

ABSTRACT

This study compares the implementation and performance of monolithic and microservice architectures on the Dekstra village service backend under increasing and sudden workloads and examines implications for architecture selection. A comparative quantitative experiment used equivalent business functions, endpoints, test data, workload distribution, and server environment. Locust generated workloads for load, stress, and spike testing, while Prometheus monitored CPU and RAM usage. Metrics covered response time, throughput, error rate, and CPU and RAM usage. The monolithic implementation ran all functions in a single application and primary database, whereas the microservice implementation separated them across three services with separate databases and synchronous inter-service communication. Both implementations used Redis and Celery workers for asynchronous processing. Microservice produced lower response times, higher throughput, and lower error rates. During load testing, microservice maintained successful service delivery while throughput increased. Monolithic showed a sharp increase in response time, limited throughput growth, and errors throughout testing. Performance declined in both architectures during stress testing, but the proportion of successful requests was higher for microservice. During spike testing, microservice recorded lower error rates. After workload reduction, the microservice response time decreased and it maintained more successful requests, although error rates increased in both architectures and service performance had not returned to normal conditions. CPU usage was similar during load testing. Microservice used more CPU during stress testing and the recovery phase of spike testing and required more RAM throughout the tests. Under the tested configuration, microservice was more suitable for workloads prioritizing processing capacity and successful service delivery during increasing loads. The monolithic architecture required less memory and provided a more centralized management structure.

Keywords: *monolithic, microservice, backend, performance testing, load handling.*