# **Catch2 Test Cases**

## **test\_filesystem.cpp**

#define CATCH\_CONFIG\_MAIN  // This tells Catch to provide a main() - only do this in one cpp file

#include "catch.hpp"

#include "FileSystem.h"

TEST\_CASE("File System Operations", "[FileSystem]") {

    FileSystem fs;

    SECTION("Create File") {

        REQUIRE(fs.handleCommand("create\_file testfile.txt") == "File 'testfile.txt' created successfully.");

    }

    SECTION("Delete File") {

        REQUIRE(fs.handleCommand("delete\_file testfile.txt") == "File 'testfile.txt' deleted successfully.");

    }

    SECTION("Read File") {

        fs.handleCommand("create\_file testfile.txt");

        fs.handleCommand("write\_file testfile.txt Hello, world!");

        REQUIRE(fs.handleCommand("read\_file testfile.txt") == "Contents of 'testfile.txt': Hello, world!");

    }

    SECTION("Write to File") {

        fs.handleCommand("create\_file testfile.txt");

        REQUIRE(fs.handleCommand("write\_file testfile.txt Hello, world!") == "Successfully wrote to 'testfile.txt'.");

    }

    SECTION("Create Directory") {

        REQUIRE(fs.handleCommand("create\_directory testdir") == "Directory 'testdir' created successfully.");

    }

    SECTION("Delete Directory") {

        fs.handleCommand("create\_directory testdir");

        REQUIRE(fs.handleCommand("delete\_directory testdir") == "Directory 'testdir' deleted successfully.");

    }

}

### **Explanation:**

* **Catch2 Configuration:** #define CATCH\_CONFIG\_MAIN tells Catch2 to generate a main function, which is where all the test cases will run.
* **Test Cases:** Each SECTION corresponds to a different operation (create\_file, delete\_file, etc.). Within each section:
* **Setup:** Perform any necessary setup (like creating files or directories) using fs.handleCommand.
* **Assertions:** Use REQUIRE to assert the expected output of fs.handleCommand.
* **Error Handling:** Ensure each operation produces the expected output when successful. Consider adding additional test cases to handle error scenarios (like attempting to delete non-existent files).

## **Compiling and Running Tests:**

After including catch.hpp and setting up your test cases as shown above:

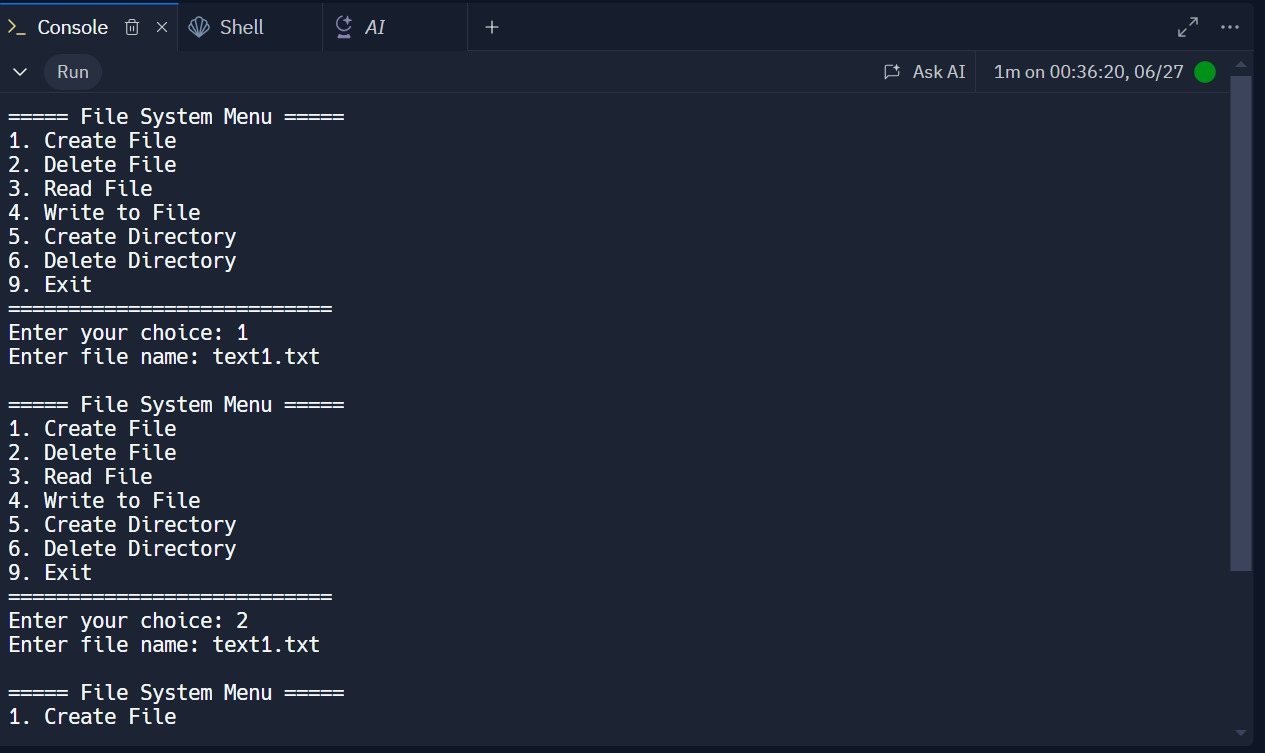
* **Compile:** Compile test\_filesystem.cpp along with your project files and Catch2. Ensure -std=c++11 or appropriate C++ standard flag is included if needed.

g++ -std=c++11 -Wall -o test\_filesystem test\_filesystem.cpp

* **Run:** Execute the compiled test executable to run your tests.

./test\_filesystem

## **Test Results:**



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated