***CSV File Handler***

*Software Design Document*

|  |  |
| --- | --- |
| **Group Members** | Joe Griffin |
| **Faculty Advisor** | Dr. Filippos Vokolos, Ph.D. |
| **Teaching Assistant** | Philip Jones |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Reasons for Revision | Revision |
| Joe Griffin | January 28th, 2020 | Original first outline, some information | 0.8 |
| Joe Griffin | January 30th, 2020 | Adding information because of design changes | 0.9 |
| Joe Griffin | February 2nd, 2020 | Final Draft | 1.0 |

# Table of Contents

[**Table of Contents**](#_z51x5r6f8oa) **3**

[**[1] Introduction**](#_lhqxkmuog6ar) **4**

[[1.1] Purpose of Document](#_l84n6hkb0f4e) 4

[[1.2] Scope of Document](#_9ke3ijk871s8) 4

[[1.3] Definitions, Acronyms, and Abbreviations](#_upmolp97wtna) 4

[[1.3.1] Computer Internals](#_x76ume1j1rcm) 5

[[1.3.2] Program Runtime](#_e93ts42dkqk) 5

[**[2] System Overview**](#_dqwjpg1ygbj5) **5**

[[2.1] Description of Software](#_uctmhfvjtivz) 6

[[2.2] Technologies Used](#_7zt549y0zmm9) 6

[**[3] System Architecture**](#_2sul1aef1za7) **6**

[3.1 Architectural Design Components](#_qyccw54jkwsq) 6

[[3.2] Design Rationale](#_iddwukjb5rqn) 7

[Why only use one main class and multiple functions?](#_hu4n768n1tmh) 7

[**[4] Component Design**](#_lza1dyw8gul3) **7**

[[4.1] Overview](#_l7p7ibrgl946) 7

[[4.2] CSVFH Launcher Architecture Design](#_bnwnk4dagbgh) 8

[[4.2.1] Attributes](#_g12p7d2iuo3c) 8

[[4.3] Source Code Architecture Design](#_1kkymhjeyfo) 8

[[4.3.1] Attributes](#_5fayxsdk9ujl) 8

[[4.3.2] Methods](#_il20brg3xve1) 9

[[4.4] Source Bank Architecture Design](#_st5yoo2oh21o) 10

[[4.4.1] Attributes](#_jort68rye2tp) 10

[[4.4.2] Methods](#_32b2qk8c5v7b) 10

[**[5] Human Interface Design**](#_az0gsahwpg5j) **11**

[[5.1] Overview of User Interface](#_9hi4jnjt7rr3) 11

[[5.2] Screen Objects and Actions](#_dqhb0un7resh) 12

[[5.3] Program Menu Flow](#_tnyfspr7islo) 12

[**[6] References**](#_q72teiwr63uv) **12**

# **[1] Introduction**

## **[1.1] Purpose of Document**

This design document is to describe the implementation of the ***CSV File Handler*** program as described in the *CSV File Handler Software Requirements document*. The ***CSV File Handler***  program is meant to be a multiple time use tool for managing and editing CSV Files on any given computer.

## **[1.2] Scope of Document**

This design document describes the implementation of the *CSV File Handler*software. The program relies on a few functional systems that are split between two main namespaces being the source code and a source bank file responsible for containing the classes and important functions for running the home page options. The source code file would essentially be responsible for the homepage of the program as well as connecting home page options to the functions that allow them to run.

## **[1.3] Definitions, Acronyms, and Abbreviations**

This section is to clarify the terminology used throughout this document. The *CSV File Handler* Program contains many smaller pieces that work together to make a full product.

### [1.3.1] *Computer Internals*

**Python:** A programming language that was used in the construction of the source code for the *CSV File Handler* software

**Bash:** a command language used with **Command Prompt** or **Terminal** that comes with an operating system.

**Command Prompt**: An application that comes with certain operating systems like Microsoft Windows that allows easier access and more options to interact with the files and directories on a computer; Also refer to **Terminal**

**Terminal:** An application that comes with certain operating systems like Apple Mac OS X that allows easier access and more options to interact with the files and directories on a computer; Also refer to **Command Prompt**

### [1.3.2] *Program Runtime*

**CSV:** comma separated values; stored in a file that is provided by the user for handling and modification of the values inside.

**Home Page:** The first menu that is presented to the user that allows the user to select different options to perform different tasks involving the CSV data

**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

# **[2] System Overview**

## **[2.1] Description of Software**

*CSV File Handler* is a program for managing and editing CSV Files of any size on personal computers. Users are given options such as displaying the CSV data, editing specific values, and saving updated versions of the data into the original file.

## **[2.2] Technologies Used**

The CSV File Handler program will require personal computers and or laptops using any type of OS as input devices. So long as the computer has access to a terminal or command prompt, the user should be able to interact/operate with the program. The CSV File Handler program should be able to run independently on any device filling out those requirements with little to no errors regardless of the size file that is given to it.

# **[3] System Architecture**

## **3.1 Architectural Design Components**

**CSVFH Launcher:** A simple system for redirecting command line input into valid input for python scripts.

**Source Code (srcecode.py):** The main source code file responsible for carrying the user into and about the functions of the program.

**Source Bank (srcebank.py):** The secondary source code file responsible for containing the classes and functions used by the main source code in order to carry out specific requests.

**Gatekeeper:** A system for checking if a file was correctly sent through to the internal scripts.

**Test:** A system for checking if the file that was sent in is in the current working directory.

**Main Menu:** The home page that the user is directed to in order to navigate to specific operations.

**Data Transfer:** A system responsible for transferring the CSV data from the file into a class inside the file.

**Menu Return:** A system for prompting the user if the wish to continue using the program or if they want to quit out.

**Display CSV Data:** A simple system for displaying the CSV data values in an organized fashion.

**Edit CSV Data:** A system made for prompting a user for a value they wish to change and then prompting for the value they wish to change it to.

**Save CSV:** A system made for saving the current version of the CSV data into the file.

**CSVData:** The main database class for storing the current version of the csv data. This version of the data can be updated after anytime you navigate to the **Edit CSV Data** menu.

## **[3.2] Design Rationale**

### Why only use one main class and multiple functions?

When trying to design the structure of my code, I realized it would be more or less worthless to create extraneous classes for pieces that could be solved through functions. I saw that the only part that needed to be stored and communicated with was the CSVdata and accessories of that and that allowed me to design any number of functions to communicate with the class and each other. Despite being seen as a non recommended practice, I ran into significantly less issues when it came to connecting systems and error testing.

### Why make so many procedures for checking the file name when its inputted into the program?

Because an important aspect of this project was error handling, I knew going into it that I would need to be making adjustments in my programming that would account for human error. That error handling began with implementing many different ways that the wrong file name had been sent in by the user. Honestly, I was going to make it trigger sooner in the programming but I realized that if I wanted to organize the functions of my code I needed to start in my Source Code File rather than through the launcher.

### Why do so many of your functions cram into python scripts instead of running in their own files

I know its common practice to make specialized scripts for certain procedures but I knew going into this if I wanted to make my code accessible and efficient, I would have to stack functions on top of eachother in order to get the most out of each file.

# **[4] Component Design**

## **[4.1] Overview**

In this section, More Details on each component are given. For each component, a brief description and example tables are given.

## **[4.2] CSVFH Launcher Architecture Design**

The CSVFH Launcher is responsible for launching the program and correcting placing the user into the program. Launching the program seems like a redirection considering the input file is run as an argument for the running of CSVFH; however, inside of the launcher there is a slightly more complicated structure. The CSVFH Launcher checks the run to make sure that the user has sent in a file and if they haven’t, the launcher will, instead of launching the program, display a help message in the command prompt/terminal that gives instructions on how to activate the program.

### [4.2.1] Attributes

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| $1 | String | A string variable that carries the value of the first argument passed in with the code |
| srcecode.py | Python Script | The python script that contains the home page as well as running capabilities with functions |

## **[4.3] Source Code Architecture Design**

The Source Code File is responsible for the accepting of the launcher and initiation of the home page of the program. This means the source code file is the essential hub for the entire *CSV File Handler* Program. The importance of the Source Code File can not be overrated as it serves as the one and only connection between all operations under the *CSV File Handler* Program

### [4.3.1] Attributes

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| file | String | The file name to be operated with |
| CSV | CSVData | The object that holds the data from the file |
| sys.argv | List | A python list containing all the arguments that were passed into the script |
| user | String | The users input into the program; used for determining which menu selection they made |

### [4.3.2] Methods

|  |  |
| --- | --- |
| gatekpr() |  |
| Input: | Void |
| Output: | Void |
| Description: | A Method that checks, before starting the program, if the program was given arguments and activates test() |

|  |  |
| --- | --- |
| test |  |
| Input: | A string that is meant to be a file name |
| Output: | Either a Boolean True or an Error message |
| Description: | A Method that checks, before starting the program, if the file sent in is accessible for the program |

|  |  |
| --- | --- |
| mainMenu |  |
| Input: | A string that is meant to be a file name |
| Output: | Void |
| Description: | The home page menu that displays for users allowing them to choose different options procedures |

## **[4.4] Source Bank Architecture Design**

The Source Bank file is the main storage file for a lot of the functions and classes that are utilized in the Source code file. The Source bank file contains the one main CSVData class as well as the operations for Editing and Saving CSV Data.

### [4.4.1] Attributes

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| file | String | The file name to be operated with |
| data | List | A list of other lists that holds the rows of the CSV data in order |
| CSV | CSVData | The object that holds the data from the file |
| user | String | The users input into the program |
| pageturn | String | A simple display string for resetting the display on the screen |
| uRow & uCol | String | String variables used for when the user is selecting a value to edit in the CSV Data |
| uValue | String | String variable used for when the user is inputting a number to insert for the selected value in the edit function |

### [4.4.2] Methods

|  |  |
| --- | --- |
| editCSVdata |  |
| Input: | CSVData object |
| Output: | Void |
| Description: | The edit CSV data page that allows users to select a value from the CSV data and change it to another value |

|  |  |
| --- | --- |
| menuReturn |  |
| Input: | Void |
| Output: | Void |
| Description: | The buffer page after running an operation which asks the user if they wish to quit the program or continue using it |

|  |  |
| --- | --- |
| dataTransfer |  |
| Input: | String |
| Output: | CSVData object |
| Description: | This is the method that takes the file name of the CSV file and then transfers all of CSV data into a CSV data object to hold it. |

|  |  |
| --- | --- |
| saveCSV |  |
| Input: | String, List |
| Output: | Void |
| Description: | This is the method that takes the CSV List and file name and saves it into the file. |

# 

# **[5] Human Interface Design**

## **[5.1] Overview of User Interface**

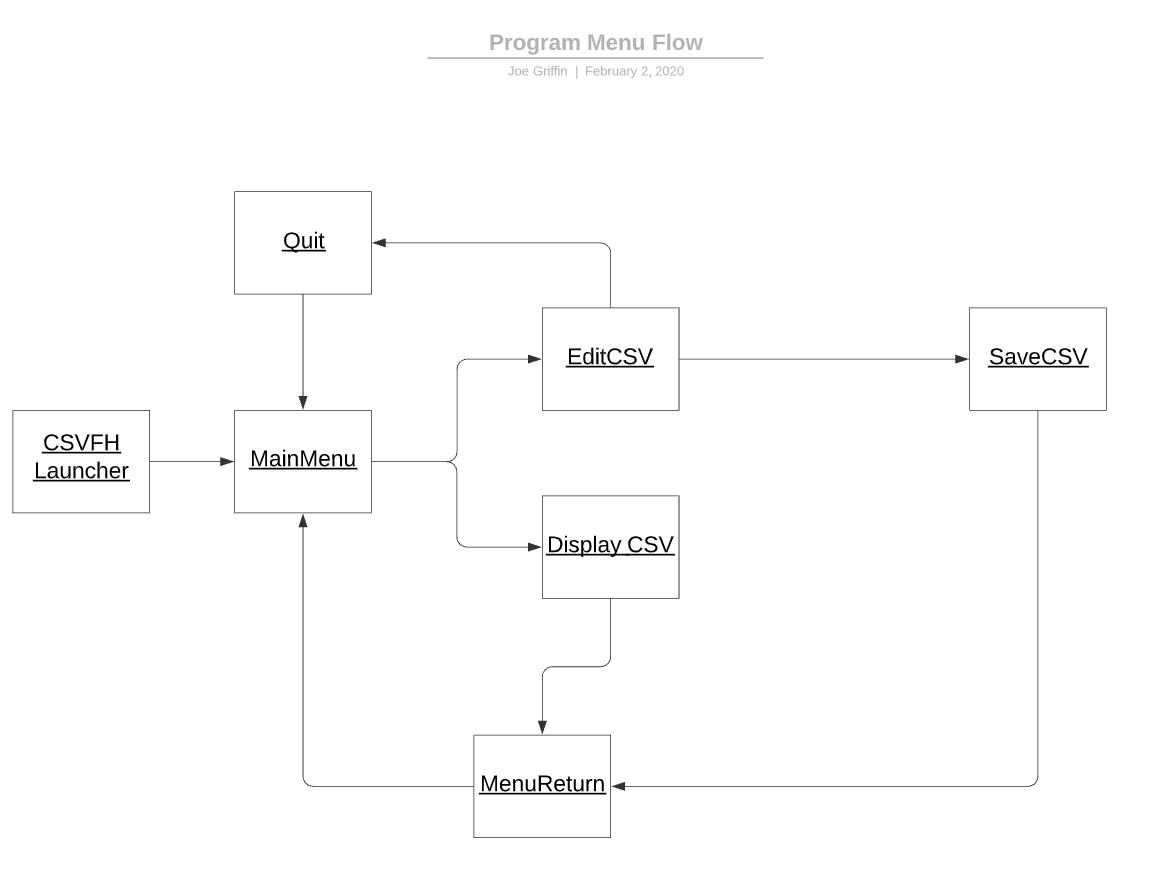
The Screenflow of the CSV File Handler goes into more detail in Requirement Specification Document[1]. All of the menus will flow through the command prompt/terminal software provided through operating systems and will allow user input almost every step of the way.

## **[5.2] Screen Objects and Actions**

The *CSV File Handler* program will use the keyboard and command prompt/terminal capabilities of the computer; however, the program as a whole will not be demanding on long strings of input from the user.

## **[5.3] Program Menu Flow**

Below is a listing of the server’s screen flow for convenience. Implementing these menus in Python involves mostly my **Source Code File** and **Source Bank File** in order to correctly move data from input into specified areas . See the Requirements Specification for more details [1].



# **[6] References**

1. Griffin, Joe. (CSV File Handler Software Requirements Specifications Document). Philadelphia, PA, United States, 2019.