Implementation Overview:

Data Structures:

Directories and Files Storage:

The file system uses a combination of dictionaries (directories and files) to represent the hierarchical structure of directories and files.

The directories dictionary stores the directory structure, where each key represents a directory, and the corresponding value is another dictionary representing the contents of that directory.

The files dictionary stores file content, where each key is the path to the file, and the value is the content of the file.

Current Directory:

current\_directory variable keeps track of the current working directory.

Path Normalization:

The normalize\_path function ensures that paths are consistently formatted with a leading '/'. It filters out empty components in the path to handle edge cases.

Command Processing:

The main loop takes user input commands and processes them accordingly.

Commands are processed using various methods like mkdir, ls, cd, touch, cat, echo, mv, cp, grep, rm. For each command, appropriate methods of the FileSystem class are called.

Error Handling:

Errors related to file operations, such as attempting to create a directory that already exists (mkdir), attempting to move a nonexistent file or directory (mv), etc., are handled and appropriate error messages are displayed.

File System State Persistence:

The save\_state and load\_state methods enable the persistence of the file system state to and from a JSON file. This is useful for saving and restoring the file system state across different sessions.

Printing File Contents with grep:

The grep method searches for a pattern within a file and highlights the matched pattern in red, providing a visual indication.