**Pozitiva:**

**test\_kuchynske\_potreby.py**

* Použití Playwrightu – Moderní a výkonný nástroj pro automatizované testování webových aplikací.
* Použití sync\_playwright() v pytest.fixture() – Zajišťuje správnou správu browser session.
* Použití headless=False, slow\_mo=1500 – Usnadňuje vizuální ladění testu.
* Použití wait\_for\_load\_state("networkidle") – Správné čekání na načtení stránky.
* Použití pytest.fixture() pro správu browseru a stránky – Minimalizuje duplicitu kódu, zajistí správné zavírání browseru.
* Použití yield v page() fixture – Správná práce s instancí prohlížeče.
* Použití timeout=30000 pro načítání obsahu – Zajišťuje stabilitu testu při delších načítáních.
* Krok za krokem pokrytí uživatelského chování: Přechod na webovou stránku. Odmítnutí cookies. Navigace do sekce „Bloky na nože“ a filtrování podle značky. Řazení produktů od nejlevnějšího. Ověření, zda produkty jsou správně seřazeny podle ceny.
* Použití assert prices == sorted(prices) – Ověřuje, že ceny jsou správně seřazené.

**test\_kuchynske\_potreby\_2.py**

* Použití pytest.mark.parametrize() – Umožňuje snadné testování více variant přihlášení.
* Použití pytest.fixture() pro správu browseru – Správně spravuje session prohlížeče.
* Podpora více prohlížečů (chromium, firefox, webkit) – Testy běží na všech hlavních prohlížečích.
* Testy zahrnují validní i nevalidní přihlášení: Špatné uživatelské jméno. Špatné heslo. Prázdné pole (heslo i uživatelské jméno).
* Ověření chybové zprávy – Zajistí, že se zobrazí správná zpráva při neúspěšném přihlášení.
* Použití page.fill() místo locator().click().type() – Efektivnější a méně náchylné na chyby.
* Použití page.wait\_for\_selector() pro čekání na odpověď serveru – Minimalizuje flaky testy.
* Použití page.inner\_text('body') pro ověření výstupu – Test se nefixuje na konkrétní umístění chybové zprávy.

**test\_kuchynske\_potreby\_3.py**

* Každý test se zaměřuje na jednu věc: test\_cart\_count() řeší počet produktů v košíku. test\_cart\_sum() řeší správnost celkové ceny.
* Použití pytest.fixture() pro browser a stránku – Zajišťuje správnou správu prohlížeče a minimalizuje opakování kódu.
* Správné zavírání prohlížeče pomocí yield – Zabrání problémům s nesprávným ukončením testů.
* Interakce s prvky stránky (click(), press()) – Test simuluje skutečné chování uživatele.
* Použití page.wait\_for\_load\_state("networkidle") – Zajišťuje, že stránka se načte před interakcí.
* Použití page.locator("button:has-text('Koupit')") – Dynamické vyhledávání tlačítek koupit (místo pevného CSS selektoru).
* assert No\_of\_products\_in\_basket == "2" – Kontroluje, zda v košíku jsou skutečně 2 produkty.
* Porovnání cen produktů v test\_cart\_sum() – Ověřuje, že celková cena odpovídá součtu cen jednotlivých produktů.

**Co by šlo vylepšit:**

**test\_kuchynske\_potreby.py**

* Momentálně jsou všechny selektory přímo v testovací funkci. Každý krok testu obsahuje page.locator("CSS\_SELECTOR"), což není dobře udržovatelné. Pokud se změní struktura HTML, je nutné opravovat celý test. Lepší řešení: Přesunutí selektorů do jedné třídy
* Některé click() operace nemají wait\_for\_selector(). Např. nejlevnejsi\_button.click() se může provést před načtením stránky.
* Momentálně je cena parsována ručně (replace(" Kč", "").strip()). To je náchylné na změny ve formátování HTML. Co když některé ceny mají Kč a jiné ne? Co když je formát jiný (1 290 Kč vs. 1290 Kč)?

**test\_kuchynske\_potreby\_2.py**

* Momentálně test očekává, že chybová zpráva je vždy viditelná. Pokud se stránka chová jinak (např. přihlášení nevyvolá chybovou zprávu, ale jen zůstane na stránce), test selže. U testu s prázdným heslem (("", "", "")) není jasné, co se očekává.
* page.click() a page.fill() by měly mít předem wait\_for\_selector(). Momentálně test klikne na přihlašovací tlačítko bez ověření, zda je dostupné. Na pomalejších serverech to může způsobit flaky testy.

**test\_kuchynske\_potreby\_3.py**

* Pokud test test\_cart\_count() proběhne, v košíku zůstanou produkty. Druhý test test\_cart\_sum() spoléhá na předchozí test, ale neprovádí vlastní přidání produktů. Co když test test\_cart\_count() selže? test\_cart\_sum() nebude mít správná data.

**Závěrem:**

**test\_kuchynske\_potreby.py** Tento test je dobře zpracovaný. Určitě jsou zde oblasti, které lze zlepšit, ale to vše přijde praxí. Dobrá práce

**test\_kuchynske\_potreby\_2.py** Opět test je skvěle napsaný. Není v něm zásadní chyba. Dobrá práce.

**test\_kuchynske\_potreby\_3.py** Budu se opakovat. Test je skvělý. Jsou v něm oblasti, které zlepšit. Ale jinak Dobrá Práce

Celkově je jedná o skvěle zpracovaný projekt. Gratuluji, projekt schvaluji.