**🖥️ Java Mini PowerShell**

This project is a lightweight simulation of a command-line shell written in Java.  
It enables basic file system operations such as viewing, navigating, and creating files and directories through familiar commands (ls, cd, mkdir, touch, pwd, help, exit).

**1. 🎯 Project Overview**

The Mini PowerShell project demonstrates practical file system manipulation and user input handling in Java.  
This educational implementation helps understand how command-line interfaces work by simulating essential shell functionality in a simplified environment.

**2. 🚀 How to Compile and Run**

**Requirements**

* Java Development Kit (JDK) 8 or higher
* Terminal/Command Prompt or an IDE (IntelliJ IDEA, Eclipse, VS Code)

**Setup Instructions**

1. Create a new Java project in your preferred IDE or a new directory
2. Add these two files to your project:
   * MiniShell.java — Main application class that processes user commands
   * ShellCommandHandler.java — Implementation class containing all file system operations

**Compilation and Execution**

# Navigate to your project directory

cd path/to/project

# Compile both Java files

javac MiniShell.java ShellCommandHandler.java

# Run the application

java MiniShell

**3. 💻 Sample Execution**

When you run the program, you'll see a command prompt showing your current location:

C:\Users\yourname > help

Shows all available commands  
**Output:**

Commands you can use:

pwd = see current folder

ls = list files

cd [folder] = change folder

mkdir [name] = make new folder

touch [name] = make new file

help = see this message

exit = quit program

C:\Users\yourname > mkdir projects

Creates a new directory named projects  
**Output:** Made new folder: projects

C:\Users\yourname > cd projects

Navigates into the projects directory  
**Output:** You are now in: C:\Users\yourname\projects

C:\Users\yourname\projects > touch app.java

Creates a new file named app.java  
**Output:** Made new file: app.java

C:\Users\yourname\projects > ls

Lists all files and directories in the current location  
**Output:** [FILE] app.java

C:\Users\yourname\projects > cd ..

Returns to the parent directory  
**Output:** You are now in: C:\Users\yourname

C:\Users\yourname > exit

Terminates the application  
**Output:** Goodbye!

**4. 🔍 Code Structure & Functionality**

**Main Class: MiniShell.java**

* Creates the command processing loop
* Parses user input into commands and arguments
* Routes commands to appropriate handler methods
* Manages the application lifecycle

// Key implementation - Command parsing

String command = "";

String argument = null;

int spaceIndex = line.indexOf(" ");

if (spaceIndex == -1) {

command = line;

} else {

command = line.substring(0, spaceIndex);

argument = line.substring(spaceIndex + 1);

}

**Handler Class: ShellCommandHandler.java**

This class manages all file system interactions through these core methods:

| **Method** | **Description** |
| --- | --- |
| printWorkingDirectory() | Displays the absolute path of current directory |
| listDirectory() | Shows all files and folders with type indicators |
| changeDirectory(String folder) | Navigates to specified directories (including parent with ..) |
| makeDirectory(String name) | Creates new directories with validation checks |
| createFile(String name) | Generates empty files with error handling |
| printHelp() | Displays all available commands with descriptions |
| getCurrentDirectory() | Returns the current directory location as a File object |

**5. ✅ Available Commands**

| **Command** | **Description** | **Example** |
| --- | --- | --- |
| pwd | Print current working directory | pwd |
| ls | List files and directories | ls |
| cd [dir] | Change to specified directory | cd documents or cd .. |
| mkdir [name] | Create a new directory | mkdir photos |
| touch [name] | Create a new empty file | touch notes.txt |
| help | Display help information | help |
| exit | Terminate the shell | exit |

**6. 🛠️ Error Handling**

The application includes validation for common scenarios:

* Missing arguments (Usage: mkdir [directory\_name])
* Navigation to non-existent directories (Can't find that folder: xyz)
* Creating duplicate files or directories (That file already exists!)
* IO exceptions during file operations (Error making file!)
* Invalid commands (Unknown command. Type 'help' to see available commands.)

**7. 📝 Future Enhancements**

Potential improvements for future versions:

* File content editing capabilities
* Command history with up/down arrow navigation
* File copying, moving and deletion operations
* Wildcard support for file selection
* Tab completion for directory and file names
* Color-coded output for different file types