

Exposé

Internationale Hochschule Duales Studium

Studiengang: Informatik

**Generative Programmierung mittels künstlicher Intelligenz**

Müller Korbinian

Matrikelnummer: 102302316

Adresse

Lohwald Straße 59

86356 Neusäß

Abgabedatum: 30.09.2024

Inhalt

[Einleitung 3](#_Toc191381531)

[Theoretische Fundierung 3](#_Toc191381532)

[Forschungsfrage 3](#_Toc191381533)

[Methodik 3](#_Toc191381534)

[Gliederungsentwurf 3](#_Toc191381535)

[Zeitplan 3](#_Toc191381536)

[Vorläufiges Literaturverzeichnis 3](#_Toc191381537)

Einleitung

Theoretische Fundierung

Forschungsfrage

In den Vorgestellten Wissenschaftlichen Arbeiten wurden Aspekte der Asynchronen Programmierung, Performance vergleiche zwischen verschiedenen Programmiersprachen dargestellt. Jedoch beschreiben die Quellen den Zusammenhang von Asynchroner Programmierung und dem daraus Resultierenden Performance Einfluss nur

Die Forschungsfrage, die in der Wissenschaftlichen Arbeit, gestellt und beantwortet werden soll ist: Wie beeinflusst die Asynchronität von Operationen innerhalb eines PHP Rest Endpunkt die Performance in Lasttests?

Methodik

Die Forschungsfrage soll mithilfe eines Experimentes überprüft werden. In dem Experiment werden die beiden Endpunkte eines Lastentestes unterzogen. Dabei soll herausgefunden werden wie die Performance der API sich verhält, wenn das Anfrage Volumen sich erhöht. Dabei wird mit einem API Test-Tool Anfragen an den Endpunkt generiert. Das Volumen der Anfragen wird während des Testes stück für stück erhöht. Insbesondere wird Überprüft wie die Antwortzeit in Relation zu den Anfragen steht.

Für das Experiment wird in PHP mit dem Framework Slim eine RestAPI entwickelt. Die API stellt zwei Endpunkte zur Verfügung. Der Synchrone wird mit Standard PHP entwickelt. Im Asynchronen kommt zusätzlich noch Guzzle zum Einsatz. Dadurch werden die Asynchronen Features ermöglicht. Die beiden Endpunkte sind gleich Strukturiert. Die Anfrage wird angenommen und anhand der Parameter eine Query in einer MySQL Datenbank ausgeführt. Dabei wird darauf geachtet, dass es sich um eine sehr Komplexe Query handelt. Dadurch wird sichergestellt, dass die Abfrage eine Zeit für die Ausführung benötigt. Die Ergebnisse der Datenbankabfrage werden in JSON-Format aufbereitet und als Response zurückgesendet. Dies bietet eine realistische Nutzung eins Rest Endpunkte.

Das Experiment wird auf einem Desktop PC ausgeführt. Die Systemumgebung, MySQL-Datenbank und Apache Webserver, wird mittels Docker erzeugt. Dafür werden die Bereits bestehenden Container webdevops/php-apache und mysql verwendet. Dadurch ist die Systemumgebung klar von der Maschine getrennt. Des weiteren ist es dadurch möglich die Systemumgebung für jeden Durchlauf zurückzusetzen. Dadurch wird sichergestellt, dass die vorherigen Experimente die Nachfolgenden nicht beeinflussen.

Die Überprüfung der Antwortzeit erfolgt mit der Anwendung Artillery. Dabei handelt es sich um eine Node CLI Anwendung, die dafür geschaffen wurde um die Performance von Webanwendungen zu testen. Es wird definiert wie Lange die Testphase dauert und wie viele Anfragen pro Sekunde gesendet werden. Im ersten Schritt wird über eine Zeit von 5 Minuten die Anzahl der Anfragen auf 1000 pro Sekunde erhöht. Dies dient der ersten Einschätzung. Falls dabei festgestellt wird, dass beide Endpunkte bei 1000 Anfragen pro Sekunde keine Nennenswerte Antwortzeit aufweisen, dann wird die Maximalbelastung in 1000 Schritte erhöht bis Nennenswerte Ergebnisse vorliegen. Bei dem Testen wird fünf Sekunden als maximal Zugelassene Antwortzeit betrachtet.

Artillery liefert Daten für die Minimale, Maximale und Mittlere Antwortzeit zurück. Ebenso auch das 99 Perzentil. Die Ergebnisse der Endpunkte wird miteinander Verglichen. Falls die Asynchrone Programmierung einen Einfluss auf die Performance besitzt, dann werden die Messdaten für die Asynchronen Endpunkt niedriger Ausfallen, als für den Synchronen. Im Falle das die Mittlere Antwortzeit und das 99 Perzentil bei beiden Endpunkten gleich ist, können die Minimalen und Maximalen Antwortzeiten einen Einblick in das Verhalten im Extrembereich liefern.

Gliederungsentwurf

1. Einleitung, 1.5 Seiten
2. Theoretische Fundierung, 4 Seiten
   1. PHP und Slim
   2. Asynchrone Programmierung
   3. Lasttest und Performance-Metriken
3. Methodik, 5 Seiten
   1. Aufbau des Experimentes
   2. Die Verwendeten Daten
   3. Strukturierung der Lasttests
   4. Die Implementation der RestAPI
4. Ergebnisse des Experimentes, 3 Seiten
5. Fazit, 1.5 Seiten
6. Literaturverzeichnis

Zeitplan

|  |  |
| --- | --- |
| 1.5.2025-29.06.2025 | Quellen Suche und Validation |
| 30.06.2025-13.07.2025 | Verfassen der Theoretischen Fundierung und Methodik |
| 14.07.2025-20.07.2025 | Bufferzeit, für die Überarbeitung und Verbesserung des Verschriflichten |
| 21.07.2025-27.07.2025 | Implementierung der Entpunkte |
| 28.07.2025-03.08.2025 | Bufferzeit, bei Problemen in der Implementierung |
| 4.08.2025-10.08.2025 | Durchführen des Experimentes |
| 11.08.2025-24.08.2025 | Analyse, Implikation und Fazit verfassen |
| 25.08.2025-31.08.2025 | Verfassen der Einleitung |
| 1.09.2025-14.09.2025 | Korrekturlesen, Ausbessern der Formulierungen |
| 15.09.2025-21.09.2025 | Bufferzeit, für Nachbesserungen nach dem Korrektur lesen. |
| 22.09.2025 | Abgabe der Wissenschaftlichen Arbeit |
| 30.09.2025 | Deadline für die Abgabe |

Vorläufiges Literaturverzeichnis

Textquellen

Internetquellen

Technology | 2024 Stack Overflow Developer Survey. (2024)

https://survey.stackoverflow.co/2024/technology#most-popular-technologies