****

**ZÁRÓDOLGOZAT TESZTELÉS**

Készítették:

Kovács Dániel

Braczkó Tamás

Novák Dominik Viktor

Konzulens:

Farkas Zoltán

Miskolc

# **Tesztelési Dokumentáció**

A webshop tesztelése kiemelkedő fontosságú, mivel az online vásárlások általában összetettek, és a felhasználói élmény kritikus szerepet tölt be. Ez a dokumentum összegzi az alkalmazás működésének tesztelésére alkalmazott módszereket.

A tesztelési folyamatok célja a webshop stabilitásának, funkcionalitásának és felhasználói élményének biztosítása, valamint az esetleges hibák és problémák azonosítása.

* Unit tesztek a Backend funkcionalitásokra:
  + A backend rész egységtesztelése során ellenőrizzük a webshop logikáját és adatfeldolgozási folyamatait. Ezek a tesztek biztosítják, hogy a webshop minden része megfelelően működjön, és az elvárt eredményeket produkálja.
* Teljesítményelemzés a Frontend részről:
  + A frontend teljesítményének elemzése segít azonosítani az oldalak betöltési sebességével és a felhasználói interakciókkal kapcsolatos problémákat. A Lighthouse Chrome bővítmény használatával mérjük és értékeljük a frontend teljesítményét, hogy a webshop gyorsan és zökkenőmentesen működjön.
* Selenium automatizált tesztek a regisztráció és belépés funkciókra:
  + A Selenium segítségével automatizált teszteket futtatunk a webshop regisztrációs és belépési funkcióira. Ezek a tesztek ellenőrzik, hogy a regisztráció és belépés folyamatai megfelelően működnek-e, és az elvárt eredményeket produkálják-e.
* Főoldal Teljesítményének Értékelése Lighthouse Segítségével.
* Felhasználói Interakciók Tesztelése CommentForm Komponenssel Jest Használatával.

# **Unit tesztek a Backenden**

## **4 teszt volt értékélve, mind a 4 sikeresen lefutott**

## **A screenshot of a computer Description automatically generated**

### **GetCategory\_ValidId\_ReturnsCategory**

A screenshot of a computer code

Description automatically generated

A teszt célja annak ellenőrzése, hogy a GetCategory metódus helyesen visszaadja-e a megfelelő kategóriát a megadott érvényes azonosító alapján. A teszt először létrehoz egy érvényes azonosítót és egy várt kategóriát. Ezután hozzáadjuk ezt a kategóriát az adatbázishoz és mentjük a változtatásokat. Ezután meghívjuk a GetCategory metódust a létrehozott érvényes azonosítóval. Végül ellenőrizzük, hogy a visszakapott eredmény nem null, és hogy megegyezik-e az elvárt kategóriával.

### **CreateCategory\_ValidCategory\_ReturnsCategory**

A screenshot of a computer code

Description automatically generated

A teszt célja annak ellenőrzése, hogy a CreateCategory metódus helyesen létrehoz-e egy új kategóriát és visszaadja-e azt a megfelelő módon. A teszt először létrehoz egy új kategóriát a "New Category" névvel. Ezután meghívjuk a CreateCategory metódust a létrehozott új kategóriával. Végül ellenőrizzük, hogy a visszakapott akció eredménye nem null, és hogy a létrehozott eredmény megfelelő típusú és tartalmú-e. Ezen belül ellenőrizzük, hogy a visszakapott kategória neve megegyezik-e az új kategória nevével.

### **UpdateCategory\_ValidIdAndCategory\_ReturnsNoContent**

A screenshot of a computer code

Description automatically generated

A teszt célja annak ellenőrzése, hogy az UpdateCategory metódus helyesen frissíti-e a meglévő kategóriát az adott érvényes azonosító alapján, és hogy a megfelelő visszajelzést adja-e vissza a frissítés sikerességére. A teszt először létrehoz egy meglévő kategóriát az adott érvényes azonosítóval és az "Existing Category" nevével. Ezután hozzáadjuk ezt a kategóriát az adatbázishoz és mentjük a változtatásokat. Ezután létrehozunk egy frissített kategóriát az eredeti azonosítóval és egy új névvel. Ezután meghívjuk az UpdateCategory metódust a frissített kategóriával és az azonosítóval. Végül ellenőrizzük, hogy a visszakapott eredmény típusa egy NoContentResult objektum-e, amely azt jelzi, hogy a frissítés sikeresen megtörtént, és nincs szükség további tartalomra a válaszban.

### **DeleteCategory\_ValidId\_ReturnsNoContent**

A computer screen shot of a code

Description automatically generated

A teszt célja annak ellenőrzése, hogy a DeleteCategory metódus helyesen törli-e a meglévő kategóriát az adott érvényes azonosító alapján, és hogy a megfelelő visszajelzést adja-e vissza a törlés sikerességére. A teszt először létrehoz egy meglévő kategóriát az adott érvényes azonosítóval és az "Existing Category" nevével. Ezután hozzáadjuk ezt a kategóriát az adatbázishoz és mentjük a változtatásokat. Ezután meghívjuk a DeleteCategory metódust az adott érvényes azonosítóval. Végül ellenőrizzük, hogy a visszakapott eredmény típusa egy NoContentResult objektum-e, amely azt jelzi, hogy a törlés sikeresen megtörtént, és nincs szükség további tartalomra a válaszban.

# **Selenium a Regisztrációhoz és Bejelentkezéshes**

A screenshot of a computer

Description automatically generated

*Részlet a regisztrációból*

Ez a Selenium Python szkript automatizálja a felhasználói felületen végzett műveleteket egy webalkalmazásban. A szkript feladata a felhasználó regisztrációs és bejelentkezési folyamatának szimulálása. Először inicializálja és maximalizálja az Edge böngészőt, majd elnavigál a megadott URL-re. A regisztrációs részben megvárja, hogy a "My Account" gomb kattintható legyen, majd a "Sign Up" opcióra kattintva kitölti és elküldi a regisztrációs űrlapot. Ezután a bejelentkezési folyamatban a szkript a regisztrált hitelesítési adatokkal kitölti és elküldi a bejelentkezési űrlapot. Végül a szkript a felhasználó fiókjához navigál, megnyitva az "Account" opciót. Ennek eredményeként a szkript segíti a fejlesztőket a regisztrációs és bejelentkezési folyamatok tesztelésében és ellenőrzésében, miközben ezeket a feladatokat automatizálja, lehetővé téve a gyors és hatékony tesztelést.

**Főoldal Teljesítményének Értékelése Lighthouse Segítségével**A screenshot of a computer

Description automatically generated

A Lighthouse egy Google által fejlesztett automatizált eszköz, amely segít a webes alkalmazások teljesítményének, hozzáférhetőségének, legjobb gyakorlatainak és SEO-jának értékelésében. Segítségével gyorsan és hatékonyan lehet tesztelni és javítani a webhelyek teljesítményét és felhasználói élményét.

**Felhasználói Interakciók Tesztelése CommentForm Komponenssel Jest Használatával**A screenshot of a computer code

Description automatically generated

A CommentForm React komponens tesztjei a @testing-library/react és @testing-library/user-event könyvtárak segítségével vannak megvalósítva. Az első teszt célja, hogy ellenőrizze, hogy a CommentForm komponens rendeltetésszerűen megjelenik-e, és nem okoz-e hibát a renderelés során. A második teszt az állapot frissítését teszteli felhasználói interakciók segítségével, ellenőrizve, hogy a beviteli mezők és a csillagok kiválasztása helyesen működik-e. Ezek a tesztek segítenek biztosítani, hogy a CommentForm komponens megfelelően működjön, javítva az alkalmazás stabilitását és megbízhatóságát.