# Design Document

# Brief Overview

The Program that has been created is a currency converter. The currency converter converts a selected currency and amount of that currency specified to another currency that also can be selected. This program helps to covert a specified amount of a certain currency to another currency according to the real-time exchange rates that are from google finance web site using only python.

The problems that this program solves is getting real-time exchange rates and returns any amount specified in a selected currency. The getting of this real-time data was difficult but by using an external python module called beautiful soup the program web scrapes the google finance web page for the relevant exchange rate data and then handles this data as a float.

A second problem that was solved was using a Graphical User Interface (GUI) to display results of the web scrap and to preform actions in the GUI that the program would recognise and then use the information supplied by these interactions with the GUI. TKinter GUI was used from the standard python library. This problem was solved using functions to handle the interaction with the GUI and with research into what TKinter could be used for and how to use the functions that TKinter provides.

# Explanation of logic of solutions

# Explanation of get\_currency module

This module is written to fetch the exchange rate between any two currency. There is a class created call currency that has a function called get\_anycurr. This function has two variables that are parsed into the function, these variables are added to the end of the URL that is used to get the exchange rate, therefore these variables are used to access the correct website. Inside the function there is a ‘try except’ that handles any errors that might have happened. If there is an error it would be from a loss in connection. An error message will print out no connection if there is no connection.

The meaty bit of the function fetches the exchange rate from google finance webpages. The program does this by firstly importing (at begging of programme) import requests which is a module that is externally downloaded that helps the programme to reach the web site. The second import is Beautiful Soup which is also a external module that must be downloaded. This module is used to sort through the source code of the website, which is the HTML and scripts that make the website function as it does.

The way Beautiful soup achieves its goal of getting the exchange rate is by firstly getting the source code of the website. Next a for loop is used to go through all the source code of the website and find all the tags in the HTNL source code that have the class name of ‘bld’. These results are then saved to a list. The first item of this list is then extracted as it is the exchange rate. The exchange rate then has the first 7 characters extracted as this is the part of the exchange rate that is only numbers. The result of this is then returned as floating number so the decimals numbers remain. This allows for the function to be called and the specified exchange rates will be returned.

# Why done this way

The get\_currency module is done in a separate module so it can be used by the main class and does not take up space and makes the layout easier. The use of requests and beautiful soup is used to simplify the process of getting the exchange rate. The use of a class was because the class allows for specifying a function and for making the programme code to look neater.

# Explanation of Main module

The main module uses the TKinter module for the graphical user interface (GUI) and imports the get\_currency module. The class exFrame contains five functions. The first function called \_\_init\_\_ which is the constructor of the class exFrame. This function is called when the class is called and its parameters are met. The function of this constructor is to create the frame for the GUI when the programme runs by calling the function initUI() which is where the graphics on the frame are created and by inheriting the functions of the TKinter parent class.

The second function is the initUI function which is where the graphics such as buttons, labels and list boxes are created. The most important function of the init function is handling user input in the GUI. These inputs are dealt with when the push\_button sub function is called. This function takes the input from the two selection boxes and the amount from the entry field and then passes it into the conversion function which will be discussed soon. The inputs are converted into the required output of the exchange rate. Which is displayed with the use of a label. The outputs are received with the use of variables that are made global in the exFrame class. These variables are then inputted into the labels which are displayed by the GUI at real-time as they can be changed.

The next two functions are on\_select\_lb1 and on\_select\_lb2. These two functions deal with the selection of items in the list boxes. When they are called, they use the selected items in the list boxes and then assign the index of the selected item to the push\_button function and they are used to update the symbol labels of the currencies.

The next function is the conversion function. This function takes the inputs from the push\_button function as parameters. The get\_currency module is then called which takes in the parameters of the function and outputs into a variable the exchange rate. The exchange rate is then multiplied by the amount entered into the entry field and then updates the a label which will then display the output on the GUI.

After the exFrame class ends then there is a main() function that basically calls TKinter and implements there framework for the GUI. The size of the frame is set here and the TKinter framework is told to loop until closed. The final if statement checks to see if this is the main module and imported by another module and then when this is true it runs the programme which it does by calling the function main.

# Why was is done this way

The code was done this way so the functions could be in the same class which would allow them to share variables and make the code easy to manipulate into the desired outcomes. The code is setup in a way that is more stable as it can have more error handling code. The use of the constructor and the main function allow for pythonic code and the final if statement allows for error handling and to run the programme when the code is run and is more pythonic.

# Retrospection

# Problems Faced

The first problem that was faced was getting the exchange rate of the of the currency to be converted too. This was solved through research for an external module, Beautiful Soup was found and is used. The thorough research went into the different functions that the Beautiful soup could do and eventually through trial and error I was able to identify the correct tag and class name to get the exchange rate from the google finance website.

The second problem that was faced was implementing the GUI. Research went into the different types and in the end the TKinter internal module was used. Through examinations of documents of the different functions that could be performed by TKinter the GUI was created and constant reference to those documents on the TKinter functions was used to figure out how to set commands and update label text.

The third problem that was faced was the receiving of inputs. Trying to get the index of the current selected item in each list box proved the most challenging. The use of a global variable within the exFrame was used to eventually store the index of the selected items. The variable needed to be assessed in another function which is why it become global. It was need so the correct dict key was slected and the correct value of the key was inputted into the conversion function from the push\_button function.

The fourth problem faced was implementing a picture into my GUI. Through many tries the picture would not pack itself into the GUI and did not work. TKinter has functions to implement images but through trial and error I could not get the image to show up in the programme. The possible reason for this is that the images could not be handled properly by TKinter as it only takes in certain formats of images.

# Improvements

The improvements that could have been made to the project include a add more currencies button that would allow the programme to get more exchange rates that the user might need. The overall code structure could have been more object orientated which would have allowed for neater and more understandable code. The layout could have been a lot better but is difficult to work with when using TKinter. The Layout is very simple and the design is not boring and could have used more pictures and more design elements.