**Software Design Document**

**For**

**KLWINES SCRAPPER**

**Version 0.5**

Prepared by:

* **Aly Mohammed Reda**

**SDD TABLE OF CONTENTS**

[1. INTRODUCTION 27](#_Toc517134720)

[1.1 Purpose 27](#_Toc517134721)

[1.2 Scope 27](#_Toc517134722)

[1.3 Aim 27](#_Toc517134723)

[1.3Overview 28](#_Toc517134724)

[1.4 Definitions andAcronyms 28](#_Toc517134725)

[1.6 People 29](#_Toc517134726)

[2. SYSTEM OVERVIEW 29](#_Toc517134727)

[3. SYSTEM ARCHITECTURE 30](#_Toc517134728)

[ArchitecturalDesign 30](#_Toc517134729)

[3.2 Decomposition Description 32](#_Toc517134730)

[3.3DesignRationale 38](#_Toc517134731)

[4. DATA STRUCTURE DESIGN 39](#_Toc517134732)

[4.0 DataDescription 39](#_Toc517134733)

[4.1 Data Dictionary 40](#_Toc517134734)

[5. COMPONENT DESIGN 42](#_Toc517134735)

[5.1 Feature Extractor Module 42](#_Toc517134736)

[5.2 Classifier Module 44](#_Toc517134737)

[6. HUMAN INTERFACE DESIGN 50](#_Toc517134738)

[6.1 Overview of User Interface 50](#_Toc517134739)

[6.1.1 Input Text Box 50](#_Toc517134740)

[6.1.2 Classify Button 51](#_Toc517134741)

[6.1.3 Output label 51](#_Toc517134742)

[6.2 API Screen Images 52](#_Toc517134743)

[7. REQUIREMENTS MATRIX 53](#_Toc517134744)

**sdd table of figures**

[Figure 12: Definitions and abbreviations 29](#_Toc517137659)

[Figure 13: System Context diagram 30](#_Toc517137660)

[Figure 14: Layered architecture for SCMS 32](#_Toc517137661)

[Figure 15: DFD Level 0 34](#_Toc517137662)

[Figure 16: DFD level 1 -Training phase 35](#_Toc517137663)

[Figure 17: DFD level 2 -Training phase 36](#_Toc517137664)

[Figure 18: DFD level 1-Testing phase 37](#_Toc517137665)

[Figure 19: DFD level 2-Testing phase 38](#_Toc517137666)

[Figure 20 :Data dictionary 42](#_Toc517137667)

[Figure 21 : Feature extractor Module 43](#_Toc517137668)

[Figure 22: Feature extractor module activity diagram 43](#_Toc517137669)

[Figure 23: Classifier class 46](#_Toc517137670)

[Figure 24: Classifier Class activity diagram 46](#_Toc517137671)

[Figure 25: Text Box Input 52](#_Toc517137672)

[Figure 26: Classify button 52](#_Toc517137673)

[Figure 27: label 53](#_Toc517137674)

Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Version** | | **Description** | **Author** |
| 2018-6-1 | | 0.1 | Initial version | Aly |
| 2018-6-2 | | 0.2 | update structure | Aly |
| 2018-6-5 | | 0.3 | update structure | Aly |
| 2018-6-7 | | 0.4 | update structure | Aly |
| 2018-6-8 | | 0.5 | update structure | Aly |

**1. INTRODUCTION**

**1.1 Purpose**

The Software Design Document describes the architecture and system design of our system (Klwines scrapper)and captures the design constraints and assumptions as well as the detailed design of the subsystems and components of the system.

* **Intended Audience**

Following stake holders are the intended audience for this document

* Developers
* System tester
* Future developers

**1.2 Scope:**

* This project aims to detect any change occur to the website (Klwine),and send the new scraped file periodically to the client in E-mail form attached by the Excel file & Json file contain the new scraped data .
* The main functionality isScrape website table manually and possible to store it on Excel and Json file and send email with attaching file and status of file comparison and automatically

( Scheduling ) Scrape website table of spirit type and send email every period of time

## 1.3Aim

The design description defined in this document serves multiple purposes:

• To describe the functional structure, data and algorithms to be implemented.

• To identify required system resources.

. • To be used to verify compliance with requirements

**1.4 Acronyms and abbreviations**

|  |  |
| --- | --- |
| *Syncopation* | *Definitions* |
| Info | Information |
| IDE | *Integrated Drive Electronics* |
| Web Scrapping | a technology solution to extract data from web sites, in a quick, efficient and automated manner, offering data in a more structured and easier to use format, either for B2B or for B2C processes. |
| IEEE | Institute of Electrical and Electronics Engineers |
| OOA | Object Oriented Architecture: is the latest version of call-and-return architecture and consist of the bundling of data and methods. |
| SOA | Service Oriented Architecture. |

**1.5-Overview Of Document**

* section 1:is introduction to the system contain main points:
* Purpose  : Identify the  purpose  of this SDD  and  its intended  audience.
* Scope : Provide a description and scope of the software and explain the goals, objectives and benefits of your project. This will provide the basis for the brief description of your product.
* Overview: Provide an overview of this document and its organization.
* Reference Material  : List any documents, if any, which were used as sources of information for the test plan.
* Definitions and Acronyms: Provide  definitions of all terms, acronyms, and  abbreviations that might exist to  properly  interpret the SDD. These definitions should be items used in the SDD that are most likely not  known to the audience.
* Section 2:show the used external system
* Section 3: Give a general description of the functionality, context and design of your project. Provide any  background information if necessary.
* Section 4 :is the System Architecture that is an abstract description of the system. This is a very high level overview of how responsibilities of the system were partitioned and then assigned to subsystems and explains the relationships between these subsystems. Provide a diagram showing the major subsystems and their interconnections ,it consist of :
* Architectural Design :Develop a modular program structure and explain the relationships between the modules to  achieve  the  complete  functionality of the  system.
* Decomposition Description :Provide a decomposition of the subsystems in the architectural design. Supplement with text  as needed.
* Design Rationale :Discuss the rationale for selecting the architecture described in Architectural Design including critical issues and  trade/offs that were  considered.
* Section 5 :is the data designed   section
* Data Description: Explain how the information domain of your system istransformed into data structures.
* Data Dictionary : Alphabetically list the system entities or major data along with their types and descriptions.
* Section 6 : In this section, we take a closer look at what each component does in a more systematic way.
* Section 7 : consist of :
* Overview of User Interface :Describe the functionality of the system from the user’s perspective.
* Screen Images :Display screenshots showing the interface from the user’s perspective.
* Screen Objects and Actions: A discussion of screen objects and actions associated with those objects..
* Section 8:Provide a cross­ reference that traces components and data structures to the requirements in your SRS document. Use  a  tabular  format to show  which system  components satisfy each of the  functional  requirements from the SRS

**1.6 Refrances**

**2. External Interface**

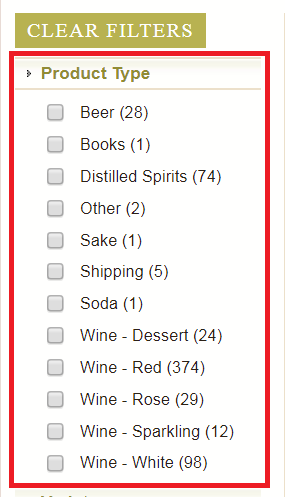
There are one external system that our system will be interfacing to. this is :

* Website (Klwines).
* G-mail.

**Website (Klwines)**

****

**Choose Category in a Website**

****

**You can do the same thing of selecting category in a website by changing the identifier number of category like in this case 10**

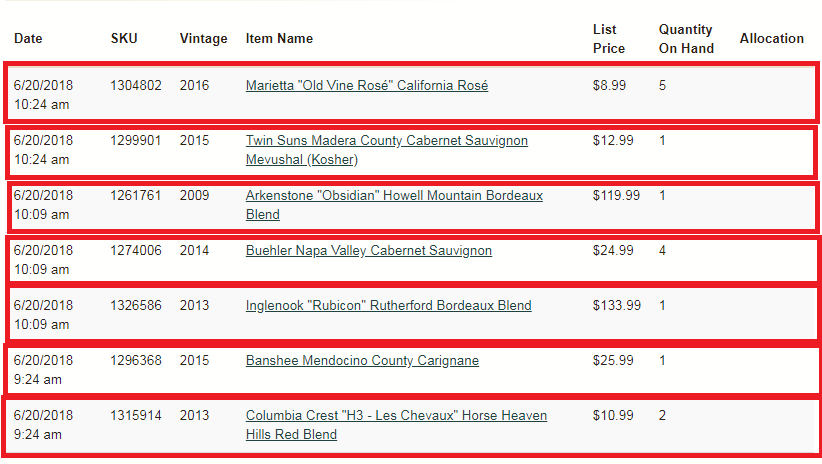
**For Distilled Spirits and in the last Number of Page**

**C:\Users\Aly\Desktop\web scraping\python win\GuI\gui\Git Klwines v 4.0\klwines V 5\klwines\version 9\klwines\New folder\New folder\klwines\klwines 9\klwines\Images\Website\URL.png**

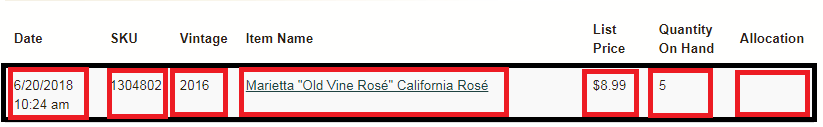
**We detect the last page by scrape the number near to the category name separately**

**C:\Users\Aly\Desktop\web scraping\python win\GuI\gui\Git Klwines v 4.0\klwines V 5\klwines\version 9\klwines\New folder\New folder\klwines\klwines 9\klwines\Images\Website\item count.png**

**Row of the website we need to scrape**

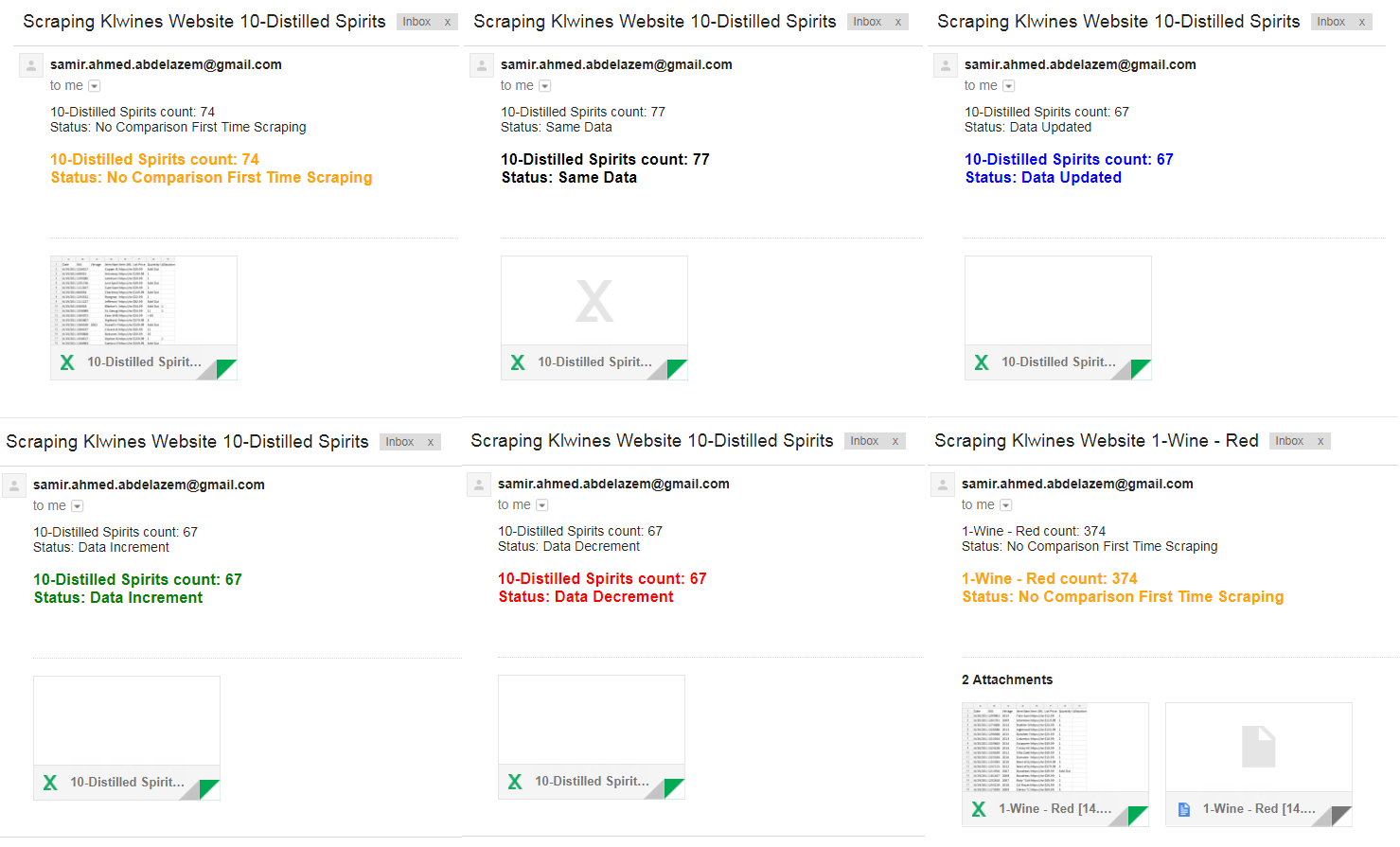
****

**Take each row and make a loop to take the 7 col from each row**

****

**Gmail**

**The output Email**

****

**3. SYSTEM OVERVIEW**

As mentioned in our SRS, KLWINES SCRAPPER has two types of End Users: **External System**(klwines website )and **user(DRUNKEN DIPLOMACY),** firstly the user choose the wine type and send a request to the system ,system will connect with the website to get data , scrap tables ,store it in Excel file and json file ,and finally send an E-mail with latest file and file status .

Figure 13: System Context diagram

**4. System Architecture**

## 4.1-ArchitecturalDesign

* + - * The Architecture pattern to be used for this system is **pipe-filter** architecture to defines the system as a whole.
      * **Down a Level, Layered** architecture to be used to define how system components interact directly with the adjacent layer, and each layer can communicate only with the one below as a set of services and direction of dependencies is clear as shown in figure 3.1, it consists of 4 layers:

1. Choose type layer: An interface layer at which end user interacts with system through.
2. Scrap and Store Layer: Layer that scrap the website and store it into Excel file and convert Excel into Json if needed.

3. Compare Layer: Layer at which the system compare between latest Excel file if exist

4- Send Layer: Layer at which the system send E-mail attached with the latest to files

5. Scheduling Layer: Layer at which the system scheduling scrape and send E-mail attached with the latest to file

**Down level layers** based on **OO architectures** that would help **increasing cohesion**(Object cohesion) and maintaining **loose coupling (Data coupling)** as every class has a **single responsibility** and their communication is via parameters as objects is being invoked by other objects.

* + - * Each Layer consists of set of modules or at least one module (object).
      * OO Architecture emphasis SOLID principles as mentioned before.
      * Design Patterns to be used for system

Figure 14: Layered architecture for SCMS

## 4.2 Decomposition Description

Our system is interact with an external system and has two phases:

**Scraping Phase and send phase**. The input of the system could be given from the user.

* In **scrapping phase**  at which user choose the type, then click scrap button ,the system connect with the website and scrap tables and store it in Excel file .
* In **send phase** at which the system compare the latest to file ,convert the Excel file into Json file if the user want and send an E-mail attached by the latest two files .

All the previous mentioned in Data flow diagram level 1 in figure 3.2.1.

Figure 15: DFD Level 0

Figure 16: DFD level 1

Figure 16: DFD level 2

## 4.3Design Rationale

* + - * The reasons for using **pipe -filter** Architecture are:

● Simple composition

● Reuse

● Prototype

● Easy growth & evolution

● Concurrency & parallelism

* The reasons for using Call and return architectures in system (OO and Layered):

1. Increasing cohesion for modules due to OOP concept and parts of each module fits together very well.
2. Reusability.
3. Extensible.
4. More closely related to how we view world objects.
5. Low coupling between components as each module have it’s specific.
6. Good ability to manage construction and destruction of objects, centralizing these operations in one place.
7. Relatively easy to distribute objects.
8. Can easily add, modify or replace an object for the system.
9. As for Layered System:
10. **Design**: based on increasing levels of abstraction.
11. **Enhancement**: since changes to the function of one layer affects at most two other layers.
12. **Reuse**: since different implementations (with identical interfaces) of the same layer can be used interchangeably.

* Trade-off for using that architectures:

1. Adding layers increase the risk of error
2. When the identity of an object changes it is necessary to modify all objects that invoke it.
3. The distinction between an object changing its content, and becoming a new object are too harsh.

# 4. DATA DESIGN

## 4.1DataDescription

The data is handled in files. We are using the following formats (Excel, Json ) in our system. We save the scraped tables in Excel file and if the user want we can convert the Excel file into Json file.

## 4.2 Data Dictionary

System has several modules, each module has set of functions that has some inputs and outputs. Some modules share common data structures.

|  |  |  |  |
| --- | --- | --- | --- |
| **Module** | **Function** | **Attribute Name** | **Type** |
| check | Internet\_Connection | REMOTE\_SERVER | string |
| check | Internet\_Connection | socket\_host | Boolean |
| check | Website\_Connection | request\_status | integer |
| check | Website\_Connection | request\_connect | string |
| check | Gmail\_Login | Gmail\_Status | string |
| check | Files\_Found | list1 | array |
| check | Files\_Found | new | array |
| check | Files\_Found | files\_path | string |
| check | Files\_Found | files | array |
| check | Files\_Found | files\_name | array |
| compare | latest\_two\_Excel | Old\_Excel | String |
| compare | latest\_two\_Excel | New\_Excel | String |
| compare | latest\_two\_Excel | one\_file | String |
| compare | Excel\_Compare | file1 | String |
| compare | Excel\_Compare | file2 | String |
| compare | Excel\_Compare | flage | Integer |
| compare | Excel\_Compare | dataFrame1 | object |
| compare | Excel\_Compare | dataFrame1 | object |
| compare | Excel\_Compare | data1 | integer |
| compare | Excel\_Compare | data2 | integer |
| compare | base\_email | base | integer |
| compare | base\_email | body | String |
| compare | base\_email | html1 | String |
| compare | base\_email | subject | String |
| files | latest\_one\_file | Excel\_Path | String |
| files | latest\_one\_file | Excel\_name | String |
| files | latest\_two\_files | Excel\_Path | String |
| files | latest\_two\_files | Excel\_name | String |
| files | latest\_two\_files | Json\_Path | String |
| files | latest\_two\_files | Json\_name | String |
| get | scrape class | identifier | Integer |
| get | scrape class | wine\_list | array |
| get | scrape class | identifier\_value | String |
| get | Row | url | String |
| get | Row | row\_blocks | array |
| get | Col | col\_text | String |
| get | Col | status | String |
| get | Col | col\_href | String |
| get | Data | Wb | Object |
| get | Data | Ws | Object |
| get | Data | current\_page | Integer |
| get | Data | out of | Integer |
| get | Data | td1 | String |
| get | Data | td2 | String |
| get | Data | td3 | String |
| get | Data | td4 | String |
| get | Data | td5 | String |
| get | Data | td6 | String |
| get | Data | td7 | String |
| get | Data | td8 | String |
| get | Json | file | String |
| get | Json | file2 | String |
| schedule | schedule\_time | schedule\_status | string |
| send | email\_send | filename | String |
| send | email\_send | excel\_name | String |
| send | email\_send | Base | Integer |
| send | email\_send | subject | String |
| send | email\_send | body | String |
| send | email\_send | html1 | String |
| send | email\_send\_two\_attachments | filename | String |
| send | email\_send\_two\_attachments | excel\_name | String |
| send | email\_send\_two\_attachments | Base | Integer |
| send | email\_send\_two\_attachments | subject | String |
| send | email\_send\_two\_attachments | body | String |
| send | email\_send\_two\_attachments | html1 | String |

**5. COMPONENT DESIGN**

## 5. Modules

## Example : 5.1 check Module

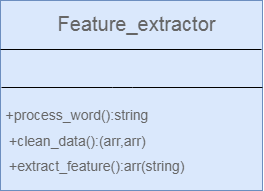


Figure 21 : Feature extractor Module

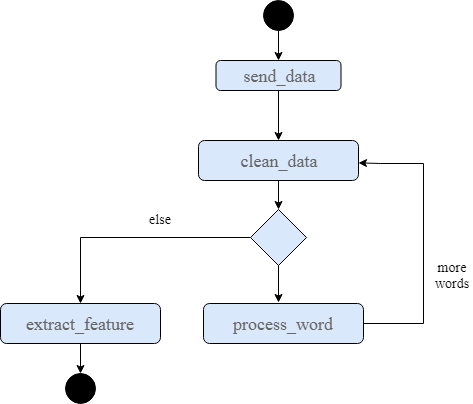


Figure 22: Feature extractor module activity diagram

The activity diagram shown in the figure 5.1.2 show the interaction between functions in that module and how they communicate together.

**5.1.1 Internet\_Connection () :**

- Purpose: check the internet connection in the computer

- Inputs: No Input

- Outputs: True if internet work and else send false.

- Called by: process\_data() from Classifier class.

- Calls: No Calls.

- Algorithm:

1. Check the internet connection
2. Return True or False

**5.1.2 Website\_Connection () :**

- Purpose: check the website availability

- Inputs: No Input

- Outputs: return string 'Website Working' or 'Website Down' .

- Called by: process\_data() from Classifier class.

- Calls: No Calls.

- Algorithm:

1. Send a request to a website server
2. If the request is 200
3. Return website working
4. Else Return Website Down

**5.1.3 Gmail\_Login () :**

- Purpose: check the the Gmail login

- Inputs: User\_Email , User\_Password

- Outputs: string Gmail\_Status can be 'Login Successful' or 'Login Failed'

- Called by: process\_data() from Classifier class.

- Calls: No Calls.

- Algorithm:

1. Try to connect gmail server
2. Request by email and password
3. If true send 'Login Successful'
4. If false 'Login Failed'

**5.1.4 Files\_Found () :**

- Purpose: check the excel files found in Folder named (Files)

- Inputs: No Input

- Outputs: return array of number that found in the name of every folder

- Called by: process\_data() from Classifier class.

- Calls: No Calls.

- Algorithm:

1. Check the current path
2. Add files path to the current path
3. Detect all files with .xlsx extension
4. Split file name to take just first letter
5. Convert letter into integer
6. Return all number in array

## Example : 5.2 compare Module

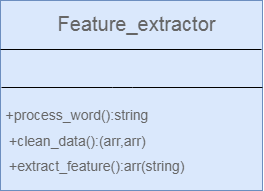


Figure 21 : Feature extractor Module

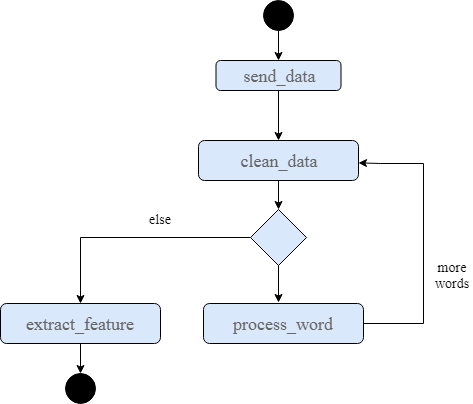


Figure 22: Feature extractor module activity diagram

The activity diagram shown in the figure 5.1.2 show the interaction between functions in that module and how they communicate together.

**5.2.1 latest\_two\_Excel () :**

- Purpose: get the name of the latest exist excel file in the folder named (Files) according to the identifier select by user and return flage to use it in Excel compare

- Inputs: int identifier , string identifier\_value

- Outputs: 3 string Old\_Excel ,New\_Excel , one\_file

- Called by: Excel\_Compare () from same module base class.

- Calls: No Calls.

- Algorithm:

1. Make a initialization of one\_file
2. Get current path
3. Get the files that has excel extension and with identifier
4. Arrange it in reverse order in the files array
5. Take the 0 in the files array as New\_Excel
6. Take the 1 in the files array as Old\_Excel
7. if there is no 1 in the array as Old\_Excel

**5.2.2 Excel\_Compare () :**

- Purpose: check the website availability

- Inputs: string file1 , string file2 , string one\_file

- Outputs: return string 'Website Working' or 'Website Down'.

- Called by: base\_email () from same class.

- Calls: latest\_two\_Excel() from same class

- Algorithm:

1. take latest two excel files and the flag check if there are just one Excel file with that identifier

2. if one flage return flag 4 and that mean there are just one Excel file

3. compare between the two excel file

4. if the two files have same length of column go through another

5. if the two excel file have the same all data return flag 0

6. if the there are a difference return flag 1

7. else the two file have not the same length and the first greater than second return 2

8. else the two file have not the same length and the second greater than first return 3

**5.2.3 base\_email () :**

- Purpose: prepare the data add to the email like subject , body and html

- Inputs: integer base , integer Page\_Count\_Number

- Outputs: string subject, string body, string html1

- Called by: send\_email() from send module.

- Calls: Excel\_Compare() from same class and page\_count() from get module

- Algorithm:

1- Take base from Excel compare according that body and html change

2- if base equal 0 same data

3- If base equal 1 data updated

4- If base equal 2 data increment

5- If base equal 3 data decrement

6- If base equal 4 No Comparison First Time Scraping

7- Return subject , body , html1

## Example : 5.3 Files Module

**5.3.1 latest\_one\_file () :**

- Purpose: search for the latest Excel file created in (Files) folder

- Inputs: no inputs

- Outputs: strings Excel\_Path , String Excel\_name

- Called by: email\_send() at send module

- Calls: No Calls

- Algorithm:

1- Detect the path of the system

2- add Files to go through Folder Files

3- check all the Excel file in the Files directory

4- take first one after make reverse arrange of array

**5.3.2latest\_two\_files () :**

- Purpose: search for the latest Excel file and the Json convert of the same excel file created in (Files) directory

- Inputs: no inputs

- Outputs: String Excel\_Path, String Json\_Path , String Excel\_name , String Json\_name

- Called by: email\_send\_two\_attachments () at send module

- Calls: No Calls

- Algorithm:

1- Detect the path of the system

2- Add Files to go through Folder Files

3- Check all the Excel and Json file in the Files directory

4- Take first one after make reverse arrange of array

## Example : 5.4 get Module

**5.4.1 row () :**

- Purpose: scrape all the row of the table from website klwines with specific identifier

- Inputs: integer Page , integer identifier

- Outputs: array row\_blocks

- Called by: data() from same module

- Calls: No Calls

- Algorithm:

1- Request the data from specific identifier for specific category and page number

2 - Extract the data of rows in form of array of html in string format

**5.4.2 col () :**

- Purpose: go through each row and scrape the 7 col of the table data

- Inputs: row\_one\_block , col\_number , status

- Outputs: 1 string col\_text or col\_href

- Called by: data() from same module

- Calls: No Calls

- Algorithm:

1- Check the status if text it will scrape text if href it will scrape url

2- Try to scrape all data if found else return empty string

**5.4.3 Page\_Count () :**

- Purpose: to scrape the count of product item for each category

- Inputs: no inputs

- Outputs: int(count)

- Called by: data() from same module and email\_send() and email\_send\_two\_attached() from send module

- Calls: No Calls

- Algorithm:

1-url to be requested

2- Take number from request by beautiful soap

3- Send the count number

**5.4.4 data () :**

- Purpose: to scrape data of a website table of specific catergory and store it into Excel file

- Inputs: integer identifier , string identifier\_value ,

- Outputs: no output

- Called by: tab2\_action\_auto\_click() and ClickAction3() from klwines module

- Calls: Page\_Count() and row() and col() from same module

- Algorithm:

1- detect the path of the directory

2- add Files to the directory if not found create the folder

3- open excel and add the table title

4- move through a loop start of one and this one for first page

5 – and get the page\_count to make the outof and this outof is the last page

6 – for loop to go through rows of row function

7- col function to take the 7 text and 1 href from table

8- if the first col not empty or No results with filters selected

8- save all of that in excel file

**5.4.5 json () :**

- Purpose: convert Excel file into json

- Inputs: string latest excel file

- Outputs: no output

- Called by: email\_send\_two\_attached() from send module

- Calls: latest\_one\_file() from file module

- Algorithm:

1- Try to convert excel into json

2- if the identifier file is not have Excel file

3- conver the latest excel file

## 5.5 Schedule Module

**5.5.1 schedule\_time () :**

- Purpose: to make function of scudling\_scraping\_time be scduleling every period of minutes take from user

- Inputs: no input

- Outputs: int Every\_Minutes

- Called by: tab2\_action\_auto\_click() from klwines module

- Calls: scudling\_scraping\_email from same class and run\_pending() from schedule module

- Algorithm:

1- Take schedule ever period of minutes from user

2- Print schedule\_status

3 – While 1 schedule.run\_pending

4- Return schedule\_status

**5.5.2 scudling\_scraping\_email () :**

- Purpose: faction to be scheduling scrape and send email

- Inputs : string User\_Email , string User\_Password , string Send\_To - Outputs: no output

- Called by: schedule\_time from same class

- Calls: data() from get module and email\_send() from send module

- Algorithm:

1- Scrape from function scrape

2- Send email with attached scraped Excel file

## 5.5 Send Module

**5.6.1 email\_send () :**

- Purpose: sending email with attached Excel file , subject , body , html

- Inputs: stringe User\_Email , stringe User\_Password , stringe Send\_To , stringe identifier\_value, integer identifier , string file1 , string body , string html1 , string subject

- Outputs: no outputs

- Called by: ClickAction3() from klwines module and scudling\_scraping\_email from schedule module

- Calls: latest\_one\_file() from file module base\_email() from compare module

- Algorithm:

1- Get file name want to attached to the email

2 – Get email data from compare module from base\_email function data is subject body html

3- Prepare message take from and to and subject

4- Attached body and html

5- Open the excel file to attached it

6- Login into the server

7- Send email with all the previous information

8- Quit server

**5.6.2 email\_send\_two\_attachments () :**

- Purpose: sending email with attached Excel and json file , subject , body , html

- Inputs: stringe User\_Email , stringe User\_Password , stringe Send\_To , stringe identifier\_value, integer identifier , string file1 , string file2 , string body , string html1 , string subject

- Outputs: no outputs

- Called by: ClickAction3() from klwines module

- Calls: latest\_two\_file() from file module base\_email() from compare module

- Algorithm:

1- Get file name want to attached to the email

2 – Get email data from compare module from base\_email function data is subject body html

3- Prepare message take from and to and subject

4- Attached body and html

5- Open the excel file and Json file to attached it

6- Login into the server

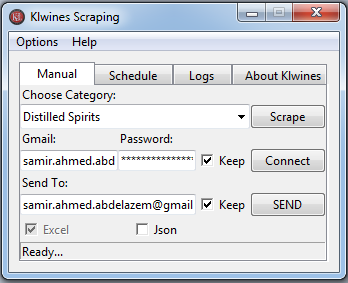
7- Send email with all the previous information

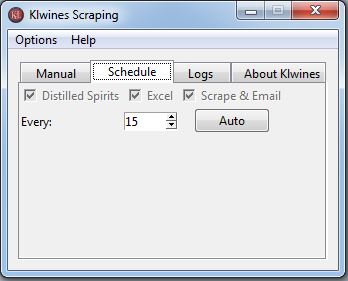
8- Quit server

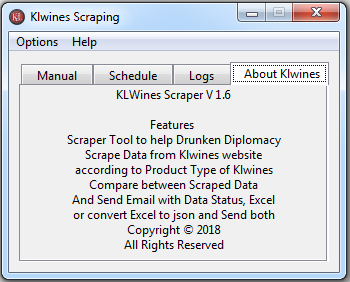
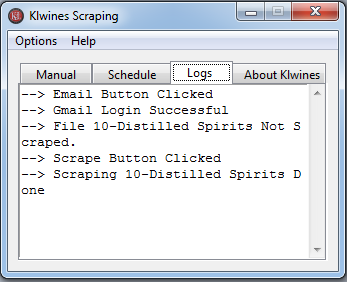
**7. HUMAN INTERFACE DESIGN:**

* The User Interface is a crucial aspect of the system in terms of both what the client wants and needs. For this reason, there is an overview of the User Interface in the Software Requirements Specification (SRS). This section will detail all aspects of the interfacing of system and its design, and thus will be more extensive than the SRS section. In This document different User Interface sections are presented.
* The Graphical User Interface (GUI), for the purpose of this description, has been broken up into three main sections. These are:
* Tab1
* Categories combo box
* scrap Button
* connect Button
* send Button
* Gmail Entry Widgets
* password Entry Widgets
* send to Entry Widgets
* keep Gmail – Password checkbox
* keep Send To checkbox
* Excel checkbox
* Json checkbox
* Status bar

* Tab2
* Auto Button
* Every: Spin
* Distilled Spirits checkbox
* Excel checkbox
* Scrape & Email checkbox
* Tab3
* Logs
* Tab4
* About Label







## 6.1 Manual Tab

## 6.1.1 Category Combo box

* Through this combo box the user can choose the category needed to be scrapped

## 6.1.2 Scrap Button

* Through this button the scrapping function is called where the wanted data is scrapped and placed in a new file inside the "files" folder with the name of the selected category

## 6.1.3 Connect Button

* Checks Connection of Gmail account

## 6.1.4 Send Button

* Through this button sending and comparing functions are called to send the scrapped data through email and to compare the files to alert and inform the user in case of new data.

## 6.1.5 Gmail Entry Widget

* Where the user enters his email address

## 6.1.6 Password Entry Widget

* Where the user enters the password for his email address

## 6.1.7 Send to Entry Widget

* Where the user enters the email address to where the data is supposed to be sent

## 6.1.8 Keep Gmail – Password Checkbox

* By checking this checkbox the application now is enabled to remember the user's Gmail and password for further usage

## 6.1.9 Keep Send To Checkbox

* By checking this checkbox the application now is enabled to remember the Gmail to where the data is supposed to be sent for further usage

## 6.1.10 Json Checkbox

* To make the file convert from excel into Json

## 6.1.11 Status Bar

* Shows the status and progress of the process

## 6.2 Schedule Tab

* To schedule send scrape and send email of Distilled Spirits category every period of minutes

## 6.2.1 Auto Button

* Sets a schedule for a scrape and email process

## 6.2.2 Every: Spin

* Through which the user sets the time interval for the application to automatically repeats the process of scraping the data and sending it through email and time in minutes

## 6.2.3 Distilled Spirits Checkbox

* Enabled by default the category user want to scrape

## 6.2.4 Excel Checkbox

* Enabled by default the format user want the email file attached with

## 6.2.5 Scrap & Email Checkbox

* Enabled by default because user want scheduling scrape and send scraped file as attached file

## 6.3 Logs Tab

* A log for documenting and showing each step taken.

## 6.4 About Klwines Tab

* A simple description for what this application is, what it is for and how it works.

**9. Requirement Matrix**

|  |  |
| --- | --- |
| **Component (class) satisfy that requirement** | **Requirement** |
| **get module** | Scrapping tables |
| **get module** | Store the scrapped data |
| **get module** | Convert Excel file into Json file |
| **check module** | Check for file exist |
| **check module** | Check for internet connection |
| **check module** | Check for website availability |
| **check module** | Check for Gmail login |
| **compare module** | Compare the latest two files |
| **files module** | Prepare Email Data |
| **send module** | Send E-mail |
| **schedule module** | Scheduling |