

API Performance Optimization using Caching in Django Framework

Author: Gaddam Swagath

Bachelor of Technology, Computer Science and Engineering

Vivekananda Institute of Technology and Science, Karimnagar, India

Email: sswagath46@gmail.com

GitHub: <https://github.com/swagath088/>

Abstract

Modern web applications require fast and efficient backend systems to provide good user experience. One of the major causes of slow performance is repeated database access. Caching is a technique used to store frequently accessed data in memory, which reduces response time and improves performance. This project analyzes the performance improvement of web APIs using caching techniques in Django framework. The system measures response time with and without caching. Experimental results show that caching significantly reduces response time. The project demonstrates how caching can optimize backend performance and improve system efficiency.

Introduction

Web applications rely heavily on backend systems to process user requests and return data. When the number of users increases, the backend system may become slow due to repeated database queries. This leads to increased response time and poor user experience. To solve this problem, caching techniques are used. Caching stores frequently accessed data in memory, allowing faster

access compared to retrieving data from a database. This research focuses on analyzing API performance and measuring the improvement achieved using caching in Django framework.

Problem Statement

Backend systems often experience performance issues due to repeated database access. Every time a user requests data, the system queries the database, which increases response time. This project aims to analyze and compare API response time with and without caching to demonstrate performance improvement.

Methodology

The system was developed using Django, a Python-based web framework. An API endpoint was created to provide student data. Response time was measured using Python's time module. Initially, the system retrieved data without caching, and response time was recorded. Then, caching was implemented using in-memory storage. The response time was measured again. The results were compared to analyze performance improvement.

Implementation

The backend system was implemented using Django and Python. An API endpoint was created to return student information. To simulate real-world database delay, a time delay was introduced using the `time.sleep()` function. A caching mechanism was implemented using in-memory cache to store API responses. When the API is called for the first time, data is retrieved from the database and stored in cache. For subsequent requests, data is retrieved from cache instead of database. This reduces response time significantly.

Results and Analysis

The performance of the system was measured with and without caching.

Without Cache:

Source: DATABASE

Response Time: 1.0004 seconds

With Cache:

Source: CACHE

Response Time: 0.0 seconds

The results clearly show that caching improves system performance. Retrieving data from cache is much faster than retrieving data from database. This demonstrates the effectiveness of caching in backend systems.

Conclusion

This project demonstrates the importance of caching in improving backend performance. The experimental results show that caching significantly reduces API response time. The system achieved faster performance after implementing caching. This research highlights how caching can be used to optimize web application performance. The project provides a practical example of backend performance optimization using Django framework.

Future Scope

Future improvements include integrating real database systems such as MySQL and implementing advanced caching mechanisms such as Redis. This will further improve system scalability and performance.

References

1. Django Official Documentation – <https://docs.djangoproject.com>
 2. Python Official Documentation – <https://docs.python.org>
 3. Web Application Performance Optimization Techniques
-

Project Repository

GitHub Link: https://github.com/swagath088/research_project