

עבודת בית: אפליקציית Web למסחר עם אוטומציה (Playwright ← עגלת ← Checkout (באמצעות UI

1. מטרה

לבנות אפליקציית Web קטנה אשר:

1. מאפשרת למשתמש להזין שאלת חיפוש (query + פילטרים).
2. מפעילה אוטומציה דרך דפדפן (Selenium או playwright) כדי לבצע:

- חיפוש באתר מסחר (amazon או אתר דומה)
- איסוף תוצאות דרך ה-DOM
- הוספה לעגלת
- מילוי תהליך Checkout

3. מדגימה שימוש אחראי ב-AI, ארכיטקטורה נקייה, ולוגים ובדיקות.

2. אתר יעד (חובה לבחור אתר אחד)

- רק אם האתר כולל:
 - קטלוג מוצרים
 - אפשרות בחירה / חיפוש / סינון
 - עגלת

3.דרישות פונקציונליות

3.1 אפליקציית Web (המערכת שאתם בונים)

האפליקציה שלכם היא "מעטפת" להפעלת האוטומציה ולהציג התוצאות. מסכים מינימליים:

- מסך חיפוש: שדה query + פילטרים (למשל מחיר מקסימלי)
- מסך תוצאות: רשימת מוצרים (שם, מחיר, תמונה אם קיימת) + כפתור Buy
- מסך סטטוס / עגלת: הציג מצב הריצה של האוטומציה
- מסך תוצאה: מילוי שדות רכישה

ה-UI אינו מציג נתונים מ-API חיצוני.

כל הנתונים מגיעים מהאוטומציה (scraping דרך UI).

3.2 שכבה אוטומציה (חוובת Playwright או Selenium)

וש לבנות מודול אוטומציה שמבצע תהליך End-to-End מול האתר.

שלבים נדרשים:

1. פיתוח דפדפן (Debug Headless או Headed במצב

2. Login לאתר (אם נדרש)

3. ניווט לעמוד הקטלוג / החיפוש

4. הפעלת חיפוש או סינון בהתאם ל-query מה-UI שלכם

5. שיליפה תוצאות דרך ה-DOM והחזרתן בפורמט מנורמל:

`id`

`title`

`price`

`currency`

`product URL`

`source`

6. בחירת מוצר לפי מדיניות מוגדרת (לדוגמה: המוצר הזול ביותר או הראשון ברשימה)

`Add to cart` .7

8. מעבר ל-Checkout

9. מילוי פרטי משלוח

10. צילום מסך (Screenshot)

חוובת לשמור Screenshot של מסך הרכישה C-proof.

4. דרישות הנדסיות

4.1 ארכיטקטורה

נדרשת הפרדה ברורה בין שכבות:

Playwright, selectors, retries – תרחישי automation •

(Product, Cart, Order) – domain •

(Search, Purchase) – לוגיקה עסקית •

api שמבצעים את האוטומציה – endpoints •

ui – frontend •

Robustness 4.2

- שימוש ב-sleep explicit waits (ולא קבוע sleep)
- timeouts מוגדרים
- retry / backoff לפעולות שבירות
- וlidציה ל-input מה-UI
- טיפול שגיאות ברור עם הודעות יידוחיות למשתמש

Observability 4.3

- לוגים מובנים עם:
 - requestId
 - שם השלב
 - זמן ריצה לכל שלב
 - שגיאה במקרה כשל
- הציג trace בסיסי ב-UI: אילו שלבים הושלמו ואייפה הייתה תקלה.

4.4 שיקיפות שימוש ב-AI (חובה)

- יש לצרף קובץ בשם pd USAGE_AI הכלל:
- באילו כלי AI השתמשתם
 - 3-5 פרומטטים (כפי שנכתבו בפועל)
 - דוגמאות להמלצות AI שגויות או מסוכנות (selectors לא יציבים, waits שגויים וכו') ויכן תיקנותם
 - כיצד מנעתם דליפת סודות (סיסמות, טוקנים)

5. בדיקות (חוובה)

בדיקות ייחודית / שירות:

- נורמליזציה של מוצר (price כ-float, מטבע)

- מדיניות בחירת מוצר

- חישובי עגלת (אם קיימים)

בדיקות אינטגרציה / E2E (פחות אחת):

- ריצה מלאה: חיפוש → הוספה לעגלת → Success

- הבדיקה חייבת ליצור Screenshot של מסך האישור.

מה להגיש

- קישור לריפורזיטורי GitHub

- : README.md

- הוראות הריצה

- משתני סביבה (credentials לאתר דמו)

- תיאור זרימת האוטומציה

- AI_USAGE.md

- README_AI_BUGS.md

- פלט בדיקות

- Screenshot של מסך אישור הזמנה

קריטריוני הערכה (100%)

- 35% איקות האוטומציה (waits, selectors, retries, proof)

- 25% נכונות ה-flow (Checkout → עגלת → עגלת)

- 20% ארכיטקטורה והפרדת שכבות

- 10% בדיקות

- 10% תיעוד ופתרונות שימוש ב-AI