



Offline Route planner using Web Automation

Prabdeep Singh Bagga¹ | Mrs. Narinder Kaur²

¹B. Tech Scholar, Information Technology Department, Maharaja Agrasen Institute of Technology, New Delhi, India

²Faculty, Information Technology Department, Maharaja Agrasen Institute of Technology, New Delhi, India

To Cite this Article

Prabdeep Singh Bagga, Narinder Kaur, "Offline Route planner using Web Automation", *International Journal for Modern Trends in Science and Technology*, 6(12): 365-369, 2020.

Article Info

Received on 16-November-2020, Revised on 09-December-2020, Accepted on 12-December-2020, Published on 17-December-2020.

ABSTRACT

Automation refers to decreasing repetitive human work, tedious tasks, and minimizing the errors. With the correct automation tools, it's possible to automate browser tasks, web testing, and online data extraction, to fill forms, scrape data, transfer data between applications, and generate reports.

The research project focuses on automating the task of placing an order of particular set of items from an online website. The main aim of the project is planning the most efficient route to visit all your stoppages and reach your destination. It automates the process of finding the most optimum route and saves a PDF of commute details to your disk.

KEYWORDS: Automation, Web Automation, Testing Tools, Web Testing, Puppeteer, Selenium

I. INTRODUCTION

Web automation is that the concept of letting software robots perform actions, tasks, and processes that involves an online browser or web application. it's a process of automatically performing operations on an internet browser, so as to attain speed and efficiency levels that wouldn't be possible with human intervention.

The uses of browser automation are practically limitless, and if you are doing a big number of tasks throughout your work day, an automation tool may help reduce the time spent on these tasks while retrieving accurate and conclusive data. as an example, upon starting a browser you wish five different websites to be loaded to a particular point. rather than manually opening website, entering login credentials and navigating to a selected page, the automation tool can perform all of these tasks automatically.

Web Automation can even be used for testing purposes. Testing is a vital a part of every software development process on which companies devote considerable time and energy. With the appearance of internet revolution and therefore the colossal rise within the development of web applications additionally as their corresponding usage, it's becoming mandatory for quality testing of web applications.

Web Application Testing [1] is gaining importance given the most important stake of economic relevance within the contemporary society. The cost of fixing a bug is directly proportional to the time of its discovery. The longer the time it takes to unearth a bug, the costlier it becomes to fix it because the software would have been distributed or under use by the customers.

STRUCTURE OF PAPER

The paper is organized as follows: In Section 1, the introduction of the paper is provided along with the structure, important terms, objectives and overall description. In Section 2 we have the complete information about automation, tools used in automation. Section 3 share information about Puppeteer, its advantages and disadvantages. Section 4 tells us about the methodology and the process description. Section 5 acknowledgement and references.

OBJECTIVES

Google Maps is one of the most reliable navigation services available, but it's not without its faults. Probably one of the biggest issues is that it works best with a solid data connection, something we can't always have. If you're going where the Internet is slow, mobile data is expensive, or you can't get online, you can save an area from Google Maps to your phone or tablet and use it when you're offline

The main aim of the project is planning the most efficient route to visit all your stoppages and reach your destination. It automates the process of finding the most optimum route and saves a PDF of commute details to your disk.

II. AUTOMATION

Browser automation helps you in preventing tedious and menial tasks, like filling out long HTML forms. Various skill levels can be accommodated using the automation tools. A non-programmer might simply record some test scripts, while software developers and programmers can make more sophisticated scripts and libraries.

Web browser automation is just like recording the steps involved in completing a particular transaction, and play it back by injecting JavaScript into the target sites, providing the required results. These web automation tools resemble macros, but are way more flexible and complicated.

Most things that you simply can do manually within the browser will be done using Puppeteer [3]

Here are some examples to induce you started:

- Generate screenshots and PDFs of pages.
- Crawl a (Single-Page Application) and generate pre-rendered content (i.e. "SSR" (Server-Side Rendering)).
- Automate form submission, UI testing, keyboard input, etc.
- Create an up-to-date, automated testing environment. Run your tests directly within the latest version of Chrome using the most recent JavaScript and browser features.
- Capture a timeline trace of your site to assist diagnose performance issues.
- Test Chrome Extensions.

Automation Tools [4]

1. Puppeteer: Puppeteer is a Google product which allows you to manage Chrome with node scripts. you'll be able to automate headless instance of Chrome using this node library. It doesn't unlock anything new, but it abstracts many details you have got to cater to, without using it.

2. Selenium: Selenium also supports headless browser automation. because it doesn't have the GUI, user cannot see the screen of test execution. Selenium relies on Selenium IDE and Webdriver.

3. Protractor: Protractor could be a web automation framework to check angular applications. It supports locating elements which are specific to angular websites. Protractor also supports headless browser automation.

4. Katalon Studio: it's another automation tool which you'll use to check Web Application, API, Desktop applications and Mobile applications. Katalon studio also supports Headless browser testing. This tool is additionally built on top of Selenium.

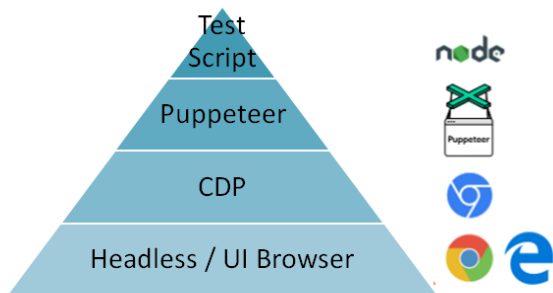
III. PUPPETEER

We will be using Puppeteer [5] for our web automation because it provides the ability of headless chrome. Puppeteer is a Node library that gives a high-level API to manage Chrome or Chromium over the DevTools Protocol. It runs headless by default but may be configured to run full (non-headless) Chrome or Chromium.

Headless Chrome [6] is essentially a Chrome browser without UI. A Headless browser is more helpful for the programmers as they'll write the automation script very easily with up-to-date rendering of the scripts. It may be used for network throttling, device emulation and code coverage.

How Puppeteer Outweighs Selenium [7]

Selenium may be a widely used web automation tool, which supports different languages. Parallel testing is additionally possible in Selenium, and it's most useful after you automate an oversized number of browser instances, which are remotely distributed making it heavy. Also, object recognition in Selenium also becomes very difficult when the application is very big and changes are continuously occurring.



Advantages of Puppeteer over Selenium

- Puppeteer requires zero set-up effort and comes bundled with the Chromium version with which it works best, making it very easy to begin with. It's an event-driven architecture, which removes lots of potential synchronization issues.
- Puppeteer is a great tool for debugging: flip the “headless” bit to false, add “slowMo”, and you'll see whatever is happening in the browser. Open Chrome DevTools to examine the test environment. It helps you to crawl a Single-Page Application and generate pre-rendered content (i.e. “SSR”-Server-Side Rendering). It can easily automate form submissions, computer program testing, inputs from keyboard, etc.
- It can create an up-to-date, automated testing environment, run your tests within the latest version of Chrome using the most recent JavaScript and every one new browser features. Puppeteer can capture a timeline trace of website to assist tracking performance bottlenecks and it's used for testing Chrome extensions.

- Puppeteer gives more power to control the Chrome browsers than Selenium Webdriver. Puppeteer excludes the outbuilding on an external driver to run the tests. Despite this problem in Selenium, it can be reduced by using Boni García's WebDriverManager dependency.

- You can test without loading the pictures within the application using Puppeteer tool which isn't possible in Selenium.

- By default, Puppeteer is about to execute in headless mode, but you can change/alter it for watching the execution board non-headless mode. It will be used for checking the proportion of CSS/JS files which are used for loading a page which isn't feasible in Selenium.

- Puppeteer also allows you to check the time taken to load the page but the identical feature isn't available in Selenium. It helps in testing new DevTools Protocol features and identifying bugs early. Executing the test in numerous devices using the emulators is feasible in Puppeteer but emulating a tool in Selenium is difficult.

Puppeteer Limitations

The complexity and automation context are changing with each passing day and hence, one tool may not be the answer for all. As every automation tool, Puppeteer has some limitations like, it supports only Chrome browser and Puppeteer Firefox is a work in progress. If the user base is more varied in its browser preferences, it should be wise to opt for other testing frameworks like Cypress.io, TestCafe or Selenium Web Driver.

IV. METHODOLOGY

The aim of the project is to provide hands-free experience to the users by automating each and every task that is done manually. The project focuses on taking input from the user, the list of products that the user wishes to order.

The idea is to take the input from user either from keyboard or by using Google's Speech API (to convert speech to text). With the integration of Google Speech API input is taken from the user in the form of speech input and then converted to text. All the products that are available at the

website are added to the cart with the help of automated script and order is executed by filling up all the necessary details, forms etc. using the script.

Therefore, the project eases the task of placing an order on a website for a user by automating everything from speech input to adding desired products to the cart and filling up all the forms. This provides a completely hands-free experience to the users and they can just sit back and relax.

The project would also provide the user with real time notifications for a Price Drop of various products that the user wishes to buy. The project can also be used to compare prices of products from different sellers.

Process Description

- The user is required to input the source destination and stoppages in a json format.
- The input is then processed by the automation script on a Headless Browser (such as Google Chromium).
- User will be automatically redirected to the google maps navigation using the source, destination and stoppages.
- The automated Script would search the different routes by permuting the stoppages provided.
- As different results are displayed on the browser automation script takes screenshots of different routes
- All the routes are saved into an offline PDF and is saved locally on the users machine

We can keep the project running on our local system to get real time notification for Price Drops of various products. Further we can buy these products by using the above automated script.

Limitations

The whole process of automation is dependent on scraping/reading data from the websites on which we want to automate things. As these websites are Dynamic and their structure may change over time, so as a result our scripts would fail in that case.

We constantly need to keep the script updated so that it can find the required DOM elements from the webpage and perform automated tasks. The method of taking input in the form of speech can vary from person to person and can sometimes lead to incorrect results. The best possible results can thus be obtained by text input or water floating temperatures. In recent years, it has been reported that heat causes changes in the mechanisms involved in regulating body fluids.

Pournot (2011) examined the short-term impacts of various immersions in recovery from intense activity. In this study, the effects of various immersion recovery techniques on the power and maximum power of athletes have been investigated, and the inflammatory response has been evaluated.

CONCLUSION

The uses of browser automation are practically limitless, and if you are doing a large number of tasks throughout your work day, an automation tool may help reduce the time spent on these tasks while retrieving accurate and conclusive data.

With the advent of internet revolution and also the colossal rise within the development of web applications moreover as their corresponding usage, it's becoming mandatory for quality testing of web applications.

Web Application Testing is gaining importance given the main stake of economic relevance within the contemporary society. The value of fixing a bug is directly proportional to the time of its discovery. It provides speed and efficiency levels that wouldn't be possible with human intervention.

V. ACKNOWLEDGEMENT AND REFERENCES

Acknowledgement

I am very thankful to Mrs. Narinder Kaur who have contributed towards development of this template and have motivated and helped me in each and every step. I would also like to thank the pioneers of research in automation and testing of web applications who proposed several novel techniques and inspiring me to build a project using automation and further pursue a direction for effective testing of web applications.

REFERENCES

1. Brian Marick. "When Should a Test Be Automated?". StickyMinds.com. Retrieved 2009-08-20.
2. Browser Automation Tutorials, Green Technology-<http://www.biodesignautomation.org/browser-automation-defined.html>
3. Puppeteer. (2020, June 24) Puppeteer Documentation. <https://pptr.dev/>
4. Software Test Automation Abdul Rauf EMa,*, E.MadhusudhanaReddybaResearch and Development Centre, Bharathiyar University, Coimbatore-641014
5. Google Web Developer Tools-<https://developers.google.com/web/tools/puppeteer>.
6. Headless Testing with Browsers <https://docs.travis-ci.com/user/gui-and-headless-browsers>
7. AathiraShaji V S. (2020, Feb 17) "Puppeteer – Web Automation with Headless Chrome". RapidValue. https://www.rapidvaluesolutions.com/tech_blog/puppeteer-web-automation-with-headless-chrome/

I J M T S T