# Quantum Resistant Algorithm Encryptor

*FOR CISC-4900-VC1B WITH PROFESSOR CHUANG*

*SUPERVISED BY MATTHEW MCNEILL*
*<MCNEILL@BROOKLYN.CUNY.EDU>*

*BY AHMED OMER*
*<AHMED.OMER68@BCMAIL.CUNY.EDU>*

# ABSTRACT

## *Quantum Chips are on the rise;*

And they seek to completely invalidate our cyber security. To combat this, many cryptographers are working time in and out around the globe, to workshop solutions. Of these solutions, there resulting were many quantum resistant algorithms; but they are of little public awareness; which is an issue. As we stand; we are ahead of what could legitimately be called a decryption crisis- and consequently, all we *need* to do to combat it, is spread awareness.

## *The project's core response;*

Which leads to my project, specifically- my project is a runnable windows application with its own install wizard; focusing primarily on the *educational* function of introducing, and explaining, multiple encryption types, while also fulfilling the *exemplary* function of working wholesale also as a file encryptor *using* said types.

The projected end-result of the project is a Windows Application (7+) with it's own install wizard, internal-file-storage within the user's directory(ies), and a hookup to Window's uninstall manager system client.

# ABSTRACT(cont.)

## *The project's core response(cont.);*

From there, the user will have access to a minimum of three encryption types; AES standard encryption, a Quantum-Resistant-Algorithm Based Vector-Lattice encryption ( exemplifying IRL minor usage via algorithms such as KYBER or DILITHIUM ), and an unchosen third. Then, beyond access to the base encryption & decryption of their files ( such as when in using the apps the likes of WinRAR or 7-Zip ); there will additionally be a blurb *explaining* each algorithm, its current history, common places of usage, and, if a *Quantum-Resistant* Algorithm, how it's stated in in tech literature to stack up to projected findings and thoughts on Quantum-*Chips*.

For complete accuracy of the project, while unflattering, it must be said that this is majorly a *learning project*. Before this project, I have had never made a proper Windows application, much less an install-wizard; and while my knowledge of cyber-security is *possibly* better than most; it is not my major for a reason. As such; the projected completed project, while *looking* to be satisfactory in its totalized development; it will most likely NOT be anything… striking.

# TOOLS & REQUIREMENTS

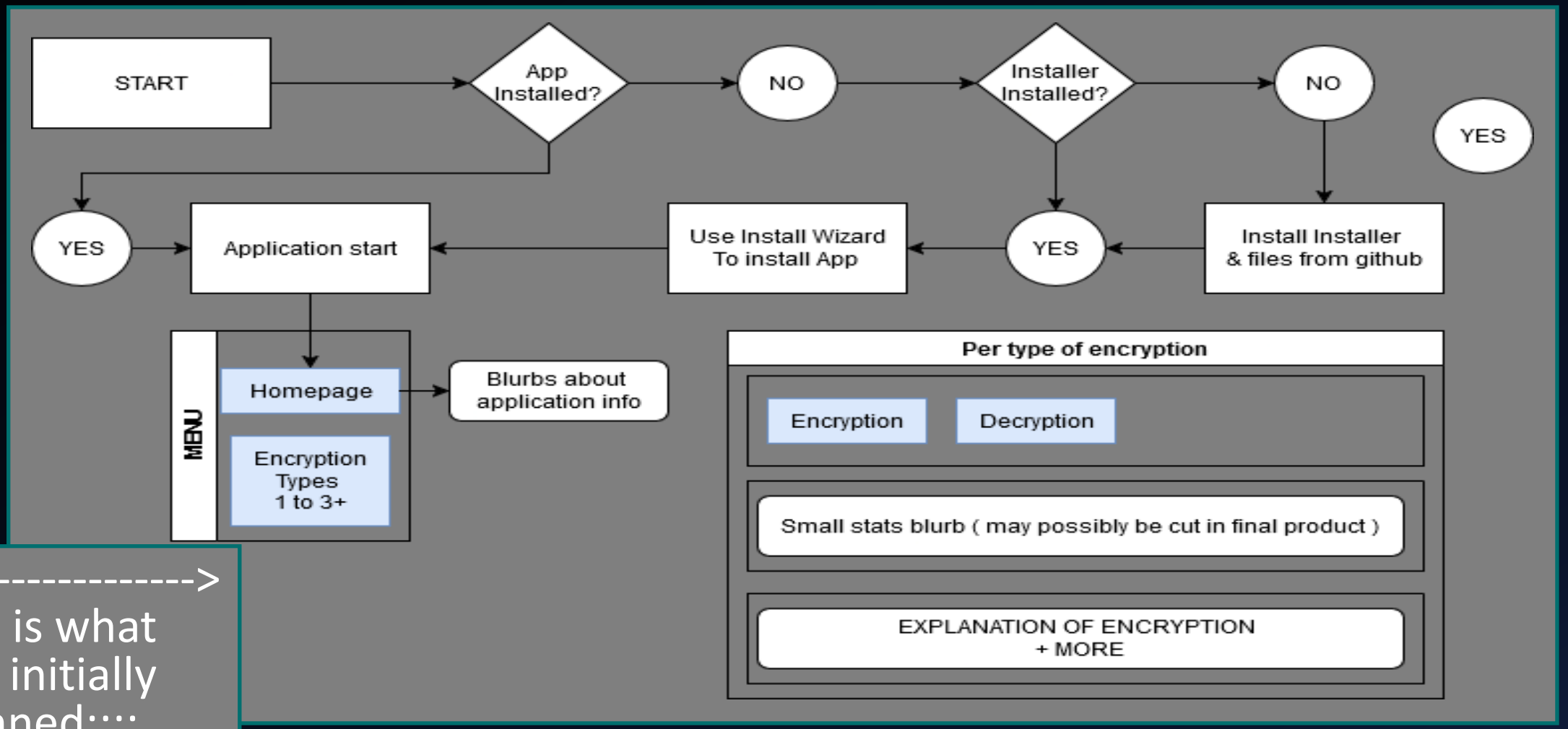
*User-end;*

- Windows 7+
- Upwards to 3 MB of free space ( looking to be much, much smaller; >1MB, but overestimation is safer )
- Internet access to download from Github, and nothing else! Every other file/thing required is packaged within the installer.

*Project-end;*

- Visual Studio 2022, version 17.12.4+
- My Laptop, & it’s Windows OS
- Microsoft’s development resources ( online )
- BouncyCastle’s cybersecurity library,
- And unironically, at least one actual band-aid ( I papercut myself on my computer. End me. )
- And nothing else!

# PROPOSED USER INTERFACE FLOW: INITIAL PROJECTIONS




----------->
This is what was initially planned;;;;

# PROPOSED USER INTERFACE FLOW: WHY I HAD TO SCRAP THAT, IN A MANNER;;;

*Encryption is more complex than that;*

When I first drafted that application flow, I didn't have the best grasp on what was involved, when it came to encryption versus decryption, versus types of either, versus both! Heck, I didn't even know the *very important* difference between key generation and validation, and direct file encrypting/decrypting. As such- when I first made that flowout-chart, I neglected to include, or plan for very important schema for each algorithm type. Such as an input-password/key textbox! Without such, all generation of encryption would either have to always use the same static key ( dumb ), or only provide encryption- no decryption. And, while either would be fine for an *educational* showing of how each algorithm worked- it wasn't satisfactory, or logical enough to me, to simply leave out such's inclusion.

Speaking of things not included in the flowout-chart- a more direct explanation of *how to use the application, specifically*! It was pointed out by my supervisor, Professor Mcneill, that while *I* knew how to navigate my application inherently, since I built it- others *not* so familiar wouldn't have that advantage. And such so, the inclusion of the 'How to use' box on each page was born!

Another thing the flowout-chart failed to mention, was a box to hold & print all the error, success, and status changes and messages! While the "Encryption" and "Decryption" boxes on the thing was a symbolized spacing for input & button boxes for each category, I also failed to roughly plan *where* the total outputs would go.

To be fair here though, I believe not having done so eventually went to my benefit- instead of having a plan I would feel compelled to stick to beforehand ( an issue I'll get to later ), I simply did what made sense. As such- I made the encryptions overwrite the appropriate input-textboxes, for easy testing, and *also* included an option for printing your results to a Desktop-.txt file- a feature that ended up expanding and being apart of every tab of the project!

# PROPOSED USER INTERFACE FLOW:
# WHY I HAD TO SCRAP THAT, IN A MANNER;;;

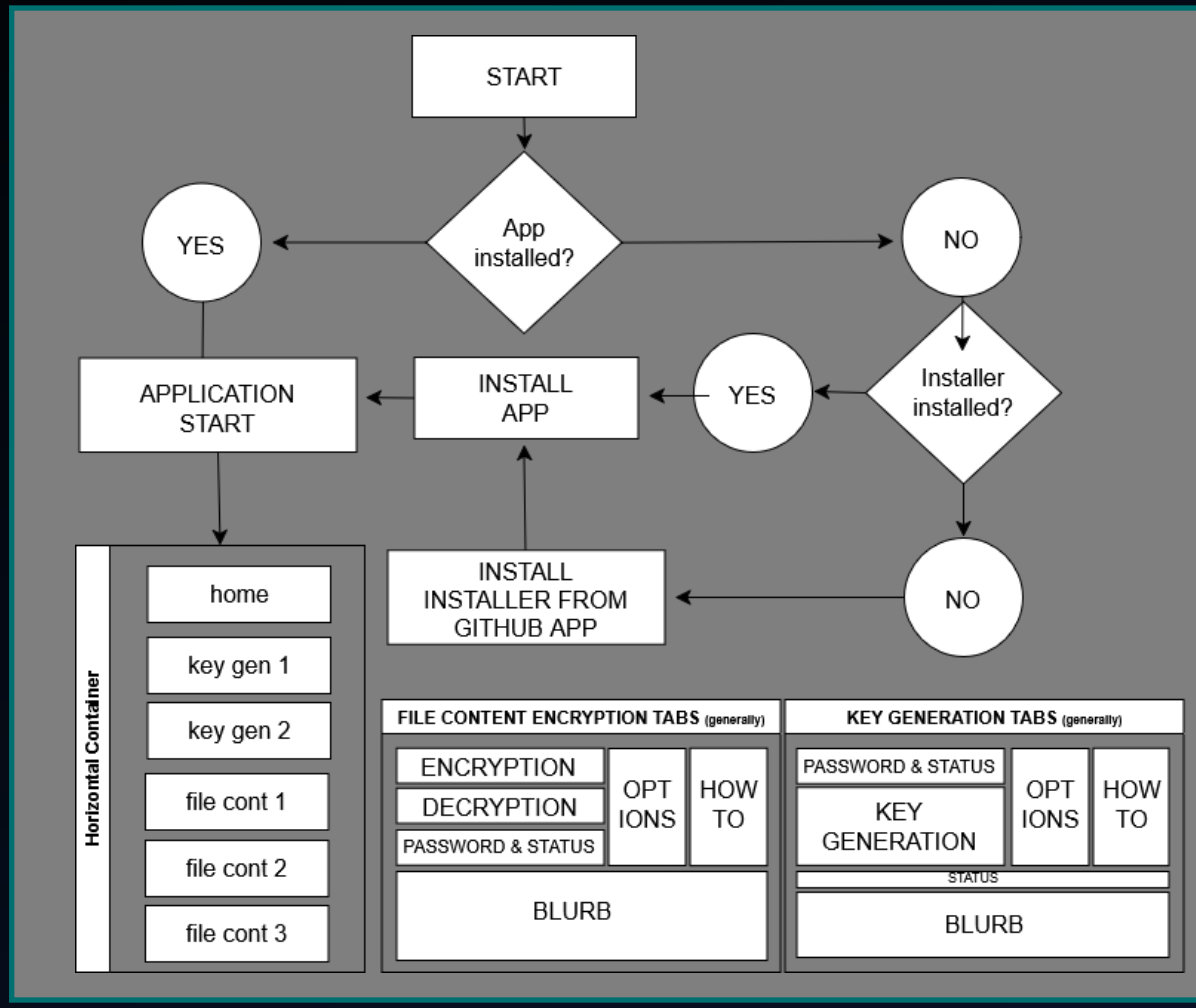
*Other changes;*

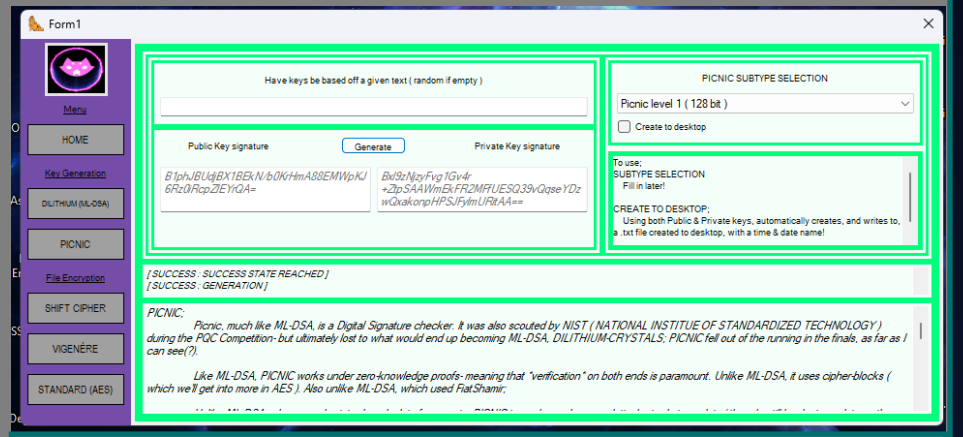
To be fair here though, I believe not having done so eventually went to my benefit- instead of having a plan I would feel compelled to stick to beforehand ( an issue I'll get to later ), I simply did what made sense. As such- I made the encryptions overwrite the appropriate input-textboxes, for easy testing, and *also* included an option for printing your results to a Desktop-.txt file- a feature that ended up expanding and being apart of every tab of the project!

Beyond that all, the flowchart had another issue- one stemming from my lack of knowledge. You see, when I planned this project out from the beginning, I wanted one Quantum algorithm, one common algorithm, and one lesser-known one- all as content-file encryptors. However, what I didn't realize was that a LOT of cryptography encryption- especially those who are Post-Quantum, either dealt solely with Digital Signature generation and verification, for plain key generation. And those which didn't, for some reason, no matter how hard I tried, was simply beyond me. As such- I ended up having to minorly compromise my original idea- although I think I ended up doing more- instead of three algorithms, I now have 5; 3 considered post quantum, two of which being Key generators that can also double as verifiers ( and do so, inside my code- I simply could-not detach the notion from the internal function, so I couldn't separate it, to show it, without reference, so I didn't- but it IS in there );; And 2 extra ciphers for content encryption- both serving the educational-aspect of my project greatly, as they serve explicitly as stepping stones to talking about AES, from which, you can then go back to dive into the key-generator's blurbs- sort of like reading levels.

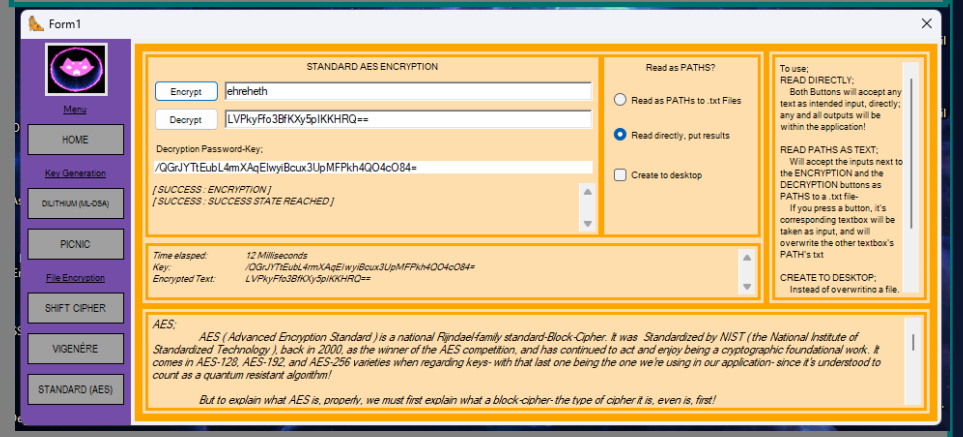
But that's getting a bit ahead of myself, now- the revised flowout-chart for my application ended up looking like;

# PROPOSED USER INTERFACE FLOW: CURRENT PROJECTIONS

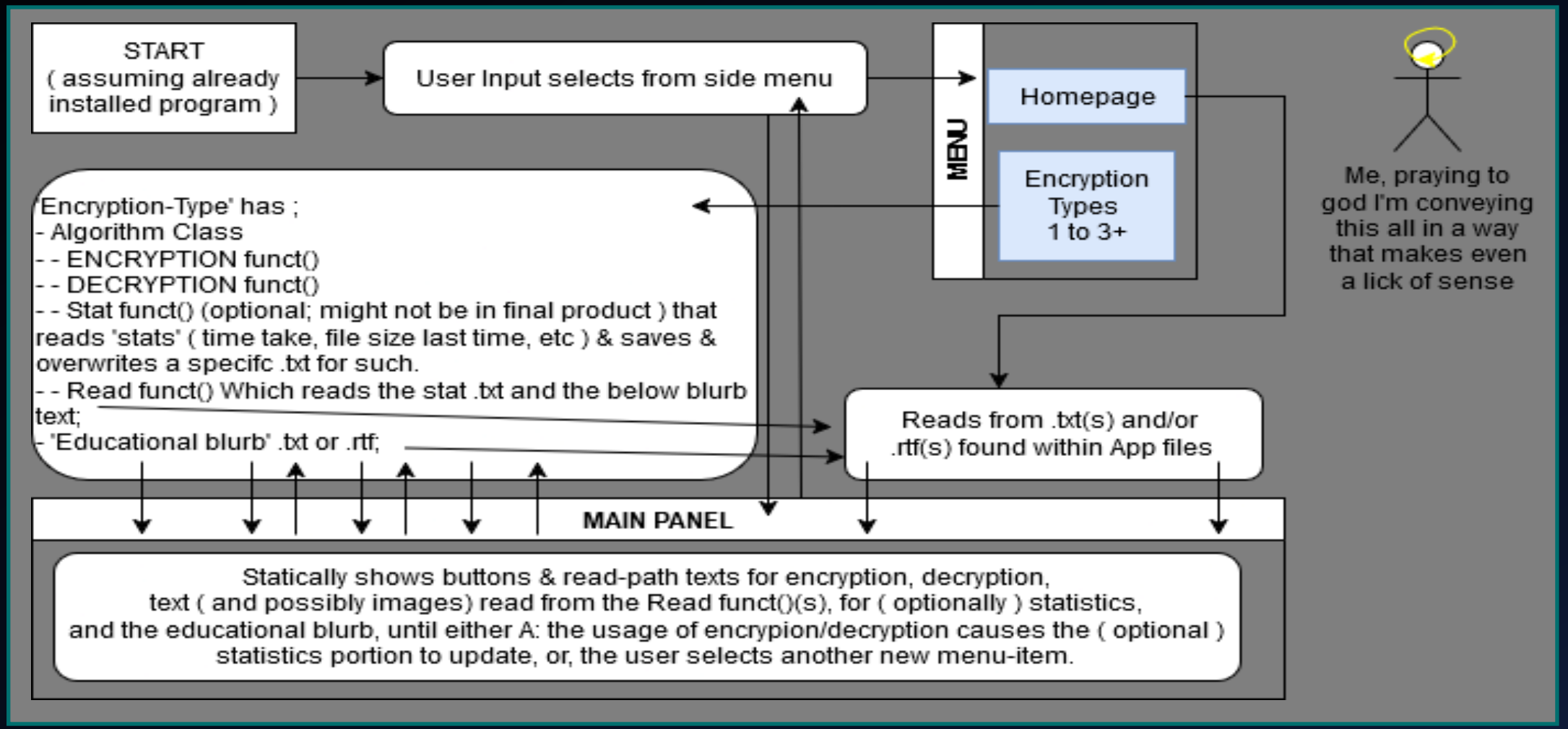


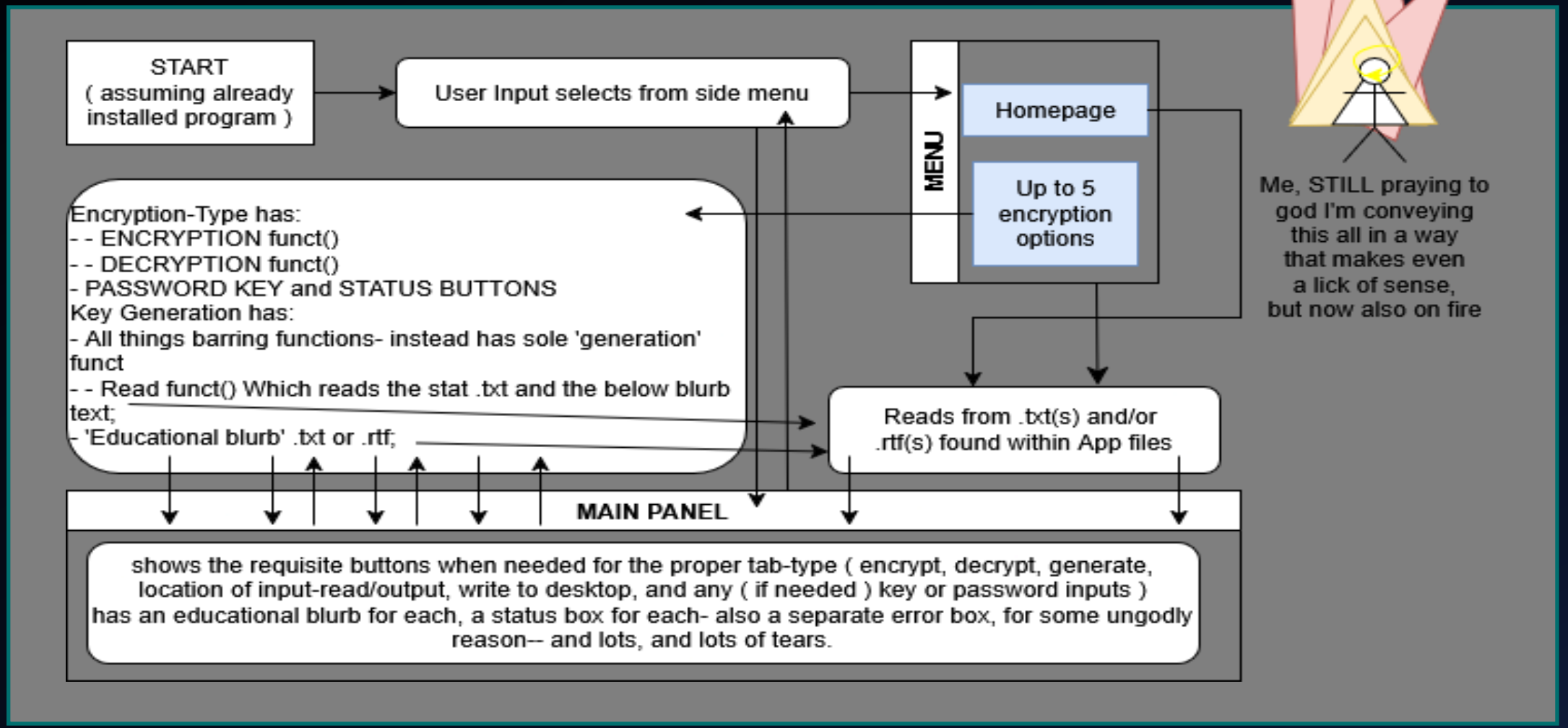

**Above: Key gen; Below; file content encryption**



# PROPOSED GENERAL APPLICATION FLOW; INITIAL PROJECTIONS



# PROPOSED GENERAL APPLICATION FLOW; CURRENT PROJECTIONS ( AND STATE )



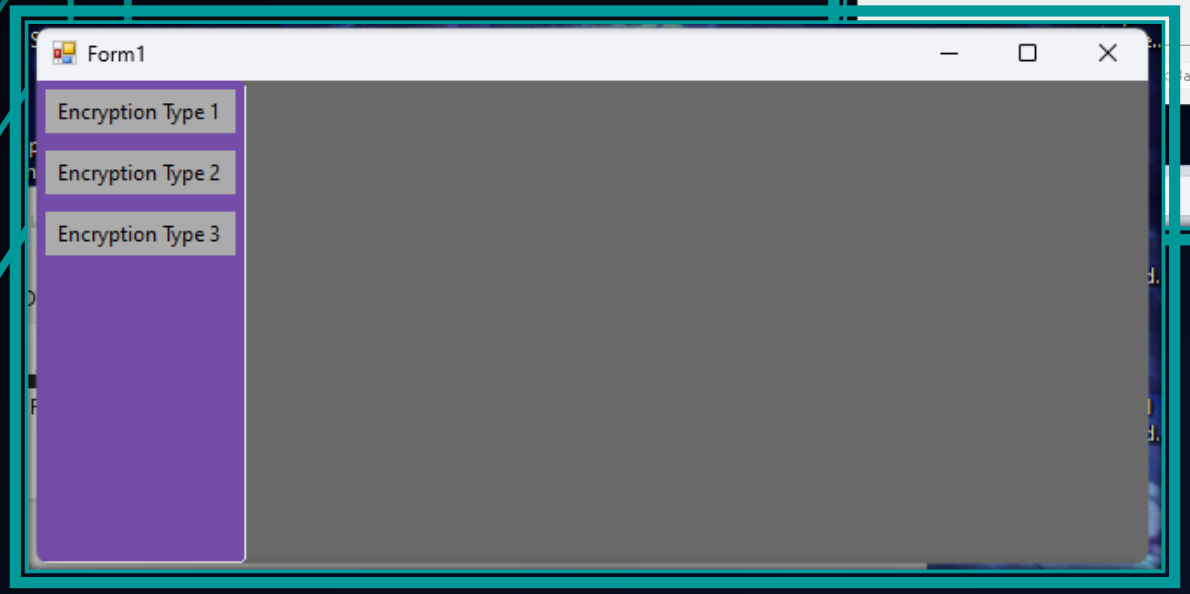


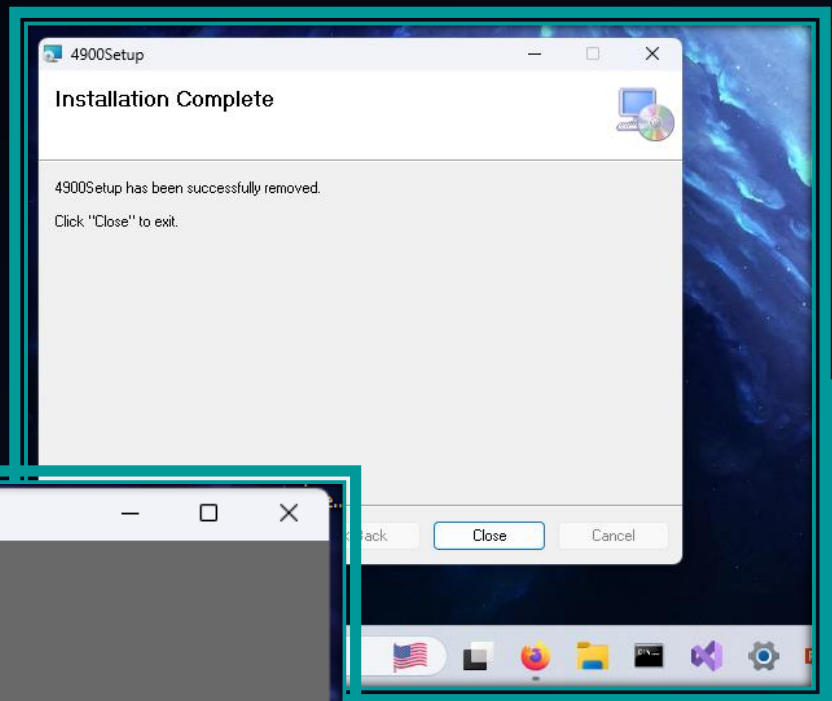

Me, STILL praying to god I'm conveying this all in a way that makes even a lick of sense, but now also on fire

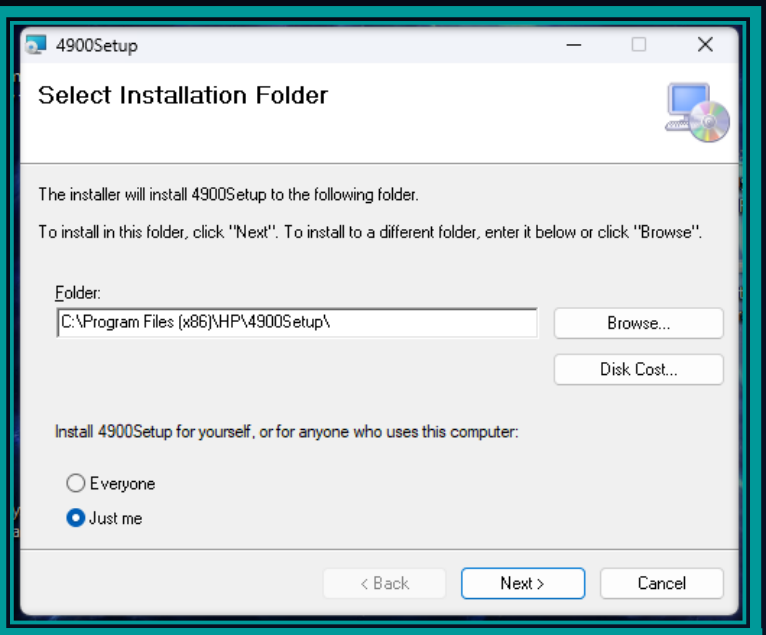
# SO FAR IMAGE INSTANCES;

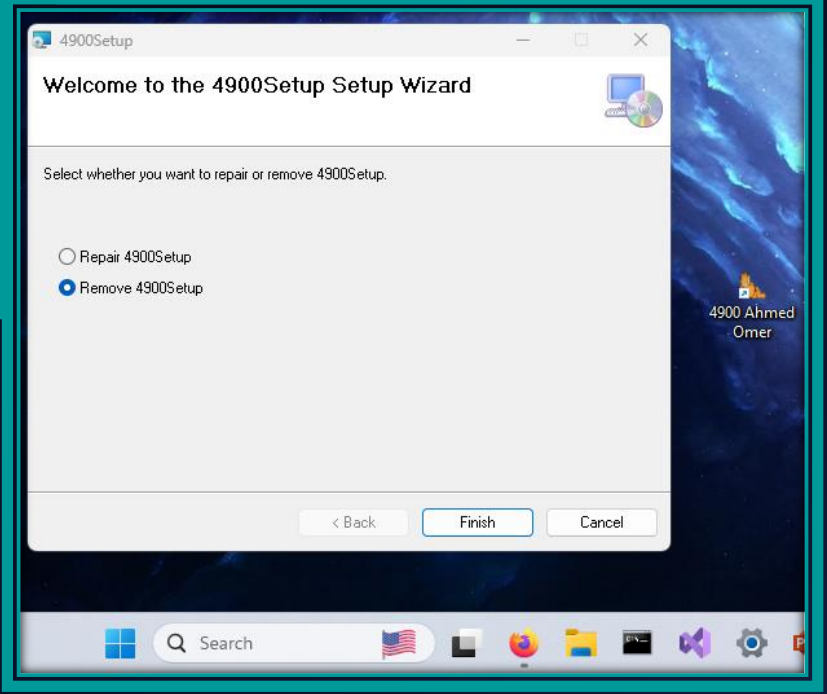
*Left to right;*

- Older version
- Install example
- Shown existence
- Uninstall example
- The growing UI of the app itself!

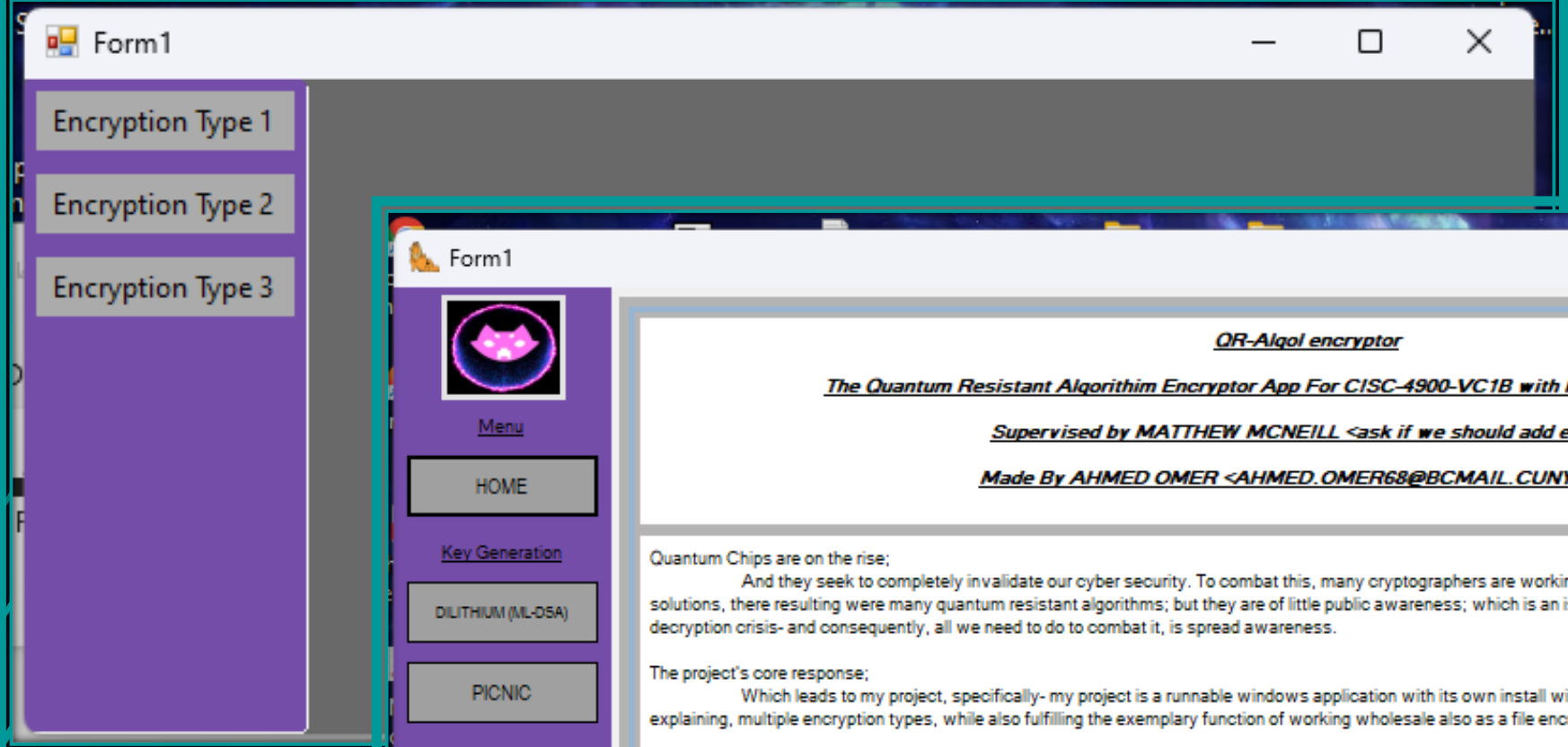


# BEFORE;




# AFTER;

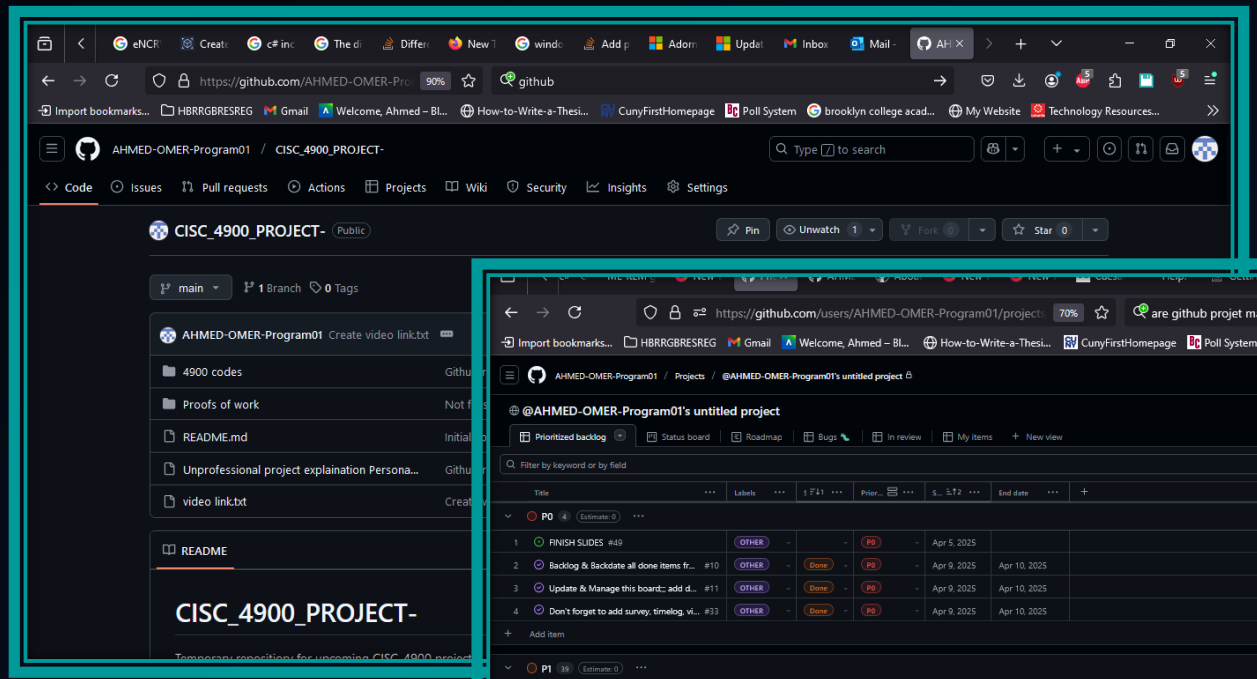
# SO FAR IMAGE INSTANCES(cont.);

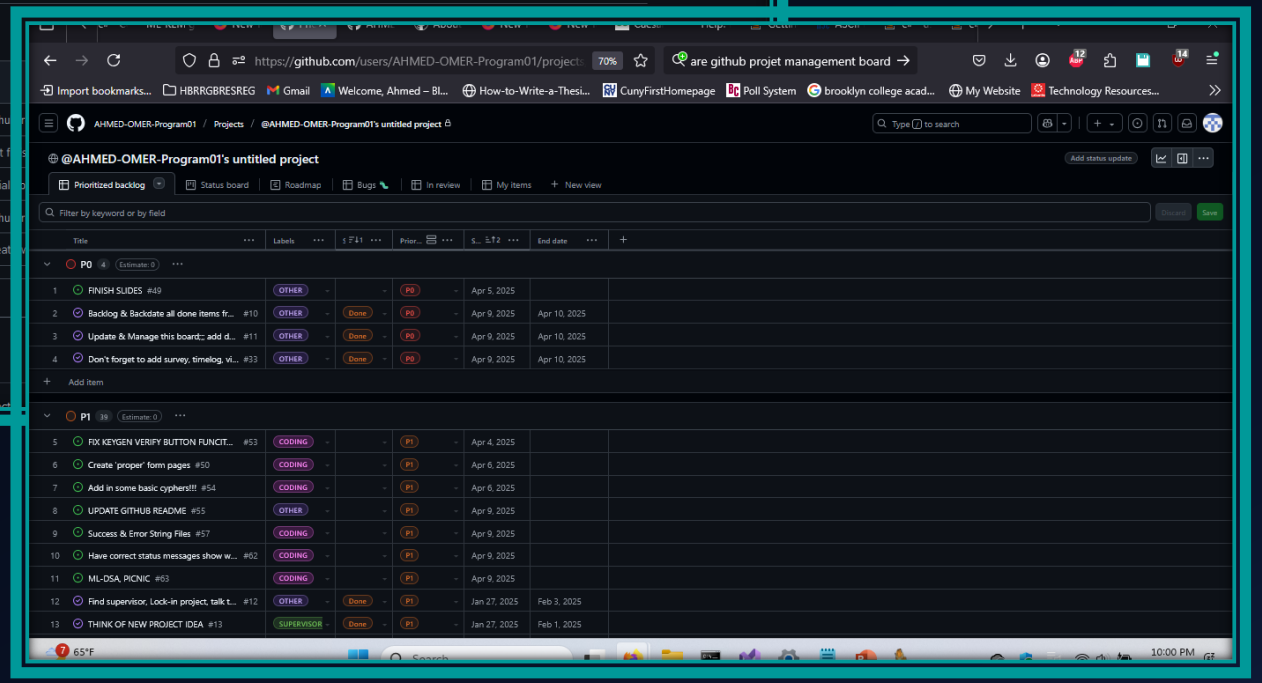
## *Notes;*

Github and the project management board! Oh jeez, those were doozies!

Since I'm more used to working on my own ( and this being a solo project ), I struggled to remember to update often to non-local. This was a tendency that that ultimately once cost me a bunch of progress, as one time in mid-Aprill attested!

Beyond that, I was more busy being admittedly stingy and spiteful to GPM for an unrelated prior experience, and mostly allotting my time regarding the internet and the project to research, and attempting to find help while doing bugfixing




Github Link;
https://github.com/AHMED-OMER-Program01/CISC_4900_PROJECT-
Github Management Board Link;
https://github.com/users/AHMED-OMER-Program01/projects/4/views/1

# SUMMARIZING; THE PROJECT, IN CHUNKS;

*In general;*

From the beginning, I have had split my project into three ‘general chunks’; each being expected to mostly take 4 weeks each, maximum. That way, for the last four weeks, I can focus more on general improvements, having already completed the focus-items.

In execution, however, while I kept to completing such items in order, I mostly found myself flying by the seat of my pants- in relation to outside issues and occurrences I’ll bring up later. Overall, I found myself often either vastly ahead, or vastly behind my planned schedule- and each main ‘chunk’ of the project- the application wizard, the encryption & decryption, and the ‘educational’ segments, all mutated over time on their own, too.

To begin with;;

# SUMMARIZING; THE PROJECT, IN CHUNKS;

*The application wizard, windows, and the App itself;*

For instance; the install wizard, and it's install capability was finished nearly instantaneously- leaving the remaining weeks of that first segment of the semester time for me to start a head-start lead.  During that time, I found myself a supervisor in Professor Mcneill, taught myself .NET frameworks and tendencies, and went from a state of not knowing how even build my project, to being able to completely redo it in a fraction of the time ( .NET wise ) if I needed to!

Yes- the application wizard, windows-installation, and the application itself? Those were easy- it was everything else, that was, tedious—

*The encryptors, encryption, and decryption(1);*

The second chunk, and the second chunk I wanted to focus on in order, was the ability to encrypt, and decrypt, in-of-itself- to give my application an *action* it can achieve, as well as meet the goals I both set for it- personally, and as-in the project plan. I wanted at *least* three algorithms- one to be quantum based, and safe to say, I knocked that portion of this out of the water- goings as to far as to add two entire more ones in- due to the whole, key-generation and signature verification, fiasco.

# SUMMARIZING; THE PROJECT, IN CHUNKS;

*The encryptors, encryption, and decryption(2);*

In regards to that, actually- was the, beginning, of my gripes with this chunk of the project. There were, and still really aren't, that many resources- especially such so to dive in head-first, without a primer. And to be fair- Cybersecurity *isn't* my major- and that's apart of the reason why I chose the quantum-encryption, and the encryption angle in general- to *learn*.

However, I greatly mis-underestimated the bounds of what I was getting into, I think it still shows! I feel like if cybersecurity WAS my major, I'd at least had doubled my algorithms present- if not at *least* attempt to recreate one of the more simple protocols- considering how a massive chunk of each standardized Post-Quantum algorithm has their schemes laid out and about, publicly.

No- seriously; Every competitor to NIST's PQC competition had completely open-source code- some even just straight up hosting to github- and I couldn't really read or understand a thing! I ended up just sticking more towards function-calls, and attempting to gleam what I could from things like government released pdfs. When I dream, however, I still see *"Module-Lattice-Based Key-Encapsulation Mechanism Standard"*

# SUMMARIZING; THE PROJECT, IN CHUNKS;

*The educational segment, and tidying up(1);*

And finally, what was yes, both the easiest, and simultaneously hardest, segment; writing the educational blurbs. No user testing. While I can't be bothered to showcase using a picture, my laptop is incredibly fragile, and I couldn't find a willing tester who owned windows.

Regardless- I am not a writer; it's why I didn't go for a CW degree- that being said; I *did* have fun. You see, throughout my semester, I put a lot of *work* into actually attempting to fundamentally understand what I was doing, and using- a lot of work I didn't actually legitimately log, just leaving to the wayside, actually. I didn't want every week reading- "attempted to read on ML-DSA for an hour while waiting in the clinic", "Tried and failed to find a new resource", "Watched an entire series by a professor on youtube while cooking dinner over the last week", etc—

So, overall-

I had a LOT of work put in- and next to no way to express it- no one in my family's in tech, and I'm self-aware enough that I'd choke, if I tried talking about it and explaining in person to like, a classmate or whatever ( also, who brags like that??? ). So- actually having a *vector* for all this stuff made the entire experience basically a free-time trip.

# SUMMARIZING; THE PROJECT, IN CHUNKS;

*The educational segment, and tidying up(2);*

**That being said;;;**

It then became hard- because I remembered a very, very vital fact.

I can't write for my life.

And so basically, my time writing all this out became another time-inefficient exercise, where I would write out something in/for about an hour, come back to it the day after, and then attempt to re-write the whole thing again.

Getting frustrated on a Saturday morning, I  eventually, I just brought the issue up to an old highschool friend on discord- and gave them all the files, and asked them to pick blind which version they thought looked best.

I thought they picked one I had recently written, and was feeling slightly better about. Halfway through the conversation, I realized they had, in-fact actually, started singing the praises

# SUMMARIZING; THE PROJECT, IN CHUNKS;

*The educational segment, and tidying up(3);*

of the first draft- since it had in-text diagrams for some of the blurbs ( the file-content tabs ), and presented themselves in a complexity order  order ( SHIFT was written with very basic words in mind, Vegenere-Beaufort built upon it, since it was basically just introducing a more complicated  way of shifting, again, and then AES stood as a middle-point; introducing the concept of BLOCKS in ciphers, and then diving headfirst into more detail- with my writing on ML-DSA and PICNIC being the, 'hardest' ).

And while I did not, and still *don't* agree with the notion- I trust their judgement more on these types of things- plus, by technicality, they served as my only actual technical 'test-user'.

They count, now- I'm counting them. No one else showed up- so, they got to chose.

And that's what happened with the educational segment. I was basically, a microcosm of the project as a whole, really- started fun, got frustrating, then basically got a sign from the universe telling me to stop.

# SUMMARIZING; THE PROJECT, IN CHUNKS;

*The educational segment, and tidying up(3);*

But that's besides the point- the project had finished- a bit later than intended- a diversion into things like SABER and KYBER took up entire swaths of time- and a bit unfinished- but even if no one else agrees, it's a miracle it finished. Not because it was hard, mind- but legitimately, I think I'm a bit more superstitious than before this semester now, since thing has been unreasonably haunted since the beginning, and no one can say I didn't do my darnest!

# TASKS AND CHALLENGES;;;;;;

*AHHHHHHHHHHHHHHHHHHHHHHHHHHHHH(1);*

But this wasn't all fun; I had a, LOT of issues throughout my semester; logistical, contextual, and seemingly *conceptually so*, it feels like, recently.

Throughout this semester, whenever I've seriously either attempted to plan out, or sit down, upon this project I am either interrupted, distracted, or straight up kept from doing so, due to logistical issues. Some of such including, but not being limited to:

- ALMOST all my data twice, because my hard-drive pseudo defaulted ( onedrive error wanted to delete everything, I hate windows )
- Having family members call me with emergency issues directly before I'm slated to have a meeting
- ACTUALLY losing all large chunks of my progress thrice, twice- once during the beginning of the semester, when I realized that C++ was a bad codebase, and switching to C#, with it's .NET implementation was wiser; and another time, moe recently ( last month(?) ), when I learned my the reason my computer was always so laggy was because it was always attempting to run & start a large, forgotten application continuously- but then when dealing with that, and deleting the folder I had then moved to my desktop, I didn't notice it also tagged two others- a mostly empty classfolder, and MY ENTIRE GODDAMN PROJECT AHHHHHHHHHHH Which set me back a time, as I had to start from the last github update again
- BouncyCastle being the only available library with consistent PQC implementation, but next to NO C# implementation introduction anywhere- leaving me to have to guess function-names as I've, so far, translated a BUNCH of stuff from their JAVA segments GOD WHY
- And much, much more. Hooray.

# TASKS AND CHALLENGES;;;;;;

*AHHHHHHHHHHHHHHHHHHHHHHHHHHHHH(2);*

But more seriously; I was having a hard time attempting to adjust post-quantum bouncycastle functions using damn context alone- and a decent set of time ago, I hyper-focused on attempting to get file-encrypting working on SABER, and call me amateur, it's fine since it's true, I hit a wall- HARD. And I kept *getting* at it, since when my brain sets up a path for me to follow, I tend to automatically chase it, least I feel intensely wring/unsatisfied.

Overall, this lead to a surprisingly lark waste of time, before I called it quits, and my supervisor, Professor Mcneill, advised me to put it aside, and focus on other items on my agenda.  I think ultimately, my issues was SABER was more or less a result  of all my frustrations with my project, my luck with it scheduling wise, even though I was actively attempting to deal with the work, and life conflicts.

And speaking back to Professor Mcneill for a bit- he might say otherwise, but he's helped me greatly on this project!

Well- not so much code-wise ( lol ), but advice wise, 10k/10 for sure! Throughout the semester, he's given me cheerfully straightforward advice- which helped give my project a more actionable direction, and curb my mistakes ( the How-to blurb, gave me advice to read up upon similar, but failing things ( Pretty Good Privacy ), etc ) and in general, was just. Normal? He was normal about all this- and I needed that, seriously- especially with how this project has gone so far.

# TASKS AND CHALLENGES;;;;;;

*AHHHHHHHHHHHHHHHHHHHHHHHHHHHHH(3);*

AND, to be fair- I've also learned a lot- I've learned the ins and outs of a lot of things! I've made my own application ( first time! ), and taught myself .NET frameworks. And, considering how little I knew to begin with, I went from not really knowing encryption & decryption even *entails*, to knowing the difference in zero-knowledge-proof approaches between MPC-in-the-head type methodology, and a more fiat-Shamir dynamic.

All such, you can read about in the application itself, to boot! Although- to be honest, I think I should've given the PICNIC blurb another look through.

But yes- I taught myself about windows applications, cryptography, *Quantum* cryptography, and revitalized my slightly out of practice C#. AND, that's not even mentioning the skills I gained unintentionally in project management, crisis management, pacing myself, self-structure, and overall attitude when revolving around what I can, and can't control.

I've learned a lot, this semester! And it's mostly all been surprising, to boot. For instance, did you know how AES, the standard introduced in 2000, mind you, was evaluated to be quantum resistant? It makes sense, considering the amount of block-shift and manipulation rounds that occur during AES-256, but still!

# TASKS AND CHALLENGES;;;;;;

*AHHHHHHHHHHHHHHHHHHHHHHHHHHHHH(4);*

And now, to finally close this, functionally  an academically sponsored rant segment, I want to talk about what I would want to do better, if I ever had to redo this project again, and what I would want to avoid, if I had to, again.

I think above all else- I would surprisingly want a partner. Doing this project on my own has really put to the line what I can, and cannot do- alongside what I can, and cannot effect. Alongside such, I've learned a few of my weaknesses- weaknesses I knew I had beforehand, but more now in more striking detail.

Without someone else involved to care about, I lag on updates- I tend to not see much a reason in them, even when I know they would be for me betterment, if I did- which is in stark contrast to past projects, in which I found joy in updating a subversion server, or what have you.

Also- I absolutely STINK at presenting things. Working on things, learning things, making and adapting on the fly a plan of action? I can do that- I can do that easy. I'd be angry all throughout, but--- I have before done so, and I can again.

# TASKS AND CHALLENGES;;;;;;

*AHHHHHHHHHHHHHHHHHHHHHHHHHHHHH(5);*

But it appears that, when it comes push to shove, to *show off*, of all things, I sorta flail a bit. Throughout my presentations and demo-recordings, I found myself either recording and rerecording, or fumbling up things besides. I like having my work speak for me, sometimes- despite how much of a running-mouth I am.

And having someone else on the team who could do that? Someone else, who's strengths could match my weaknesses? That would be a godsend- much less forgetting sharing the workload with someone else on this project really *would've helped oh my god*.

So, overall- If I could do one more thing different, I'd probably have roped in a friend or something into my project- I knew someone else doing this same semester- I should've reached out more.

And a thing I'd avoid would be not giving up and attaching Java to the project. I don't care if not doing so made my project less clunky, and I learned more. GOD, If I could-have used all those bouncy-castle api-examples off the cuff, I'd be *leagues* ahead of where I am now, yeesh!

# SUMMARIZING; THE PROJECT, IN CHUNKS; THAT ORGINAL PLAN;;;

*When, specifically, (did) you plan to accomplish these tasks(?);*

| | |
|---|---|
| The Application Wizard & it's basis; | weeks 3-5 |
| The AES & Lattice encryptor; | weeks 5-7 |
| Third encryptor find, & incorporation; | weeks 6-8 |
| The educational segment, & testing; | weeks 10-12 |
| Tidying up & adding anything extra | weeks 12+ |

These are the times I projected back before the first diagrams submission.
To quote;

*"You might notice these are different times than I projected for the Project-Proposal survey; this is because despite the simultaneous workings of finding, researching, securing, and setting up a project's idea, alongside the project itself, I managed to create the installer fairly quickly, thanks to a mixture of Microsoft's resources, and my willingness to switch language types ( just a jump from C++ back to C#; but the .NET basis for application-building has been so standard for so long, resources are abundant ).*

*This has been overall a major boon for my projected *best* times, since now, average case, I have some wiggle-room for time-sensitive project emergencies!"*

# SUMMARIZING; THE PROJECT, IN CHUNKS; WHAT HAPPENED;;;

*How these tasks worked out;*

| | |
|---|---|
| The Application Wizard & it's basis; | weeks 3 |
| Encryption 7 Decryption algorithms | weeks 4-12-13 |
| The educational segment, & testing; | weeks 13-14 |
| Tidying up & adding anything extra | NO TIME |

In the end, however, things ended up like this!

I don't *mind* this- considering how much more I accomplished with the project then I ever initially expected or thought to accomplish- but it's personally unsatisfying, besides.

Regardless- I always have the files, and can refurbish whenever I wish to- so, in retrospect, I think I spent my time on my project well.

I'll just- be coming back to it later- perhaps when I'm, luckier.

# HELPFUL LEARNING RESOURCES;;;;;;

## OTHER PLACES;

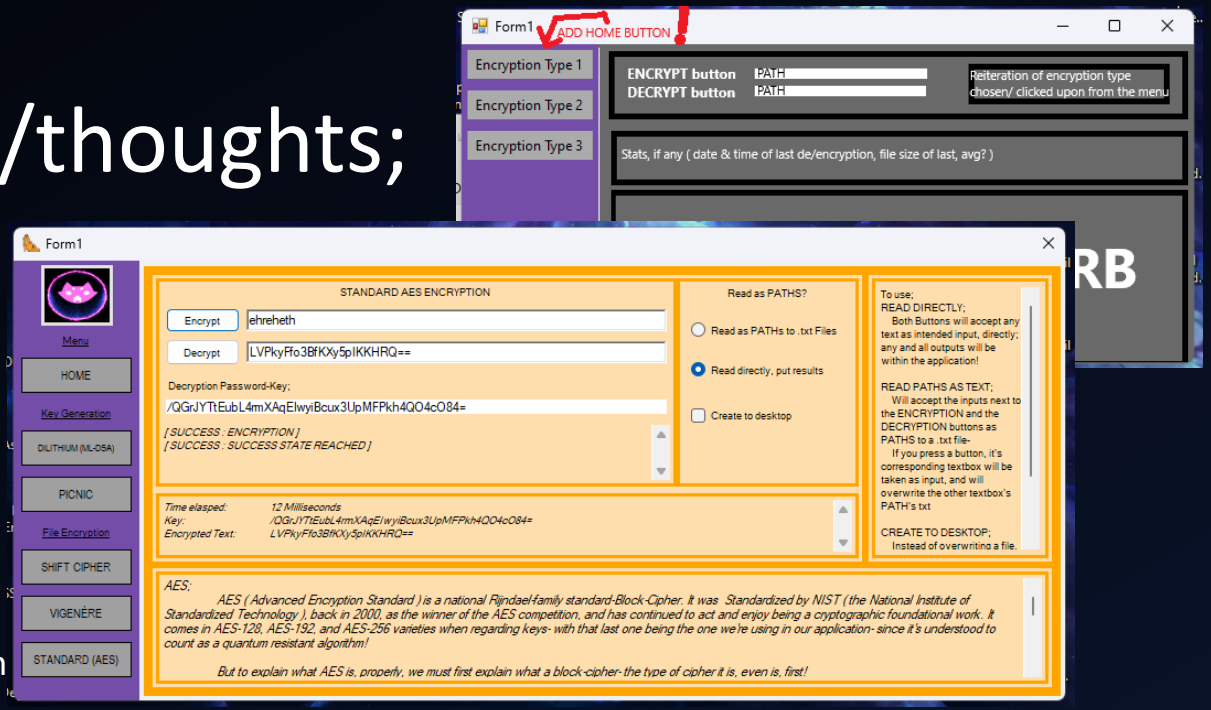
- Microsoft's own .NET documentation site
- Cornell University's papers (ex:"**A Tutorial Introduction to Lattice-based Cryptography and Homomorphic Encryption"**)
- Crystals/Dilithium's main site; https://pq-crystals.org/dilithium/index.shtml
- Asecuritysite, for giving me closure, albeit literally only as a taunt on your end; https://asecuritysite.com/
- **The Knowledge Complexity of Interactive Proof Systems, for getting me through the door; https://people.csail.mit.edu/silvio/Selected%20Scientific%20Papers/Proof%20Systems/The_Knowledge_Complexity_Of_Interactive_Proof_Systems.pdf**
- **Bouncycastle's JAVA API, when you're doing JAVA implementation**
- **Stackexchange, for helping me get my foot into the door of .NET when Mirosoft's page was outdate**
- And the many, many cryptography blogs I went through attempting to figure out the Post-Quantum Competition;( ex; https://blog.cr.yp.to/20250118-flight.html )

## AND NIST, IN ALL IT'S GLORY

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

HTTPS://WWW.NIST.GOV/

# Some quick final portions/thoughts;




*specifically, Uses for this project That I accomplished;*

- As a regular file encryption, decryption program for windows examples of such include WinRAR, 7-Zip, DiskCryptor, etc )
- A reasonable KEY generator program- an actually rare thing not just within a website- and since the program ALSO can *use* said keys directly for encryption in other tabs- makes it easier to secure your files!
- As a Working example on your windows machine of Quantum Resistant encryption, decryption; plus, since consumer versions aren't common, slightly more obtuse file-protection for the average user who choses to do so- an actually legitimate application selling point
- And as an encryption educational tool! Both as a fun little read-though for those curious just a download away- but considering the nature of github, also as a tool via my own code, and my own attempts at learning! To the github, I also pushed some pdfs I found and downloaded, and some JAVA codes I wrote, so I could work offline and still reference things– so, there's my gift back to the world, there!

*Final thoughts;*

This has been fun so far! As I've mentioned ( plenty ) prior, this was majorly a learning project for me- and I've learned a lot! Both code-base wise, and project management wise! So, I just want to say two things;

Thank you for the chance, and for this experience! It's been a fun semester, professors!

Cheers!

# Quantum Resistant Algorithm Encryptor -END

*FOR CISC-4900-VC1B WITH PROFESSOR CHUANG*

*SUPERVISED BY MATTHEW MCNEILL*
*<MCNEILL@BROOKLYN.CUNY.EDU>*

*BY AHMED OMER*
*AHMED.OMER68@BCMAIL.CUNY.EDU*

*GITHUB PROJECT LINK:*
*HTTPS://GITHUB.COM/AHMED-OMER-PROGRAM01/CISC_4900_PROJECT-*

*PROJECT MANAGEMENT BOARD;*
*HTTPS://GITHUB.COM/USERS/AHMED-OMER-PROGRAM01/PROJECTS/4/VIEWS/1*