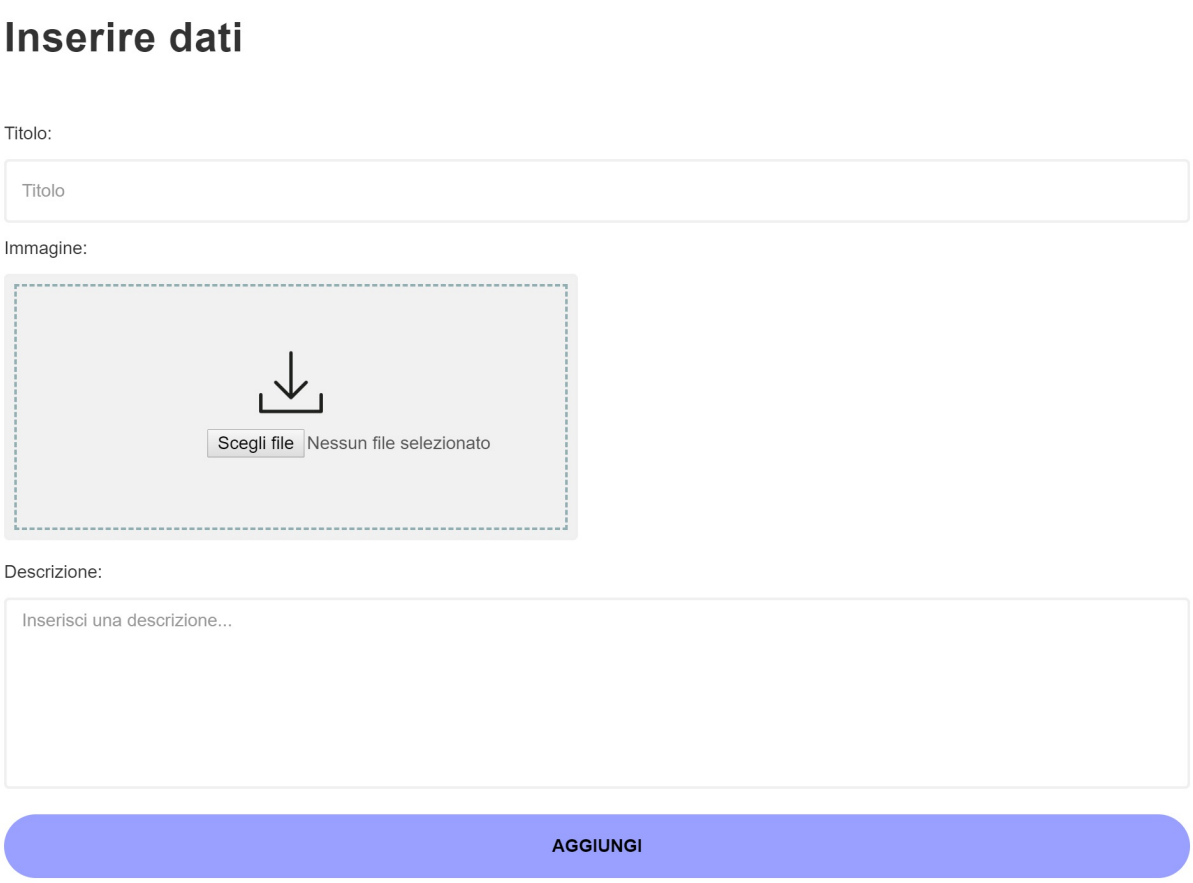
```
<div class="row">
   <div class="col-md-6">
        <div class="form-group files color">
             <input type="file" class="form-control" accept="image/*">
        </div>
   </div>
</div>
```

CSS:

```
<div class="row">
.files input {
    outline: 2px dashed #92b0b3;
    outline-offset: -10px;
    -webkit-transition: outline-offset .15s ease-in-out, background-color .15s linear;
    transition: outline-offset .15s ease-in-out, background-color .15s linear;
    padding: 120px 0px 85px 35%;
    text-align: center !important;
    margin: 0;
    width: 100% !important;
}
.files input:focus{
    outline: 2px dashed #92b0b3;
```

```
    outline-offset: -10px;
    -webkit-transition: outline-offset .15s ease-in-out, background-color .15s linear;
    transition: outline-offset .15s ease-in-out, background-color .15s linear; border:1px solid #92b0b3;
 }
.files{ position:relative}
.files:after {  pointer-events: none;
    position: absolute;
    top: 60px;
    left: 0;
    width: 50px;
    right: 0;
    height: 56px;
    content: "";
    background-image: url(https://image.flaticon.com/icons/png/128/109/109612.png);
    display: block;
    margin: 0 auto;
    background-size: 100%;
    background-repeat: no-repeat;
}
.color input{ background-color:#f1f1f1;}
.files:before {
    position: absolute;
    bottom: 10px;
    left: 0;  pointer-events: none;
    width: 100%;
    right: 0;
    height: 57px;
    display: block;
    margin: 0 auto;
    color: #2ea591;
    font-weight: 600;
    text-transform: capitalize;
    text-align: center;
}
```

Il risultato finale della pagina è il seguente:



Carlo invece si è dedicato alla parte back-end della pagina di aggiunta delle sale, giochi, promozioni e utenti. Innanzitutto adatta il titolo della pagina al contesto, nel senso che quando per dire l'admin clicca sul pulsante per modificare le sale il tipo dive essere "Modifica sale". Questa cosa è gestita grazie al metodo GET.
Poi ha creato un form che si adatta in modo dinamico per aggiungere le varie cose (sale, giochi, promozioni e utenti), questo è fatto così:

```
$get = $_GET['type'];
try{
    $result = $db->executeQuery("select * from $get");
```

```
    $n = 0;
    foreach ($result[0] as $key => $value) {
        if($n%2==0){
            echo "<div class='col-md-12 col-sm-12'>
                <span>".strtoupper($key{0}).substr($key,1,strlen($key)).":</span>
                <input name='$key' type='text' class='form-control' id='title' placeholder='".strtoupper($key{0}).substr($key,1,strlen($key))."'>
            </div>";
        }
        $n++;
    }
}catch(InvalidArgumentException $iae){}
```

Infine assieme alla cosa soprastante ha gestito l'inserimento in modo dinamico dei campi:

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $get = $_GET["type"];
        $query = "Insert into $get(";
        $n = 0;
        foreach ($_POST as $key => $value) {
            if($n+1 == sizeof($_POST))
                $query .= $key;
            else
                $query .= $key . ",";
            $n++;
        }
        $query .= ") values(";
        $n = 0;
        foreach ($_POST as $key => $value) {
            if($n+1 == sizeof($_POST))
                $query .= "'$value'";
            else
                $query .= "'$value'" . ",";
            $n++;
        }
        $query .= ")";
        //echo $query;
        $db->executeQuery($query);
        header("Refresh:0");
    }
```

Thor le prime 2 ore ha fatto il test del modulo 133, successivamente nel tempo che gli rimaneva a cercato dei modi per far andare selenium nel server linux (senza GUI), su stackoverflow :
https://stackoverflow.com/questions/34342632/will-selenium-work-on-ubuntu-with-no-gui
Dove consigliano di usare "xvfb" dopodiché ha letto anche la guida trovata da Matteo per far funzionare i test di Selenium con Jenkins.
Inoltre ha provato continuato a creare il codice Java per la registrazione degli utenti (con file csv, iniziato la scorsa lezione).

Problemi riscontrati e soluzioni adottate

Punto della situazione rispetto alla pianificazione

Rispetto alla pianificazione siamo in orario.

Programma di massima per la prossima giornata di lavoro

Matan deve cercare di far funzionare il test di Selenium completato oggi con Firefox e informarsi su come fare a eseguire il test senza un'interfaccia grafica.

Matteo deve provare ad eseguire dei test senza GUI.

Thor deve trovare un modo per far andare i test di selenium sul server e continuare a cercare codice di test(nel dettaglio quello della registrazione, se finisce ne inizierà un altro).