Testathon Participant Guide

Welcome to a hands-on event designed to help you learn and sharpen your test automation skills and explore cross-browser testing using BrowserStack’s test platform.

This guide includes everything you need to prepare and participate in this hackathon. Prerequisites

Skillset

You should be comfortable with :

- Writing test scripts in Java, NodeJs, Python, C#, PhP or Ruby.

- Using a test automation framework (example: Selenium, Playwright, Cypress, Puppeteer, etc).

- Basic knowledge of how websites work (elements, locators, etc)

Setup

- A laptop

- A GitHub account. You will create a private repository and invite us as collaborators to submit your code.

(https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-personal account-on-github/managing-access-to-your-personal-repositories/inviting-collaborators-t o-a-personal-repository)

- Access to BrowserStack tools. You will be provided access just before the event.

Tools you will use

During the hackathon, you will be required to use BrowserStack’s test platform.

| Tool | Purpose | Documentation | Video |
| --- | --- | --- | --- |
| Test Management | Document your test cases | https://www.browsers tack.com/docs/test-m anagement | https://www.browsers tack.com/docs/test-m anagement/overview/ demo |

| Automate | Run your test scripts across  browsers/devices | https://www.browsers tack.com/docs/autom ate/  https://www.browsers tack.com/docs/autom ate/selenium/sdk-ben efits |  |
| --- | --- | --- | --- |
| Test reporting and analytics | Analyze test results | https://www.browsers tack.com/docs/test-re porting-and-analytics | https://www.youtube. com/watch?v=\_J6iqR \_-7u4&ab\_channel=B rowserStack |

Event Overview

In this hackathon, you will be developing an automation test suite for a website, from design to execution. The website will be revealed on the day of the event.

You will work individually or in teams of up to 3 to complete the entire challenge in 3 hours. Step 1: Test Case Identification and Documentation

Design a test plan. Use Test Management to create, organize, and document test cases covering all the critical user flows in the application. Ensure all edge cases and error scenarios are covered, too.

Use the “Generate with AI” feature to get kick-started!

Deliverable: A well-organised suite of documented test cases in Test Management. Share a link to the project, and add it to your project’s readme.

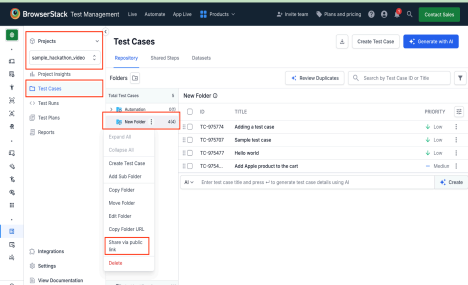
- Choose the Folder you want to share.

- Click the 3 dots beside the folder's name.

- Select Share via Public Link

- Enable Public Link

- Copy Link given the box



Helpful Documentation Links:

https://www.browserstack.com/docs/test-management/overview/getting-started Step 2: End-to-End Automation with Automate

Implement an end-to-end automation suite for the test cases using Automate. Use the BrowserStack SDK to integrate the automation test framework of your choice in the programming language of your choice with Automate.

Automate also allows you to replay your test recordings and debug them.

Deliverable: A set of functional automation scripts with successful runs across target browsers/devices. Create a new GitHub project, upload your code, and share the link. - Create a Public repo.

- Do not use BrowserStack API username and Access in the code.

- Use Placeholders or Env Variable while submission

- Share the Repo link.

Helpful Documentation Links:

https://www.browserstack.com/docs/automate/playwright/getting-started/nodejs?fw-lang =nodejs

Step 3: Integrating Your Tests with Test Management

Integrate your Automation Tests with Test Management to monitor the suite’s

performance and analyse failure patterns.

Deliverable: Link to the dashboard. Share a link to the project, and add it to your project’s README.

- Go to your Project.

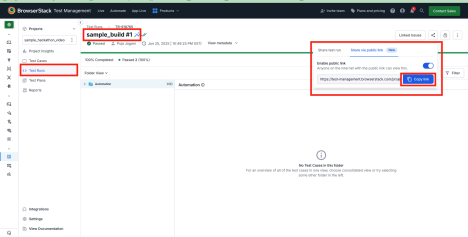
- Click on Test Runs.

- Select the Test Build you want to share.

- Click on the Share icon.

- Select Share via Public Link.

- Copy Link.

Helpful Documentation Links:

https://www.browserstack.com/docs/test-management/test-runs/test-case-tagging

NOTE: If your Language or Framework does not support Integration or if you were not able to complete, then you can skip the previous step of sharing. Go to Step 4 instead.

Step 4: Test insights using Test Reporting and Analytics

Use Test Observability to monitor the suite’s performance and analyse failure patterns. Deliverable: Link to the dashboard. Share a link to the project, and add it to your project’s README.

- Go to Build Runs

- Click the Share button in the top right.

- Select Public Link.

- Copy Link

-

Note: The website you will test emulates real-world scenarios, so you might see bugs and inconsistencies across various browser and platform combinations. The goal is to catch them via solid tests.

Link for Submission:

https://docs.google.com/forms/d/1nFS3UwROana2dqVXOhNraUc2FaoatK 1z9nY7\_6mroeE/preview

Judging Criteria

We will be judging equally across these areas.

To help you stay on track and deliver your best, here is our evaluation plan. 1. Thought through test coverage

We want to see how well you understand the website’s functionality.

Make sure your test cases cover:

a. All critical user flows (eg, login, product search, add to cart, checkout, etc.).

b. Important edge cases and negative scenarios. Don’t just focus on happy paths.

c. A clear structure in test case folders in BrowserStack Test

Management.

2. Quality of automation

a. Thorough end-to-end tests that run across different browsers and device combinations. Leverage BrowserStack SDK for this.

b. Follow best coding practices based on the language selected.

c. The test automation should reflect your proposed test scenarios, but also should be easy to follow. (Think of this as developing a maintainable test suite.)

d. Ensure a clear project structure and naming.

Happy testing!