Alzheimer’s currently affects around 850,000 individuals in the UK with approximately 210,000 people contracting dementia per year aswell as this, people above the ages of 80 have been found to have a 1 in 6 chance of having the disease, Dementia encompasses of various brain disorders that cause a deterioration of brain function. All of these conditions being generally gradual though and ultimately severe. Dementia symptoms include memory loss, confusion and problems with speech and comprehension problems and is a terminal condition. Overall treatment and care expenses in the UK alone has cost around £35 billion (Alzheimer’s Society, 2020)

Alzheimer’s has been identified to correlate with protein Aβ containing a glycine zipper motif, in which is deemed have an important role in the tissue formation of amyloid fibril formation which is thought to have a toxic pathogenic effect(Fonte, 2011).

Using a custom regex of (G|A)...(G)...(T|G) these stronger and weaker membrane proteins in which the motif is commonly found in can be used to search through Uniprot accession codes to find the matching motifs.

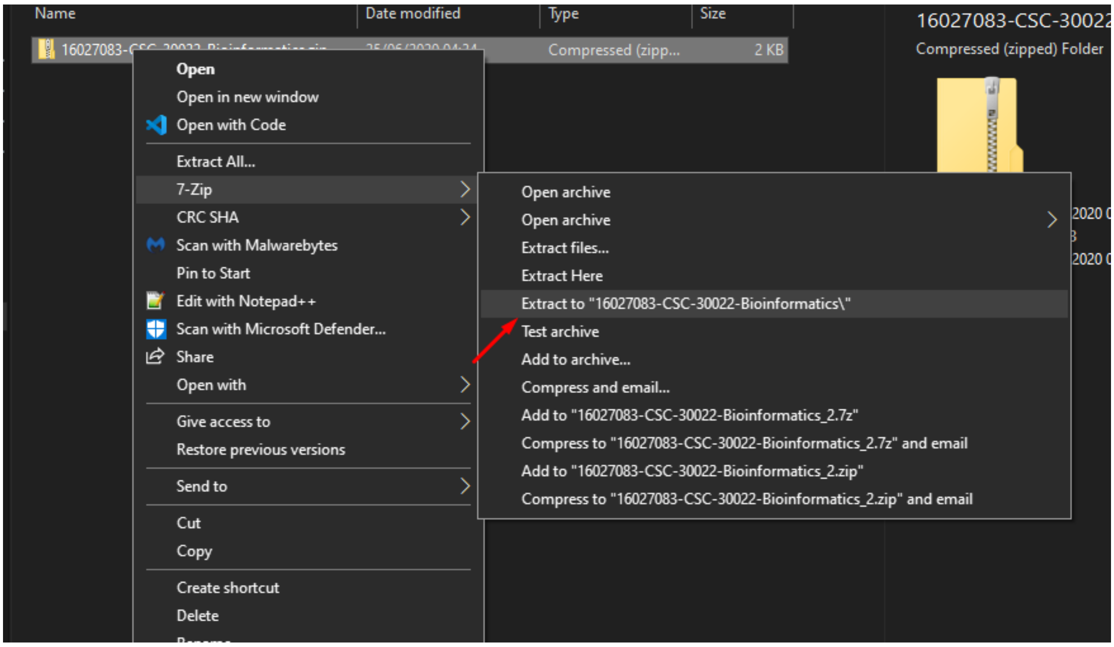
This regex expression for checking motifs was generated using the Expresso program, which provides a toolbox that enables a user to construct regular expressions using a collection of tabs from which you can pick all of the syntactical components desired. Test data may be interpreted or entered manually after creating an expression, and the standard expression can be performed against the data after which the search results are then shown (Ultrapico, 2010).

The tool utilises python due to being a relatively easy to learn programming language due to the syntax being easy to read and having a large community leading to lots of resources to be used meaning it can be used to solve a range of problems.

The program utilises Biopython a open source python library with functions for reading and writing various sequence file formats and sequence alignments within batches aswell as being able to access key online databases such as SwissProt(Cock, 2009).

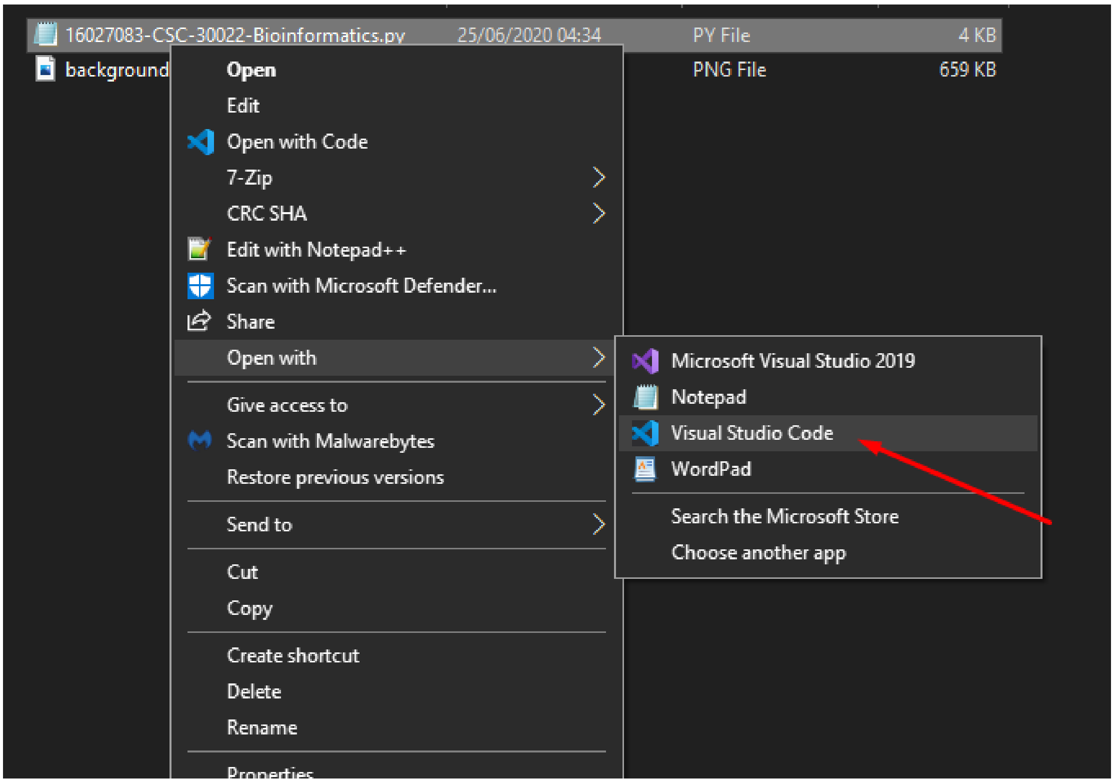
User Guide

Step 1: Folder Unzipping


Name
Date modified
Type
Size
16027083-CSC-30022-Bioinformatics.zip
Compressed (zipp...
2 KB
16027083-CSC-30022
Compressed (zipped) Folder
Open
Open in new window
Open with Code
Extract All...
7-Zip
CRC SHA
Scan with Malwarebytes
Pin to Start
Edit with Notepad++
Scan with Microsoft Defender...
Share
Open with
Give access to
Restore previous versions
Send to
Cut
Copy
Create shortcut
Delete
Open archive
Open archive
Extract files...
Extract Here
Extract to "16027083-CSC-30022-Bioinformatics\"
Test archive
Add to archive...
Compress and email...
Add to "16027083-CSC-30022-Bioinformatics_2.7z"
Compress to "16027083-CSC-30022-Bioinformatics_2.7z" and email
Add to "16027083-CSC-30022-Bioinformatics_2.zip"
Compress to "16027083-CSC-30022-Bioinformatics_2.zip" and email

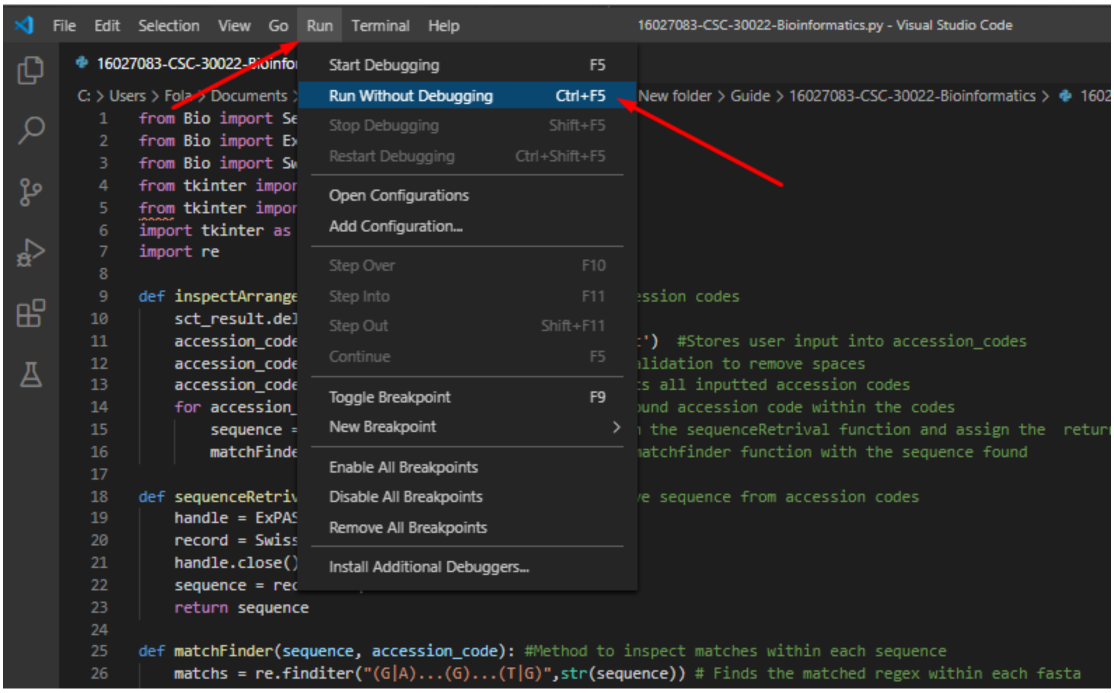
The first step after downloading the zip file containing the search tool would be to extract the zip folder this could be done through any extracting program for example 7zip.

Step 2: Opening Python file



Once installed you will want to run the extracted .py file and run it either through the command prompt for ease of use through Visual Studio Code

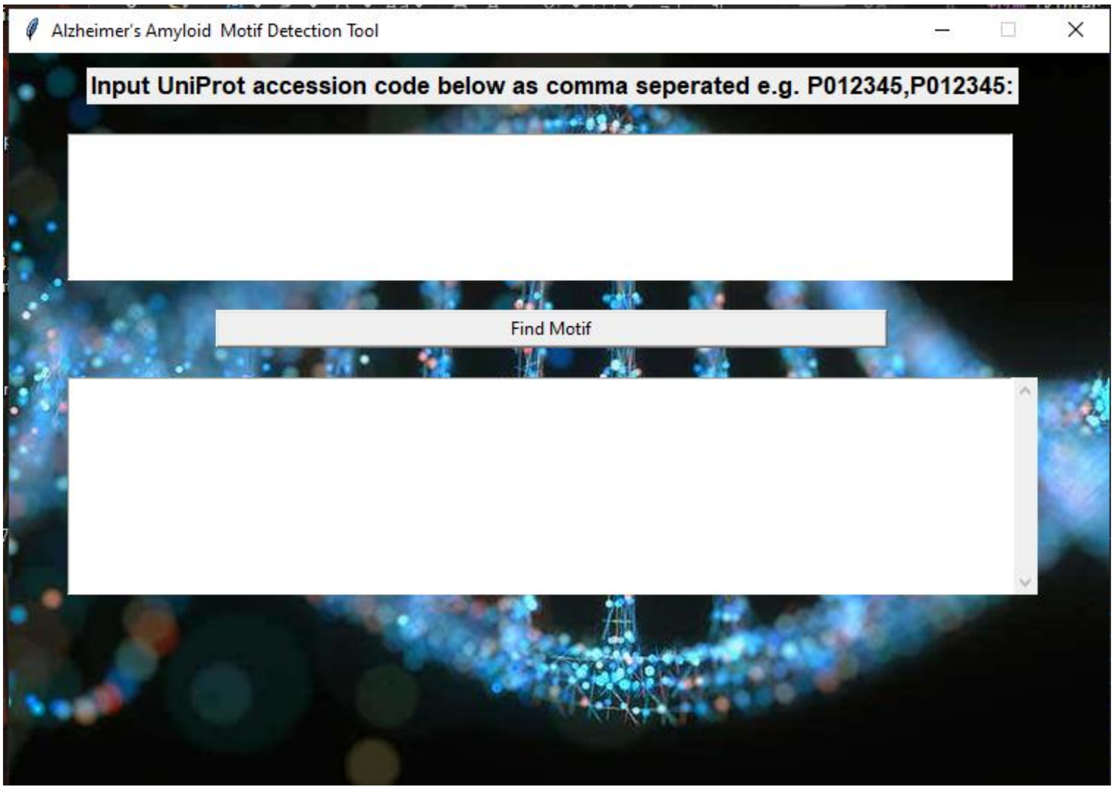
Step 3:Executing Python File



```
from Bio import Se
from Bio import Ex
from Bio import Sw
from tkinter impor
from tkinter impor
import tkinter as
import re

def inspectArrange                                  ession codes
    sct_result.del
    accession_code                                  ') #Stores user input into accession_codes
    accession_code                                  lidation to remove spaces
    accession_code                                  s all inputted accession codes
    for accession_                                  und accession code within the codes
        sequence =                                  the sequenceRetrival function and assign the  retur
        matchFinde                                  atchfinder function with the sequence found

def sequenceRetriv                                  e sequence from accession codes
    handle = ExPAS
    record = Swiss
    handle.close()
    sequence = rec
    return sequence

def matchFinder(sequence, accession_code): #Method to inspect matches within each sequence
    matchs = re.finditer("(G|A)...(G)...(T|G)",str(sequence)) # Finds the matched regex within each fasta
```

After opening file hover over the run button on the top of the application and then click ‘Run Without Debugging’ or press Ctrl+F5

Step 4:Inputting Accession codes

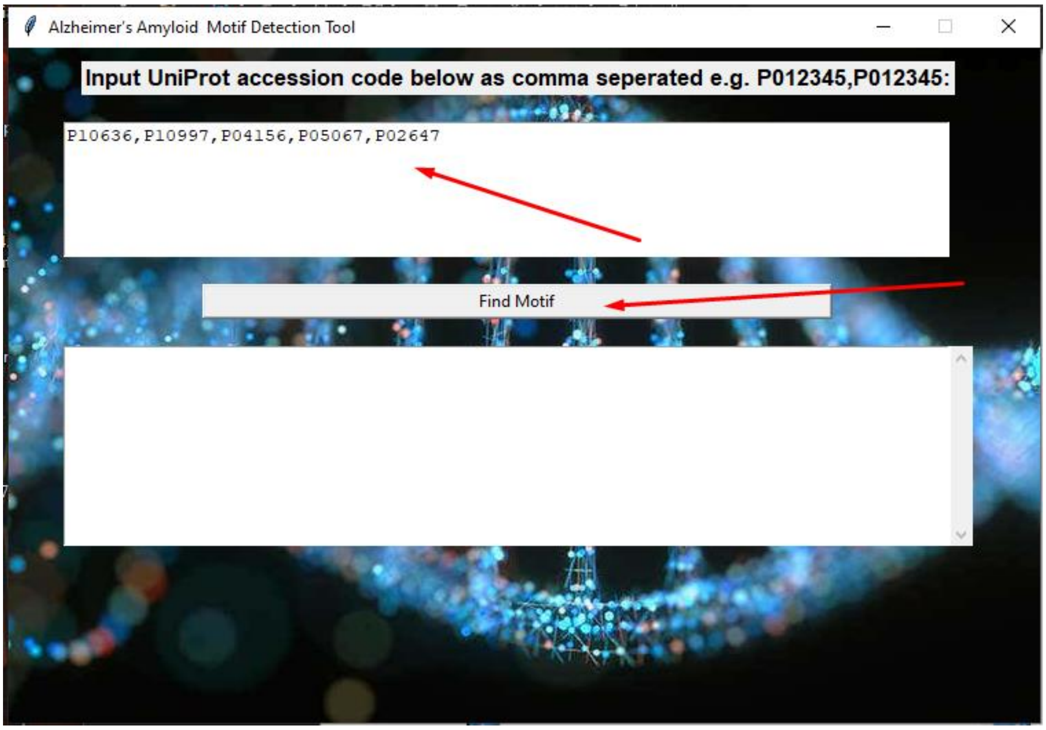


Input UniProt accession code below as comma seperated e.g. P012345,P012345:

After clicking run this tool will open up on your screen with instructions to insert your desired accession codes into the text box this can be either one or multiple accession codes though must be separated by a delimiter in this case being ‘,’

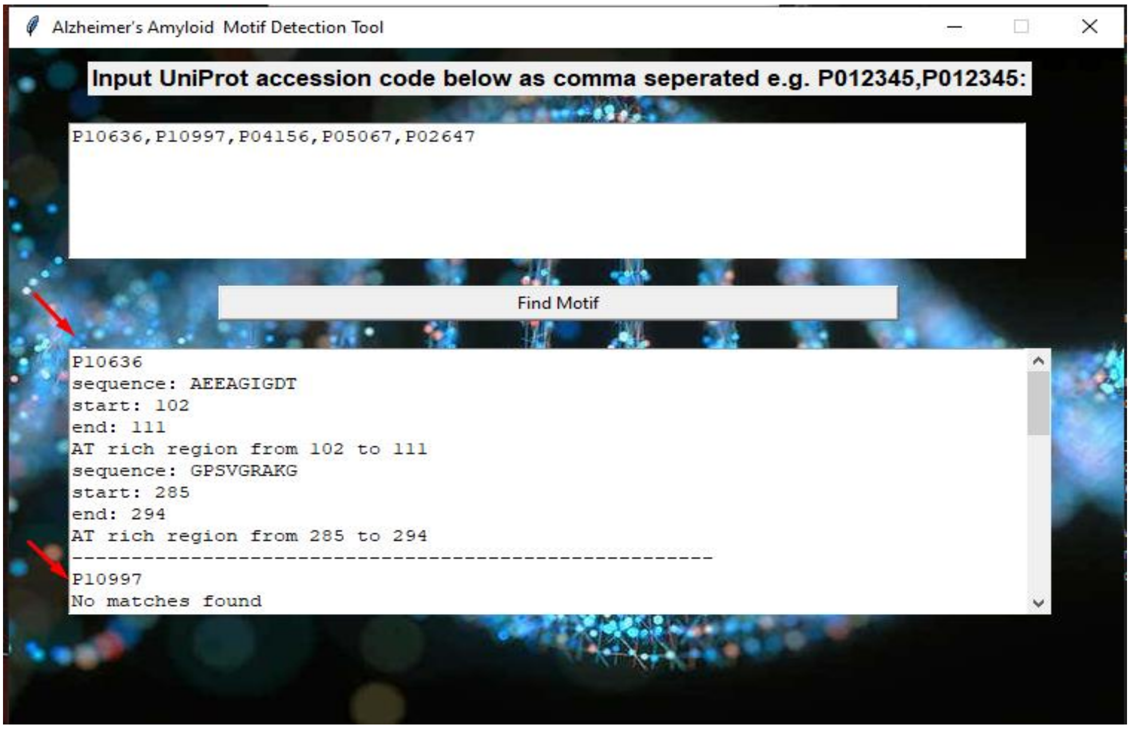
Step 5: Finding Corresponding Motifs

Once desired Accession codes have been inserted click the ‘Find Motif’ which will then print the motifs found in the box below.


Alzheimer's Amyloid Motif Detection Tool
Input UniProt accession code below as comma seperated e.g. P012345,P012345:
Find Motif

```
P10636,P10997,P04156,P05067,P02647
```

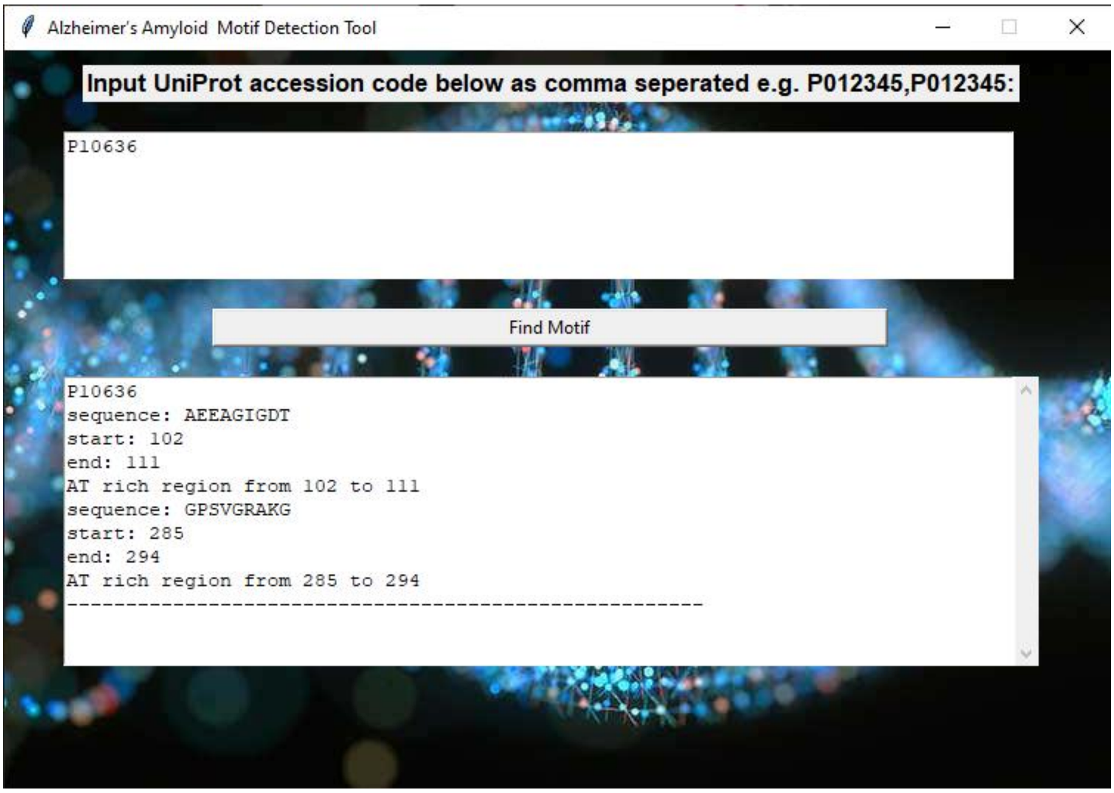
Step 6: View Results


Alzheimer's Amyloid Motif Detection Tool
Input UniProt accession code below as comma seperated e.g. P012345,P012345:
Find Motif

```
P10636,P10997,P04156,P05067,P02647
```

```
P10636
sequence: AEEAGIGDT
start: 102
end: 111
AT rich region from 102 to 111
sequence: GPSVGRAKG
start: 285
end: 294
AT rich region from 285 to 294
------------------------------------------------------
P10997
No matches found
```

Printed results for Motifs found will be shown in the box where the user can freely scroll through analysing the Accession code as well as the sequence, start and end of motif as well as informing the user of where in the sequence this match can be found. If a match isn’t found then the user will get a response of No matches found.

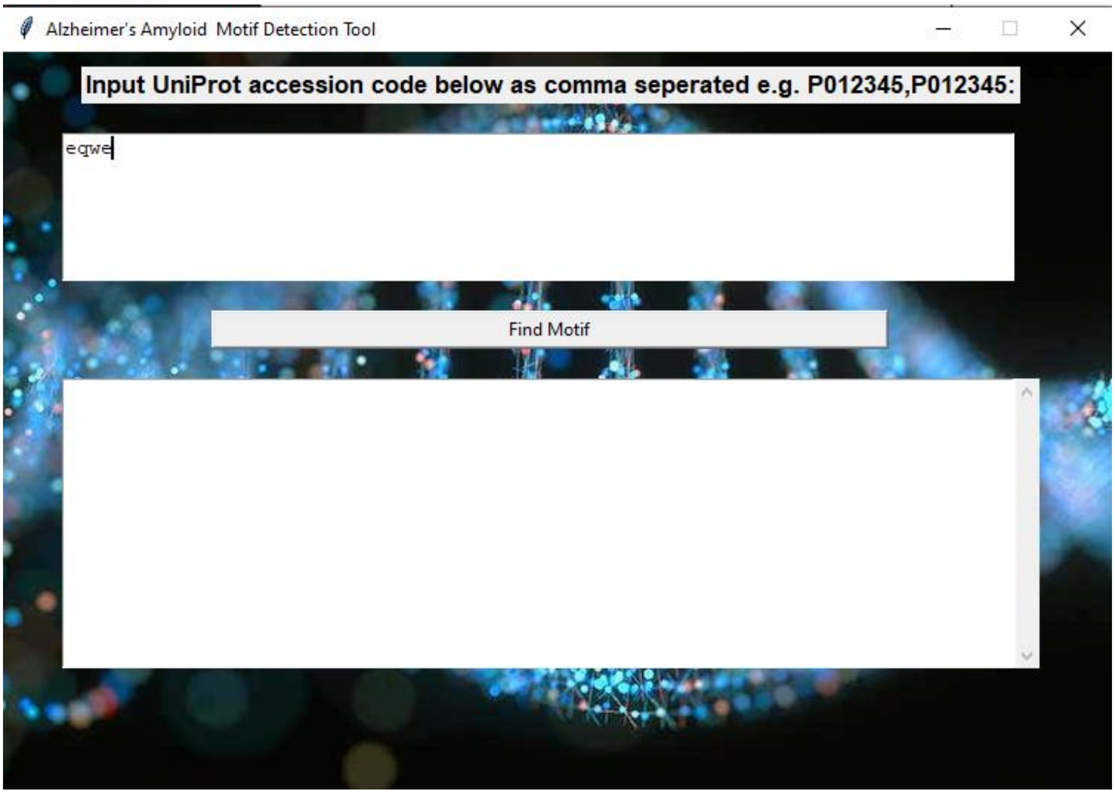


Input UniProt accession code below as comma seperated e.g. P012345,P012345:

```
P10636
```

```
P10636
sequence: AEEAGIGDT
start: 102
end: 111
AT rich region from 102 to 111
sequence: GPSVGRAKG
start: 285
end: 294
AT rich region from 285 to 294
------------------------------------------------------
```

Any previous results will automatically be cleared if new accession codes are entered on button press of the 'Find Motif'



Input UniProt accession code below as comma seperated e.g. P012345,P012345:

```
eqwe
```

If an accession code isn't inputted then no results on to whether a match was found or not will not be displayed.

Appendix

```
from Bio import SeqIO

from Bio import ExPASy

from Bio import SwissProt

from tkinter import *

from tkinter import scrolledtext

import tkinter as tk

import re


def inspectArrangement(): #Method for user to input accession codes

   sct_result.delete('1.0', END)

   accession_codes  = txt_input_field.get("1.0",'end-1c')  #Stores user input into accession_codes

   accession_codes = accession_codes.replace(" ","")#Validation to remove spaces

   accession_codes  = accession_codes.split(",") #splits all inputted accession codes

   for accession_code in accession_codes: #for every found accession code within the codes

      sequence = sequenceRetrival(accession_code) #run the sequenceRetrival function and assign the  returned value to the sequence variable

      matchFinder(sequence, accession_code)#runs the matchfinder function with the sequence found


def sequenceRetrival(accession_code): #Method to retrieve sequence from accession codes

   handle = ExPASy.get_sprot_raw(accession_code)

   record = SwissProt.read(handle)

   handle.close()

   sequence = record.sequence

   return sequence


def matchFinder(sequence, accession_code): #Method to inspect matches within each sequence

   matchs = re.finditer("(G|A)...(G)...(T|G)",str(sequence)) # Finds the matched regex within each fasta

   sct_result.insert(INSERT, accession_code + "\n")
```

```
    count = 0
    if matchs:
        for match in matchs:
            matchs_start = match.start()
            matchs_end = match.end()
            matchs_group = match.group()
            sct_result.insert(INSERT, "sequence: " + str(matchs_group) + "\n")
            sct_result.insert(INSERT, "start: " + str(matchs_start) + "\n")
            sct_result.insert(INSERT, "end: " + str(matchs_end) + "\n")
            sct_result.insert(INSERT, "AT rich region from " + str(matchs_start) + " to " + str(matchs_end
)+ "\n")
            count +=1
    if count == 0:    # if matches are empty responds with no matches found
        sct_result.insert(INSERT, "No matches found \n")
    sct_result.insert(INSERT,"---------------------------------------------------- \n")


if __name__ == '__main__': # Entry point for application
    window = tk.Tk() #Creates tkinter object


    window.title("Alzheimer’s Amyloid  Motif Detection Tool") # Settting up window and name
    window.geometry('750x500')
    window.configure(background = 'Grey')
    window.resizable(width=FALSE, height=FALSE)


    background_image = PhotoImage(file="background.png")
    background_label = Label(window, image=background_image)
    background_label.place(x=0,y=0, relwidth=1, relheight=1)


    lbl_instruction = Label(window, text="Input UniProt accession code below as comma seperated
e.g. P012345,P012345:", font = ('articulat',13,'bold')) #Creates and places instruction label
    lbl_instruction.grid(column=0, row=1, padx=40, pady=10)
```

```
txt_input_field = Text(window) #Creates and places input field
txt_input_field.grid(column=0, row=2, padx=40, pady=10, sticky="W")
txt_input_field.config(width=80, height = 6)

sct_result = scrolledtext.ScrolledText(window) #Creates and plces result field
sct_result.grid(column=0, row=4, padx=40, pady=10, sticky="W")
sct_result.config(width=80, height=12)

btn_run = Button(window, width=93, text="Find Motif", command=inspectArrangement) #Creates and places button and binds it to inspect arrangment method
btn_run.grid(column=0, row=3, padx=0, pady=10)
btn_run.config(width=64)

window.mainloop() #Stops the window from closing
```

References

1. Alzheimer's Society,(2020).*Facts for the media*
2. Fonte, V., Dostal, V., Roberts, C. M., Gonzales, P., Lacor, P., Magrane, J., ... & Link, C. D. (2011). *A glycine zipper motif mediates the formation of toxic β-amyloid oligomers in vitro and in vivo. Molecular neurodegeneration.*
3. Ultrapico.(2010). *Expresso 3.1*
4. Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., ... & De Hoon, M. J. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, *25*(11), 1422-1423.