Puppeteer:

מזה?

ספרייה שרצה בסביבת נוד, רצה על כרום\כרומיום, נועדה לכתיבת טסטים ולהעביר אותם אוטומציה.

בנוסף, ניתן להשתמש בפאפטייר בשביל לעשות סקרייפינג לאתר רשת מסויים.

הבדלים בין פאפטייר ופאפטייר קור:

פאפטייר – לרוב בשביל אוטומציות, רוב ההגדרות כבר מוגדרות (אפשר לשנות אותן) והוא אוטומטית מתקים כרומיום.

פאפטייר קור – משמש לרוב במידה ואתה משתמש בדפדפן ספציפי שלך, ההגדרות שלו אינן מוגדרות וצריך להגדיר כל דבר קטן.

לפאפטייר יש 2 דרכים לרוץ:

HEADLESS – כשאי אפשר לראות מה קורה במסך (לכאורה מהר יותר)

או NONHEADLESS – כשאפשר לראות מה קורה במסך.

מה עוד אפשר לעשות עם פאפטייר?

לצלם מסך או לגנרט PDF

למלא פורומים, להספים שירות.

לקבל מידע אודות האתר שאותו אתה בודק לגבי מהירות ביצוע ומהירות טעינה.

לקדם אתרי רשת (לכאורה)

אז מה זה סקרייפינג?

גירוד תוכן מאתר אחד והצגתו באתר אחר, במידה ועושים אותו חכם ממש אפשר לגרד כל מידע ברחבי הרשת.

גבולות חוקיים:

ברמת החוק, כל עוד אין גירוד של פרטים שלא חשופים לכל אחד באינטרנט וכל עוד אין עניין של הפרת זכויות יוצרים השמיים הם הגבול!

איך כותבים?

כל פונצקיה של PUPPETEER היא אסינכרונית, וכל פקודה צריכה לקבל AWAIT.

נתחיל בלפתוח דפדפן עם const browser = await puppeteer.launch();

במידה ונרצה להשתמש במצב NONHEADLESS נרשום : {headless: false} בתוך הLAUNCH

וניצור כרטיסייה חדשה: const page = await browser.newPage();

בשביל לנווט אל האתר הרצוי:   
 await page.goto('https://developer.chrome.com/');

בשביל לעשות פעולות של ממש נצטרך לקבל PATH מאלמנטים ברחבי הדף.

בשביל זה יש כמה שיטות:

XPATH, CSSSELECTOR, CLASSNAME,BYID

במידה ונרצה לחכות לאלמנט שייטען למשל אלמנט עם CLASS מסויים:

const searchResultSelector = '.search-box\_\_link';  
await page.waitForSelector(searchResultSelector);

במידה ונרצה ללחוץ על אלמנט:

נקרא לו ונכתוב .CLICK()

במידה ונקרצה להקליד בתוכו: נקרא לו ונוסיף .TYPE()

בסוף כל פונקציה של פאפטייר צריך להוסיף:

await browser.close();

בשביל לסגור את הדפדפן.

פונקציות של JS בתור הקוד:

נשתמש במילת המפתח EVALUATE למשל:

const three = await page.evaluate(() => {  
 return 1 + 2;  
 });

שליפה של נתונים מהדפדפן:

יש כמה דרכים לשלוף נתונים, כמה קיצורים של EVALUATE:

יש את page.$eval(“path”,text=>text.Content)

או אם כבר יש לנו משתנה שמצביע על PATH אז פשוט:pathvar.$eval(text=>text.textContent)

עוד דוגמאות:

const searchValue = await page.$eval('#search', el => el.value);

const preloadHref = await page.$eval('link[rel=preload]', el => el.href);

const html = await page.$eval('.main-container', el => el.outerHTML);

יש את $$EVAL:

שמחזירה הבטחה, זה אומר שהיא תחכה עד שהיא תקבל תשובה בשביל להחזיר את הערך הרצוי

// get the amount of divs on the page

const divCount = await page.$$eval('div', divs => divs.length);

// get the text content of all the `.options` elements:

const options = await page.$$eval('div > span.options', options => {

return options.map(option => option.textContent);

});

זהו זה הבסיס.