Implementing Multithreading in the Parking System Server

for

Master of Science

Information Technology

Peter Fedor

University of Denver College of Professional Studies

June 1st, 2025

Faculty: Nathan Braun, MS

Director: Cathie Wilson, MS

Dean: Michael J. McGuire, MLS

Introduction

For this assignment, I was tasked with adding multithreaded handling of requests to the single threaded parking server I created last week. The goal was to make it so that the server could handle multiple registration requests at once. As we had learned, multithreading comes at a cost unless proper precautions and corrections are implemented. In this case, I used thread pooling to handle the requests.

Design

The design for this assignment was very straight forward. Previously, the design made it so that each request was queued, and the server would handle each request one at a time. This was a good example of single threaded processing on the server side. To fix this, I introduced thread pooling. The server was updated to use the new Client Handler class, which takes care of all of the communication between the clients and the server. This includes interpreting and executing all commands that the server receives. The newFixedThreadPool creates a pool of threads to be used to accept requests to the server. Each time a client sends a request to the server, the server will then send a new Client Handler task to the pool of threads. This makes it so that there is not long that aforementioned queue, and connections do not block each other, and more connections can be made to the server. With this, I also added logging messages so that important event information is reported, such as the time to connection and the request being made. This allows the user to see the multithreading in action as multiple requests are being handled at the same time.

Challenges and Problem Solving

Though this assignment did take me a while to get working properly, I did find it quite satisfying. Since the idea of multithreading was new to me, it did take me quite some time to figure out the best method to implement it in the parking system. I originally wanted to use synchronization but could not figure out how to use that with the previous code. After some research, I found that thread pooling would be the best and easiest way to implement multithreading into the system. I also realized that thread pooling would be more

scalable, as I could easily change the number of threads in the pool to allow for more connections to the server if needed. I also found that testing the multithreading was hard to do. Since I am running commands through a terminal, I had to keep running the same code with a different name than the user and car to ensure that it was working properly. This made me unsure if it was working, but after many tries and restarting the server, I was able to ensure that it was working properly.

Conclusion

Overall, this assignment was not too challenging. I now better understand the use of multithreading, the best approaches to implement it into a system. Before these updates, the system was only able to handle one request at a time. This is unrealistic to real world applications, as a server will receive many requests simultaneously. I thought of it as if Walmart only had one checkout lane, it would take forever to go shopping. Outside of the parking application, I can see how this would be great for webservers when multiple clients are sending requests to the server. The implementation of multithreading into the parking system was a valuable and interesting assignment that showed me a practical application for the use.

References

"Thread Pools (the Java Tutorials > Essential Java Classes > Concurrency)." n.d. Docs.oracle.com. https://docs.oracle.com/javase/tutorial/essential/concurrency/pools.html.

"Thread Pools in Java." 2017. GeeksforGeeks. June 8, 2017. https://www.geeksforgeeks.org/thread-pools-java/.

Paraschiv, Eugen. 2016. "Introduction to Thread Pools in Java | Baeldung." Www.baeldung.com. August 10, 2016. https://www.baeldung.com/thread-pool-java-and-guava.