

# Encryption Decryption Algorithm Project Guide

By Justin Lam  
CWID: 886399013  
justinlam0624@csu.fullerton.edu

## Getting Started:

### 1. Download and Extract the File

- Download the zipped file that contains all the Sample tests and .exe
- Extract it to the folder that you desire
  - Right click the folder “**Encryption Decryption Project App with UI.zip**”
  - Select “**Extract All**”
  - Once in this screen:

Extract Compressed (Zipped) Folders

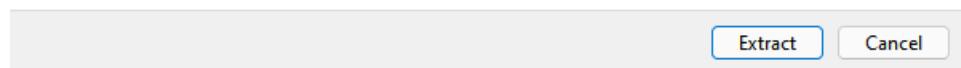
#### Select a Destination and Extract Files

Files will be extracted to this folder:

C:\Users\justi\Downloads\Encryption Decryption Project App with UI

Browse...

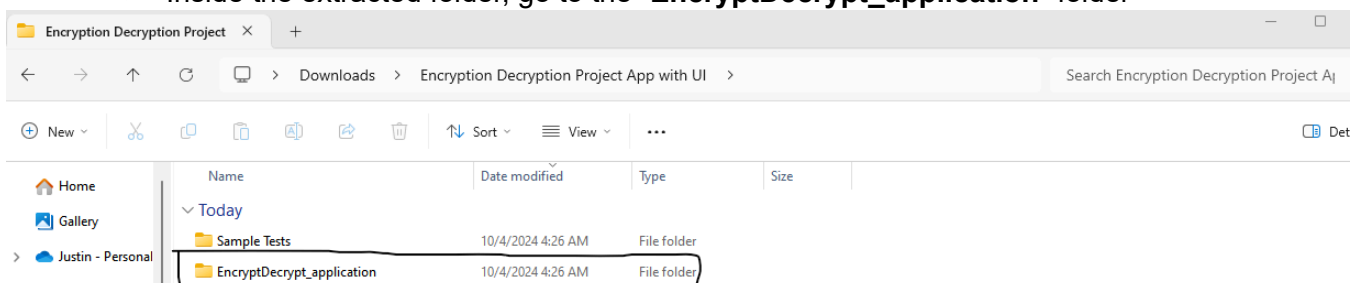
☒ Show extracted files when complete



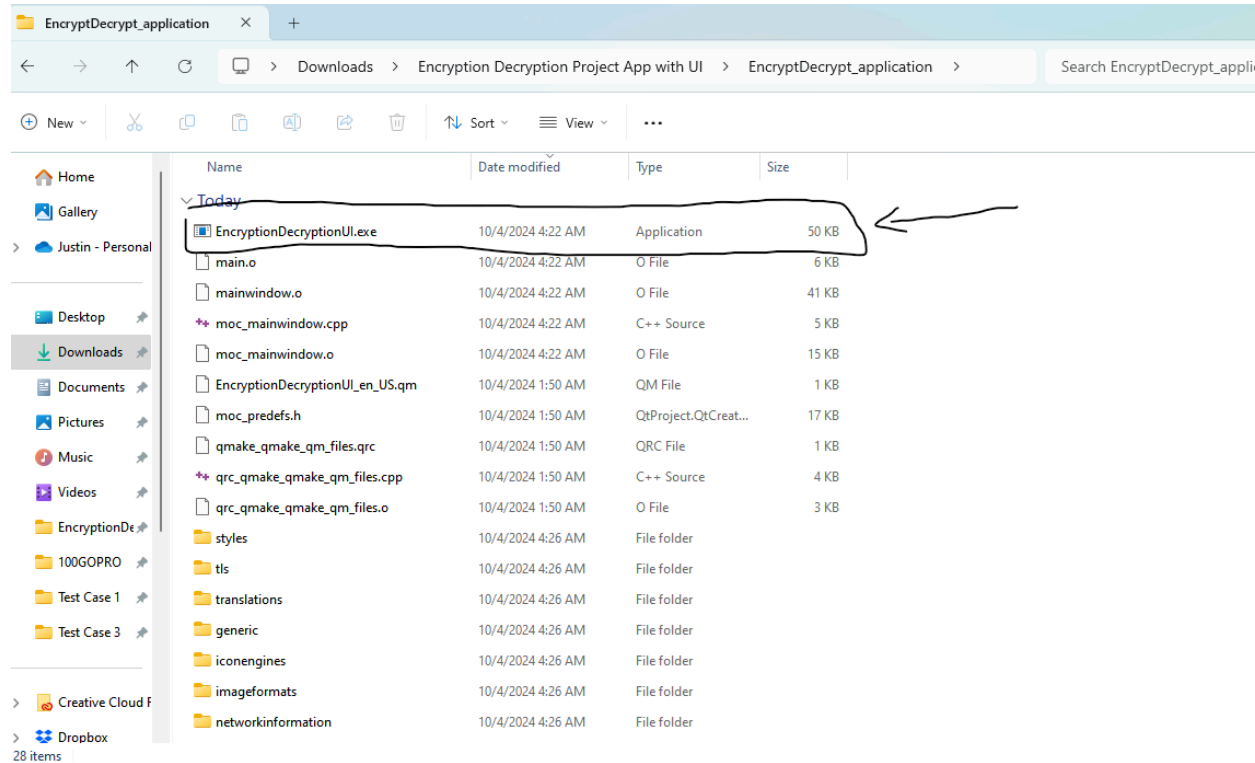
- Click “**Extract**”

### 2. Launching the Program

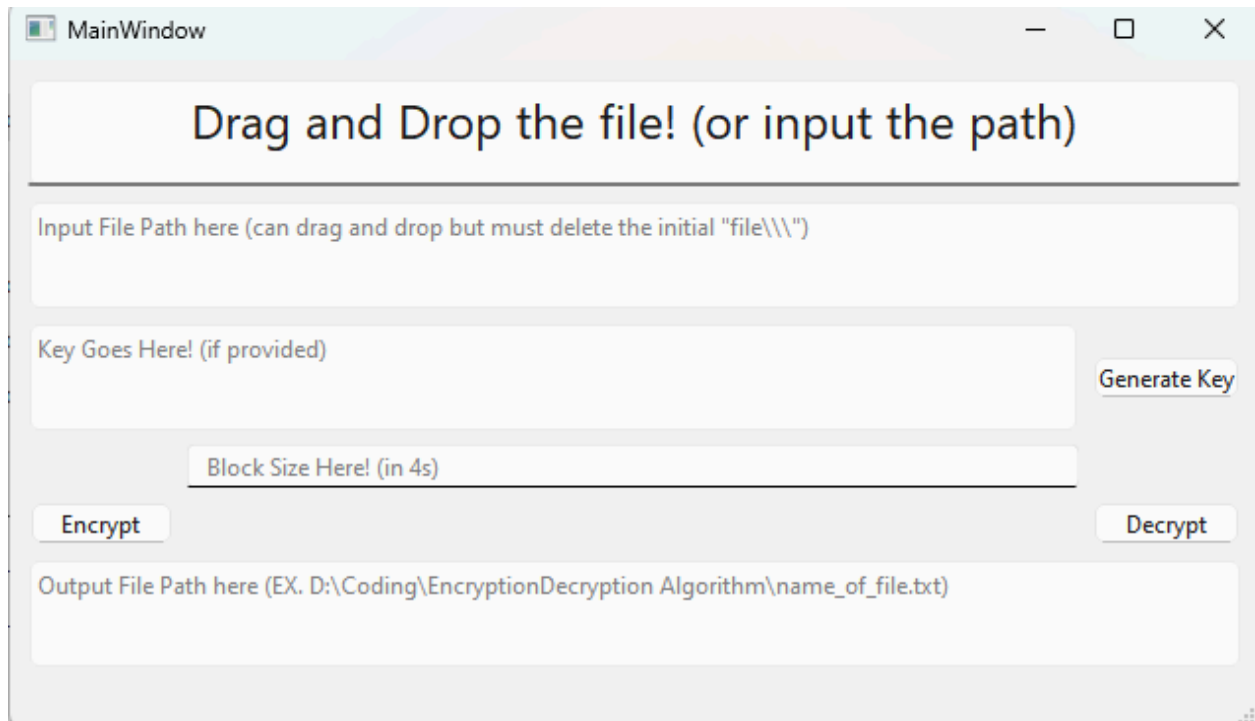
- Inside the extracted folder, go to the “**EncryptDecrypt\_application**” folder



- Within that folder, locate the file named **“EncryptionDecryptionUI.exe”**



- Open the file and it should open up to this

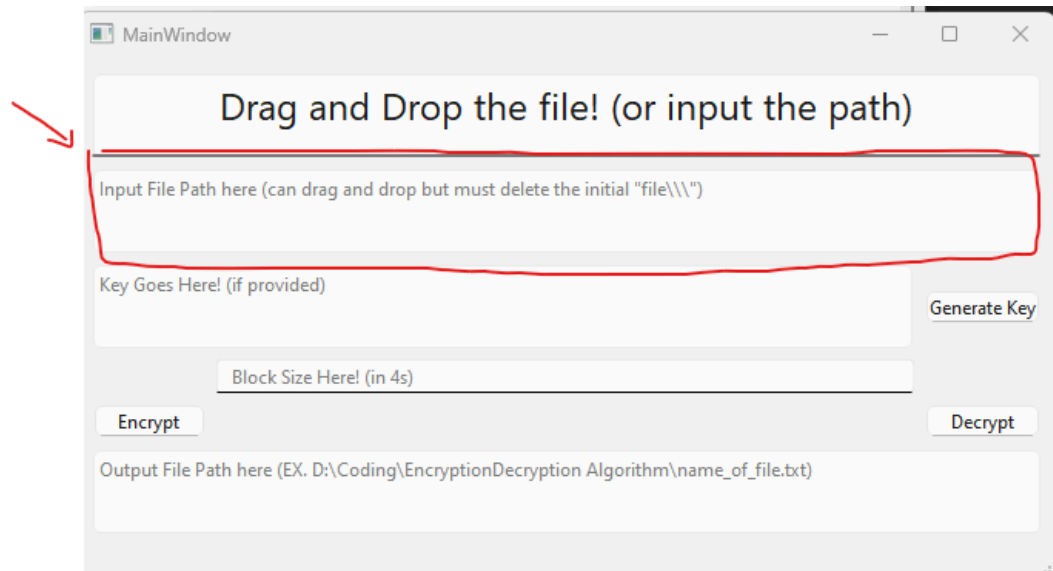


- Once you are in this screen move to step 3! **IF NOT GO BACK TO STEP 1**

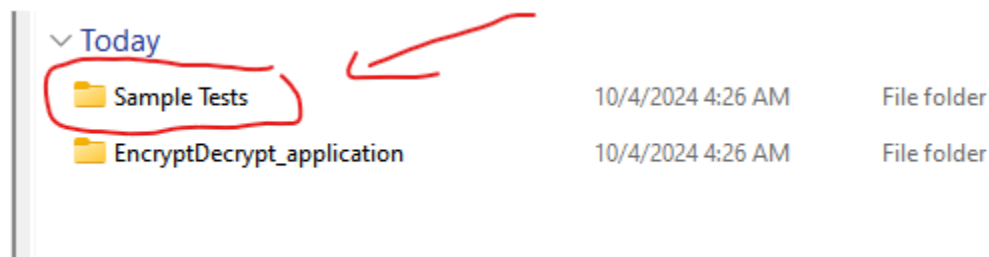
# Beginning Encryption:

## 3. Selecting the Input File

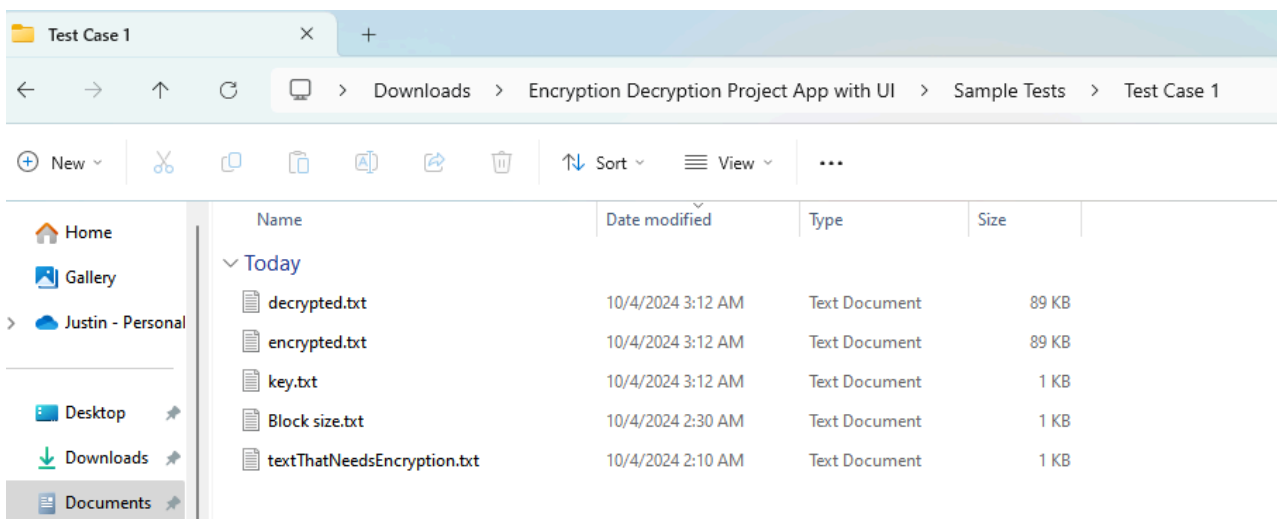
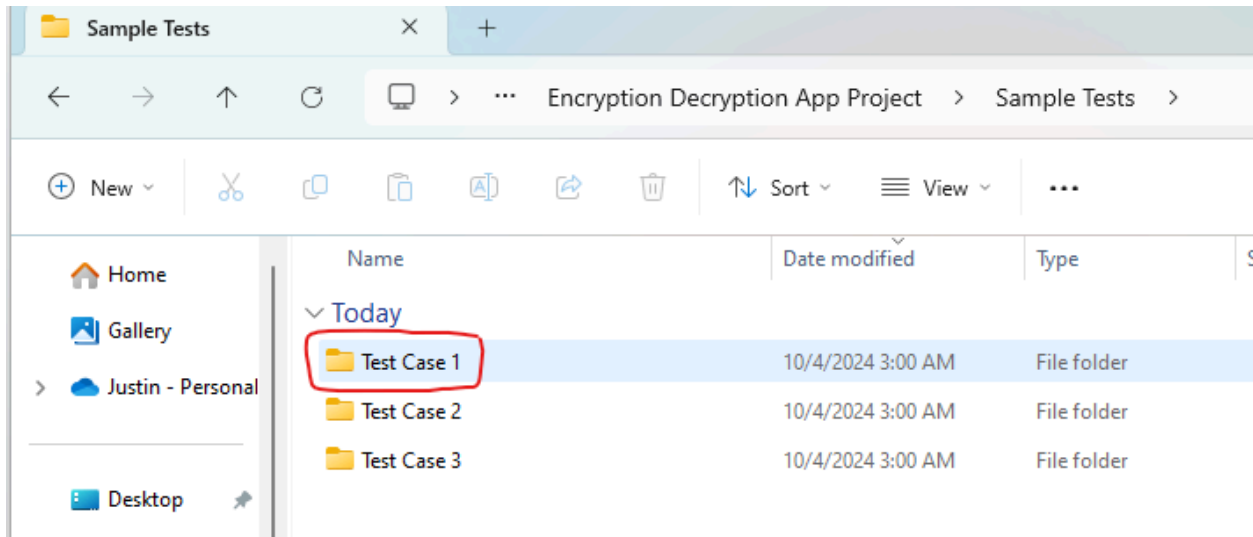
- **Note:** This is where things can get tricky be sure to follow the guide *exactly* here
- Where it says “**Input File Path here**” you can listen to the text at the top that says “Drag and Drop the file! (or the input path)”



- Where the arrow and red box is highlighted drop the path to your text file, so here go back to the first folder that contains the two folders “**Sample Tests**” and “**EncryptDecrypt\_application**”
- Select the “**Sample Tests**” and open it in another window
- 



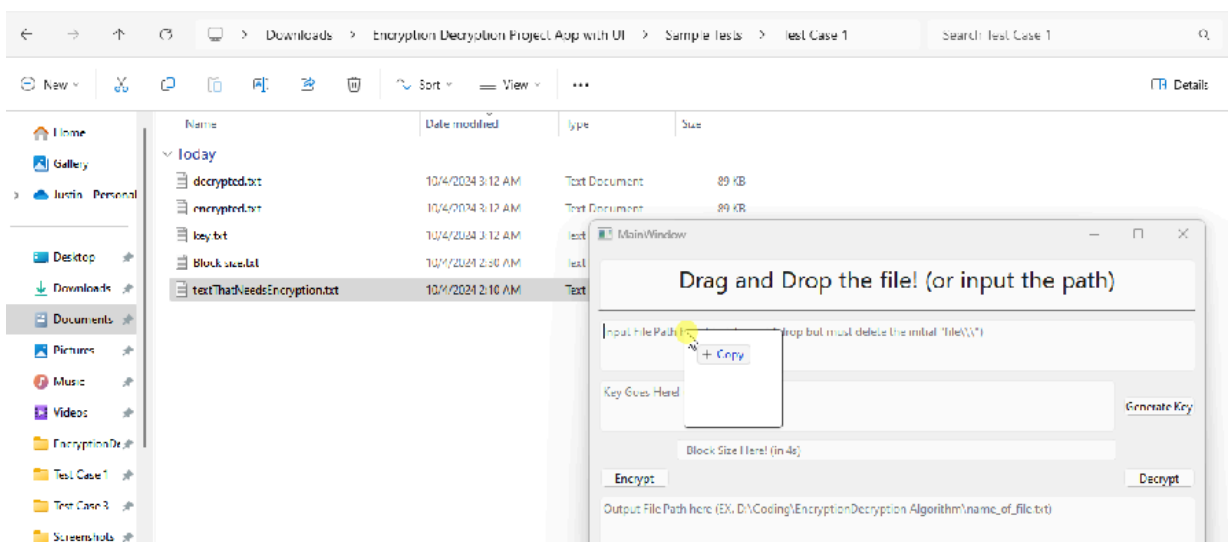
- Now go into the “**Test Case 1**” folder and you should see these files



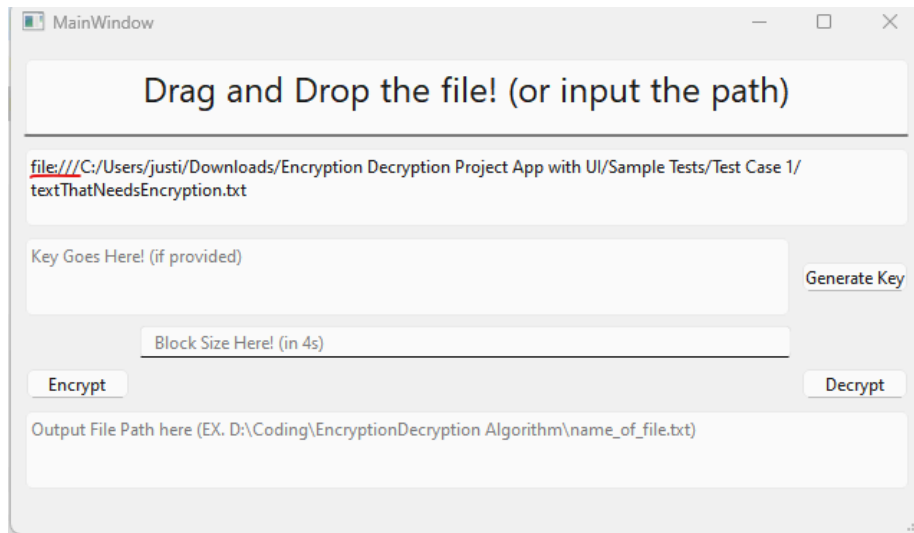
- Now you can either “drag and drop” the text file that says “textThatNeedsEncryption.txt” or you can copy its path, here is how to do both ways respectively

### DRAG AND DROP:

- To “drag and drop” the file, you must simply drag with your cursor and drop it into the window



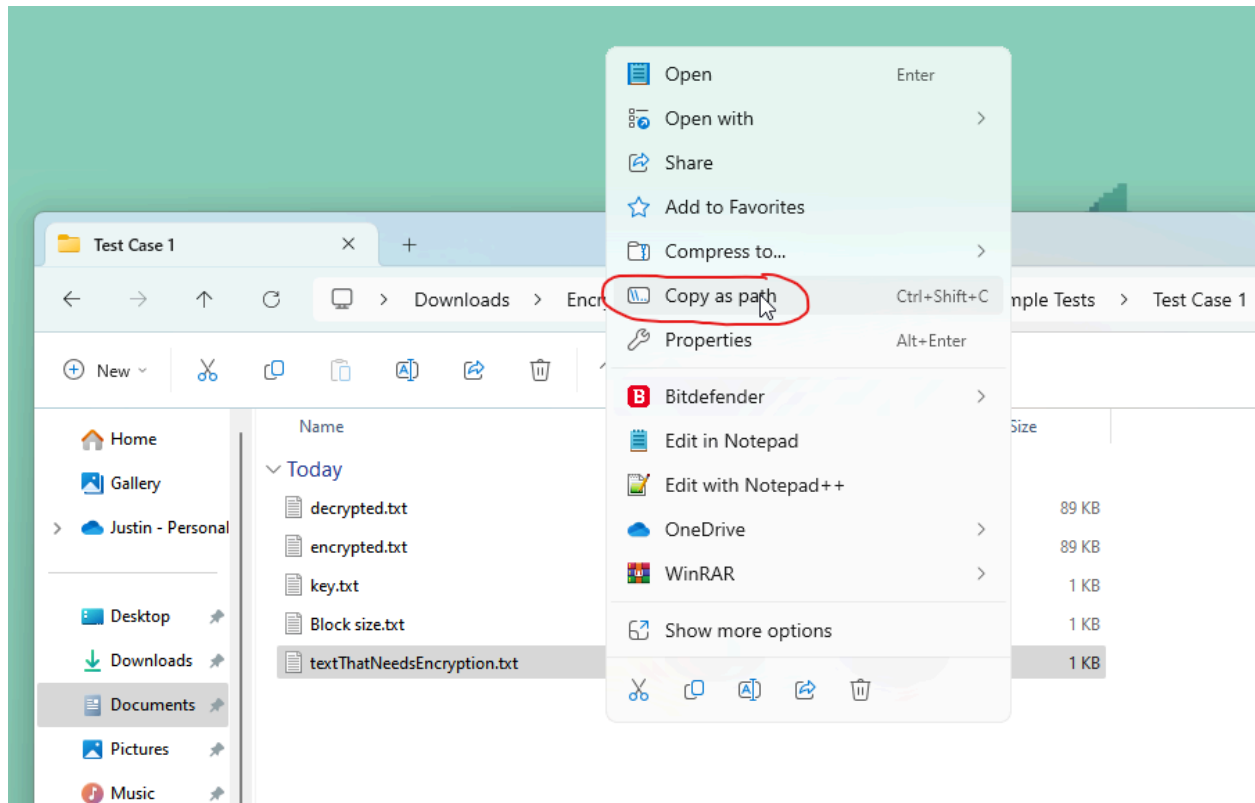
- When dropped into the window it will look like this:



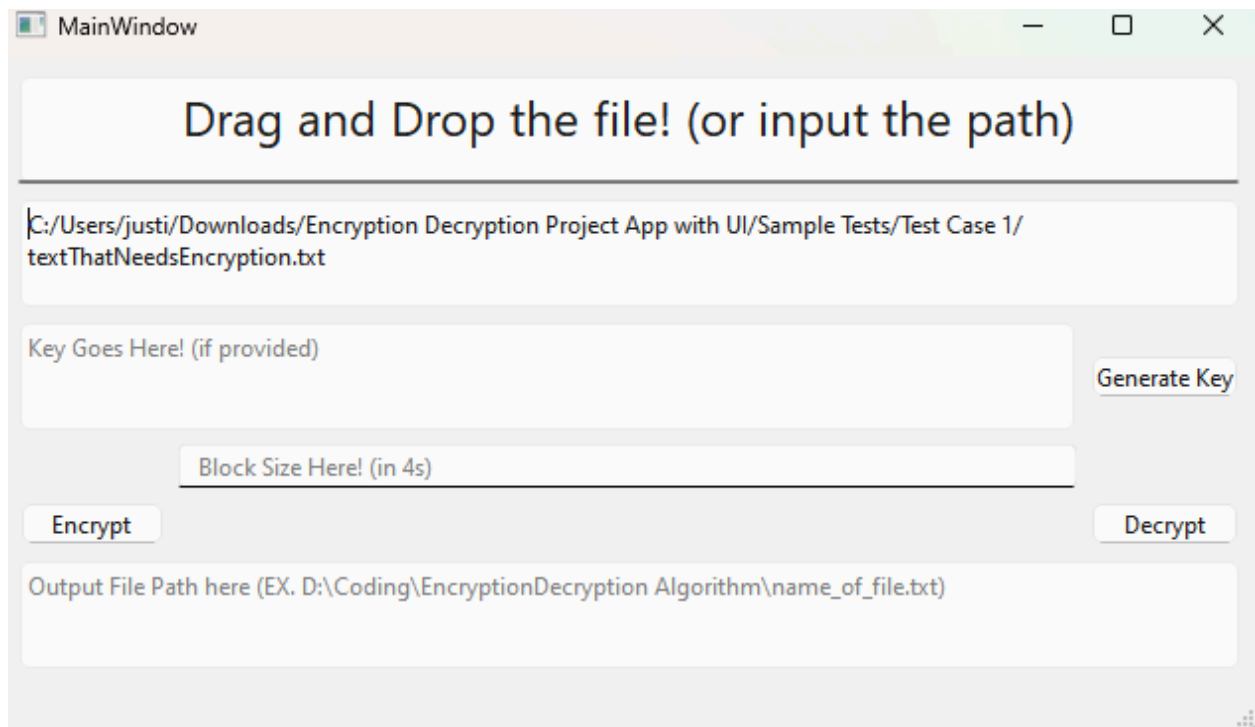
- Where it is underlined, delete it. So it should look like **“C:/Users/justi/Downloads/Encryption Decryption Project App with UI/Sample Tests/Test Case 1/textThatNeedsEncryption.txt”** instead of **“file:///C:/Users/justi/Downloads/Encryption Decryption Project App with UI/Sample Tests/Test Case 1/textThatNeedsEncryption.txt”**
  - This is mainly to correct the directory to the file since the code cannot read the initial part
- **Once past this step you can move onto step 4**

#### **COPYING THE PATH OF THE FILE:**

- If you do decide to get the path of the file, just right click the file instead and select **“Copy as path”** or on most systems **Ctrl + Shift + C**



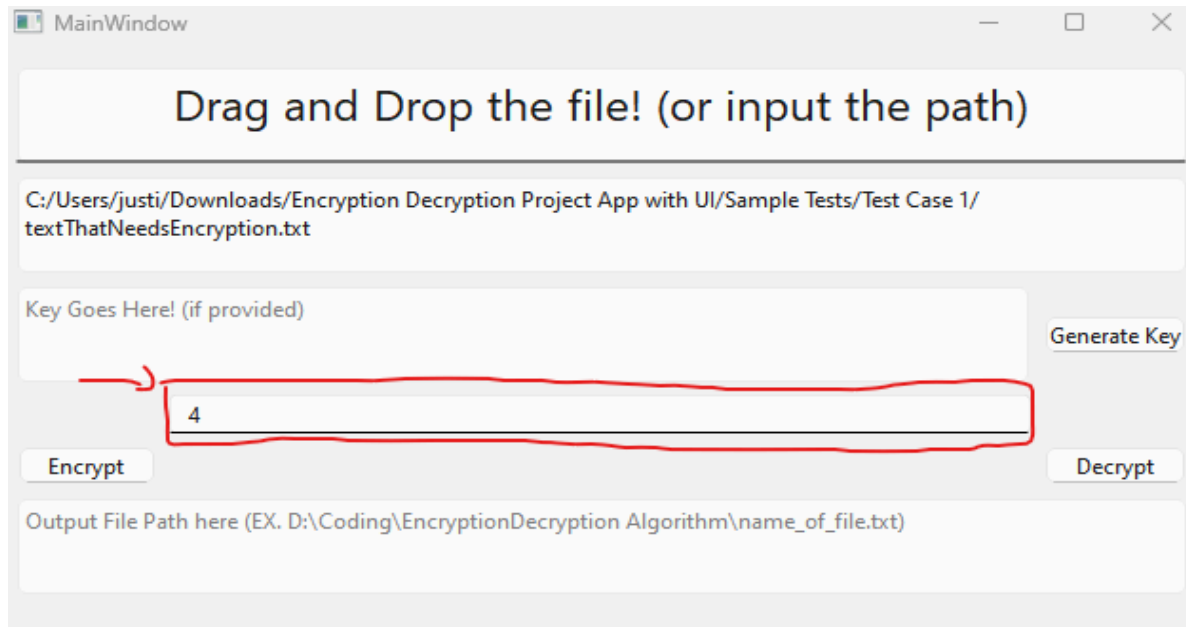
- Then with that go back to the window and paste in the path with **Ctrl + V**
- It should look like this:



- **Now we can move onto step 4**

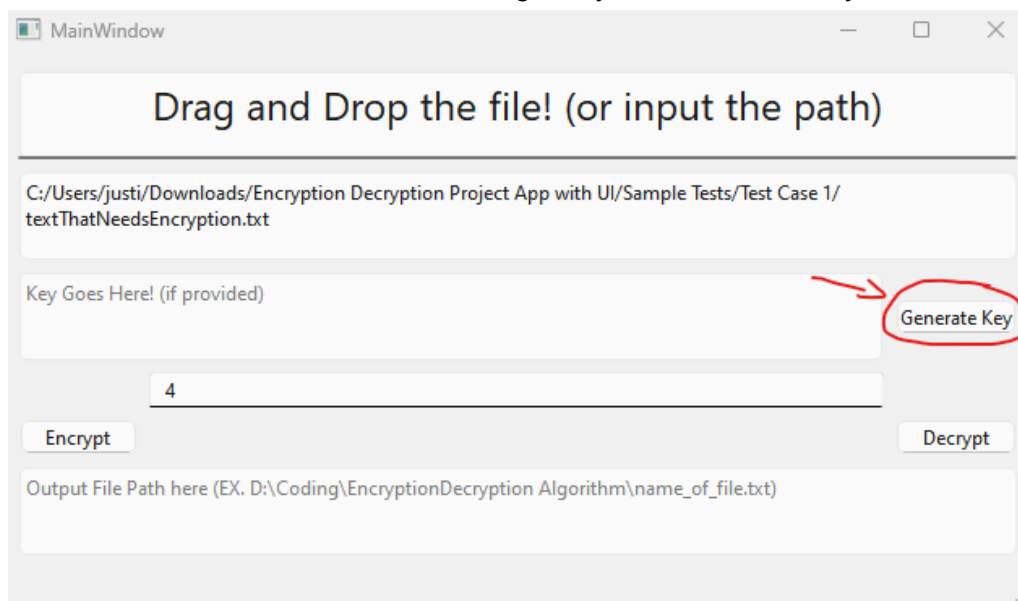
#### 4. Selecting the Block Size and Generating a Key

- For this step we can select the block size to be in “4s” so you can choose the block size to be 4, 8, 12, 16, 32, 64, or even 1024, it is up to you.
- In this example we will use 4 to start...
- Keep in mind that you must remember what block size you put, hence why there is also a file named “**Block size.txt**” for future reference in the test case folder
- Below is what it will look like when inputted



The screenshot shows a window titled "MainWindow" with a header "Drag and Drop the file! (or input the path)". Below the header is a text area containing the file path "C:/Users/justi/Downloads/Encryption Decryption Project App with UI/Sample Tests/Test Case 1/textThatNeedsEncryption.txt". Underneath is a label "Key Goes Here! (if provided)" followed by a text input field containing the number "4". A red arrow points to the input field, and a red box highlights it. To the right of the input field is a "Generate Key" button. Below the input field are two buttons: "Encrypt" on the left and "Decrypt" on the right. At the bottom is a text area for the "Output File Path here (EX. D:\Coding\EncryptionDecryption Algorithm\name\_of\_file.txt)".

- Here we are choosing to ignore the “**Generate key**” so that it may randomly generate any 16 character key for us, or you can click the “**Generate Key**” button to create one
  - When “Generating a key” it will create a key for us but will not yet save it.

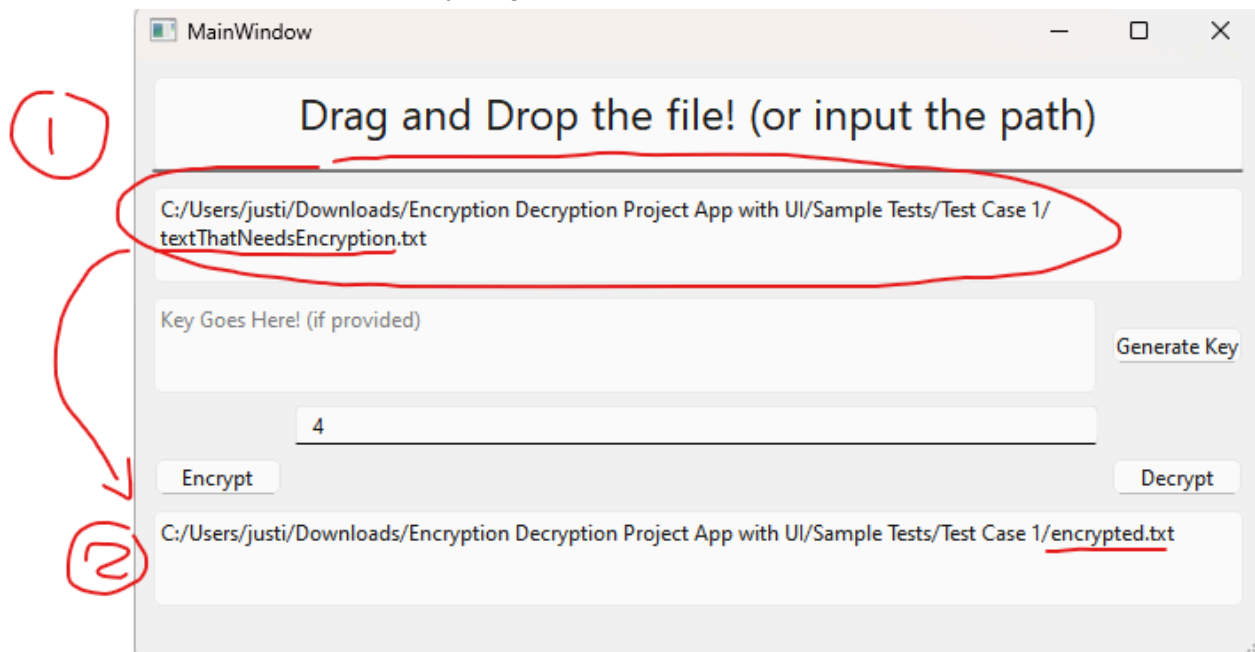


This screenshot is similar to the previous one, but the "Generate Key" button is circled in red with a red arrow pointing to it. The input field still contains the number "4".

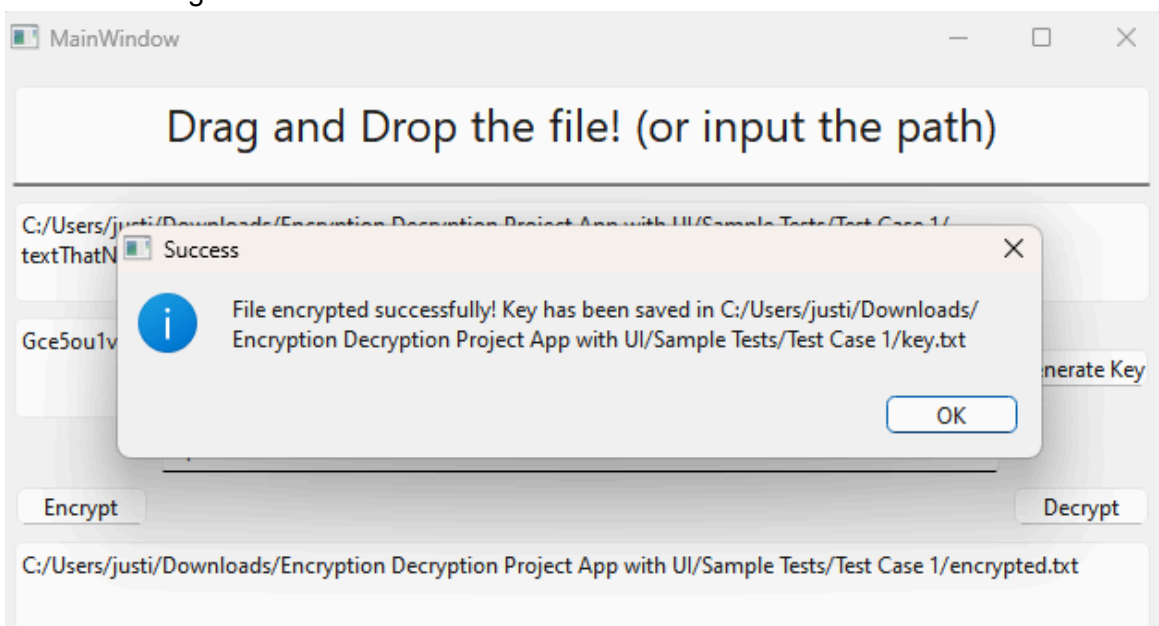
- Now let us move onto step 5

## 5. Selecting/Creating the Output File Path

- Here we are going to be making a output file path, so we can choose the name to be whatever we want and location, but for this specific example we are going to use the name “**encrypted**”, and choose it in the same folder as our Test Case 1
- With that in mind you can copy and paste the path from above the key and paste that into the Output path, only renaming the things before .txt and after “**Test Case 1/**”
  1. Copy the path from the input
  2. Paste the path onto the bottom “**Output File**” renaming it to your desired name, in this case “**encrypted**” because we are encrypting the file

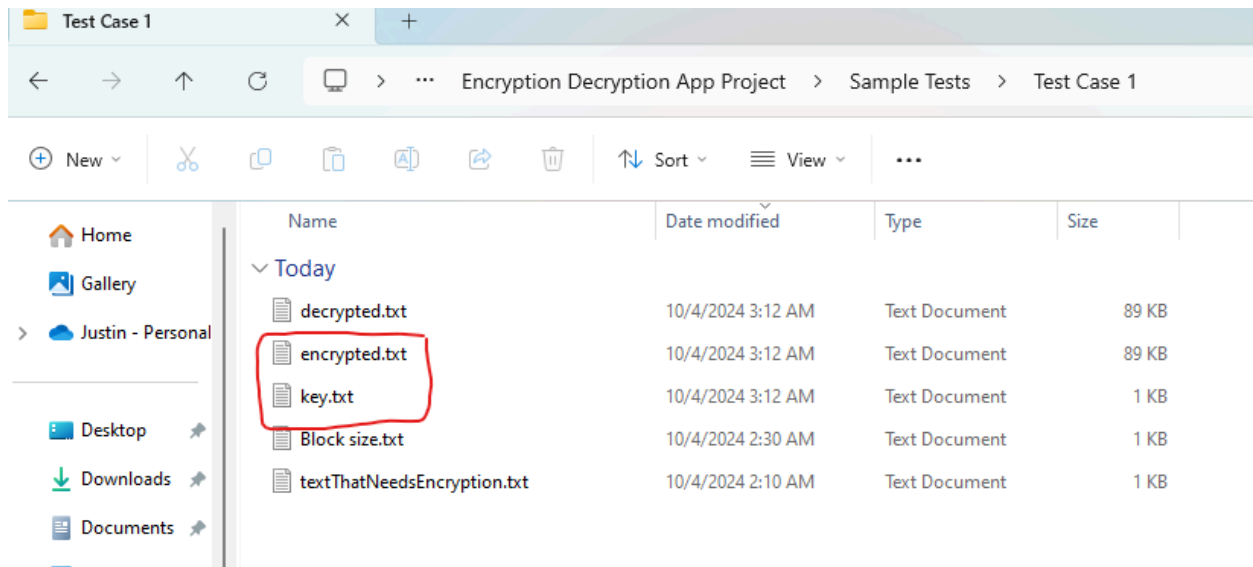


- Now you can select “**Encrypt**” and you now have an encrypted file with the key saved to the same location!
- There should be a pop out screen telling you that you have successfully encrypted the file, and that a key has been saved to the designated location alongside that file





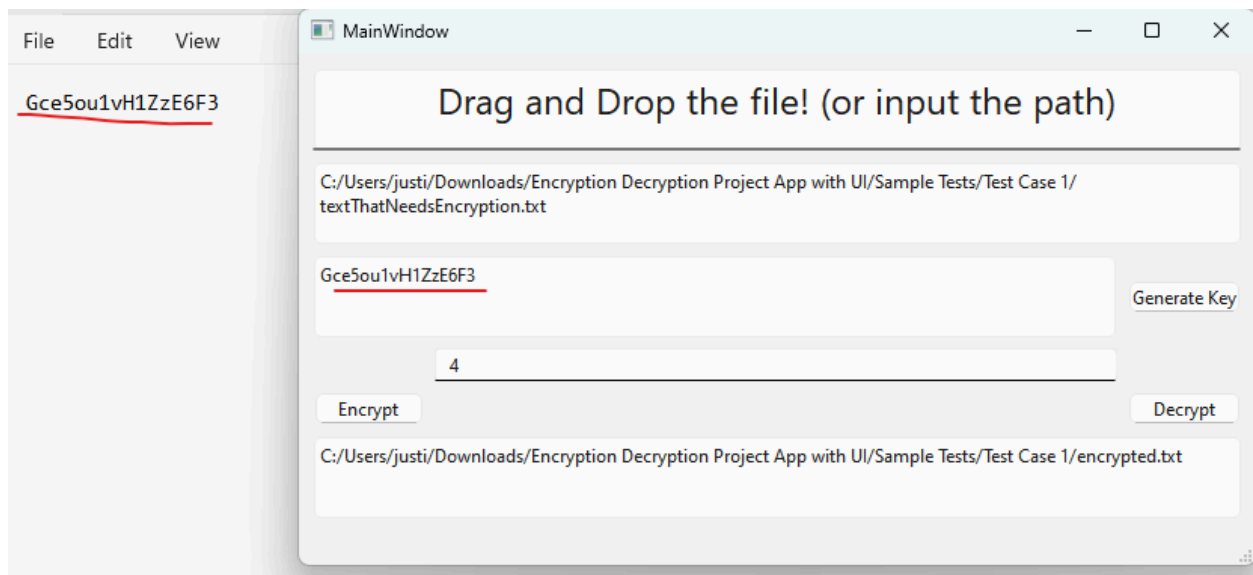
- If you navigate back to our **“Test Case 1”** folder, you will find two .txt files that have replaced your **“encrypted.txt”** and **“key.txt”**



- Now with our newly generated key and **“encrypted.txt”** if you open **“encrypted.txt”** it should showcase cipher text like this:

```
|
P0Ux0;D76*D#^g*0F00}0:T7Z
F5F*/
G
00?8B/0 Y4V*C,E0 \:'C?0e06@20)Z00\VB0)0(z)A"0E|00D00^(0(00C400y I0hx* 0[
\5000&0B0%}50 [fz700X#0C0%00
6C+0(0 X0<A0=\
```

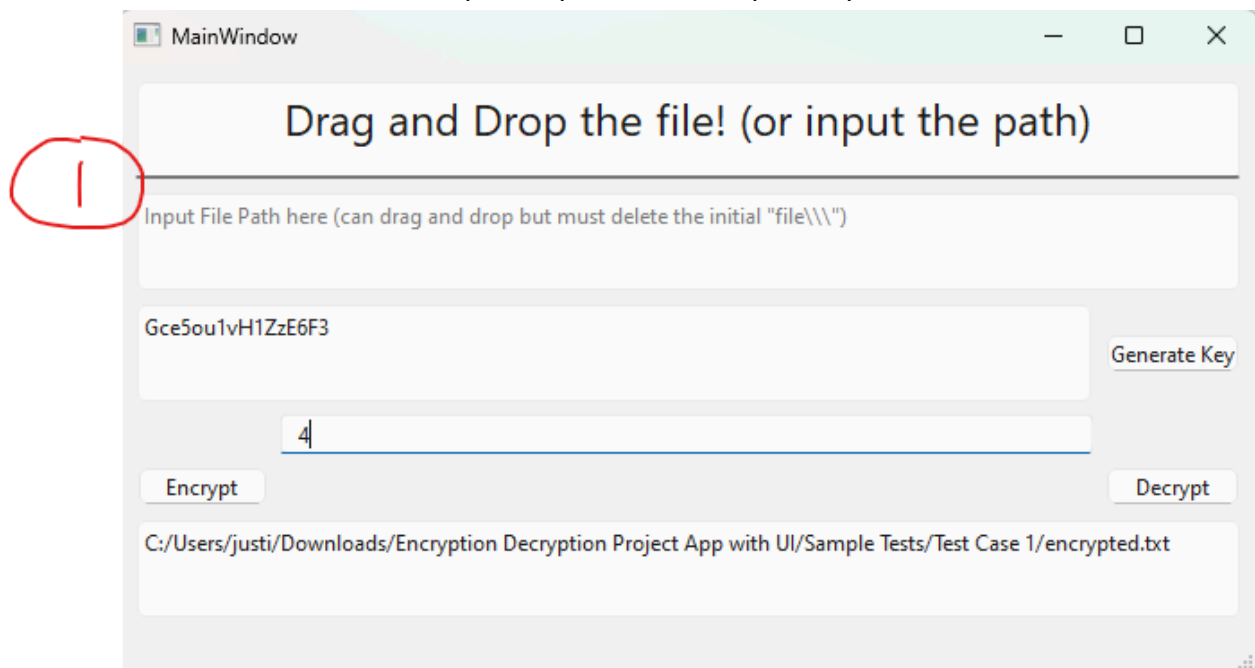
- And alongside this encrypted file you also have your **“key.txt”** which if you open should showcase exactly like what you see in your window (keys may or may not be similar due to random generation):

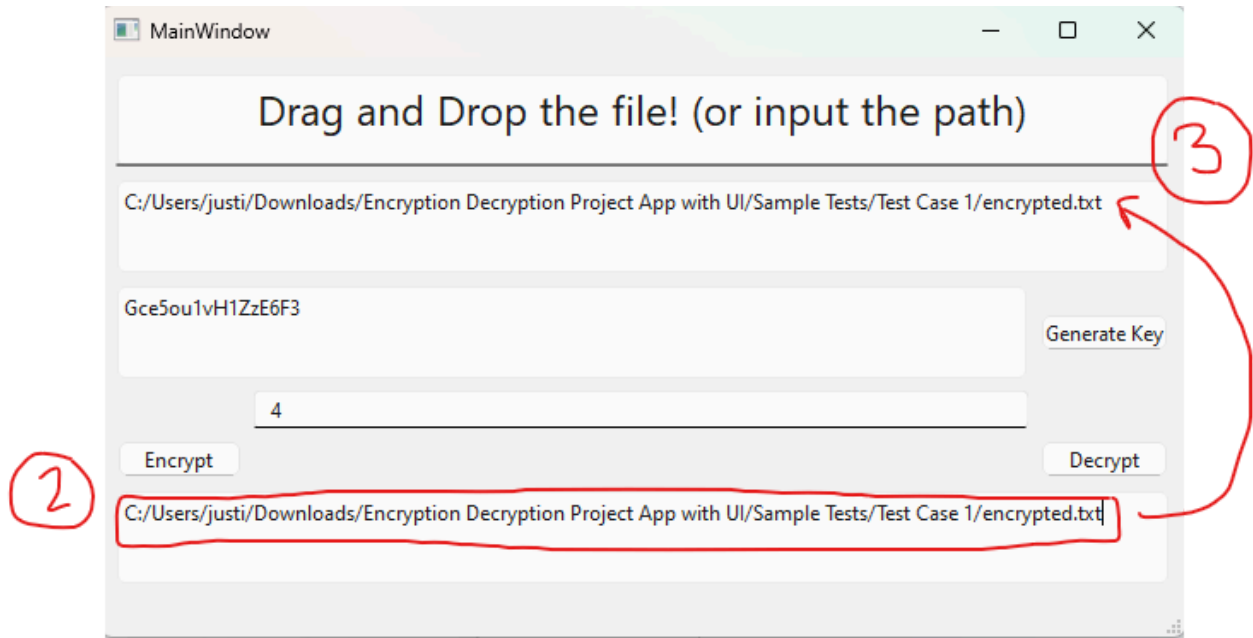


# Beginning Decryption

## 6. Reselecting the Input File

- Now in order to decrypt your encrypted file you must copy the path location to the “**encrypted.txt**”, so you can simply delete the input file and copy and paste the output file path into the input file path:
  1. Delete the input file path
  2. Copy the output file path
  3. Paste the output file path into the input file path

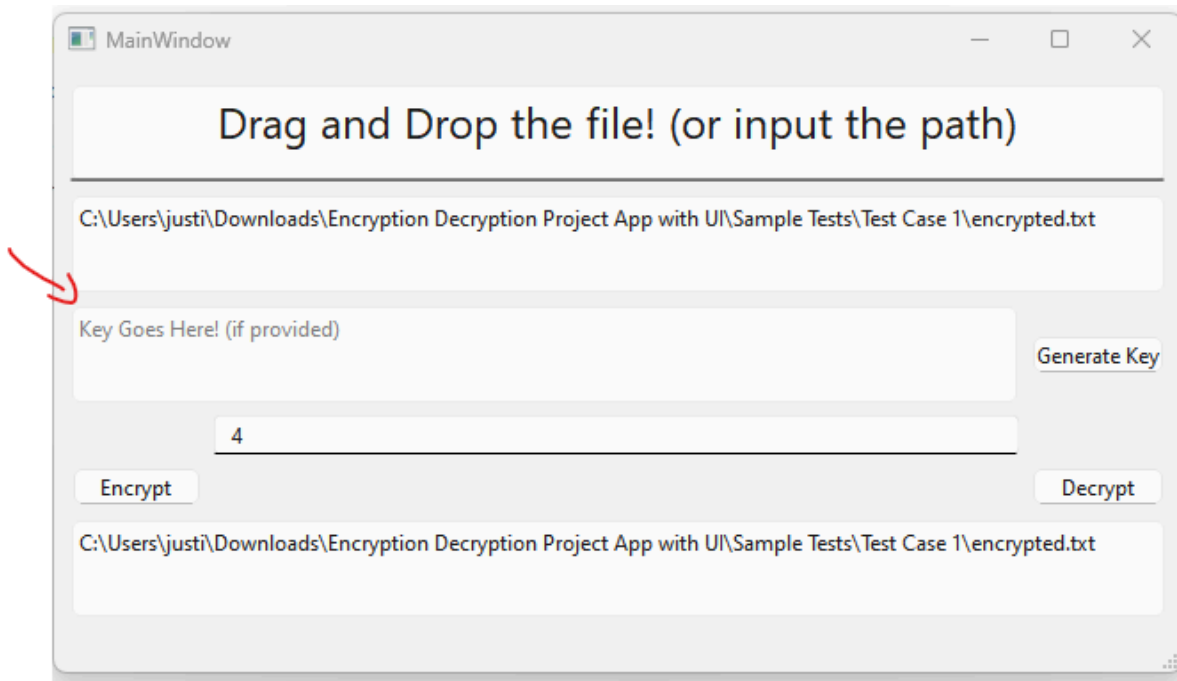




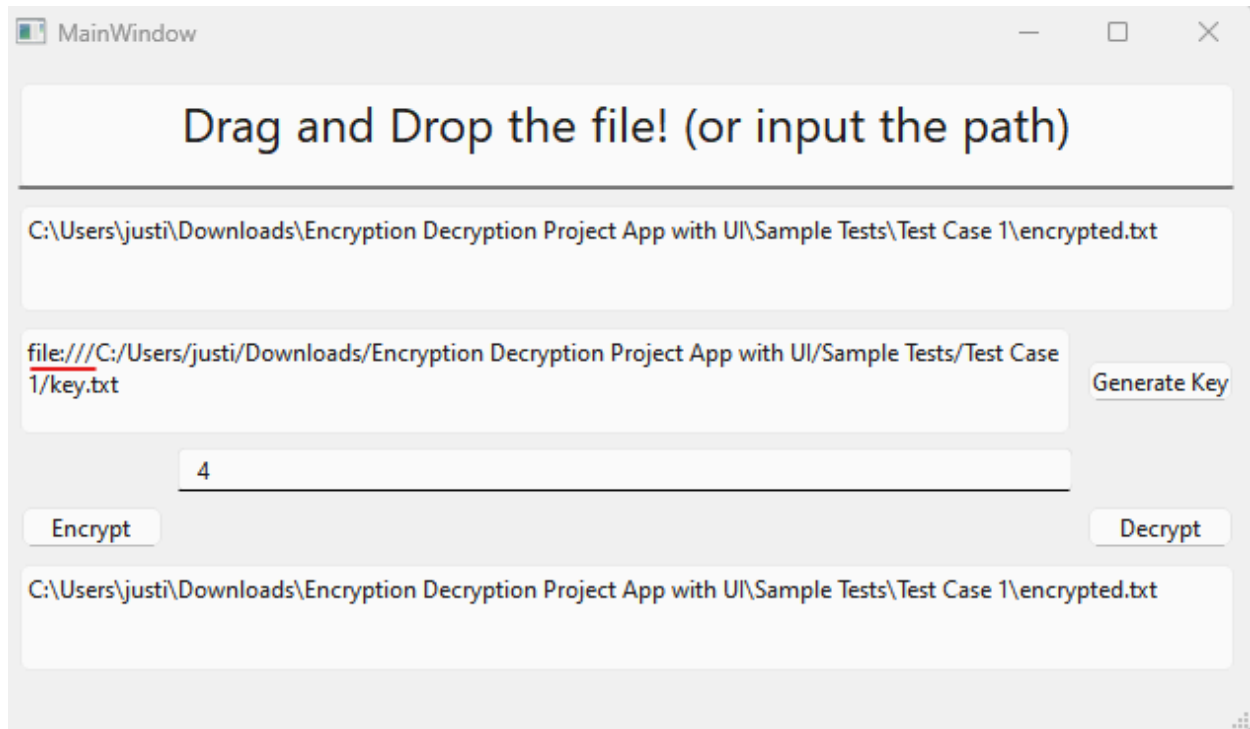
**Note:** Now that we have the input file path, we just need to create a new output file path

## 7. Reselecting the Block Size and Key

- If you have accidentally generated a new key, no need to worry, just delete the key and “drag and drop” the “**key.txt**” file into the empty space that says “**Key Goes Here!**”



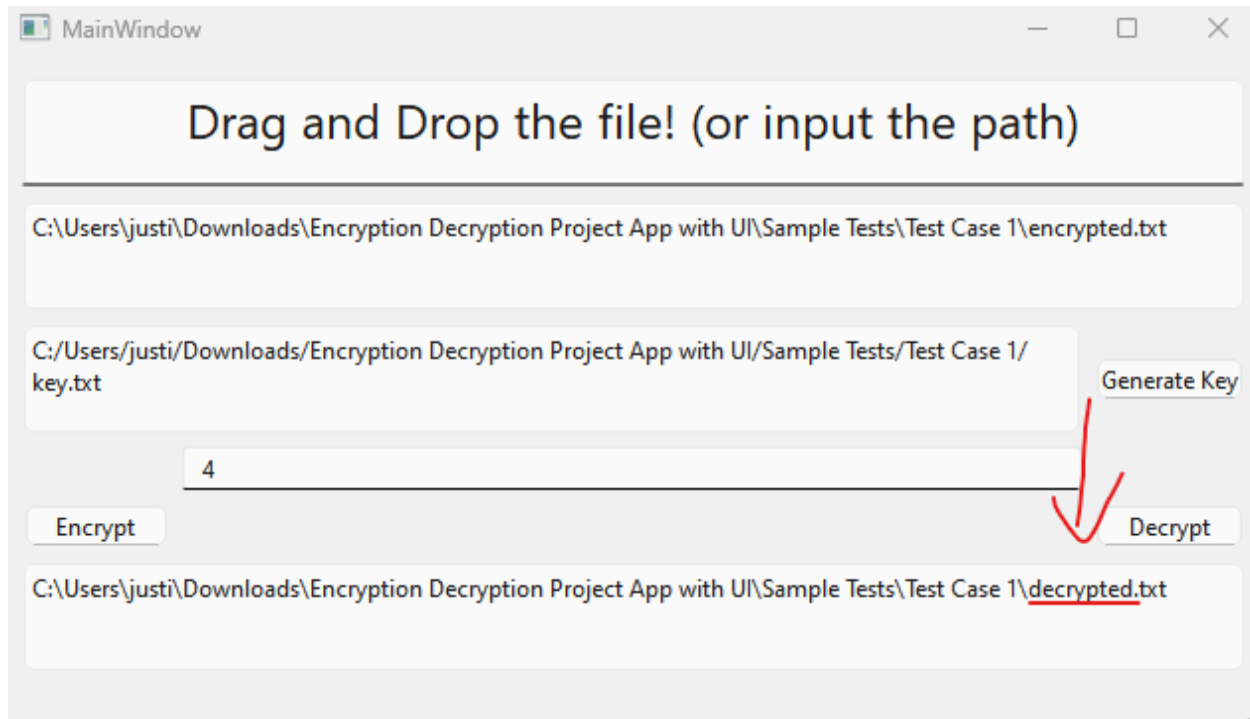
- Be sure to remove the “**file:///**” in front of the directory so that it could read the “**key.txt**” file



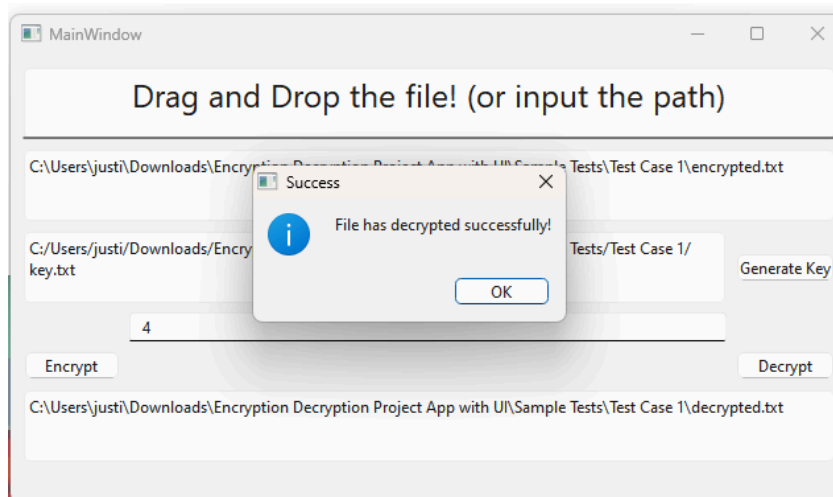
- **Note:** If you have kept the block size on a separate piece of paper you can always refer back to it and rewrite it as needed if you also accidentally close out of the application. Of course you can always brute force if you must, if you do end up losing the block size (remember multiples of 4).
- Otherwise we are finally ready to move onto step 8.

## 8. Creating a New Output File Path

- Now that we have come this far, we can repeat the same process we had when selecting a new output path. This time the key will not go with you, as you will only get the message that was originally encrypted.
- So, in this case we are going to leave the location as is for simplicity, but we are going to change just the name from “**encrypted.txt**” to “**decrypted.txt**” so it will look like this:



- Now simply press “**Decrypt**” and you have successfully decrypted your message!
- It will have a popout screen with “Success” and the actual information text stating “File has decrypted successfully!”, you may select **OK** to close the window out.



- Now if you go back to your “**Test Case 1**” folder you will see the “**decrypted.txt**” and now be able to open it and see that it showcases the original message that needed to be encrypted!
- Have fun experimenting with the encryption!

## Some Things to Note

Make sure block sizes are the same, so if you encrypt with 16 blocks you have to decrypt with 16 blocks, if you encrypt with 4 blocks, it has to also be 4 blocks that you also decrypt with.

Make sure paths are exact with no "file://" in front, has to be "D:\whatever" not "file\\D:\whatever" in order for the encryption and decryption to work

**DO NOT LOSE THAT KEY OR THAT FILE CANNOT BE DECRYPTED WITHOUT**

**SAME WITH BLOCK SIZE REMEMBER HOW MANY OR WRITE IT DOWN IN ANOTHER FILE (this one you can brute force it may just be a while)**

Make sure to put a legitimate generated key, a key that you make up will most likely not work - haven't been able to check but once out of every so often it would work when I experimented

Auto generates a key if you do not have a key when encrypting. If you want, you could generate a key, it will also create a different key if you do not like the key provided

Key file is always saved as "key.txt" and saved into the same folder as the encrypted file

KEY HAS TO BE 16 CHARS

Also it WILL overwrite files with the same name so be careful upon naming it and placing it in certain locations

In the provided test cases i also stated the block size used to test

If you read this far down I hope this project satisfies if at all possible anything... Took me a long time to create and was probably the most i've learned