Group 7- Project Proposal

Instructor-In-Charge - Dr. Kunal Korgaonkar

Group Members

Arnav Goyal - 2021A7PS2596G - f20212596@goa.bits-pilani.ac.in
Aryan Nambiar - 2021A7PS2619G - f20212619@goa.bits-pilani.ac.in
Pranav Bajpai - 2021A7PS2062G - f20212062@goa.bits-pilani.ac.in
Amey Patil - 2021A7PS2740G - f20212596@goa.bits-pilani.ac.in

Project Title

Implementing a multithreaded web server in Rust and showcasing its concurrency advantages over C.

Outline

The digital era necessitates high-performing, secure, and concurrent web solutions, especially for file storage systems. This proposal outlines the development of a multi-threaded web server using Rust to facilitate seamless and concurrent file storage operations. Built upon Rust's robust safety and concurrency guarantees, our solution aims to outclass traditional C-based web servers.

Proposed Advantages over C

1. **Memory Safety**: Rust’s ownership system guarantees safety against issues like null pointer dereferences, dangling pointers (through borrow checker), or buffer overflows — problems often encountered in C-based applications.
2. **Concurrency**: Rust’s concurrency model is designed to catch data races at compile-time. This drastically reduces the potential for race conditions, a dominant challenge in multi-threaded web servers.
3. **Immutable By Default**: In Rust, data is immutable unless explicitly specified otherwise. Combined with its ownership paradigm, concurrent programming becomes safer and more intuitive.
4. **Performance**: Rust’s performance metrics should be faster than C and C++ as it is a strongly typed language and favors compile time checks over runtime checks reducing overhead during runtime and hence improving the performance of web server.

Conclusion

By leveraging Rust for our multi-threaded web server, we anticipate a high-performing, secure, and reliable solution. Rust's inherent safety guarantees, especially in the context of concurrency, position it as a superior alternative to traditional C-based web servers.