Software Requirements Specification

for

TCSS 360 File Picker Software

**Version 1.0 approved**

**Prepared by**

**Ryder DeBack, Manjinder Ghuman, Brendan Tucker**

**Team 6**

**1/29/2025**

**Table of Contents**

[1. Introduction 1](#_Toc189155067)

[1.1 Purpose 1](#_Toc189155068)

[1.2 Document Conventions 1](#_Toc189155069)

[1.3 Intended Audience and Reading Suggestions 1](#_Toc189155070)

[1.4 Project Scope 1](#_Toc189155071)

[1.5 References 1](#_Toc189155072)

[2. Overall Description 2](#_Toc189155073)

[2.1 Product Perspective 2](#_Toc189155074)

[2.2 Product Features 2](#_Toc189155075)

[2.3 User Classes and Characteristics 2](#_Toc189155076)

[2.4 Operating Environment 2](#_Toc189155077)

[2.5 Design and Implementation Constraints 2](#_Toc189155078)

[2.6 User Documentation 2](#_Toc189155079)

[2.7 Assumptions and Dependencies 2](#_Toc189155080)

[3. System Features 3](#_Toc189155081)

[3.1 Track Changes to User Defined File System 3](#_Toc189155082)

[4. External Interface Requirements 3](#_Toc189155083)

[4.1 User Interfaces 3](#_Toc189155084)

[4.2 Hardware Interfaces 3](#_Toc189155085)

[4.3 Software Interfaces 3](#_Toc189155086)

[4.4 Communications Interfaces 4](#_Toc189155087)

[5. Other Nonfunctional Requirements 4](#_Toc189155088)

[5.1 Performance Requirements 4](#_Toc189155089)

[5.2 Safety Requirements 4](#_Toc189155090)

[5.3 Security Requirements 4](#_Toc189155091)

[5.4 Software Quality Attributes 4](#_Toc189155092)

[6. Other Requirements 4](#_Toc189155093)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Ryder DeBack Manjinder Ghuman Brendan Tucker | 1/30/25  3/21/25 | Creation of Documentation  Update of Documentation | 1.0  2.0 |

# Introduction

## Purpose

The purpose of this document is to give an overview of the purpose and requirements for the creation of a File Watcher program. It is a GUI-based application designed to track file system events in real time, focusing on user-defined file extensions. It records various file activities such as creation, modification, deletion, and renaming into an SQLite database.

## Document Conventions

SRS document body text will be composed of Times New Roman text. Bold times text for section headings. Font size 18 for section headings, size 14 for subsection headings, and size 12 for body text. Incomplete text is marked with a yellow highlighter and is intended to be updated before project completion. Text which is expected to be expanded or changed in the future will be bolded.

## Intended Audience and Reading Suggestions

This SRS document is intended for use by the developers working on the TCSS360 File Watcher project and the class instructor for grading and navigating the project build. This document is intended to be fully read sequentially, starting with the project introduction, and finishing with program requirements. File Watcher developers are encouraged to focus on system features and requirements sections.

## Project Scope

The File Watcher Software is a Windows-based application with Java-based development and SQLite database. The applications monitor file changes and records them into SQLite database. It is designed for people who need real time tracking of file modifications, deletions, and creations. It will assist organizations in keeping an audit trail for sensitive files.

## References

**No references yet may be filled in later.**

# Overall Description

## Product Perspective

This product is being designed as a standalone application. It is intended to be an additional component to the already existing Windows file management system that gives added functionality in the form of tracking changes to the file structure within its user-defined purview.

## Product Features

It enables users to track and log file operations within a specified directory, storing event data in an SQLite database. Supports user-defined file extensions for filtering which files to monitor. Provides a GUI for managing monitoring sessions. Stores detailed file events including timestamps.

## User Classes and Characteristics

Anticipated user classes include system administrators for file system security. In this case, the program continually runs, acting as a tool that can announce when any modifications are made to file system structure or contents.

An additional user class would be a personal file tracker that can announce when the system must backup a user file. Upon moving, editing, or deleting a file, the program announces to separate backup software where a new backup should be made or make changes to a preexisting file.

## Operating Environment

The File Watcher program should work on all currently Microsoft supported Windows machines. There should be nothing outside of that that requires the program to work properly.

## Design and Implementation Constraints

The Software is designed exclusively for windows operating systems and will not support macOS or Linux. Requires JRE. Database must be implemented using SQLite. User interface must be developed in Java Swing. Must use Visual Studio Code for development. The application must follow Java standard coding and must be properly documented with comments.

## User Documentation

A readme document that is attached to the application detailing different parts of the application and its functionality.

## Assumptions and Dependencies

One assumption is that the user will know what directory it is they want to have monitored, that is the File Watcher will not by default choose a directory for the user.

Another assumption is that the user will **NOT** intentionally try to crash the database or do any form of SQL injection. This is due to the database being purely local to a machine and would provide no benefits to that user.

Dependency wise is a sqlite.jar file that comes with the program, without this file it is impossible for the database to interact with the software.

# System Features

## Track Changes to User Defined File System

3.1.1 Description and Priority

This program feature allows users to monitor changes made to files within a user-defined directory and any subdirectories contained within. It provides a log of events, containing information such as what file was modified, how it was modified, and updates its location if necessary.

This feature is of extremely high priority (9).

3.1.2 Stimulus/Response Sequences

This feature requires users to select a directory they wish to track within the File Watcher program and click a “start tracking” button to officially start tracking events within that directory. Users can end file tracking by selecting a “stop tracking” button.

3.1.3 Functional Requirements

No functional requirements apart from having File Watcher installed and running.

# External Interface Requirements

## User Interfaces

The application will have a GUI with a menu bar and buttons to push that provide the same functionality.

* A button to start / stop monitoring the program
* A button that will write the current list of monitored activities to the database
* A button to open a query window allowing the user to search the database for specific instances.
* A button to reset the current list, not in the database but the list itself.
* Drop down menus to allow users to search for specific directories for monitoring and output
* Export to CSV in the query window that will take user requested database output and export it into a CSV file.
* A button that will allow the user to fully reset the database.
* A file menu option for the user to email a file from their computer to the specified email entered.

## Hardware Interfaces

No hardware interfaces will be necessary, just a machine to run the program.

## Software Interfaces

SI-1: File Tracker GUI

SI-1.1: Logs changes that occur in the file directory and stores them to the database.

SI-1.2: Will keep track of how long the timer is running as well as current time, shown in the bottom corners.

SI-1.3: User enters an input and output directory, input to monitor, output to record data.

## Communications Interfaces

**POSSIBLY TO BE ADDED LATER.** (Unsure if the File Watcher Database will ever **NOT** be a local database)

# Other Nonfunctional Requirements

## Performance Requirements

**Not applicable, the program will be lightweight and not require much for maximum performance.**

## Safety Requirements

**Not a concern, File Watcher will just monitor any changes made to a specified directory and log them in a localized database, no deletion or movement of data will occur to the actual directory itself.**

## Security Requirements

**Not applicable, the database is a localized file on the computer of the user that monitors that localized machine. If that computer is compromised, it would have been from the user themselves not the File Watcher program.**

## Software Quality Attributes

The application will always be available for the user once they have access to the original program itself.

# Other Requirements

Requires a localized database file to store all the monitored data, inside the application itself this file will be able to be created.

**Appendix A: Glossary**

Database – Location on local machine that will store changes being applied to files

Input Destination – File directory that is wanting to be monitored by the user of the application

Output Destination – File directory where all the changes that occur to the Input Destination will be stored.

Query Database – Will run a command on the database to search for specific parameters.

**Appendix B: Analysis Models**

**A computer screen shot of a diagram

AI-generated content may be incorrect.**

**Appendix C: Issues List**

When the user exits the program it will ask if there is data, if they want to register that data into the database, regardless of if that data is already in the database. Small issue but ran out of time to finalize it.