

Performance Dashboard for Continuous Benchmarking of HPC Libraries

Chingun Ariunbat, Max Schik, Walter Alexander Böttcher,
Darius Schefer, Jamil Bagga

2021-05-06

1 Introduction

Glossary acronym example:

CI

Continuous Integration

Continuous Integration (CI)

$$\underbrace{\text{http}}_{\text{protocol}} : // \underbrace{\text{web.io}}_{\text{host}} / \underbrace{\text{index}}_{\text{path}} ? \underbrace{\text{argument=somevalue}}_{\text{parameter}} \# \underbrace{\text{theAnchor}}_{\text{fragment}}$$

more placeholder

2 Goals

2.1 Required

heading	NICHT IMPLEMENTIERT	M1
yet more placeholder		
Schnelle Weiterleitung Kurz- zu Lang-URL		M2
Implementiert durch:	FR1	
Authentifizieren mit E-Mail oder Facebook		M3
Implementiert durch:	FR2	
Rechtliche Vorgaben werden eingehalten		M4
Implementiert durch:	FR3 FR4	
template		

2.2 Optional

Authentifizieren mit Github		K1
Implementiert durch:	FR2	
Seite mit Betreiberinfo	keine entsprechende Anforderung	K2
template		

2.3 Limitation

Keine Wahl Kurz-URL		A1
template		

3 Usage

template

4 Product Environment

template

5 Functional Requirements

Schnelle Weiterleitung	NICHT GETESTET	FR1
Implementiert: M2		
template		
template	NICHT GETESTET	FR2
Implementiert: M3 K1		
template		
Auf jeder Seite ist ein Link „Impressum“	NICHT GETESTET	FR3
Implementiert: M4		
template		
Auf jeder Seite ist ein Link „Datenschutz“	NICHT GETESTET	FR4
Implementiert: M4		
template		
Daten werden persistent gespeichert	NICHT GETESTET	FR5
Implementiert:		
template		

6 Nonfunctional Requirements

Modernes Design

NF1

template

Persistenz

NF2

template

Erweiterbarkeit

NF3

template

7 Tests

8 Scenarios

Scenario name: pushAndInspect

Participating actor instance: Bopp: Developer

- Bopp pushes his work to a git repository and fires off a benchmark test
- Bopp opens the web app and selects his last pushed change
- Bopp chooses a type of visualization
- The app creates the given type of visualization with the benchmark results from the selected change

Scenario name: visualizeFromTemplate

Participating actor instance: Jeremy: User

- Jeremy opens the web app
- Jeremy chooses a template for a visualization
- Jeremy chooses which commit he wants to visualize
- The app creates the given type of visualization with the commit

Scenario name: saveTemplate

Participating actor instance: Jeremy: User

- Jeremy opens the web app
- Jeremy configures a visualization
- Jeremy saves his visualization as a template for future use

Scenario name: inspect

Participating actor instance: Jeremy: User

- Jeremy wants to see the latest performance benchmarks for the project
- Jeremy opens the web app and selects the latest change
- Jeremy chooses a benchmark to compare

- Jeremy chooses a type of visualization by selecting which value to plot on the x axis and which value on the y axis
- The app creates the given type of visualization with the benchmark results from the selected change

Scenario name: compareImplementations

Participating actor instance: Jeremy: User

- Jeremy wants to know which implementation is the fastest
- Jeremy opens the web app and selects a benchmark
- Jeremy selects commits from different branches containing different implementations
- Jeremy chooses a type of visualization by selecting which value to plot on the x axis and which value on the y axis.
- The app creates the given type of visualization with the benchmark results from the selected change

Scenario name: pushAndCompare

Participating actor instances: Bopp: **Developer**

- Bopp pushes his work to a git repository, and fires off a benchmark test.
- Benchmark results are fed into the database.
- Bopp opens the webapp and selects his last pushed change>
- Bopp selects a previous change that he wants to compare to.
- Bopp chooses a type of visualization.
- The app creates the given type of visualization with the benchmark results from the selected changes.

Scenario name: badPerformance

Participating actor instances: Bopp: **Developer**

- Bopp pushes his work to a git repository, and fires off a benchmark test.
- Benchmark results are fed into the database.

- Our dashboard-backend realizes that the benchmark data for this change is far worse than usual.
- Bopp gets notified that his last pushed change significantly worsened the performance and the related details about that.

Scenario name: impossiblePerformance

Participating actor instances: Bopp: Developer

- Bopp pushes his work to a git repository, and fires off a benchmark test.
- Benchmark results are fed into the database.
- Our dashboard-backend realizes that the benchmark data for this change is theoretically impossible.
- Bopp gets notified that his last pushed change has improved the performance above the theoretical maximum and the related details about that.

Scenario name: authentication

Participating actor instance: Jeremy: User

- Jeremy opens the webapp.
- Jeremy gets prompted for a authentication.
- Jeremy logs in over Github/Gitlab/other services.

Scenario name: shareVisualization

Participating actor instance: Jeremy: User

- Jeremy found an interesting visualization for something.
- Jeremy clicks a *share* button next to the visualization.
- Jeremy gets a link he can share with others that redirects them to the exact same visualization.

Scenario name: visualizeCommitWithoutBenchmark

Participating actor instance: Jeremy: User

- Jeremy opens the webapp and wants to visualize benchmarkdata for a specific commit. This commit has no benchmark data attached to it, only the commit before and the commit after.

- Jeremy can't click on the commit because it is greyed out.

Scenario name: takeVisualizationFromHistory

Participating actor instance: Jeremy: User

- Jeremy opens the webapp and visualizes something. He then visualizes something else. His previous visualizations are stored in a list somewhere.
- Jeremy decides to take another look at a previous visualization.
- Jeremy picks his previous visualization and gets the previous visualization.

Glossar

CI Continuous Integration.

Developer Person working on the project that is to be benchmarked.