RUSH Document

Elementary System Programming

Software Engineering Program,

Department of Computer Engineering,

School of Engineering, KMITL

67011090 Chanunyu Chinnawuth
67011352 Theepakorn Phayonrat

Preface

This project, titled RUSH Rust Shell Terminal, was undertaken as part of Elementary System Programming course in Software Engineering at KMITL. Throughout this project, We have gained valuable insights into developing System Software Program, teamwork management and developing workflow. This preface serves to outline the journey that led to the final outcome, which aims to contribute to the field of Software Engineering by providing by demonstrating the power of Rust in building fast and secure system-level software.

Abstract

This project, titled RUSH: Rust Shell Terminal, presents the design and implementation of a command-line shell application developed in the Rust programming language. As part of the Elementary System Programming course in Software Engineering at KMITL, RUSH was created to develop a fast, and efficient custom shell using the Rust programming language that provide users with typical features found in standard command shells (such as Bash, Zsh, etc.), including executing programs, navigating directories, file management, environment management while utilizing Rust's safety guarantees to minimize common security vulnerabilities.

The project demonstrates a range of system programming concepts, including process management, command parsing, and file handling, showcasing the capabilities of Rust in building performant and secure applications. RUSH supports essential command-line functionalities such as a help system, history logging, and automatic configuration of necessary files, providing a user-friendly and highly customizable terminal experience.

This work aims to contribute to the field of Software Engineering by offering a flexible and safe alternative to traditional shell interfaces, especially suited for developers seeking to understand system programming through the Rust language. The final outcome highlights Rust's effectiveness in system software development, encouraging further exploration of Rust for similar high-performance applications.

Contents

1 Introduction **4**
- 1.1 Project Overview . . . 4
- 1.2 Background . . . 4
- 1.3 Objective . . . 4

2 Project Technical Overview **5**
- 2.1 Commands . . . 5
- 2.2 Services . . . 6
  - 2.2.1 Command Combination Services . . . 6
  - 2.2.2 History Log . . . 6

3 Installation and Execution Guide **7**
- 3.1 Git Clone from the Remote Repository . . . 7
- 3.2 Build and Run the program . . . 7

4 Summary **8**
- 4.1 Learning Outcomes . . . 8
- 4.2 Accomplishment . . . 8

5 References **9**

6 Appendix **10**
- 6.1 Demonstration Video . . . 10
- 6.2 Demonstration Video . . . 10

Chapter 1

Introduction

1.1 Project Overview

RUSH (Rust Shell) project aims to develop a fast, and efficient custom shell using the Rust programming language.

RUSH will provide users with typical features found in standard command shells (such as Bash, Zsh, etc.), including executing programs, navigating directories, file management, environment management while utilizing Rust's safety guarantees to minimize common security vulnerabilities.

The RUSH terminal will provide users with different customization options to make your terminal look the way you prefer. Not only looks, but also how the terminal acts too. Users can tweak the behavior of the terminal to match their preferences.

1.2 Background

We wanted to make a command shell that is cross-platform, so it can be used on any operating system without having to re-learn the shell of different systems.

1.3 Objective

This project aims to create an efficient, high-performance and cross-platform shell as an alternative to other terminal emulators. The terminal will be perfect for using across operating systems so the users don't have to re-learn the different terminals for different systems.

Chapter 2

Project Technical Overview

2.1 Commands

RUSH provides a set of commands that cover various functionalities such as:

Command	Description	Usage
`cd`	Change directory.	`cd <path>`
`clr`	Clear the terminal.	`clr`
`exit`	Exit the terminal	`exit`
`find`	Find phrase in a file.	`find <phrase> <file>`
`help`	Display help.	`help <command>`
`log`	Show command history.	`log`
`ls`	List files and directories.	`ls`
`meow`	Concatenate file and print to standard output	`meow <file1> <file2> <file3>...`
`mkdir`	Create a directory.	`mkdir <directory>`
`mkfile`	Create a file.	`mkfile <file>`
`shout`	Print a message.	`shout <message>`

2.2 Services

RUSH provides a set of services that can improve your terminal such as:

2.2.1 Command Combination Services

1. | (Piping): Insert the returned value from the previous command to the next command.
2. Redirecting
   (a) > : Sends the command's output to a file, overwriting it if the file exists.
   (b) >> : Sends the output to a file, appending to it if the file exists.

2.2.2 History Log

RUSH provides a built-in history log system which stores recently executed commands. The history log file (rush.log) is located in:

For Windows: `OS_DRIVE:\Users\USERNAME\.rush\rush.log`

For Unix Based OS: `~/.rush/rush.log`

Chapter 3

Installation and Execution Guide

3.1 Git Clone from the Remote Repository

```
git clone https://github.com/Pottarr/RUSH-Rust-Shell-Terminal.git
cd RUSH-Rust-Shell-Terminal
```

3.2 Build and Run the program

```
cargo build
cargo run
```

Chapter 4

Summary

4.1 Learning Outcomes

- We have learnt some fundamental of System Programming.
- We have learnt how to design application both in UX and UI.
- We have learnt the workflow of project developing.
- We have learnt how to use Version Control to help developing application.

4.2 Accomplishment

We have created a terminal application which beginner friendly, have history logs and customizable

Chapter 5

References

- hecrj. (2024). *Rust iced crate Documentation 0.13.1.* Retrieved from `https://docs.rs/iced/0.13.1/iced/index.html`
- alexcrichton. (2024). *Rust Home Crate Documentation 0.5.9.* Retrieved from `https://docs.rs/home/0.5.9/home/index.html`
- rust-lang-owner. (2024). *Rust Standard Crate Document.* Retrieved from `httphttps://doc.rust-lang.org/std/`

Chapter 6

Appendix

6.1 Demonstration Video

6.2 Demonstration Video

https://www.canva.com/design/DAGVhiWupX4/L_cscDEmChqqI4FdjnQ26Q/view?utm_content=DAGVhiWupX4&utm_campaign=designshare&utm_medium=link&utm_source=editor