## 

**SE 211: Software Specification and Design II**

CSV FILE HANDLER

Assignment #1:

Software Requirements Specification Document

By

Joe Griffin

**February 2nd, 2020**

# [0] Table of Contents

[**[0] Table of Contents**](#_1dp559epoxom) **1**

[**[1] Introduction**](#_nqjggnj4w7f7) **3**

[[1.1] Purpose & Overview](#_olik96d0uqqv) 3

[[1.2] Intended Audience](#_1x71a3lv26ip) 3

[**[2] Overall Description**](#_x749gb2d1kut) **5**

[[2.1] Product Perspective](#_va77fflv659z) 5

[[2.2] Product Features](#_c5h3fb7cu936) 5

[[2.3] User Characteristics](#_ewjlo7v1yld7) 6

[[2.4] Operating Environment](#_mnu94423i7yc) 6

[[2.5] Assumptions and Dependencies](#_n5igynrlrr1) 6

[**[3] Specific Requirements**](#_qme7gxkspa6o) **8**

[[3.1] External Interface Requirements](#_5w5g0mc1i6i2) 8

[[3.2] Functional Requirements](#_3qu0udu6mduw) 8

[[3.2.1] Homepage](#_27xhzjtehmj4) 10

[[3.2.2] Displaying the CSV Data](#_d7n29dndq4nw) 10

[[3.2.3] Editing Specific CSV Data](#_8hhz8o5k97fi) 10

[[3.2.4] SourceCode](#_ie5xov1gm9up) 11

[[3.2.5] Source Bank](#_tgehxcuezisc) 11

[[3.3] Non-Functional Requirements](#_ilr5qjv2wbrh) 12

[[3.3.1] Reliability](#_jcfmp71ner33) 12

[[3.3.2] Availability](#_jgu7ljhl3o26) 12

[[3.3.3] Scalability](#_ezutvqpkrfbu) 12

[[3.3.4] Usability](#_ud57xolfeqhg) 12

[[3.3.5] Performance](#_p121py8zbgdv) 12

[[3.3.6] Supportability](#_5y318w15kj2a) 12

[**[4] Appendix**](#_qs98z23cdz88) **13**

[[4.1] Appendix A: Glossary](#_rljz3vcuerw0) 13

[[4.2] Appendix B: User Scenario Cases](#_s3s63vs6nrhx) 15

[[4.3] Appendix C: Sequence Diagrams](#_lwp2oza6ixq9) 19

# 

# [1] Introduction

## [1.1] Purpose & Overview

This **Software Requirements Specifications (SRS)** document will provide the necessary details and specifications for the products being created for the *CSV File Handler* program.

*CSV File Handler* is a program which consists of mainly two (2) parts:  
 1) The *CSV File Handler* Source Code  
 2) The *CSV File Handler* Database

***CSV File Handler* Source Code:** refers to the physical program that will be run on a given computer.

***CSV File Handler* Database:** refers to the database that the program will access and communicate with while running on a computer.

There are three (3) major sections within this SRS after **[1] Introduction**:

1) **Overall Description:** provides the specification of the system model, the classes model, the main constraints, and the list of any assumed factors that used within this document

2) **Specific Requirements:** provides a list of external interface requirements used to have access to the *CSV File Handler* program and interact with CSV files on personal devices, functional requirements by major features of the platform and a list of non-functional requirements to be present throughout the platform, internally and externally

3) **Appendix:** provides more information beyond the functional and non-functional requirements on certain features of the documents, terms, phrases, more insight in how the platform was intended to be used, and provides visual mock-ups on some screens shown in the platform to visually clarify ambiguous and/or an exorbitant amount of information

## [1.2] Intended Audience

This document is intended for any client, *CSV File Handler* developer, project manager, or documentation writer that needs to understand the system from a product, developer, and user standpoint; although this document tailors more towards the developer standpoint.

Here are the potential uses for each of the reader types:

**client:** views, verifies, and agrees to the features and requirements included with the SRS. Acts as a witness to the creation of the document and, in turn, extends their good spirits and buffers expectations that the program will be completed as per the requirements located in this SRS

**developer:** needs to implement and, if necessary, provide accessibility for all requirements to be completed within the document for the *CSV File Handler* project. Also needs to verify that the requirements do not conflict with other requirements, and if they do, must notify the project manager to solve the requirements conflict

**project manager:** needs to verify and update the requirements so as there are no requirements conflicts and that they are up-to-date; needs to pass the information to the documentation writer for SRS updates. Also oversees the developer pipeline to ensure there are no production blockages and/or bottlenecks so as to ensure the efficiency of production time

**documentation writer:** maintains and updates the SRS to the most up-to-date to ensure the latest changes to the requirements, if any, are conveyed to the developers. Also responsible for requirements change control history and version control history of the SRS

# [2] Overall Description

*CSV File Handler* is a program that can be run on any device as a way of handling and managing CSV files and the data inside of them. While the program is running users will have multiple options for retrieving and managing CSV files that are passed into the program.

*CSV File Handler* only requires one file and it will allow users to edit and store CSV files through the program. The menu is tight and allows for command entry for selecting different procedures.

## [2.1] Product Perspective

There are a variety of reasons that users would like to utilize the *CSV File Handler* program and manage their csv files on their personal computers.

**Firstly**, users are able to change specific CSV data values by navigating through a menu option. **Secondly**, users are able to store data through the CSV and save it in the file immediately.

The *CSV File Handler* program is designed to be simple and easy to use for even the most technologically clueless. This is conveyed through the straightforward and easy-to-understand user interface. To use the program, a user would need to select the options with the text closest to the user’s intention(s).

## [2.2] Product Features

The major features the *CSV File Handler* program contains are the following:

* **Ability to display the CSV Data:** pulls data from the file and puts in in a custom table.
* **Ability to edit specific values:** lets the user access the pulled data directly in order to edit values.
* **Ability to save new versions of the CSV file with updated data:** takes the current edited version of the CSV data and saves it to the file.

*Figure 1.1 CSV File Handler Program Diagram*

## [2.3] User Characteristics

For conventional reasons, we name each of the actors with this format:

**Physical Actors:**

**User**: the entity that we act as to follow the flow of the program and managing CSV Files. The user can be a person or a test application.

**System Actors:**

**Source Code:** The physical source code that is triggered by the launcher application in order to run the program through a terminal.

**Source Database:** acts as the bastion and gatekeeper of information on all entities in the *CSV* program; stores information on functions and classes, file editing processes and manages the file name of the CSV file that is being managed.

**CSVFH Launcher:** acts as the launcher application that is run by the user in a terminal. The launcher needs to be prompted with the inclusion of the file name in order to run the program. Also, the file needs to have been placed inside the CSVFH folder in order to operate with the data.

## [2.4] Operating Environment

This program will operate in the following operating environments:

* Apple Mac OSX
* Microsoft Windows
* Linux/Unix

This environment is meant to be run through a command prompt using just a run prompt and a file as the argument. If the computer does not have command prompts and bash capabilities or python capabilities the program will not work and needs to be run on devices with those capabilities.

## [2.5] Assumptions and Dependencies

Users are expected to have a personal computer that possesses capabilities to run bash and python command scripts. The user must also be able to have storage space available on their device to store and edit CSV files through.

The program depends on the user to place the CSV file to be managed in the folder with the source code and database. Also, the user must also run the launcher including the filename exactly in order to run the program.

# [3] Specific Requirements

These requirements outline what is needed for each major feature of the program in order to successfully create the *CSV File Handler* program with all of its critical, conditional, and optional components to ensure an outstanding user and player experience.

These requirements also act as a guideline as to how they correlate to the development process and as a checklist for developers to understand what requirements they have and have not completed.

## [3.1] External Interface Requirements

**External interface requirements** refer to the entities used independent of the *CSV File Handler* program in order to access the *CSV File Handler* program. The unique identifier used for this specific set of requirements is shown below:

**EXT-#**

The following external interface requirements needed to use *CSV File Handler* and play the game are below:

**EXT-1:** The user shall possess a computer device that runs with either a Apple Mac OS X, Microsoft Windows, or Linux/Unix operating system with bash and python capabilities.

**EXT-2:** The user shall possess a command prompt or terminal window that can operate in their operating system in order to run and interact with the *CSV File Handler* program.

**EXT-3:** The user shall possess a computer keyboard used to input characters and form text responses for interacting with the *CSV File Handler* program.

## [3.2] Functional Requirements

These **functional requirements** describe the features of this system and what it does. Depending on the complexity of the requirement, each requirement may or may not have *sub-requirements*, further detailing out the higher-level requirement.

Unique identifiers have been named for each section of requirements as it aids in organizing and referring to each requirement. Each requirement will possess a unique identifier along with a number and may possess a letter to signify sub-requirements. An example format of these unique identifiers is shown below:

**SHOW-1**

**HOME-2A**

**EDIT-1C-3**

The following identifiers are used in the functional requirements below:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Identifier** | **Description** | | | | | |
| HOME | Refers to the homepage | | | | | |
| SHOW | Refers to the menu option on the main page for displaying the CSV file data in a table | | | | | |
| EDIT | Refers to the menu option on the main page for selecting and editing specific cells in the csv file | | | | | |
| SRCE | Refers to the interaction within the srcecode.py file and specifications on functions | | | | | |
| BANK | Used to refer to the srcebank.py file that holds class data and important function data needed for the running of the program | | | | | |

Each higher-level requirement will also utilize a **priority level**. A priority level describes the *urgency* or *critical component* status needed to achieve the minimum features in order for the *CSV File Handler* program to be usable at all. A priority level will also utilize a **highlighted color** for easy recognizability and clarity. A table of priority levels along with a description of their urgency are shown below:

|  |  |
| --- | --- |
| **Priority Level Legend** | |
| HIGH | Must be completed first and foremost; critical component of system |
| MED | Should be completed after HIGH priority requirements; nearly critical component of system, but not 100% necessary |
| LOW | Not a priority and should be completed after all HIGH and MED requirements have been completed; not a critical component of system |

### [3.2.1] Homepage

**HOME-1:** (HIGH) The *CSV File Handler* shall provide a quick text welcoming the user and offering guidance for the navigation of the program.

HOME-2: Selection and Navigation (HIGH)  
 **HOME-2A:** The user shall be able to select an option to “Display File Data,” represented through a menu option  
 **HOME-2B:** The user shall be able to select an option to “Edit File Data,” represented through a menu option  
 **HOME-2C:** The user shall be able to select an option to “Quit the Program,” represented through a menu option  
 **HOME-2D:** The user shall be navigated to their respective page after selecting the option on the homepage and inputting the value that will take them there, with the exception of the “Quit the Program” option which instead ends the program.

### [3.2.2] Displaying the CSV Data

**SHOW-1:** (MED) The data shall be displayed in a table like setting with even spacing all separated by vertical bars.   
 **SHOW-2:** (MED) The user should be able to clearly see all of the values with the rows and columns in the right places.   
 **SHOW-3:** (MED) The user should be prompted with a buffer page allowing them to decide whether to go back to the homepage or quit the program.

### [3.2.3] Editing Specific CSV Data

EDIT-1**:** Editing a Specific Value (HIGH)  
 **EDIT-1A**: The data shall be displayed in the table like setting just like in the “Display File Data” menu option.  
 **EDIT-1B:** The program shall prompt the user for the row and column of the data values that is to be changed.  
 **EDIT-1C:** No matter what type of data the user enters into the program, the selected values shall be changed to the new inputed value.  
 **EDIT-1D:** The user shall also be presented with the option to save this data as the new values.

### [3.2.4] SourceCode

**SRCE-1:** (HIGH) The user shall not be able to access or change any of these lines of code from outside the module

SRCE-2: Source Code Design (MED)  
 **SRCE-2A:** The code should be entirely hidden behind functions and classes with minimal lines of code out freely in the script.  
 **SRCE-2B:** The source code shall be simplistic in design with few needed resources like imports to reduce the number of errors across platforms.  
 **SRCE-2C:** Since the code is running through a terminal window, most of the programming done within the source code file should be for assisting the visual and UI elements like menus and display screens.

**SRCE-3:** The source code shall be commented at least minimally for developers and/or project managers to look through and verify.

### 

### [3.2.5] Source Bank

**BANK-1:** (HIGH) The user shall not be able to access or change any of these lines of code from outside the module

BANK-2: Source Bank Design(MED)  
 **BANK-2A:** The code should mostly be about defining the functions and classes that is being used by the Source Code File.  
 **BANK-2B:** The Source Bank file shall focus on the holding and collecting of important classes and should avoid containing functions that are meant to implement important procedures including the CSV Files.

**BANK-3:** The source code shall be commented at least minimally for developers and/or project managers to look through and verify.

## [3.3] Non-Functional Requirements

These **non-functional requirements** describe the method(s) as to how the features of this system would be achieved. These requirements will revolve around *quality attributes* which are mostly attributes of the system concerning reliability, availability, scalability, usability, performance, and supportability.

### [3.3.1] Reliability

The *CSV File Handler* program should be available no matter what time or system it is moved to.

### [3.3.2] Availability

The *CSV File Handler*  should be usable by anyone who owns a computer with any one of the valid operating systems. See **Section [2.4] “Operating Environment”** for more information.

### [3.3.3] Scalability

The *CSV File Handler* should be able to handle CSV files of any size.

### [3.3.4] Usability

Users who have access to a command prompt or terminal for their operating systems as well as a keyboard for inputting information should be able to operate this program. This program should not demand a deep knowledge of bash or python programming to operate.

### [3.3.5] Performance

The program should provide an under-four-second (<4.0s) response time for loading menus and saving information to files.

### [3.3.6] Supportability

The program is available on any computer for free as long as the OS is Windows, Mac or Linux. The computer also must have access to a command prompt or terminal application as well as bash and python functionality.

# [4] Appendix

The **Appendix** provides more information beyond the functional and non-functional requirements on certain features of the documents, terms, phrases, more insight in how the platform was intended to be used, and provides visual mock-ups on some screens shown in the platform to visually clarify ambiguous and/or an exorbitant amount of information.

There are four (4) appendices located below:

1. **Appendix A: Glossary:** details and explains certain ambiguous or confusing terms and phrases
2. **Appendix B: User Scenario Cases:** defines the uses of the *CSV File Handler* program between external entities or “actors” and the system itself
3. **Appendix C: Sequence Diagrams:** behaviorally models usage scenarios of the *OnlineCheckers* platform and the user(s) interacting with it
4. **Appendix D: User Interface Prototypes:** describes the “look” and “feel” of how a screen should appear to the user and how it will potentially react to player input

## [4.1] Appendix A: Glossary

The **Glossary** clarifies and describes the language in the context used throughout the document as it pertains to the requirements described.

**CSV:** comma separated values; usually stored in a file as a way of storing large quantities of information

**CSV file:** the file that holds comma separated values (CSV).

**OS:** operating system; the system as to which a computer operates and performs basic tasks

**CSV File Handler:** refers to the program in which the user is currently operating

**program:** refers to the *CSV File Handler* program and its underlying code as it responds to certain actions and events and progresses the state of the platform navigation and/or game environment; cross listed with ***system***

**system:** refers to the *CSV File Handler* program and its underlying code as it responds to certain actions and events and progresses the state of the platform navigation and/or game environment; cross listed with ***program***

**user:** any connected device who is currently using the *CSV File Handler* Program

**command prompt:** an application that comes included with Microsoft Windows operating systems in order to more efficiently interact with and operate files and directories; cross listed with **terminal**

**terminal:** an application that comes included with Apple MAC OS X operating systems in order to more efficiently interact with and operate files and directories; cross listed with **command prompt.**

**bash:** a command line language used for interacting in the command prompt or terminal. For more information see **terminal** and **command prompt.**

**python:** a general purpose programming language for creating executable files likes scripts and modules.

## [4.2] Appendix B: User Scenario Cases

These **Usage Scenarios** define the uses of the *OnlineCheckers* platform between external entities or “actors” and the system itself. In this context, the actors are the *user(s)*, *player(s)*, *game*, and *system* (otherwise known as the *OnlineCheckers* platform).

For each Usage Scenario, there are sections defining its **Pre-Conditions**, **Course of Action**, **Post-Conditions**, and **Exceptions**. Each section may be referred to in other sections by being represented through the *Numbered List* format as seen in the following example:

1. This
   1. is
   2. A
      1. numbered
2. list

## Displaying the CSV Data

### Pre-Conditions

1. The user has navigated to the *CSV File Handler* home page
2. The user selected the **Displaying The CSV Data** option on the *CSV File Handler* homepage

### Course of Action

1. The user selects the option, **Displaying The CSV Data**, by typing in the menu number in the input

### Post-Conditions

1. The system displays the CSV Data in table format
   1. The CSV Data will be equally spaced and display with bars in between the values
2. The screen will then prompt the user to see if they want to return to the home page or quit out of the program

### Exceptions

1. **Problem:** The CSV data takes up more room than the page can hold
   1. **Solution:** The data will still print fully to the command prompt but the user will need to scroll up to see the top of the table
2. **Problem:** The user closes the program mid run
   1. **Solution:** The program will be closed and the file will not be affected

## Editing the CSV Data

### Pre-Conditions

1. Please refer to **Displaying the CSV Data** -> **Pre-Conditions**

### Course of Action

1. The user selects the option, **Edit CSV Data**, by typing in the menu number in the input

### Post-Conditions

1. The display will then change to displaying the CSV Data in table form
2. The program will then provide the user with a choice as to which value they wish to change, by providing a row and column through two different prompts.
   1. If the user selects a value to edit it will then prompt the user for a value to insert
   2. The user can also input a “q” into the input to quit the process and return to the homepage
   3. The value will change and then the program will as if you want to save the current version of the code into the file or not
      1. If the user says yes to saving the current version of the file the file will save and then prompt the user if they want to return to the home page or not
         1. If the user accepts they will be delivered to the home page
         2. If the user denies then the program will end
      2. If the user says no to saving the current version of the file then they will be prompted on if they wish to return to the home page or not and the changed value will remain
         1. If the user accepts they will be delivered to the home page
         2. If the user denies then the program will end

### Exceptions

1. **Problem:** The user leaves the program before they are prompted to save the new values in the CSV file
   1. **Solution:** The new information is not saved into the CSV file and all progress is lost but the file is not corrupted and remains unaffected
2. **Problem:** The user saves the new information but quits the program before they answer the following prompt
   1. **Solution:** the new information is saved to the CSV file and no file or values are corrupted but the program is quit

## Saving a New File

### Pre-Conditions

1. Please refer to **Editing the CSV Data** -> **Post-Conditions**

### Course of Action

1. The user selects the option, **Edit CSV Data**, by typing in the menu number in the input
   1. The display will then change to displaying the CSV Data in table form
   2. The program will then provide the user with a choice as to which value they wish to change, by providing a row and column through two different prompts.
      1. If the user selects a value to edit it will then prompt the user for a value to insert
      2. The value will change and then the program will as if you want to save the current version of the code into the file or not

### Post-Conditions

1. If the user accepts they will be delivered to the main menu
2. If the user denies then the program will end

### Exceptions

1. **Problem:** The user closes the program before they can input a response
   1. **Solution:** The current CSV file is not corrupted or disrupted and the data that was going to be saved is lost
2. **Problem:** The user chooses to save the file but leaves before the next prompt can come up
   1. **Solution:** The data is saved to the file and the program is quit meaning they can not edit those values unless they turn run the program again

## [4.3] Appendix C: Sequence Diagrams

These **Sequence Diagrams** behaviorally model usage scenarios of the *CSV File Handler* program and the user(s) interacting with it. Since sequence diagrams utilize the concept of “lifelines,” the equivalency between actors of usage scenarios and lifelines of sequence diagrams is established here.

For each Sequence Diagram, there will be accompanying text describing how each diagram operates.

## Displaying the CSV Data

### 

### Documentation

Once the player chooses the option to *Display File Data*, the program contacts the CSV Data to retrieve the data. Once the program has received the data, it puts it into a table and displays it through the screen.

## Editing the CSV Data

### 

### 

### Documentation

In order to edit CSV data through the program the user must successfully navigate to the main menu and then select the **Edit File Data** menu option. Once the user has selected that option they will be prompted for the exact value they wish to replace using two input prompts for row and column. Once the value to be edited is selected, the user is then prompted to provide a new value. Then when the program has received the new value, it will replace the current value in that position. The CSV data class will be updated inside the program and then the user will be taken to the next steps (See **Saving a New File** below).

## Saving a New File

### 

### Documentation

In order to save a new file, the user must have navigated and operated the Editing The CSV Data sequence and changed values of a specified position. Then once the user has completed updating a value of the CSV data, the program will then prompt them asking if the user wants to save the current updated CSV data into the file. If the user responds yes, the program will retrieve the current CSV data from the CSV Data class and then save it to the file through a function.

**Intentionally Stated**

End