

Load Testing and Performance Evaluation of the Theia Online IDE

Bachelor Thesis Presentation

Author: Tobias Klingenberg

Supervisor: Prof. Dr. Stephan Krusche

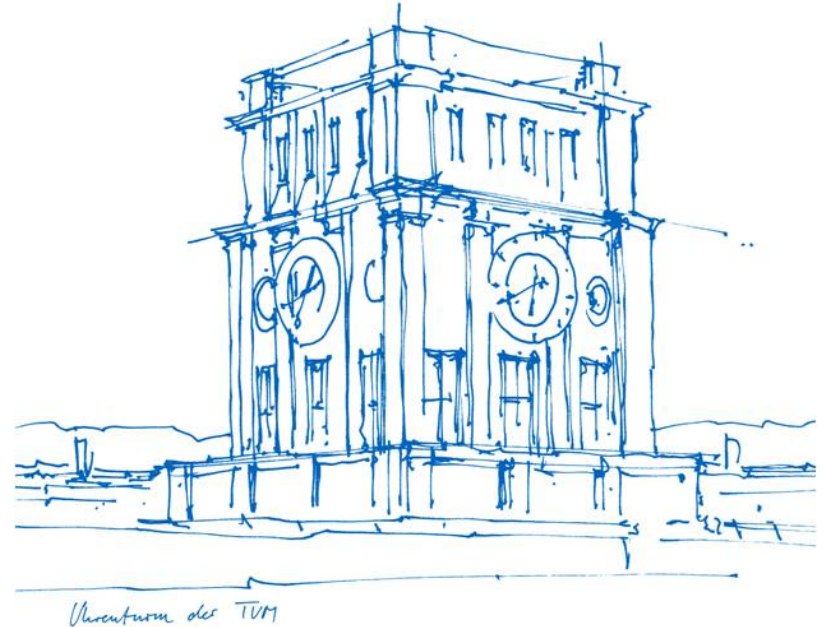
Advisor: Matthias Linhuber

Technische Universität München

TUM School of Computation, Information and Technology

Applied Education Technologies

Garching, 23. October 2025



Outline

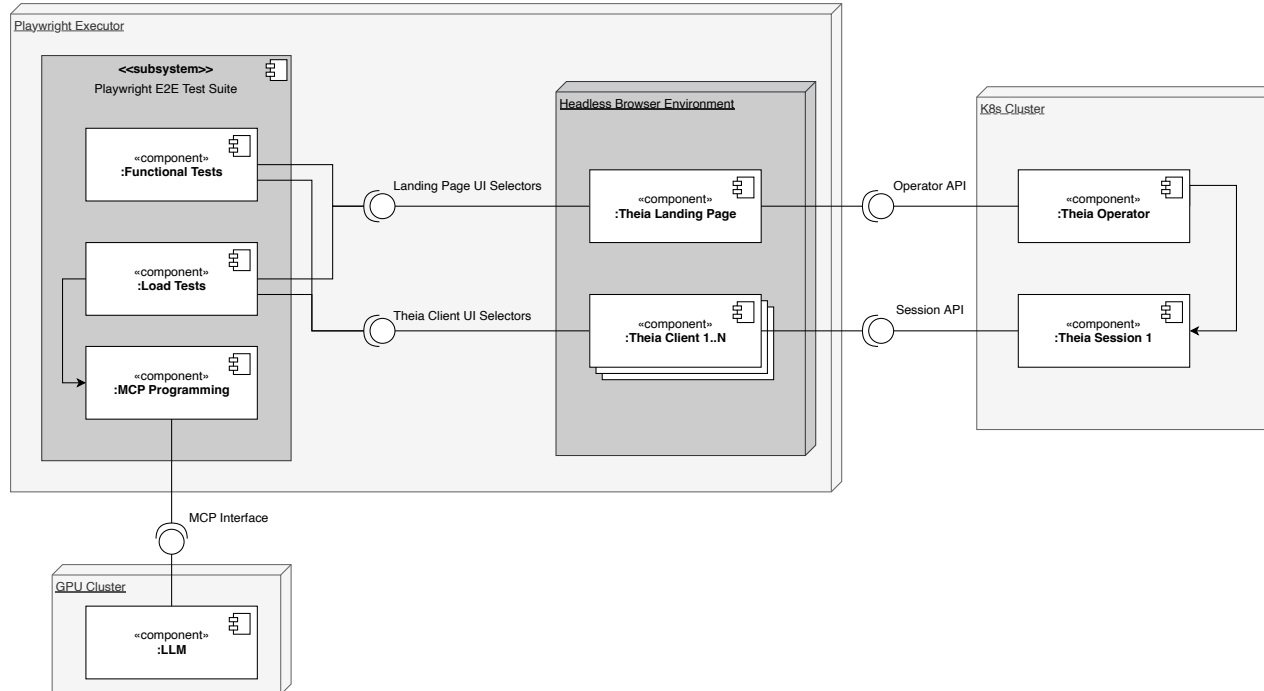
1. Introduction
2. Problem and Motivation
3. Objectives
4. Functionality Tests
5. Load Testing
6. Evaluation
7. MCP Testing / Demo
8. Questions

Introduction

Motivation and Problem

Objectives

Top Level Design



Functionality Tests

Functionality Tests



Playwright

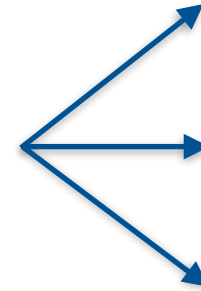


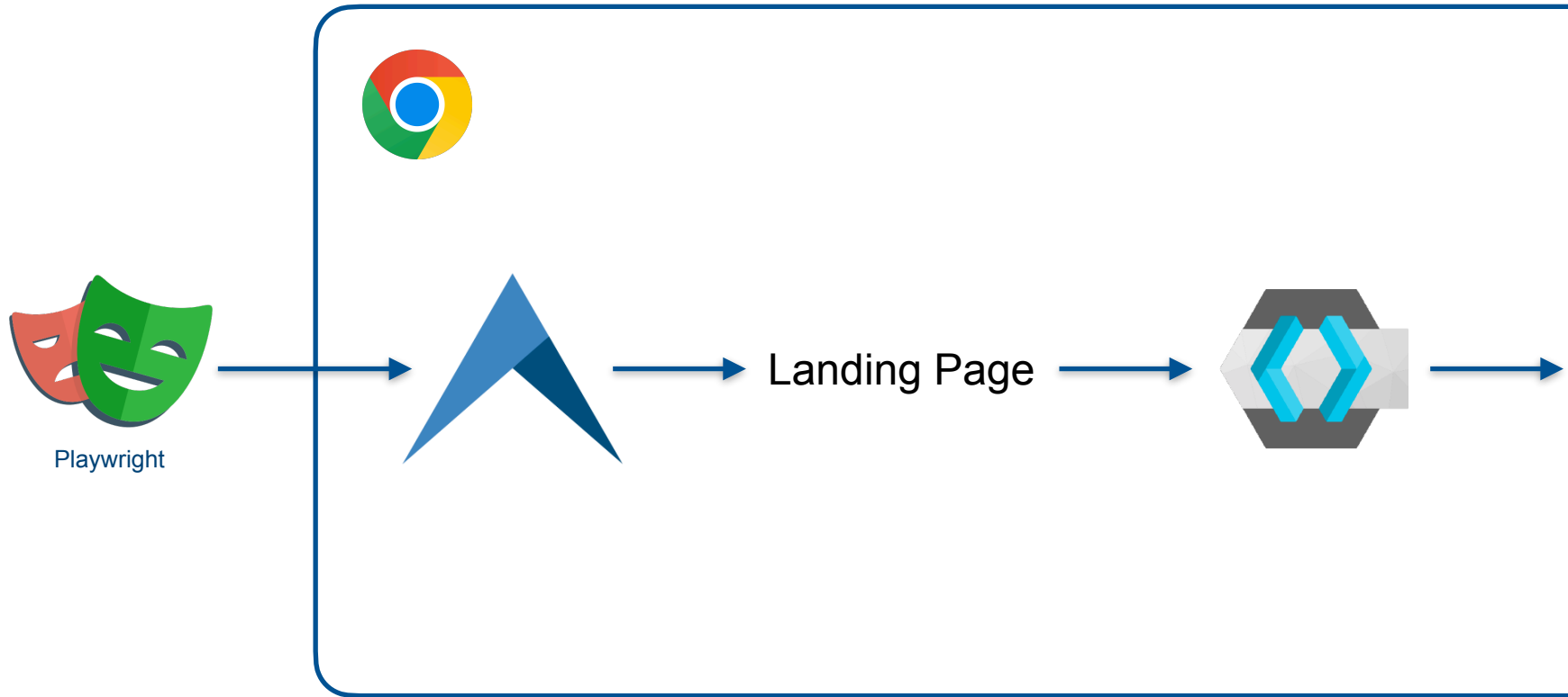
Test Suite

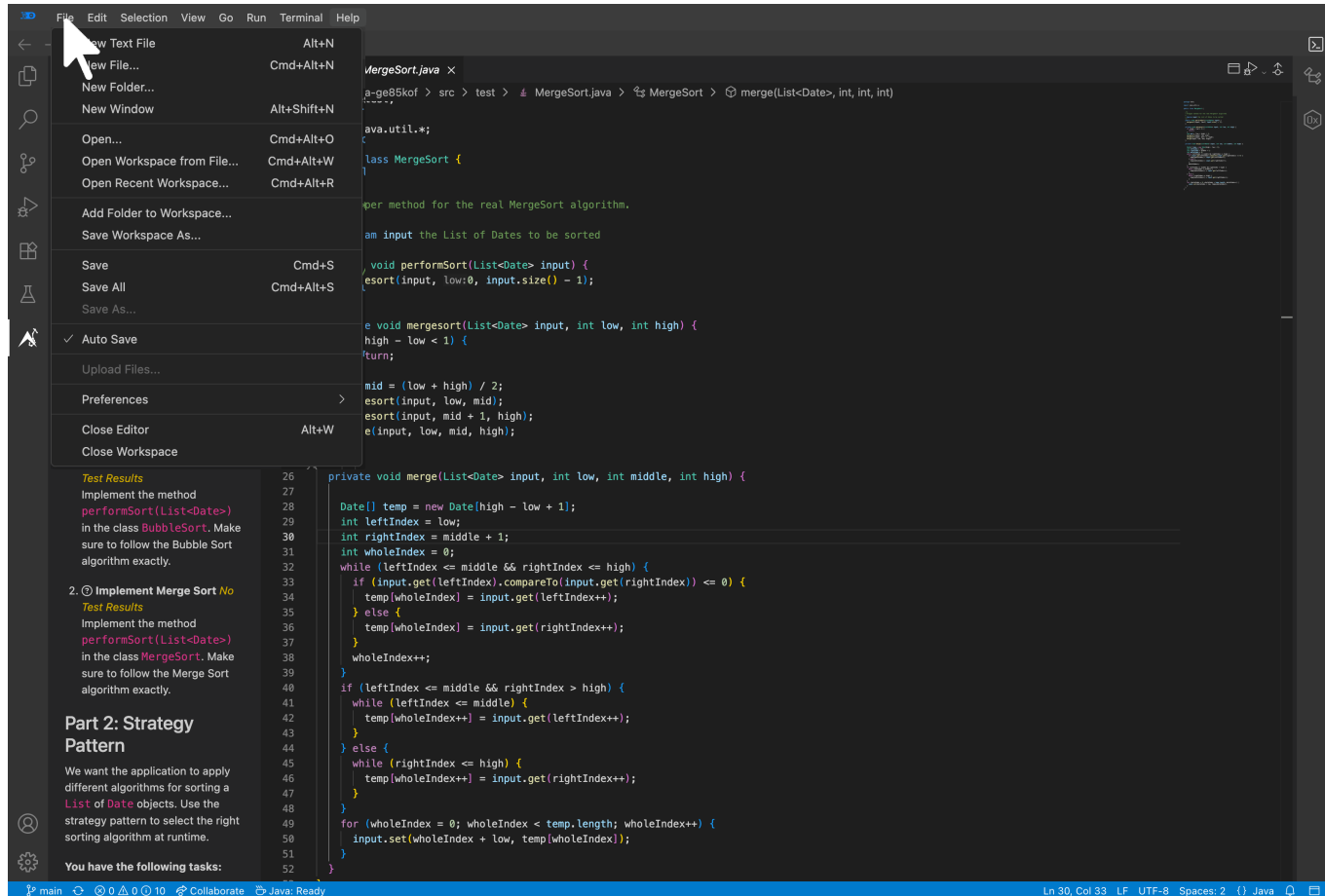
Websockets



Playwright







The screenshot shows an IDE with a Java project. The main editor displays the `MergeSort.java` file, which implements a Merge Sort algorithm. The code includes a `performSort` method and a `merge` method. The sidebar on the left contains instructions for implementing the Merge Sort algorithm.

File Menu:

- New Text File (ALT+N)
- New File... (CMD+ALT+N)
- New Folder...
- New Window (ALT+SHIFT+N)
- Open... (CMD+ALT+O)
- Open Workspace from File... (CMD+ALT+W)
- Open Recent Workspace... (CMD+ALT+R)
- Add Folder to Workspace...
- Save Workspace As...
- Save (CMD+S)
- Save All (CMD+ALT+S)
- Save As...
- ✓ Auto Save
- Upload Files...
- Preferences
- Close Editor (ALT+W)
- Close Workspace

Test Results:

Implement the method `performSort(List<Date>)` in the class `BubbleSort`. Make sure to follow the Bubble Sort algorithm exactly.

2. Implement Merge Sort No

Test Results

Implement the method `performSort(List<Date>)` in the class `MergeSort`. Make sure to follow the Merge Sort algorithm exactly.

Part 2: Strategy Pattern

We want the application to apply different algorithms for sorting a `List of Date` objects. Use the strategy pattern to select the right sorting algorithm at runtime.

You have the following tasks:

```

26 private void merge(List<Date> input, int low, int middle, int high) {
27
28     Date[] temp = new Date[high - low + 1];
29     int leftIndex = low;
30     int rightIndex = middle + 1;
31     int wholeIndex = 0;
32     while (leftIndex <= middle && rightIndex <= high) {
33         if (input.get(leftIndex).compareTo(input.get(rightIndex)) <= 0) {
34             temp[wholeIndex] = input.get(leftIndex++);
35         } else {
36             temp[wholeIndex] = input.get(rightIndex++);
37         }
38         wholeIndex++;
39     }
40     if (leftIndex <= middle && rightIndex > high) {
41         while (leftIndex <= middle) {
42             temp[wholeIndex++] = input.get(leftIndex++);
43         }
44     } else {
45         while (rightIndex <= high) {
46             temp[wholeIndex++] = input.get(rightIndex++);
47         }
48     }
49     for (wholeIndex = 0; wholeIndex < temp.length; wholeIndex++) {
50         input.set(wholeIndex + low, temp[wholeIndex]);
51     }
52 }

```

Ln 30, Col 33 LF UTF-8 Spaces: 2 {} Java

The screenshot shows an IDE with a file menu open on the left. The menu items are:

- New Text File (ALT+N)
- New File... (Cmd+ALT+N)
- New Folder...
- New Window (ALT+SHIFT+N)
- Open... (Cmd+ALT+O)
- Open Workspace from File... (Cmd+ALT+W)
- Open Recent Workspace... (Cmd+ALT+R)
- Add Folder to Workspace...
- Save Workspace As...
- Save (Cmd+S)
- Save All (Cmd+ALT+S)
- Save As...
- ✓ Auto Save
- Upload Files...
- Preferences >
- Close Editor (ALT+W)
- Close Workspace

The main editor displays Java code for a MergeSort algorithm. The code is as follows:

```

MergeSort.java
a-ge85kof > src > test > MergeSort.java > MergeSort > merge(List<Date>, int, int)
...
ava.util.*;
class MergeSort {
    // per method for the real MergeSort algorithm.
    // an input the List of Dates to be sorted
    void performSort(List<Date> input) {
        esort(input, low:0, input.size() - 1);
    }
    e void mergesort(List<Date> input, int low, int high) {
        high - low < 1) {
            'turn;
        }
        mid = (low + high) / 2;
        esort(input, low, mid);
        esort(input, mid + 1, high);
        e(input, low, mid, high);
    }
    private void merge(List<Date> input, int low, int middle, int high) {
        while (leftIndex <= middle && rightIndex <= high) {
            if (input.get(leftIndex).compareTo(input.get(rightIndex)) <= 0) {
                temp[wholeIndex] = input.get(leftIndex++);
            } else {
                temp[wholeIndex] = input.get(rightIndex++);
            }
            wholeIndex++;
        }
        if (leftIndex <= middle && rightIndex > high) {
            while (leftIndex <= middle) {
                temp[wholeIndex++] = input.get(leftIndex++);
            }
        } else {
            while (rightIndex <= high) {
                temp[wholeIndex++] = input.get(rightIndex++);
            }
        }
        for (wholeIndex = 0; wholeIndex < temp.length; wholeIndex++) {
            input.set(wholeIndex + low, temp[wholeIndex]);
        }
    }
}

```

The bottom panel shows test results and instructions for implementing the Merge Sort algorithm. The instructions state: "Implement the method performSort(List<Date>) in the class MergeSort. Make sure to follow the Merge Sort algorithm exactly." The bottom status bar shows "Ln 30, Col 33 LF UTF-8 Spaces: 2 {} Java".

`expect(page.locator(".menu-widget")).toBeVisible()`

Functionality in Test



Editor



Search



Terminal



VCS

Load Testing

Load Testing

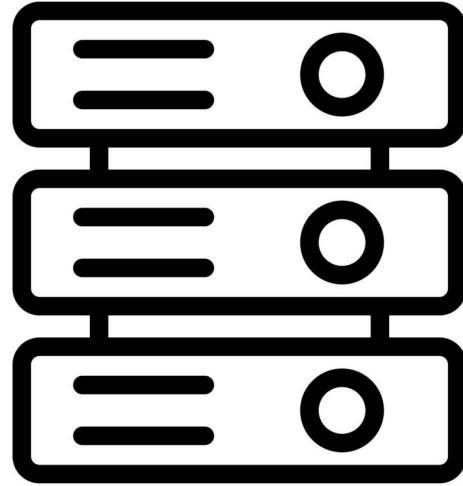
Load Testing



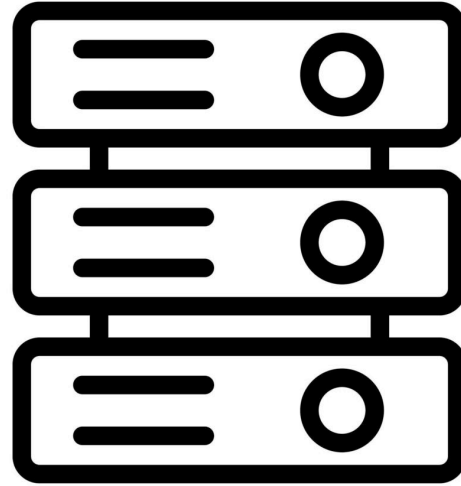




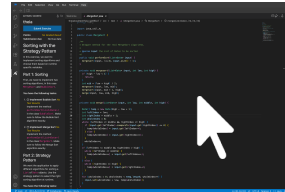
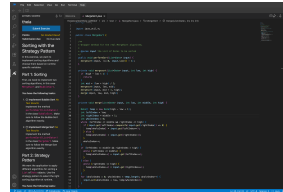
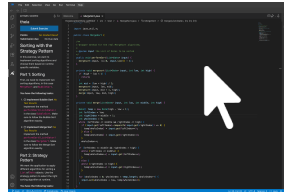
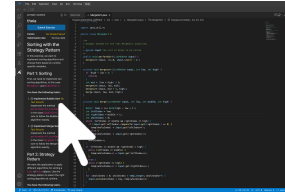
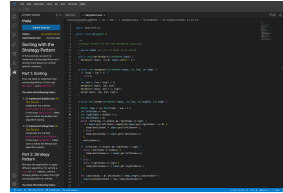
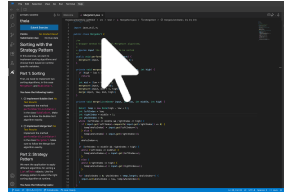
N - Instances



Prewarming (Preview...)

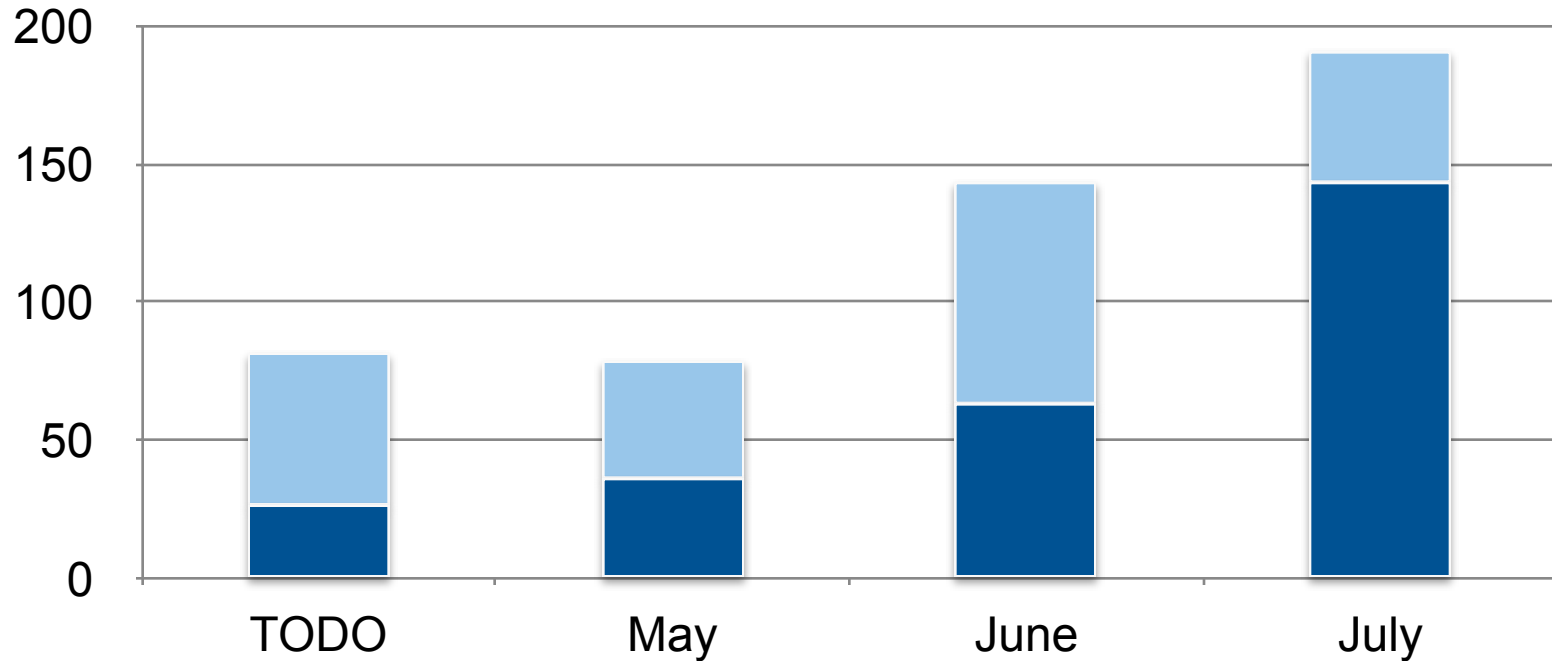


Prewarming (Preview...)



N - Instances

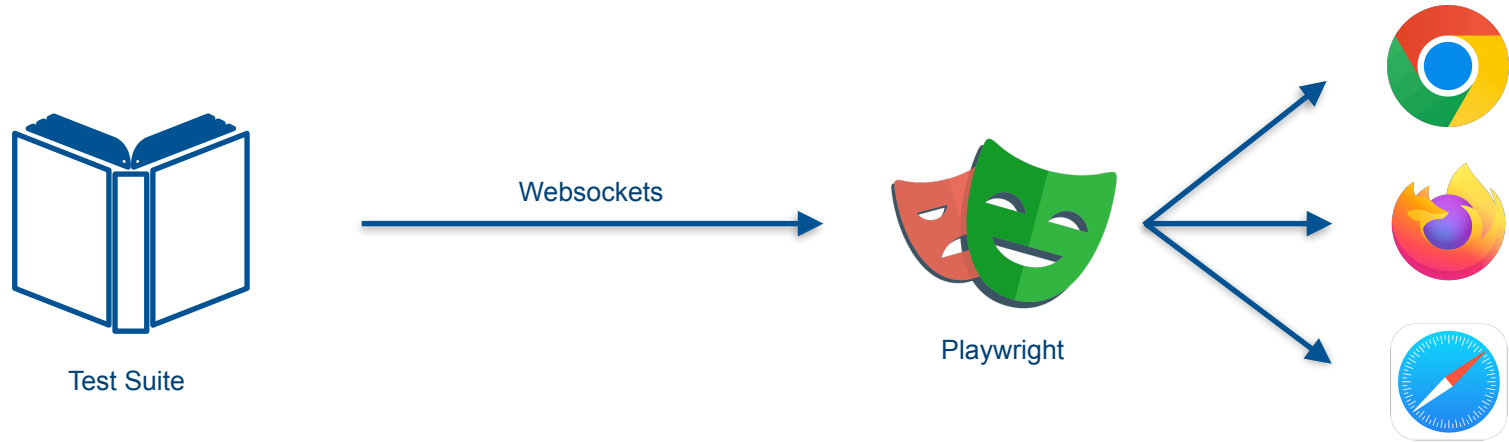
Evaluation



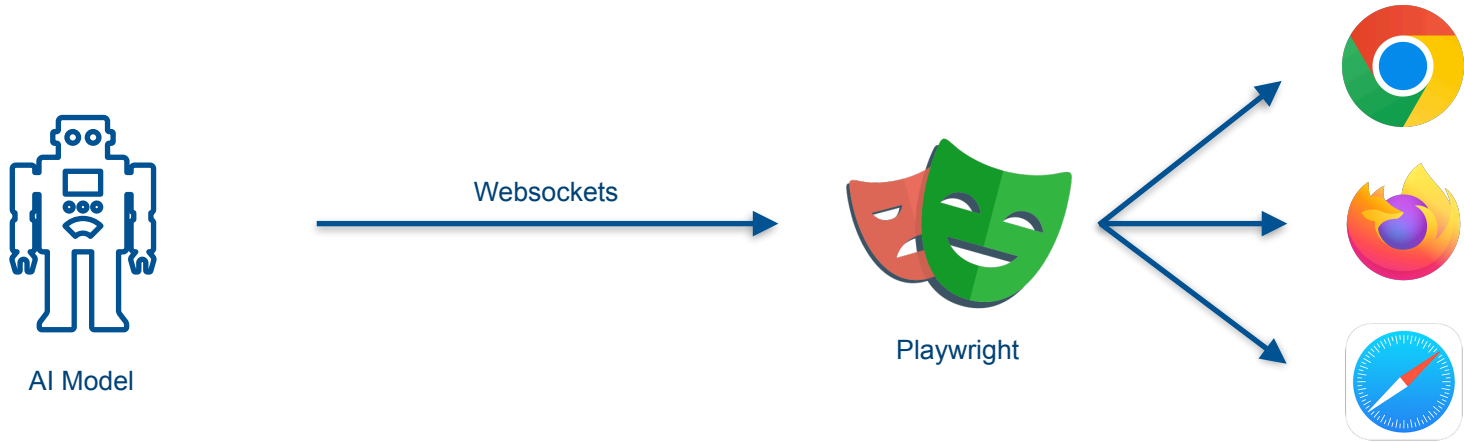
What is MCP

Model Context Protocol

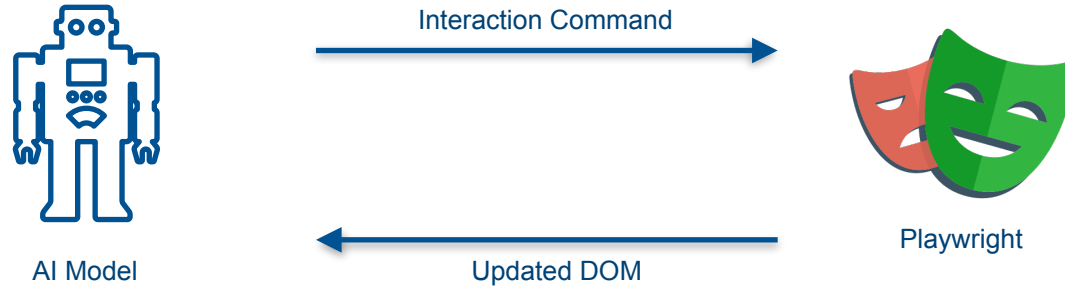
How can we utilize MCP for testing



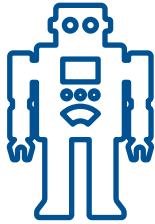
How can we utilize MCP for testing



How can we utilize MCP for testing



How can we utilize MCP for testing



AI Model

Prompt:

Imagine you are a student with a programming exercise. You are given an Online IDE with all the known functionality. Interact with the Online IDE and behave like a student trying to solve the task.

DEMO



Questions

Thank you for listening!

