### **Lab 14:** **Build with Cargo in Rust**

**Exercise: Building a Simple Rust Project**

In this exercise, We will create a Rust project, add dependencies, and build the project using cargo.

1. Create a new Rust project using cargo:
2. Open your terminal/command prompt and run the following command:

cargo new math\_operations

cd math\_operations

Open the Cargo.toml file in the root directory of your project. You can use any code editor for this.

Add a dependency to the rand crate for generating random numbers:

[dependencies]

rand = "0.8"

Open the src/main.rs file and implement a simple program that generates two random numbers and performs basic mathematical operations (addition, subtraction, multiplication, and division) on them:

use rand::Rng;

fn main() {

let mut rng = rand::thread\_rng();

let num1: i32 = rng.gen\_range(1..=100);

let num2: i32 = rng.gen\_range(1..=100);

println!("Number 1: {}", num1);

println!("Number 2: {}", num2);

let sum = num1 + num2;

let difference = num1 - num2;

let product = num1 \* num2;

println!("Sum: {}", sum);

println!("Difference: {}", difference);

println!("Product: {}", product);

}

Save the files and return to your terminal/command prompt.

Build your project using cargo build:

cargo build

The build process will download and compile the required dependencies and create an executable binary in the target/debug directory. You can run the program by executing the generated binary:

./target/debug/math\_operations

The program will generate two random numbers and perform basic mathematical operations on them. The results will be displayed on the console.

Example Output:

Number 1: 35

Number 2: 85

Sum: 120

Difference: -50

Product: 2975

Division: 0.41

Congratulations! We have successfully completed the lab exercise on building Rust projects with cargo. We learned how to create a Rust project, manage dependencies using Cargo.toml, and build the project with cargo build. The cargo tool makes it easy to manage Rust projects and handle dependencies, streamlining the development process.

**Happy coding!**