OSTİM TEKNİK ÜNİVERSİTESİ
ANKARA

Command line shell application

CENG204 Term Project

- Younes Nourzehi (210201910) - Section 1
- Faraz Azarmi (200201928) - Section 1
- Nazanin Haghdoust Talebinejad (200201929) - Section 1
- Rania M.S. Alshemarti (210201911) - Section 2

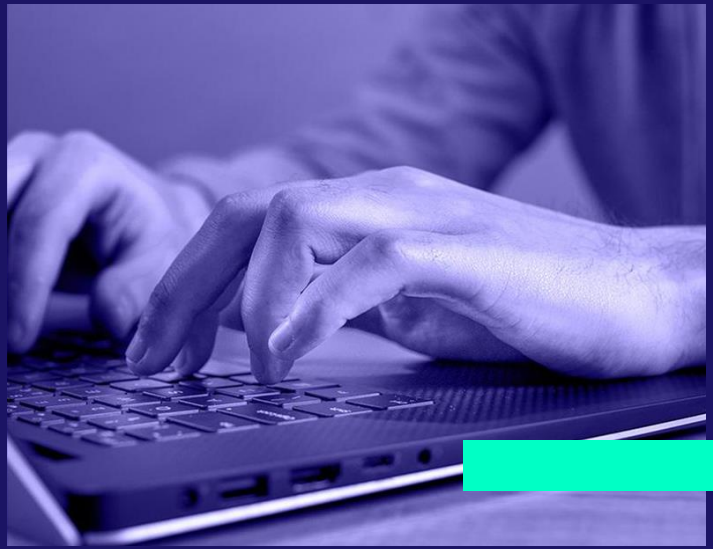
TABLE OF CONTENTS

01

Project Overview and Introduction

What is Shell?

A shell application is a program that provides a user interface for interacting with a computer's operating system. It is the first program that is loaded when a computer boots up, and it remains running until the computer is shut down. The shell provides a way for users to enter commands that tell the operating system what to do.



Types of shell Application

CLI

A Command Line Interface is a text-based interface where users type in commands and press Enter to execute them. CLIs are often used by system administrators and other technical users because they offer a high level of control over the operating system.

GUI

A GUI is a visual interface where users interact with the operating system by clicking on icons and buttons. GUIs are more user-friendly than CLIs, but they offer less control over the operating system.

“The command line is my canvas, and shell applications are my brushes.”

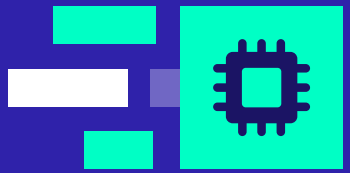
—Linus Torvalds

The creator of the Linux kernel

Advantages and Disadvantages of Using a CLI

More control

CLIs allow users to enter specific commands that tell the operating system exactly what to do.



Efficiency

CLIs can be more efficient than GUIs for tasks that require a lot of repetitive commands.

Power

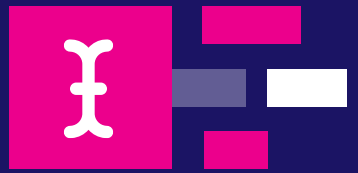
CLIs can be used to perform tasks that are not possible with GUIs.

Learning curve

CLIs can be difficult to learn

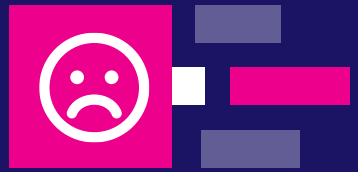
Text-based

CLIs are text-based, which can be difficult to read and use for long periods of time.



Not user-friendly

difficult to use for tasks that require a lot of interaction with the operating system.



Project Scope

The project's primary objective is to implement a set of essential commands:

`ls`

list files and directories by category

`cd`

change directory (forward and back)

`cat`

concatenate and display file contents

`rm`

remove files and directories

`mkdir`

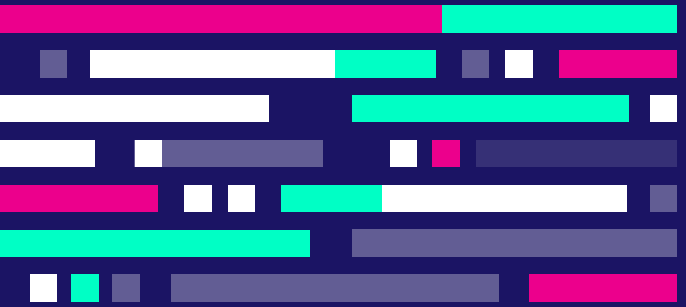
Create a new directory

`touch`

Create a new file

+Clear: Clear the console

Essentials, Limitations, and Expansions



While our focus lies on implementing these core features, we acknowledge that there are numerous additional commands and functionalities that could be incorporated into a command line shell application. However, due to the scope and time limitations of our project, we have prioritized the essential commands and key features to deliver a robust and reliable shell application.

02

Implementation

Language choice

we chose C for our project due to its **efficiency**, **portability**, and **familiarity**. Given the nature of the command line shell application, where direct access to system resources is essential, C offers the necessary control and performance. Additionally, its portability ensures that the application can be used on different platforms.

Challenges

The primary challenge encountered during this project was the implementation of the C programming language. While it would have been easier to use a higher-level language like Python or Node.js, C presented significant difficulties. Although our team had experience with Node.js, using C was particularly challenging and difficult.

One of the major challenges encountered was the inability to use libraries such as <unistd.h> and <dirent.h> because they are not supported by Windows and Visual Studio 2022. Consequently, the System function had to be used in some parts of the implementation.

Challenges

Another challenge was memory management. The program kept throwing errors related to memory problems, and the team could not resolve them. Ultimately, a completely new code had to be implemented. These problems would not have arisen in high-level languages due to their automatic memory management capabilities.

03

Code Demo

04

Discussion

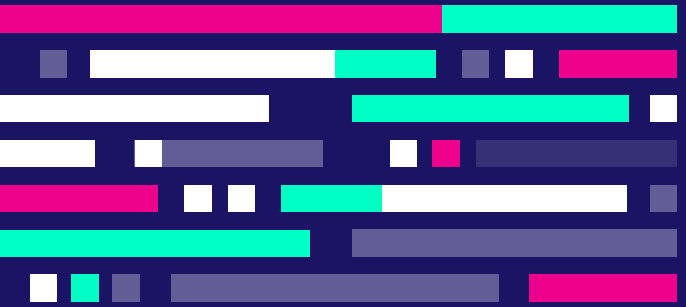
limitations or areas for improvement

Limits

The application does not support all the features that are available in a typical command-line interface

- command line editing
- wildcard expansion
  *.txt
- error handling mechanisms
- limited support for cross-platform functionality

main contributions and achievements



The program can change the current directory to the specified path, list all files in the current directory, display the contents of a file, create directories and files, and remove files and directories. The program uses the standard libraries provided by C and Windows API functions such as GetFileAttributes, CreateDirectory, and DeleteFile to perform these operations.

The program is well-structured and easy to understand, making it easy to modify and extend.

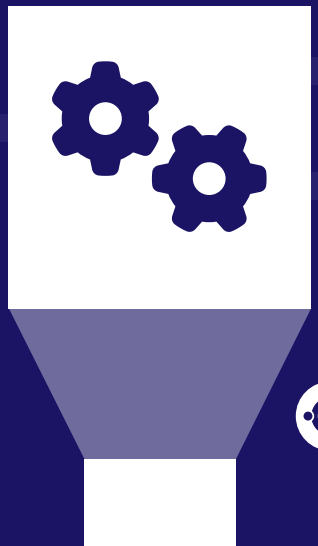
potential future work

More Commands

such as file copying, moving, and searching

support environment variables

For example, the "PATH" and “HOME" environment variables



Remote Execution

SSH integration, executing commands, and transferring files between local and remote

+ Cross-Platform Support

THANKS !

Link to Project's GitHub Repository:
https://github.com/Younesnz/CENG204-Project/blob/main/Report.md