CS350

PROJECT REPORT

Tolga Gümüşçü

Fatih Memiş

Part 2

Project Topic: Write your Python (.py) code to analyze file count, size and type information on two different operating systems.

Challenges to Solve:

- Coding the back-end side of our file analysis program
- Creating a GUI design.
- Coding the front-end side.
- Running our application on two different operating systems.

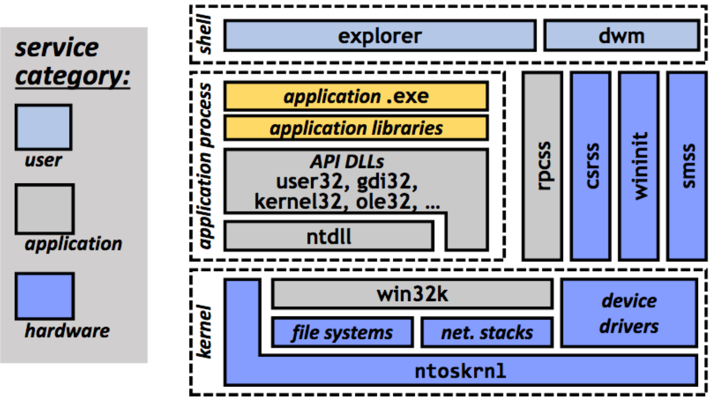
Operating Systems That We Are Using in This Project:

- Windows
- Linux

Libraries We Used

- OS Library
- Tkinter Library

OS Library

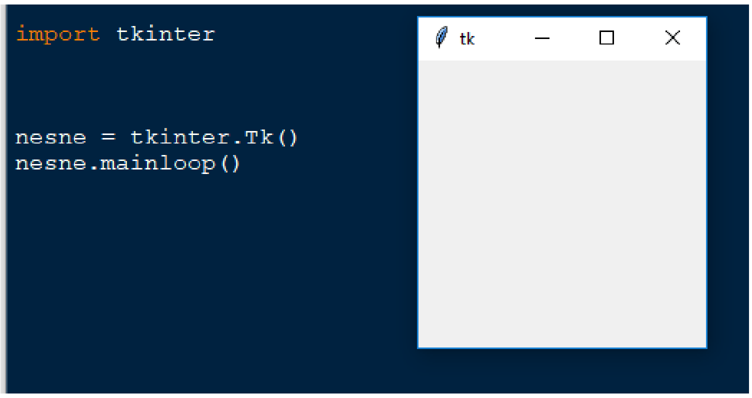


In Python, the OS module has functions for dealing with the operating system. Python's standard utility modules include OS.

Python's OS module includes functions for creating and removing directories (folders), retrieving their contents, altering and identifying the current directory, and more.

Tkinter Library

With Tkinter, we developed the GUI side of our application, which is the front-end part.



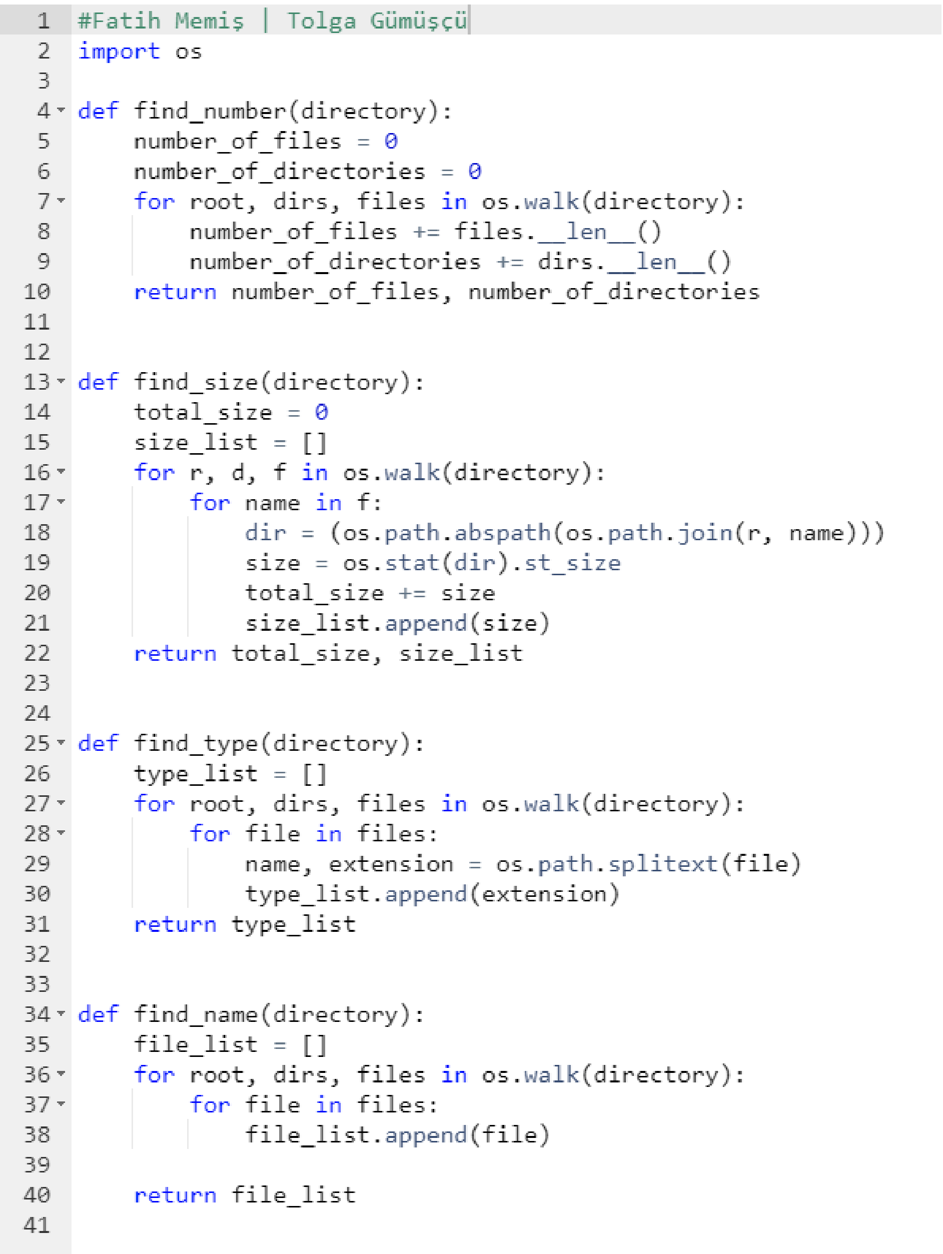
```
import tkinter


nesne = tkinter.Tk()
nesne.mainloop()
```

To create a tkinter app:

- Importing the module – tkinter
- Create the main window (container)
- Add any number of widgets to the main window
- Apply the event Trigger on the widgets.

Detailed Explanation:

BACKEND



```python
#Fatih Memiş | Tolga Gümüşçü
import os

def find_number(directory):
    number_of_files = 0
    number_of_directories = 0
    for root, dirs, files in os.walk(directory):
        number_of_files += files.__len__()
        number_of_directories += dirs.__len__()
    return number_of_files, number_of_directories


def find_size(directory):
    total_size = 0
    size_list = []
    for r, d, f in os.walk(directory):
        for name in f:
            dir = (os.path.abspath(os.path.join(r, name)))
            size = os.stat(dir).st_size
            total_size += size
            size_list.append(size)
    return total_size, size_list


def find_type(directory):
    type_list = []
    for root, dirs, files in os.walk(directory):
        for file in files:
            name, extension = os.path.splitext(file)
            type_list.append(extension)
    return type_list


def find_name(directory):
    file_list = []
    for root, dirs, files in os.walk(directory):
        for file in files:
            file_list.append(file)

    return file_list

```

find_number = In this function, we can count the number of all files in the target file and get a total number.

find_size = In this function, we access the size of the files in the target file.

find_type = In this function, we can learn the types of files we access in the target file.

find_name = In this function, we can learn the names of the files we access in the target file.

In Python, how does os.walk() work?

OS.walk() generates file names in a directory tree by walking top-down or bottom-up through it. It returns a 3-tuple for each directory in the tree rooted at directory top (including top itself) (dirpath, dirnames, filenames).

root : Prints only the directories that you specify.

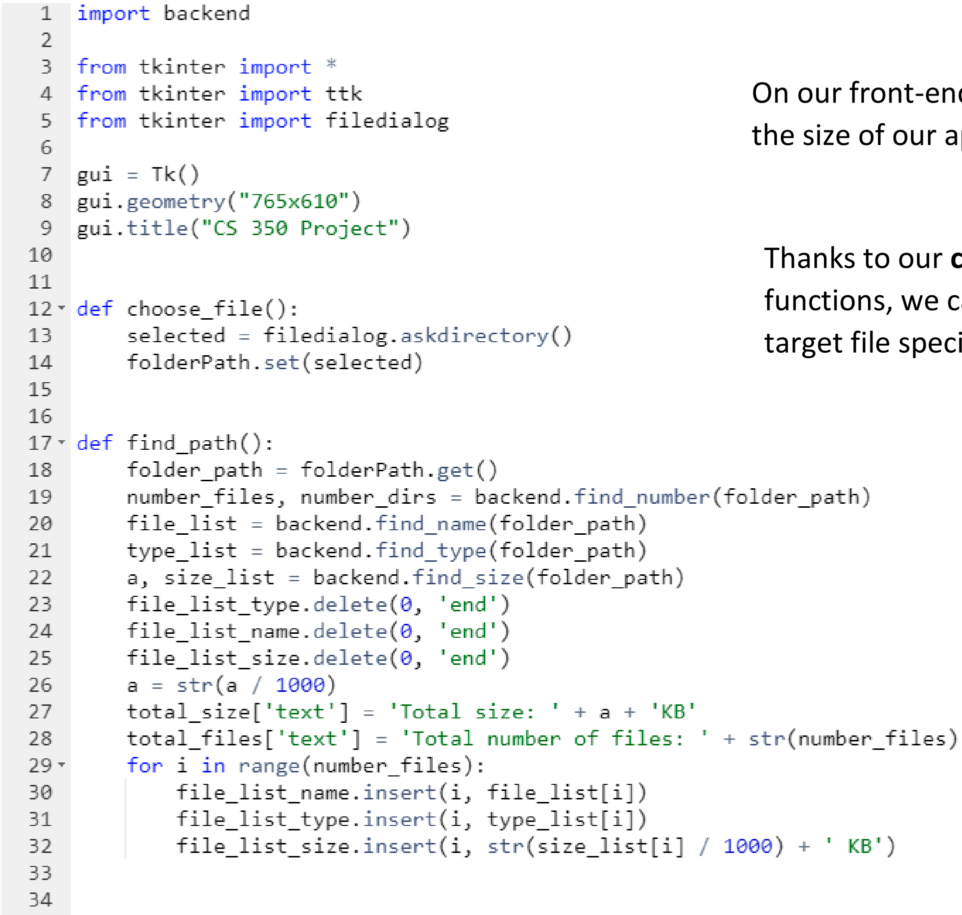
dirs : Prints subdirectories from the root directory.

files : Prints all files in the current directory and subdirectories.

FRONTEND

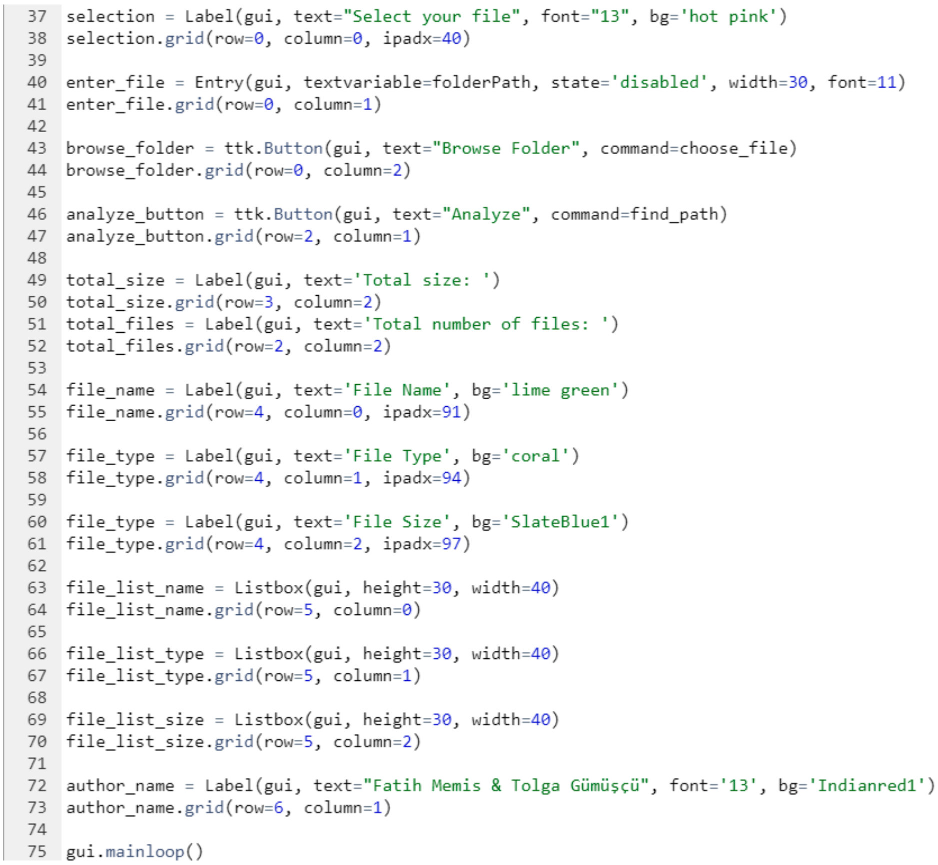
On our front-end, we first set our size and the size of our application.

Thanks to our **choose_file** and **find_path** functions, we can reach the source of the target file specified by the user.



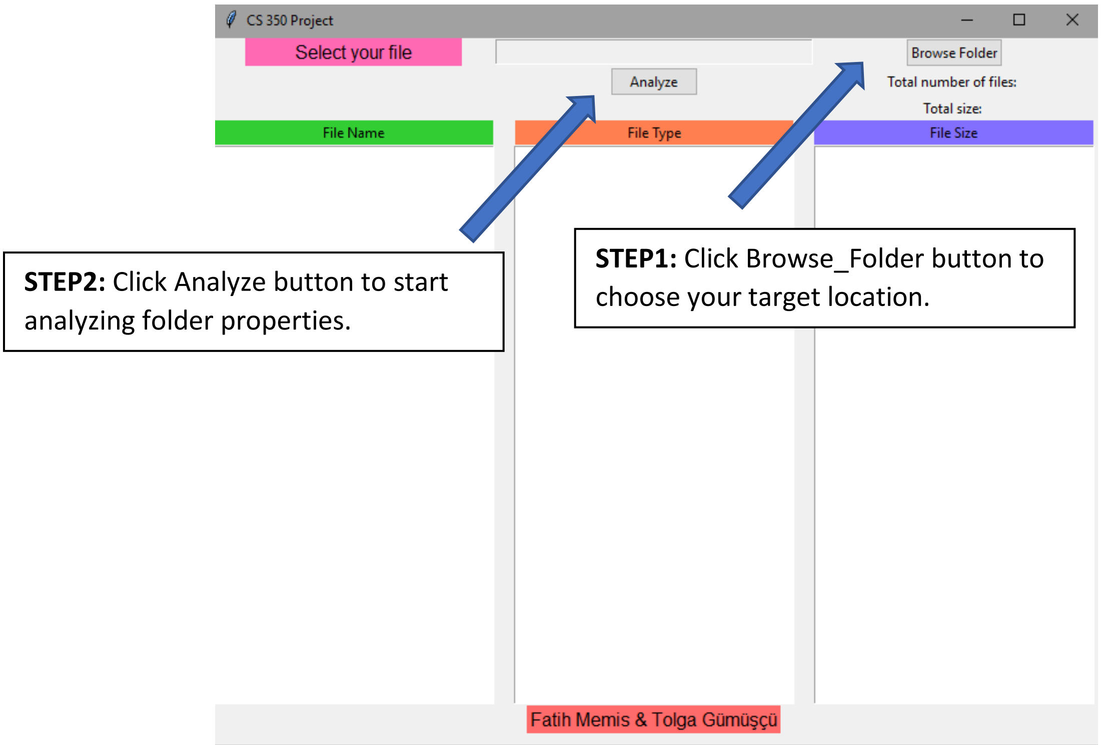
```
import backend

from tkinter import *
from tkinter import ttk
from tkinter import filedialog

gui = Tk()
gui.geometry("765x610")
gui.title("CS 350 Project")


def choose_file():
    selected = filedialog.askdirectory()
    folderPath.set(selected)


def find_path():
    folder_path = folderPath.get()
    number_files, number_dirs = backend.find_number(folder_path)
    file_list = backend.find_name(folder_path)
    type_list = backend.find_type(folder_path)
    a, size_list = backend.find_size(folder_path)
    file_list_type.delete(0, 'end')
    file_list_name.delete(0, 'end')
    file_list_size.delete(0, 'end')
    a = str(a / 1000)
    total_size['text'] = 'Total size: ' + a + 'KB'
    total_files['text'] = 'Total number of files: ' + str(number_files)
    for i in range(number_files):
        file_list_name.insert(i, file_list[i])
        file_list_type.insert(i, type_list[i])
        file_list_size.insert(i, str(size_list[i] / 1000) + ' KB')


folderPath = StringVar()
```

In the continuation of the **front-end** part, we define and create the visual components of our application. In this way, we create a program that the user can easily use.



```
selection = Label(gui, text="Select your file", font="13", bg='hot pink')
selection.grid(row=0, column=0, ipadx=40)

enter_file = Entry(gui, textvariable=folderPath, state='disabled', width=30, font=11)
enter_file.grid(row=0, column=1)

browse_folder = ttk.Button(gui, text="Browse Folder", command=choose_file)
browse_folder.grid(row=0, column=2)

analyze_button = ttk.Button(gui, text="Analyze", command=find_path)
analyze_button.grid(row=2, column=1)

total_size = Label(gui, text='Total size: ')
total_size.grid(row=3, column=2)
total_files = Label(gui, text='Total number of files: ')
total_files.grid(row=2, column=2)

file_name = Label(gui, text='File Name', bg='lime green')
file_name.grid(row=4, column=0, ipadx=91)

file_type = Label(gui, text='File Type', bg='coral')
file_type.grid(row=4, column=1, ipadx=94)

file_type = Label(gui, text='File Size', bg='SlateBlue1')
file_type.grid(row=4, column=2, ipadx=97)

file_list_name = Listbox(gui, height=30, width=40)
file_list_name.grid(row=5, column=0)

file_list_type = Listbox(gui, height=30, width=40)
file_list_type.grid(row=5, column=1)

file_list_size = Listbox(gui, height=30, width=40)
file_list_size.grid(row=5, column=2)

author_name = Label(gui, text="Fatih Memis & Tolga Gümüşçü", font='13', bg='Indianred1')
author_name.grid(row=6, column=1)

gui.mainloop()
```

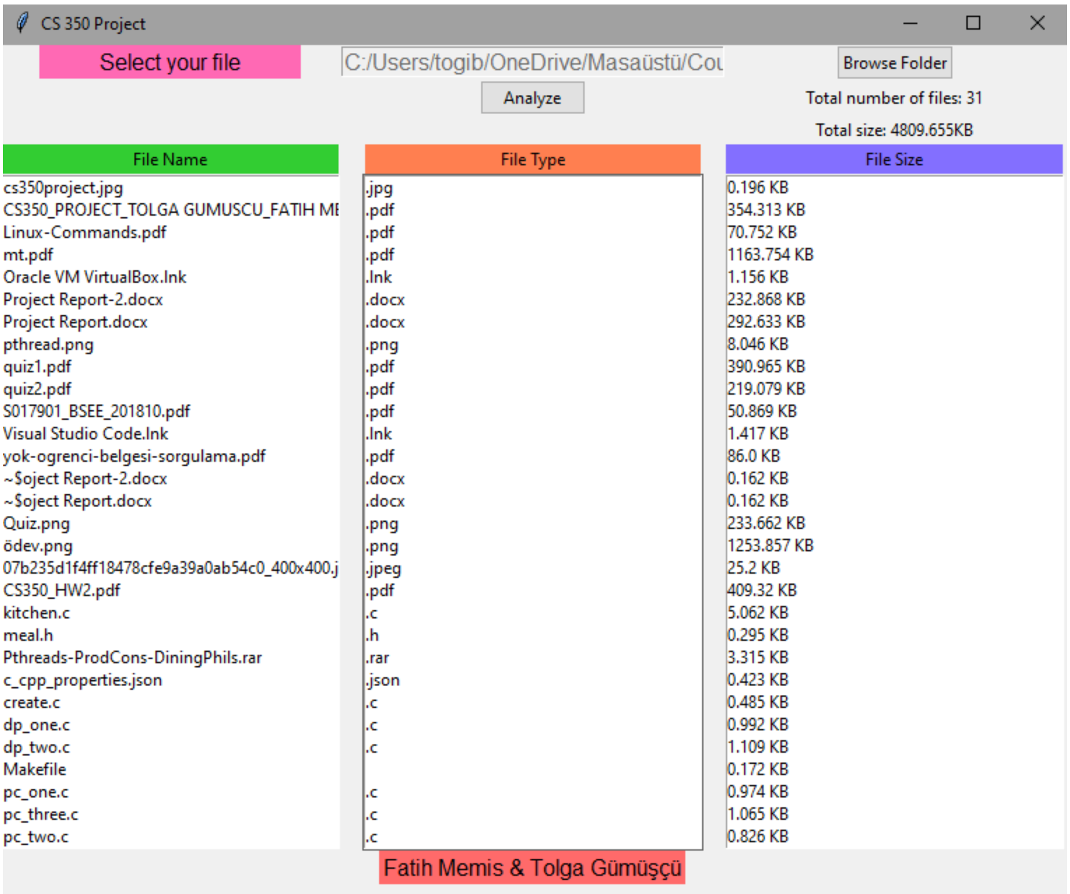
How to Execute



STEP1: Click Browse_Folder button to choose your target location.

STEP2: Click Analyze button to start analyzing folder properties.

Output:



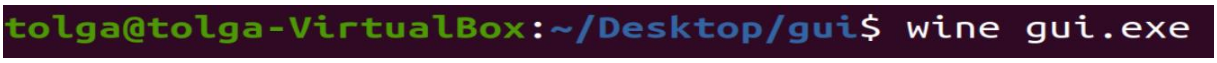
C:/Users/togib/OneDrive/Masaüstü/Cou

Total number of files: 31

Total size: 4809.655KB

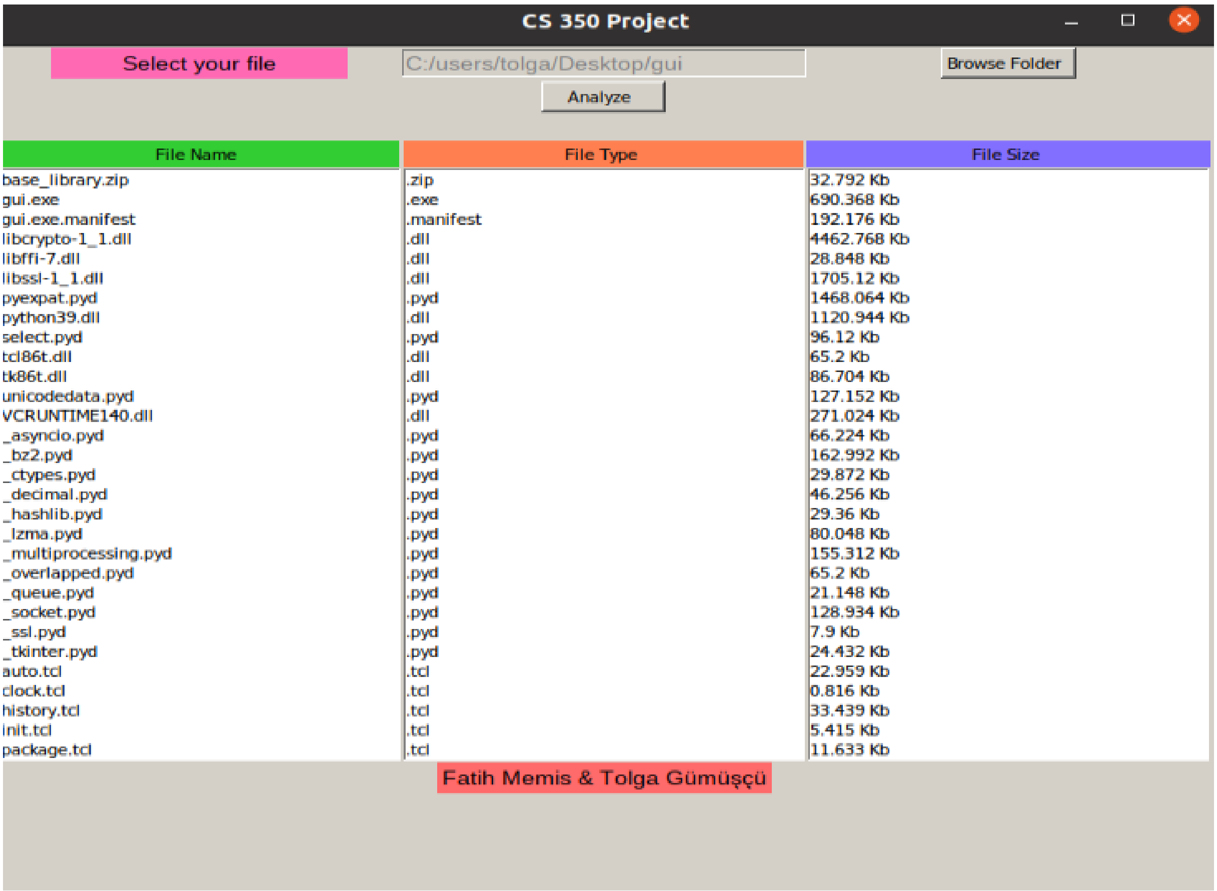
File Name	File Type	File Size
cs350project.jpg	.jpg	0.196 KB
CS350_PROJECT_TOLGA GUMUSCU_FATIH ME	.pdf	354.313 KB
Linux-Commands.pdf	.pdf	70.752 KB
mt.pdf	.pdf	1163.754 KB
Oracle VM VirtualBox.lnk	.lnk	1.156 KB
Project Report-2.docx	.docx	232.868 KB
Project Report.docx	.docx	292.633 KB
pthread.png	.png	8.046 KB
quiz1.pdf	.pdf	390.965 KB
quiz2.pdf	.pdf	219.079 KB
S017901_BSEE_201810.pdf	.pdf	50.869 KB
Visual Studio Code.lnk	.lnk	1.417 KB
yok-ogrenci-belgesi-sorgulama.pdf	.pdf	86.0 KB
~$oject Report-2.docx	.docx	0.162 KB
~$oject Report.docx	.docx	0.162 KB
Quiz.png	.png	233.662 KB
ödev.png	.png	1253.857 KB
07b235d1f4ff18478cfe9a39a0ab54c0_400x400.j	.jpeg	25.2 KB
CS350_HW2.pdf	.pdf	409.32 KB
kitchen.c	.c	5.062 KB
meal.h	.h	0.295 KB
Pthreads-ProdCons-DiningPhils.rar	.rar	3.315 KB
c_cpp_properties.json	.json	0.423 KB
create.c	.c	0.485 KB
dp_one.c	.c	0.992 KB
dp_two.c	.c	1.109 KB
Makefile		0.172 KB
pc_one.c	.c	0.974 KB
pc_three.c	.c	1.065 KB
pc_two.c	.c	0.826 KB

Ubuntu Execution:



```
tolga@tolga-VirtualBox:~/Desktop/gui$ wine gui.exe
```

We use the wine command to run it with .exe extension on Ubuntu. If we call the wine command in the file with the .exe extension file to the terminal as in the picture above, our program runs smoothly.



C:/users/tolga/Desktop/gui

File Name	File Type	File Size
base_library.zip	.zip	32.792 Kb
gui.exe	.exe	690.368 Kb
gui.exe.manifest	.manifest	192.176 Kb
libcrypto-1_1.dll	.dll	4462.768 Kb
libffi-7.dll	.dll	28.848 Kb
libssl-1_1.dll	.dll	1705.12 Kb
pyexpat.pyd	.pyd	1468.064 Kb
python39.dll	.dll	1120.944 Kb
select.pyd	.pyd	96.12 Kb
tcl86t.dll	.dll	65.2 Kb
tk86t.dll	.dll	86.704 Kb
unicodedata.pyd	.pyd	127.152 Kb
VCRUNTIME140.dll	.dll	271.024 Kb
_asyncio.pyd	.pyd	66.224 Kb
_bz2.pyd	.pyd	162.992 Kb
_ctypes.pyd	.pyd	29.872 Kb
_decimal.pyd	.pyd	46.256 Kb
_hashlib.pyd	.pyd	29.36 Kb
_lzma.pyd	.pyd	80.048 Kb
_multiprocessing.pyd	.pyd	155.312 Kb
_overlapped.pyd	.pyd	65.2 Kb
_queue.pyd	.pyd	21.148 Kb
_socket.pyd	.pyd	128.934 Kb
_ssl.pyd	.pyd	7.9 Kb
_tkinter.pyd	.pyd	24.432 Kb
auto.tcl	.tcl	22.959 Kb
clock.tcl	.tcl	0.816 Kb
history.tcl	.tcl	33.439 Kb
init.tcl	.tcl	5.415 Kb
package.tcl	.tcl	11.633 Kb

Fatih Memis & Tolga Gümüşçü

Executable File Location:

\Analyzer\app\frontend

Our .exe extension file is located in the app file in the Analyzer folder.

References:

https://www.geeksforgeeks.org/python-gui-tkinter/

https://docs.python.org/3/library/os.html

https://www.geeksforgeeks.org/os-walk-python/