**A**

**MINI PROJECT REPORT**

**on**

**FILE ORGANIZER**

**BE(IT)-III Sem**

**By**

**Rehan Ali Farooqui**

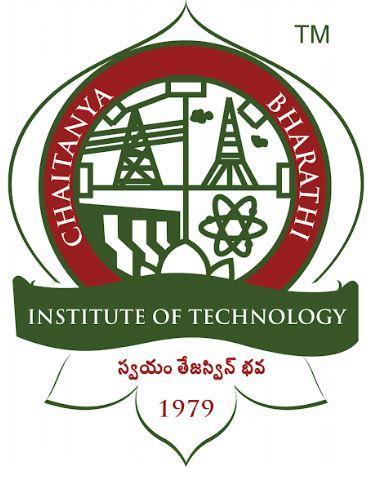
**(160120737043)**

**Under the guidance of**

**Dr. Trupthi Mukund**

**Assistant Professor**

**IT Department**



**DEPARTMENT OF INFORMATION TECHNOLOGY   
CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)**

**(Affiliated to Osmania University; Accredited by NBA(AICTE) and NAAC(UGC), ISO Certified 9001:2015)**

**KOKAPET(V),GANDIPET(M),RR District HYDERABAD - 75**

**Website:** [**www.cbit.ac.in**](http://www.cbit.ac.in/)

**2021-2022**

****

This is to certify that the project work entitled “**FILE ORGANIZER**” submitted to CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY, in partial fulfillment of the requirements for the completion of Mini Project-I of  III Semester B.E. in Information Technology, during the Academic Year 2021-2022, is a record of original work done by **REHAN ALI FAROOQUI (160120737043)** during the period of study in the Department of IT, CBIT, HYDERABAD, under our guidance.

| **Project Guide** | **Head of the Department** |
| --- | --- |
| **Dr. Trupthi Mukund** | **Dr.K.Radhika** |
| Assistant Professor, Dept. of IT, | Professor, Dept. of IT, |
| CBIT, Hyderabad. | CBIT, Hyderabad. |

## ACKNOWLEDGEMENTS

We would like to express our heartfelt gratitude to **Dr. Trupthi Mukund**, our project guide, for her invaluable guidance and constant support, along with her capable instruction and persistent encouragement.

We are grateful to our Head of Department, **Dr.K.Radhika**, for her steady support and for the provision of every resource required for the completion of this project.

We would like to take this opportunity to thank our Principal, **Dr.P.Ravinder Reddy**, as well as the Management of the Institute, for having designed an excellent learning atmosphere.

Our thanks are due to all members of the staff and our lab assistants for providing us with the help required to carry out the groundwork of this project.

**ABSTRACT**

Files are something that we use in our daily life whether it maybe on our mobile or on desktop, and during these days when many of us are still in the Work from home zone the amount of files that we use almost doubles as all the paperwork is now available in the form of a document file and for storing records and data we need ***Files.***

It becomes necessary for us to organize the files so that the information could be retrieved whenever required. The existing system usually provides users with options to locate, delete, create ,open files on the basis of the user requirement and some of them provide sorting files based on the extensions.

The system to be built can sort your files by the file name initials such as let’s say you’re a student and you have downloaded your notes and there are around 12 documents for each subject. The system intends to sort them and store them in a new folder with the subject name provided that the documents downloaded are initialized by their subject name (math for mathematics). This might seem very useful for a person who keeps most of the files and folders at one location and sometimes is confused about what all files are there. With this project, the system to be built will be useful for anyone regardless of profession; it would be a useful project for all the community.

In order to build such a system the programming language which is most suitable is **python** as we can use modules like **shutil, glob** and **OS (Operating System)** which can help us in performing operations on files.

**CONTENTS**

| **S.no** | **Topics** | **Page. No** |
| --- | --- | --- |
|  | **List of Figures** |  |
|  | **List of Tables** |  |
|  | **Abstract** |  |
| 1 | **Introduction** |  |
| * 1. Motivation | 8 |
| * 1. Objective of the Project | 9 |
| 1.3 Problem Statement | 9 |
| 2 | **Existing System** | 10 |
| 2.1 Literature survey |  |
| 3 | **Proposed Methodology** |  |
| 3.1 System Specifications | 11 |
| 3.2 System Design | 12 |
| 3.3 Proposed Work | 12 |
| 4 | **Implementation and Results** | 13 |
| 5 | **Conclusion and Future Scope** | 21 |
|  | **Bibliography** | 22 |

## 

**List of Figures**

| **Figure No** | **Name of the Figure** | **Page No** |
| --- | --- | --- |
| 4.1 | Unorganized folder with random files | 13 |
| 4.2 | Implementation for accessing file names | 14 |
| 4.3 | Displaying files of directory | 14 |
| 4.4 | Implementation to segregate files | 15 |
| 4.5 | Manipulations made in folder | 16 |
| 4.6 | Directory after running the system | 16 |
| 4.7 | Implementation of existing system | 18 |
| 4.8 | Final demonstration-1 | 19 |
| 4.9 | Final demonstration-2 | 19 |
| 4.10 | Final demonstration-3 | 20 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**List of Tables**

| **Table No** | **Name of the Table** | **Page No** |
| --- | --- | --- |
| 1. | Literature Survey | 07 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**1. INTRODUCTION**

**1.1 Motivation**

As you use your computer to create and download files, it’s easy to become buried in a sea of tiny icons with vague names. Just as it is important to organize papers so that you can find them later, it is important to organize your computer files by creating folders and putting files inside of them. If people can find files faster, they can accomplish more work than if they had to spend time trying to locate a file.

In the pandemic I personally noticed my **downloads** directory where I sometimes had 30-40 files with different extensions of various subjects. The only thing that could help me a little was just the Windows File Manager sort by option where I could access the newly downloaded files much easier.

Then I realized that the amount of work done by employees with all file manipulations is unscalable. In order to make it a bit easier for the users we must have a system which could segregate our files and fulfill our need to access the desired file within a few seconds. It shouldn’t be us who will search the file consecutively but any external system who could do it for us.

The File Organizer which I was thinking about was limited to the extensions and placing a new folder, but with my mentor’s guidance I got inclined to work with the file names and make it more precise to the user.

**1.2 Objective**

The main objective of this project is to segregate the files based on the file names or the file extension (chosen by the user) and place them into a newly created folder with the name provided by the user or the extension of the file.

The system should identify the file name by traversing through all the files and do the mapping , if the user input and the corresponding response is found then it must perform the necessary operations.

The redundant folder should not be created and if in such a case it must go with the existing folder and use it to transfer files or the required operations.

The tons of downloads of the user should turn into a directory consisting of minimal folders with relevant files placed inside it.

**1.2 Problem Statement**

Generally, my desktop is full of files. As a student, I’ve many files to download each day and my download file contains hundreds of documents of different types (having different extensions line jpg, doc, py , pdf, html ,etc). We can implement a python code to get our task done.

We can organize the files by clubbing all files of the same type in each separate folder by giving the directory where we want to store the files. This helps us when we are in search of a particular file.

**2. EXISTING SYSTEM**

**2.1 Literature Survey**

Table 2.1

| **S No.** | **Title** | **Methodology** | **Authors** | **Limitations** | **Year** |
| --- | --- | --- | --- | --- | --- |
| 1. | Conceptual File Management: Revising the structure of classification-based information retrieval | The proposed Conceptual File Management (CFM) method allows a file to be accessed from multiple folders; furthermore, it keeps the current hierarchical structure with minor changes. CFM reduces the ambiguity and redundancy, therefore the quality of information retrieval is improved. | Ali Sajedi , Morteza Ashourzadeh , and Mehregan Mahdavi. | It doesn’t allow us to arrange the files according to any format, it just provides us to access files and retrieve data. | 2016 |
| 2. | The ubiquitous digital file | There’s an option to organize the files into sub-folders.  Once I’ve picked how I want my files to be sorted and renamed, I go ahead and click apply. Folders get created, files get renamed and I’m done: | Richardo Vidal | Here the files can be organized only on the basis of type, size, and date. | 2019 |

**3. PROPOSED METHODOLOGY**

**3.1 System Specifications**

* Modern Operating System:
  + Windows 7 or above
  + Mac OS X 10.11 or higher, 64-bit
  + Linux: RHEL 6/7, 64-bit (almost all libraries also work in Ubuntu)
* x86 64-bit CPU (Intel / AMD architecture)
* 4 GB RAM
* 5 GB free disk space
* 1 gigahertz (GHz) or faster 32-bit (x86) or 64-bit (x64) processor
* 16 GB available hard disk space (32-bit) or 20 GB (64-bit)
* Any python IDLE installed of 8.2 version or above

**3.2 System Design**

The modules, components and their interfaces and data for a system are mentioned as below:

The system built consists of modules which includes:

i.**Tkinter library** :For rendering graphics on a display window

ii.**OS library** :The OS module in Python provides functions for interacting with the operating system.

iii.**Shutil library** : Shutil module in Python provides many functions of high-level operations on files and collections of files.

iv.**Glob library** :The glob module finds all the pathnames matching a specified pattern according to the rules used by the Unix shell, although results are returned in arbitrary order.

The interface provided to the end user using tkinter has an entry button to specify the file name or the substring of it. And two buttons to select the desired options of which one of the buttons named **File Name** will invoke the corresponding command functions to perform the necessary operations based on the entry by the user. And the other button named **File Extension** will do the sorting based on extensions, it doesn’t require any input from the user side.

**3.2 Proposed Work**

The system to be built can sort your files by the file name initials such as let’s say you’re a student and you have downloaded your notes and there are around 12 documents for each subject. The system intends to sort them and store them in a new folder with the subject name provided that the documents downloaded are initialized by their subject name (math for mathematics). This might seem very useful for a person who keeps most of the files and folders at one location and sometimes is confused about what all files are there.

**4. IMPLEMENTATION AND RESULTS**

This is an example folder consisting of some notes in the form of pdf and few images and our python file containing the code.

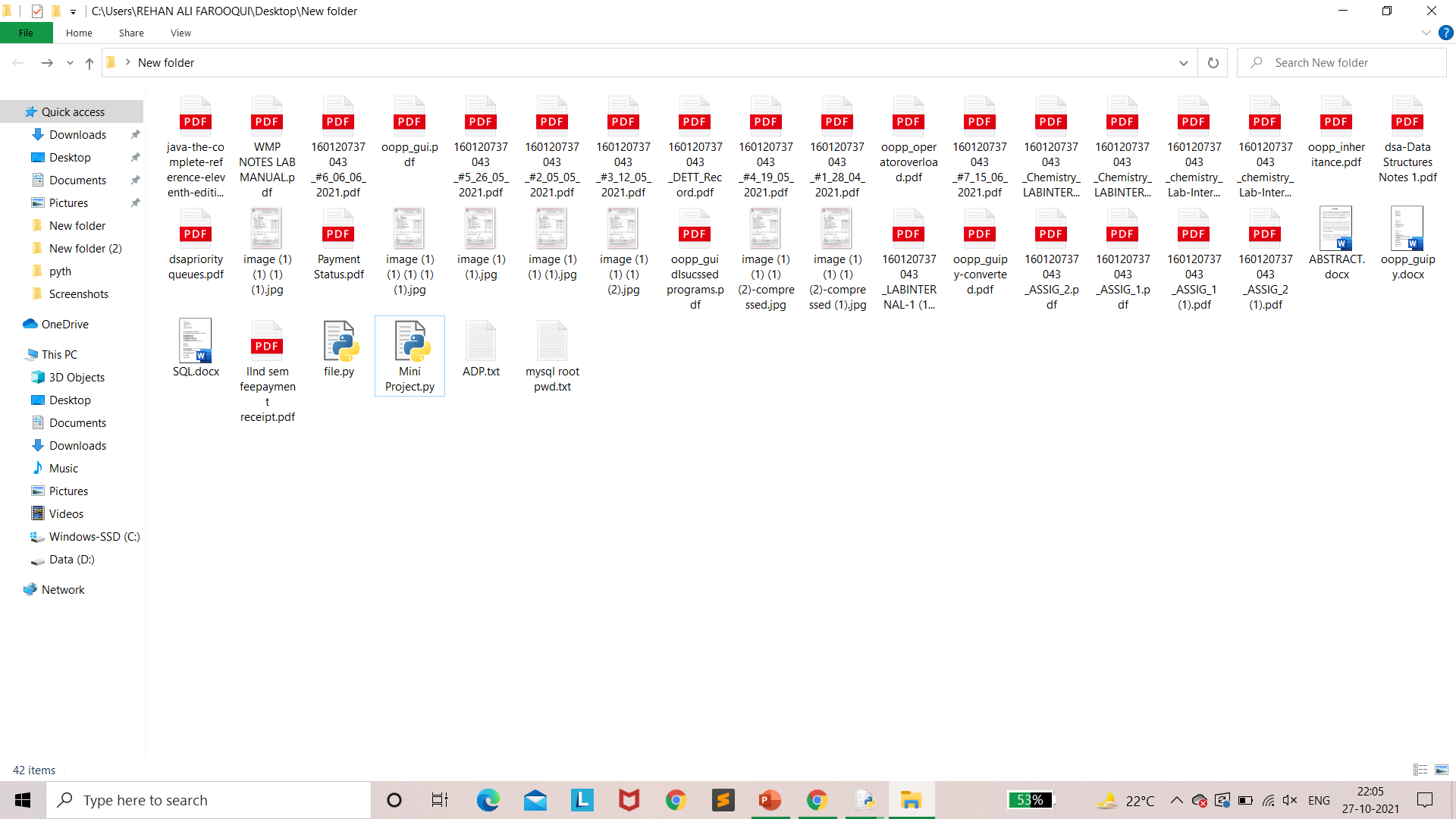
****

Fig 4.1 Unorganized folder with random files

By using the OS(Operating system), shutil, and glob module we can work with the file directories and move the file in a new folder. For the above case we need to create folders named **160120737043** and **OOPP** to club all the files having the folder name as its initials.

The implementation of the system done so far is that it is now able to access all the files in the particular directory and number of files in it using the OS module.Using list slicing it can also access a number of files of a particular initial file name provided as below. And further it is also capable of accessing the complete file names based on the initial name provided in the parentheses of glob.glob(“file\_initial\*”) .

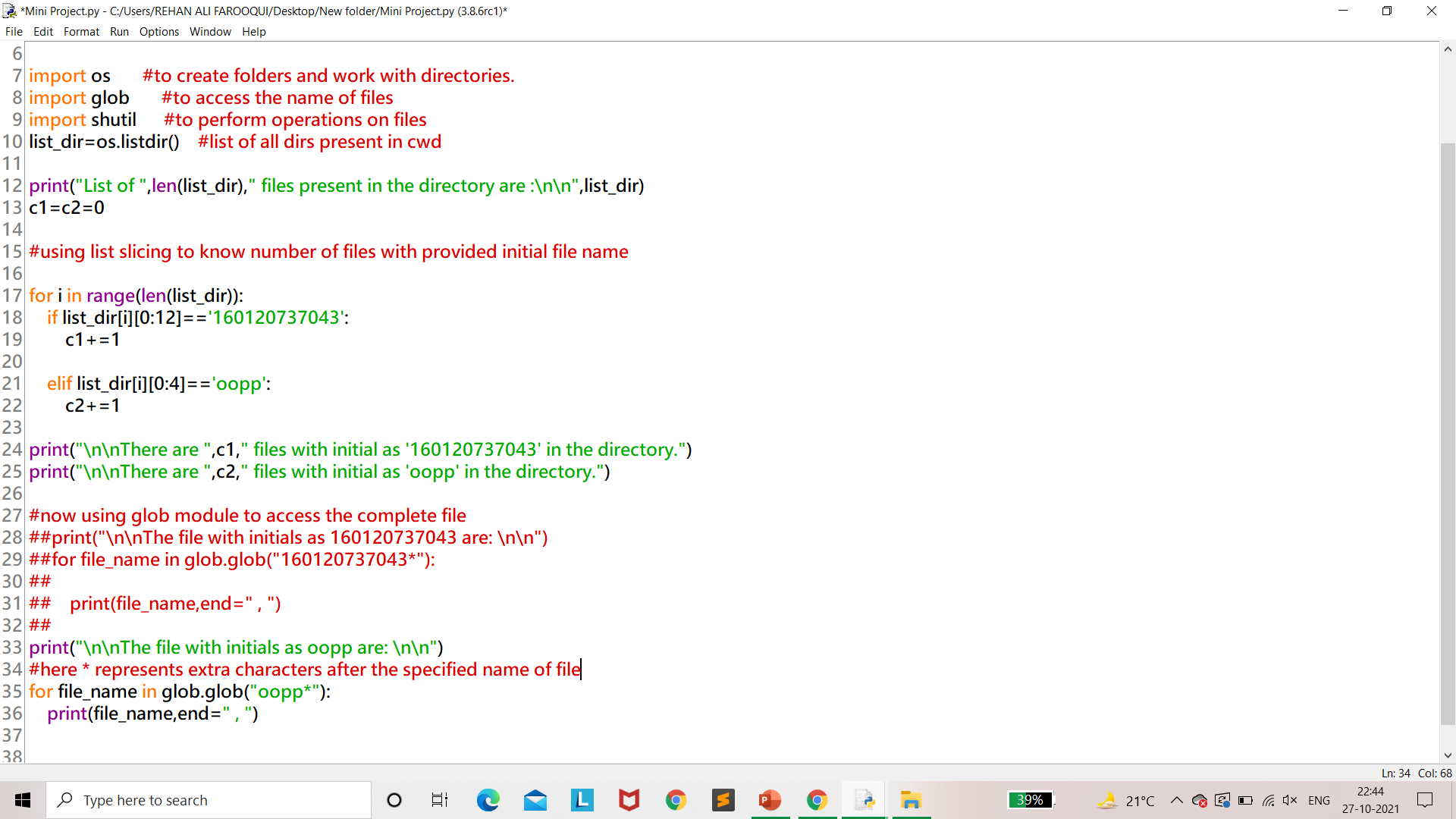


Fig 4.2 Implementation for accessing file names

Output

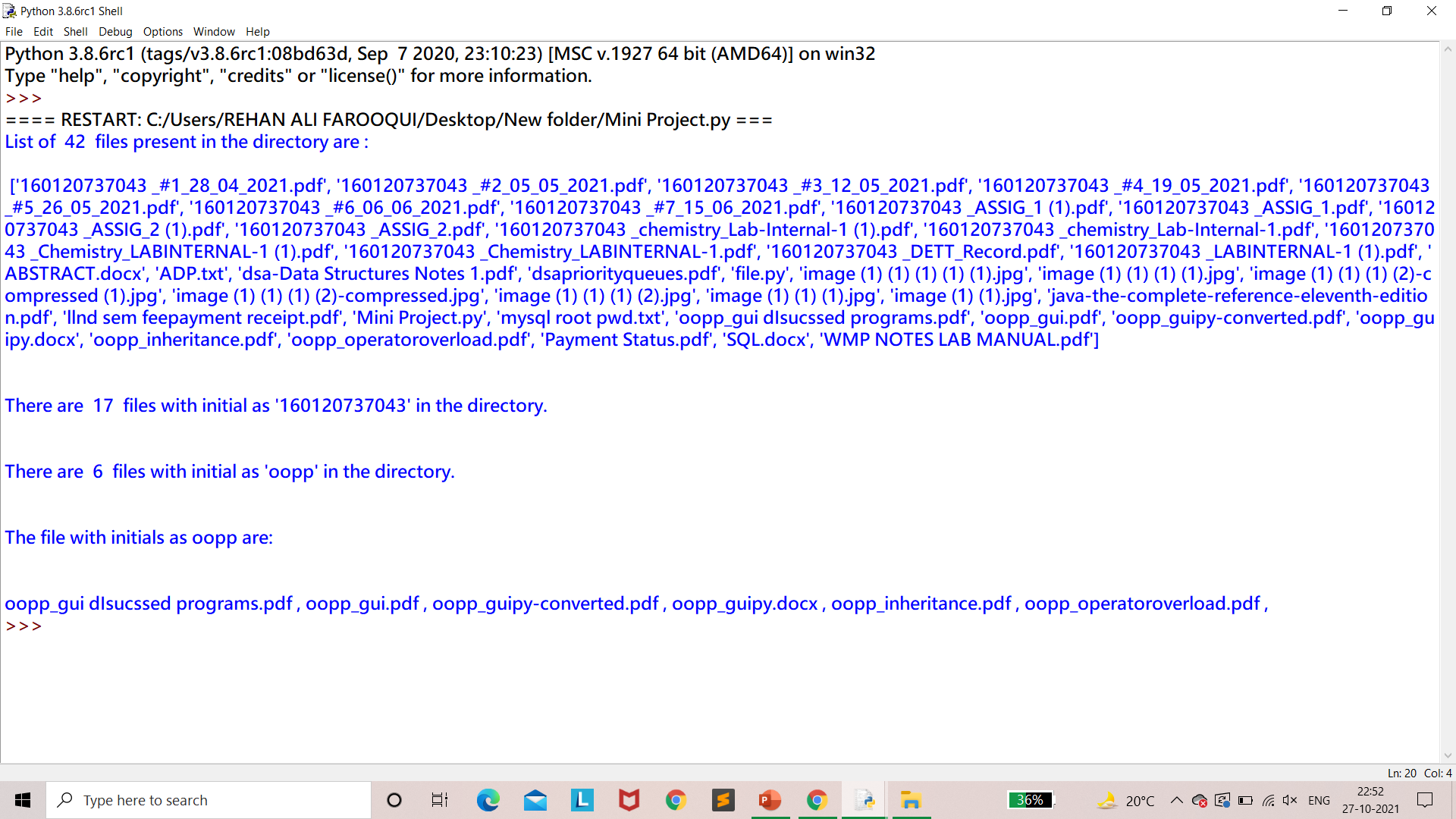
****

Fig 4.3: Displaying files of directory

Now that the system is able to segregate the files based on the file names, the next task is to create a folder and club those files in it , in the same directory. (eg : The files of oopp will be placed in a folder named OOPP). Similarly, the other files sharing the same initial names will be placed in the new folder.

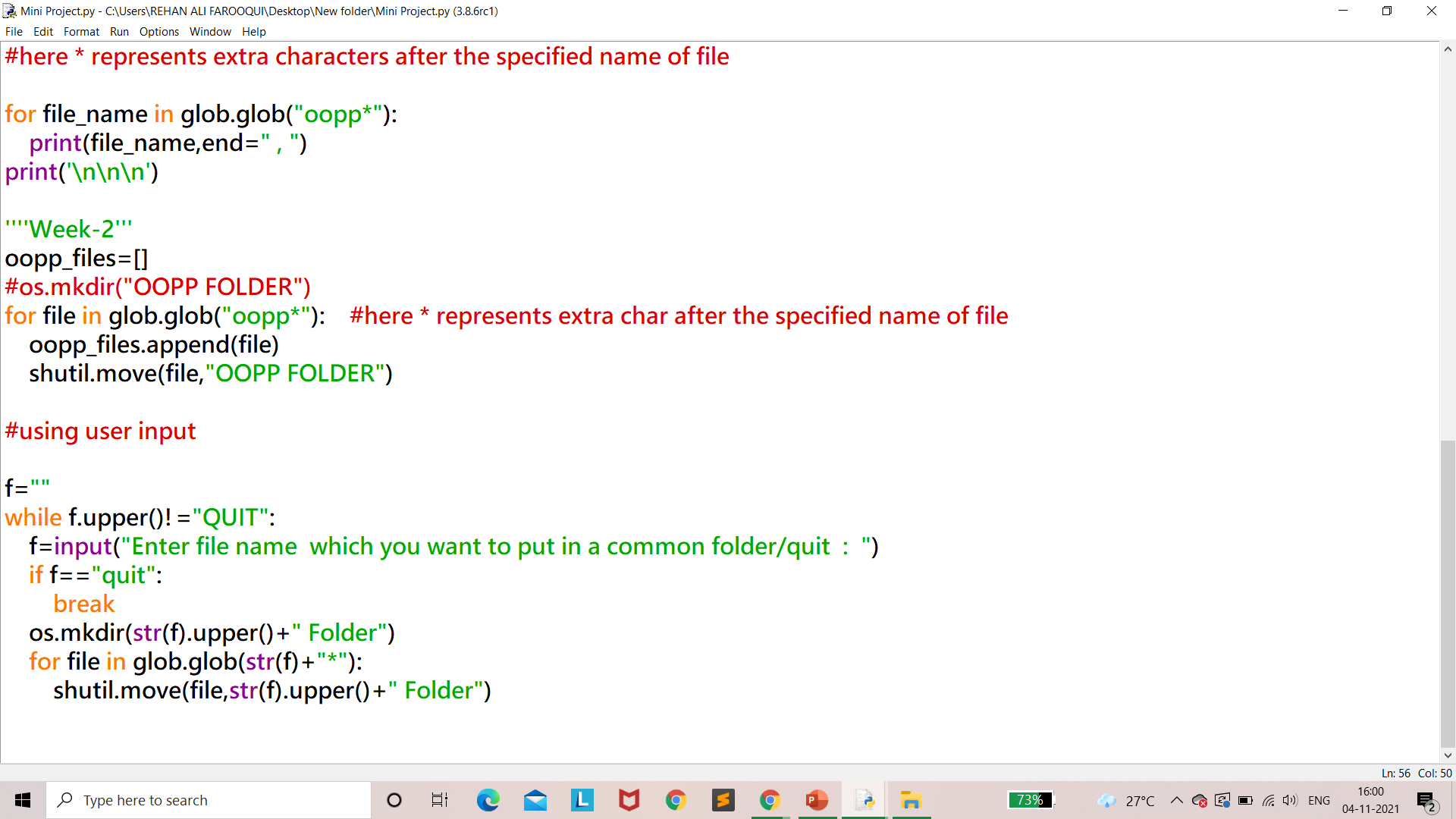


Fig 4.4: Implementation to segregate files

As we can see that the oopp directory was created by the programmer by specifying the folder name, but in the next part the system provides an option to the user to give an input file name which he wants to segregate from other files in the current working directory by placing them in a new folder.

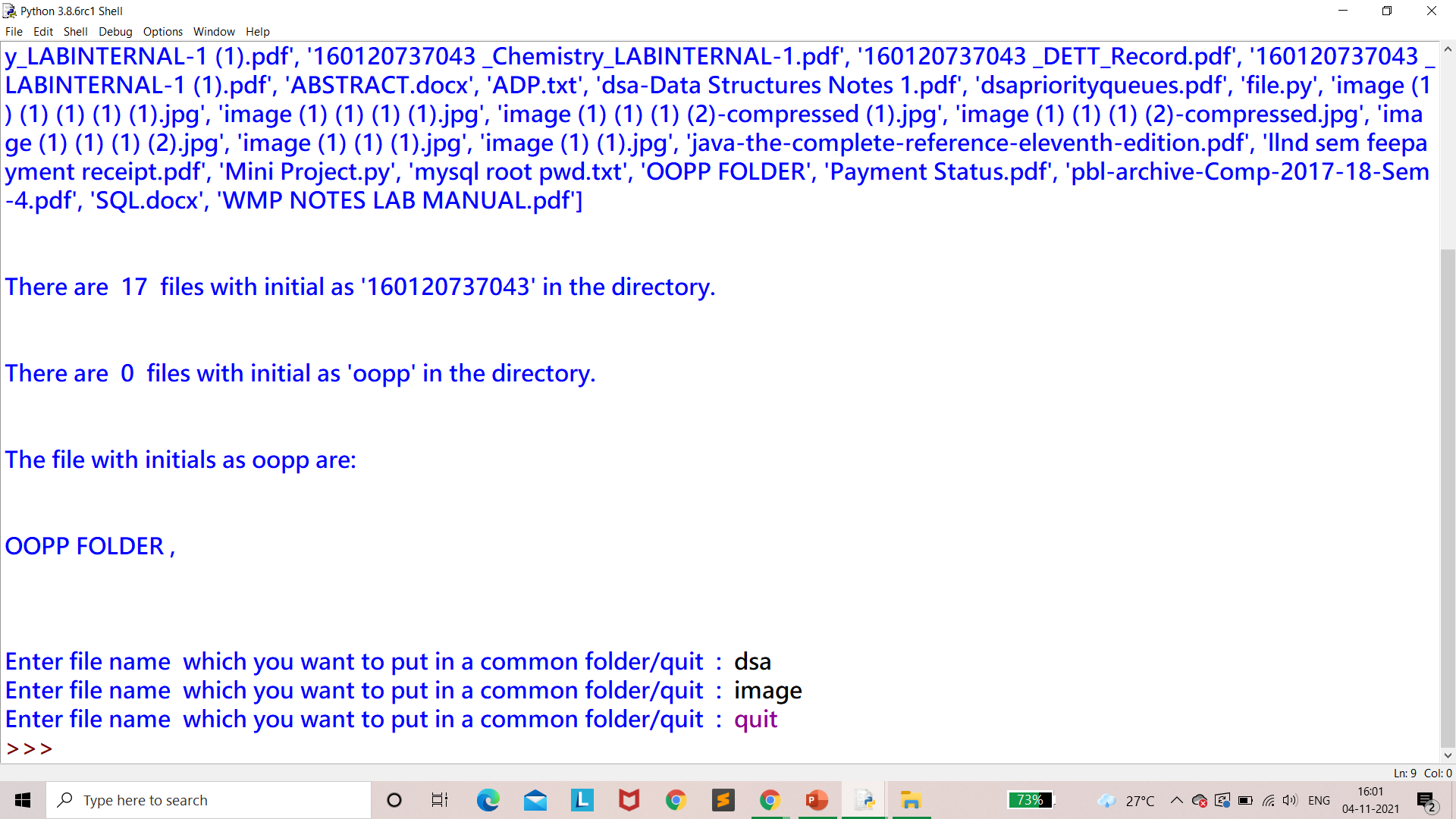
****

Fig 4.5: Corresponding Output

The user needs to enter quit if he wants to stop giving the inputs. In the above image we can see that the user has provided **dsa** and **image** as the file initials where he wants to club all files named dsa and image in the new folder. This is how our folder looks after executing the program.

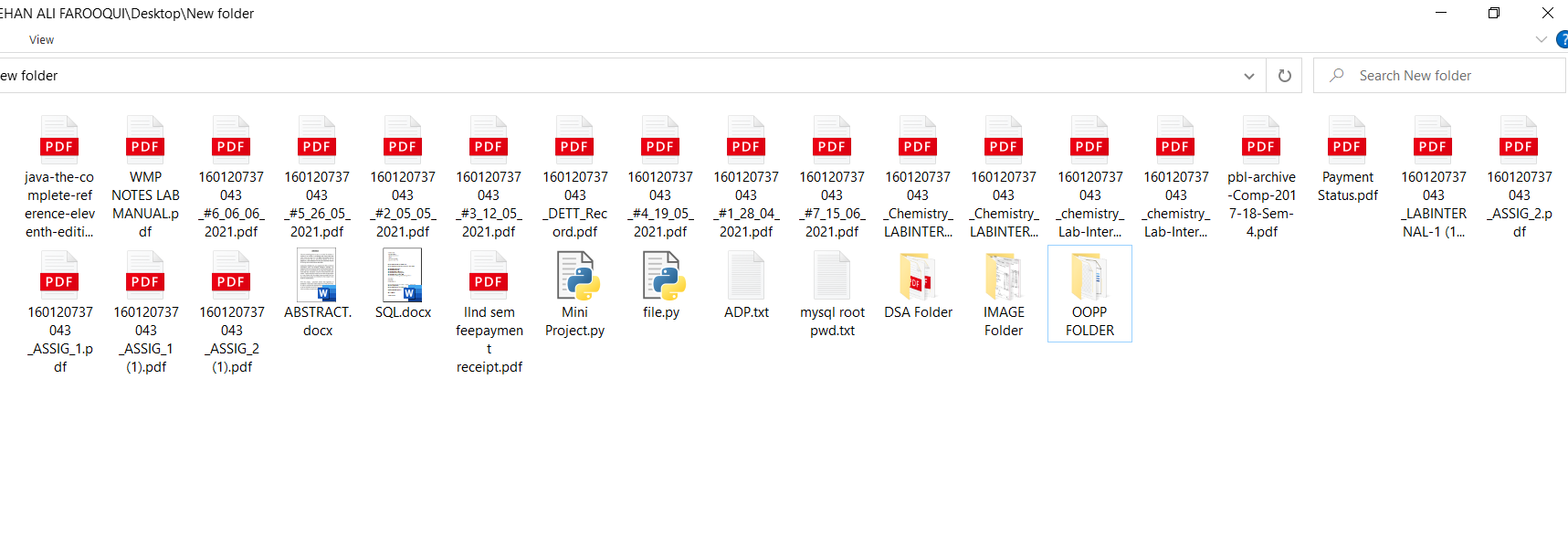


Fig 4.6:Directory after running the system

As of now, the system is able to access the file\_name (initial only) and segregate those files and put them in a newly created folder.

The next task to be implemented is that it must check if the folder already exists in the directory, if yes then it must not create the folder as it will return an error specifying that

**FileExistsError**: [WinError 183] Cannot create a file when that file already exists. So in order to avoid the following error we must only club the resulting files into the already existing folder.

And another thing is that random access of file\_name, in cases where the user wants to specify the few characters which he remembers, the system is capable of handling such a situation.

**Example:** file\_name is Image\_miniproject, miniproject.ppt, Report\_miniproject, miniproject\_code.

The random access can club all the above files in the folder **miniproject.**

Now, we need to enable the system to do the sorting on the files with respect to the file extension (eg-mini\_proj.**py** here py is the file extension).

Creating two different methods and placing them in the code , now the user will have the option to select either of the file sorting mechanisms or he/she can select both the mechanisms one after the other.

In order to make our system standout we also included the existing system. Below is the implementation for it.

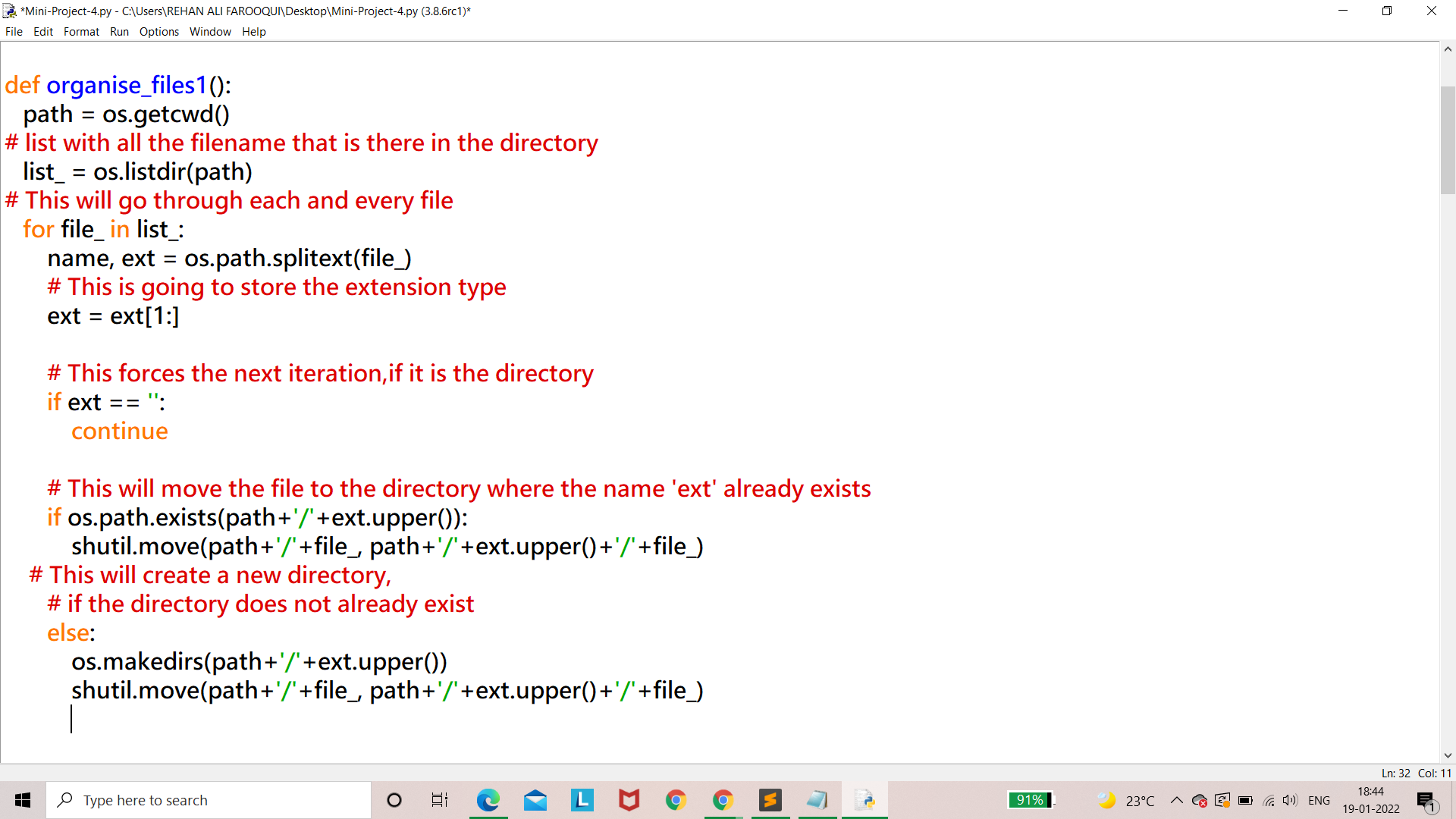


Fig 4.7:Implementation of existing system

Below is the output that the system gives which includes both, the existing system and the proposed system. In order to demonstrate it much better I have included the folder window too where in the operations will make a change in the folder by organizing the folder more efficiently. You can observe clearly that there are certain files with the same substring in their file names such as level, resource, abstract.

The system doesn’t check for the *case (*uppercase or lowercase*)* of the file name rather it checks the characters. The code file named Mini Project Code.py exists in the folder.

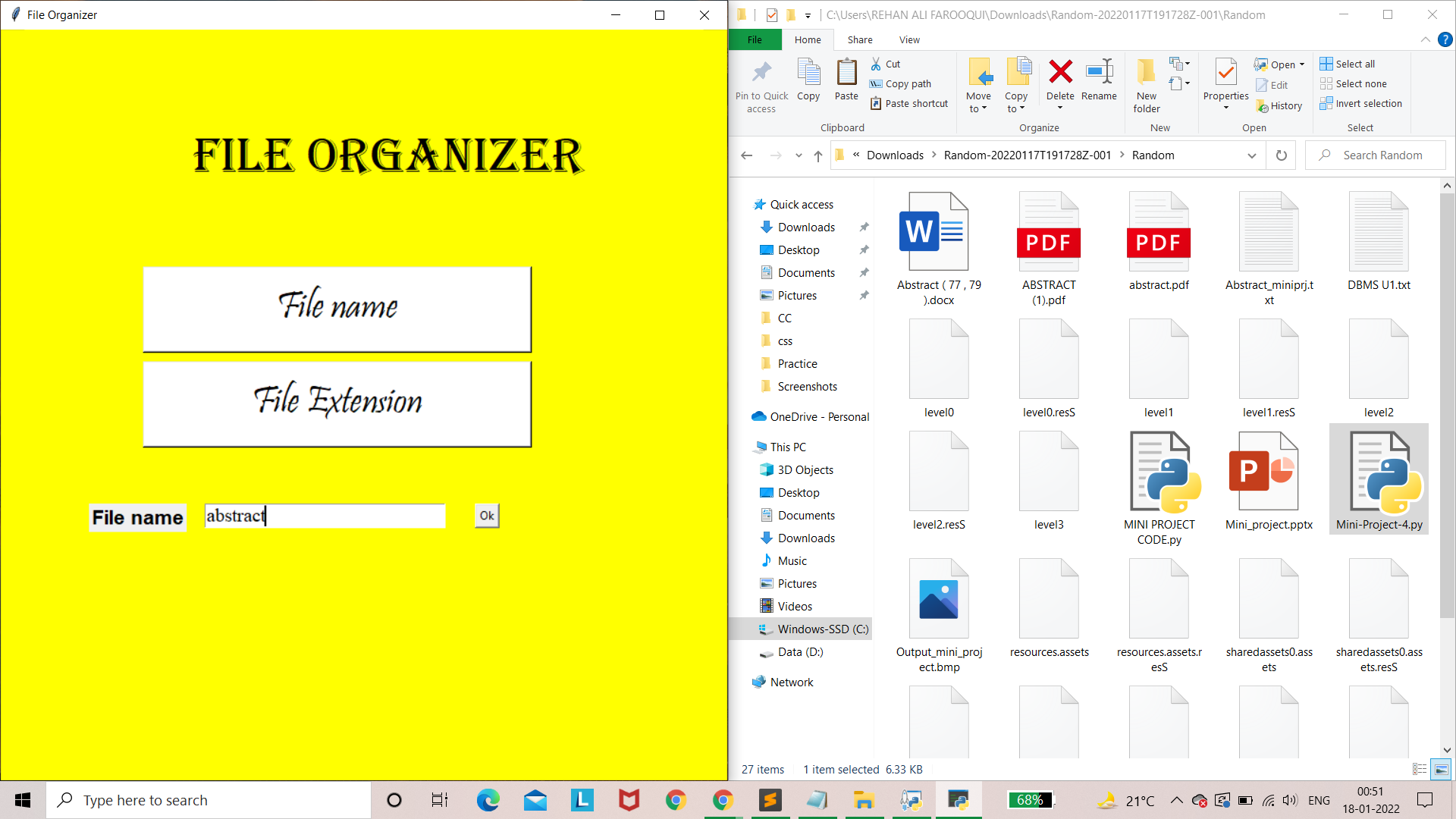


Fig 4.8: Final demonstration-1

As the input provided by the system is **abstract** the system does the organizing part. Similarly, if we provide various inputs (in this case *level,assets,project,*) and then the leftover files can be sorted based on their extension (.*txt, .ress*).

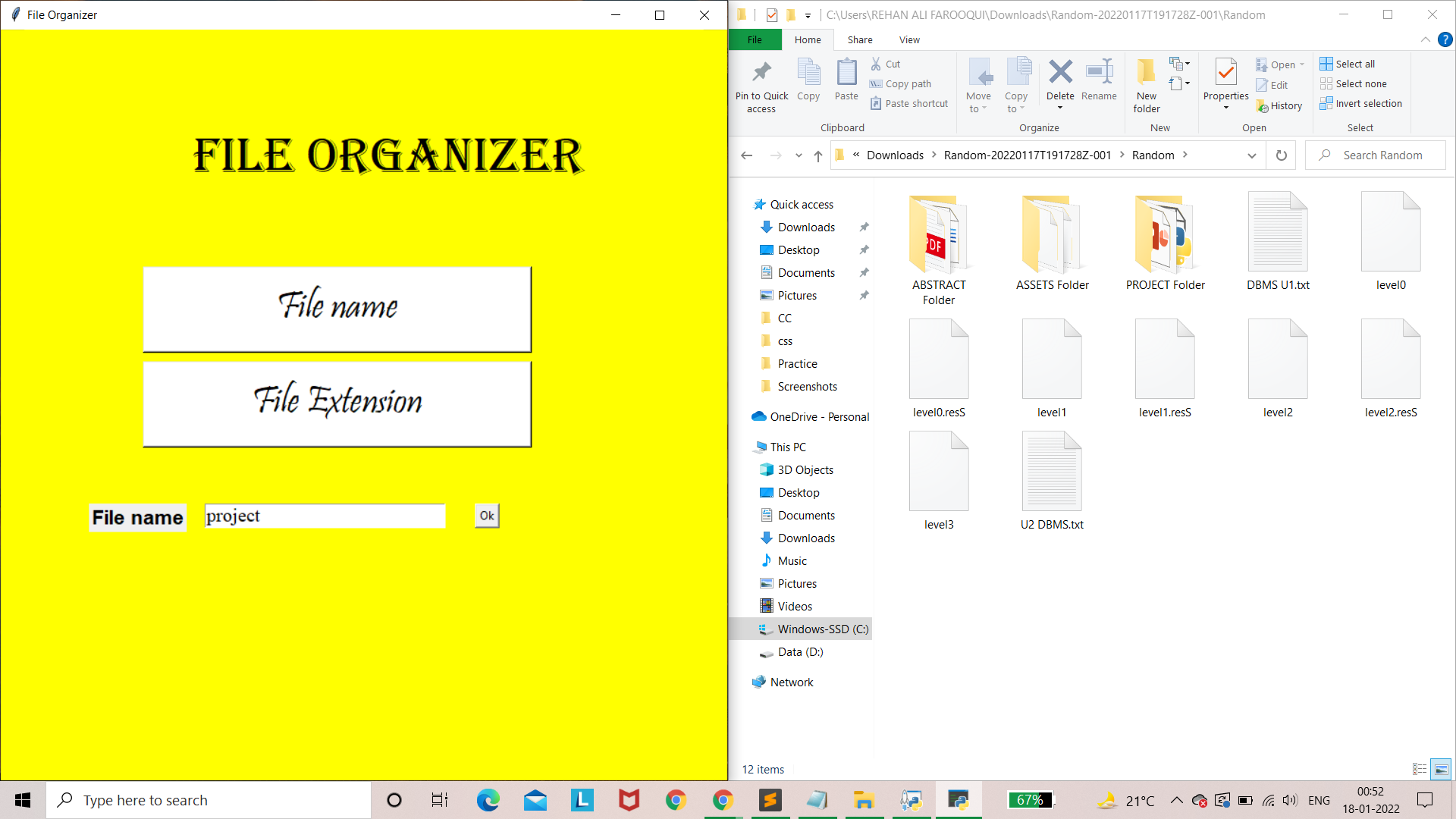


Fig 4.9: Final demonstration-2

After the user clicks the File extension option the final view of the folder organized by the system will be:

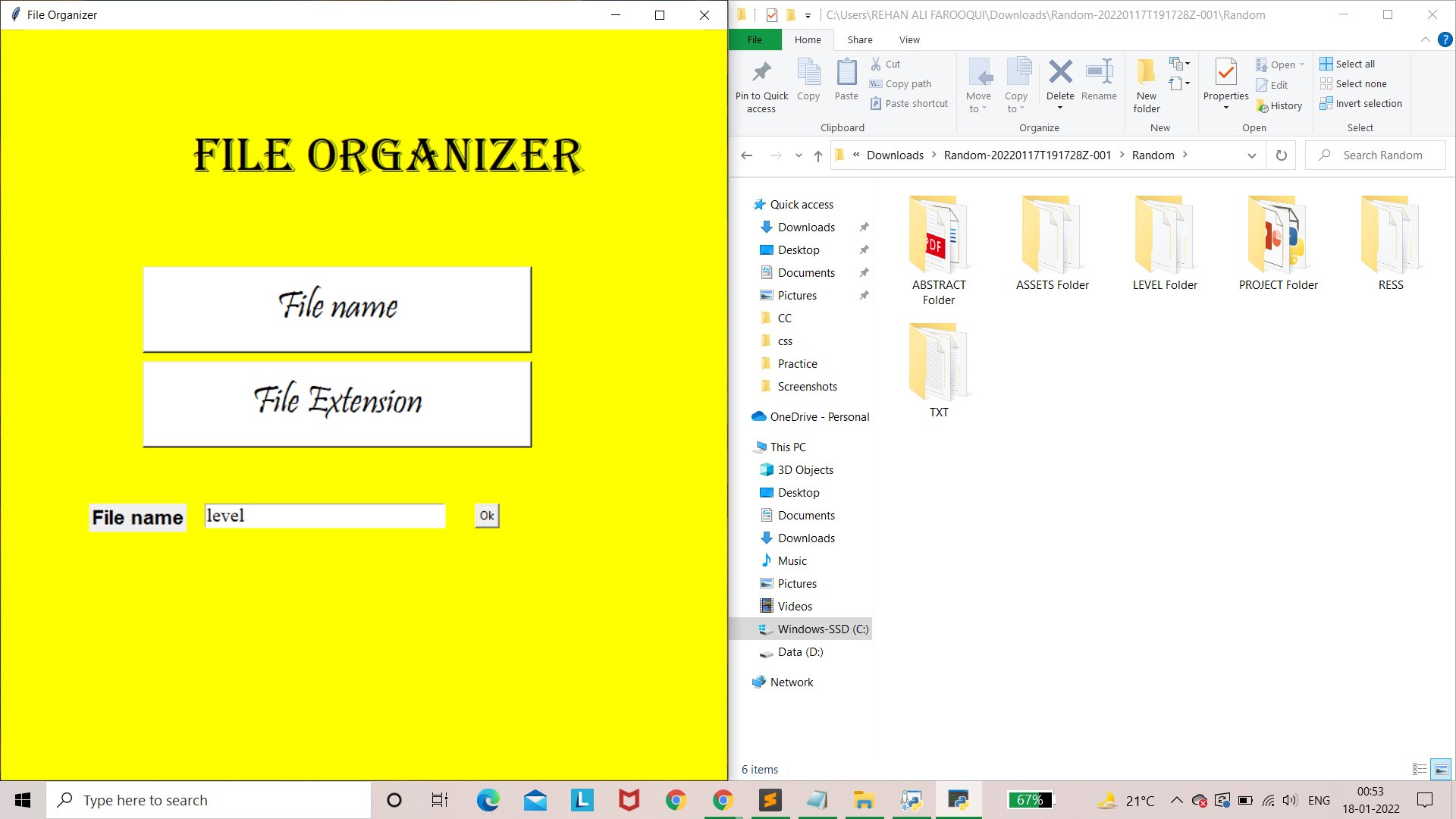


Fig 4.10: Final demonstration-3

**5. CONCLUSION AND FUTURE SCOPE**

The created file organizer system now enables the user to deal with the files and folders inside a directory. With this the user can organize the files based on his preference, the additional feature to sort the files based on file type makes it more optimized as it also includes the existing system. It is an easy to run system with general system requirements. We have achieved to club similar named files even with some common characters into a folder so that we can easily access the files when required.

The extension to the system built can be adding a database in order to keep all the file details in the database, by doing so we can know when the file was being manipulated.

The most important thing is we can take the backup of all the files in our google drive or some service cloud, this seems useful if we had any useful file which we aren’t able to restore from Recycle Bin as we prefer removing anything that has no further use.

**BIBLIOGRAPHY**

* <https://data-flair.training/blogs/python-file-explorer-project/>
* <https://www.techbeamers.com/python-shutil-module/>
* <https://www.javatpoint.com/python-os-module>
* <https://blog.devgenius.io/python-for-beginners-how-to-write-a-simple-file-organizer-code-fd6a12eb4b3d>