Final PP/SRS/SDS
5/31/21
Group 3:
Joshua Fawcett, Max Hopkins, Meghan Riehl, Yuyao Zhuge

Project Plan
Vigenere Cipher Visualization Tool

Overview:

Our goal is to create a teaching/learning aid for Professor Joe Li to use in his class on cryptography. This tool will be able to visualize the steps necessary to encrypt and decrypt text using the Vigenere cipher by animating the steps of the encryption process. The user will be able to control the speed as well as step through the steps manually. This system was designed to be used in class as well as something a student could use at home for more exploration.

Structure and Communication:

Our team lead is Joshua Fawcett, though as a small team decisions will be made as a group.

Tasks will be assigned to all group members either individually or as pairs. These assignments will be recorded along with their completion dates within a google spreadsheet document that all team members have editing privileges for.

Task completion will be reported to and confirmed by at least one other member of the team, preferably the team lead.

Our team will meet at least twice a week:
- Tuesdays @ 12:00 PM
- Fridays @ 3:00 PM
- Extra meetings to be scheduled as needed

The team will meet via zoom so that we can talk face to face and collaborate on implementations and problem solving. Day to day communication and general updates will be handled through a discord server where team members can talk via instant messaging and voice chat channels as needed.

Tuesday and Thursday scheduled meetings will have a mandatory attendance while extra meetings will not though all members who can are expected to participate. Team members will report on the completion of their assigned tasks either before or during a scheduled meeting so that other team members can confirm and double check their work.

The individual modules and codes will be kept on a GitHub repository where all team members can access all parts. Any and all edits will be recorded by GitHub at the time of the edit, including adding and removing items from the repository.

Tentative Build Plan:

1. Interviewing Professor Joe Li (5/06)
2. Research
    a. Visualization tools
    b. Existing research on algorithm visualization
    c. Existing software
3. Cipher functionality
    a. Encryption/ Decryption
4. Visualizations
    a. Pacing tool
5. User interface
    a. Executable file
6. Instructions/ Information
    a. Using the tool
    b. Cipher information
7. Additional functions if time allows
    a. Cryptanalysis (breaking the cipher)
    b. Knowledge check and testing

Build Plan Rationale:

We will start with researching various articles on algorithm visualization and existing cipher visualization tools in order to assist us in the process of creating requirements and designing our system. Then we will get started with the basic functionality of the system, with each of the separate modules being created in parallel by separate group members. This will allow us to make progress quickly towards creating an initial prototype. Our initial implementation of the system will include only the essential features such as cipher functionality (encryption/decryption), cipher visualization, and the user interface. After developing a working initial prototype then we will revise our system and add some secondary features.

(1) Initially Meghan and Yuyao looked into what library or coding system that would best suit the needs of an interactive animation of the cipher while Max and Josh looked into more specific requirements for a teaching and learning tool. (2) The work was then split depending on proclivity and knowledge, Max working on the base game functionality and Meghan helping and building some sub menus and buttons. Josh built the algorithms then assisted Yuyao with the visualization/animation. (3) The individual pieces were then brought together and bug tested. Menu buttons were mapped to the correct menus, control buttons were given functionality and extra information was added for the user.

Risks:

1. Expected
   a. Lack of knowledge
   b. Poor communication
   c. Delayed assigned tasks
   d. Time restraints
2. Unexpected
   a. Emergencies (medical, family, natural disasters)
   b. Hardware/ software malfunctions (computer breaks, internet goes down, etc.)
   c. Other unforeseen problems

1.
(a) No one was expected to know what the specific requirements of the system were going to be and as such no one was expected to know how best to meet those requirements.

We all needed to learn:

-Who this system would benefit

We contacted Prof. Joe Li who teaches the applied cryptology class and interviewed him on the necessities of his class and the type of system that he could use.

-User requirements

We got initial requirements from Prof. Joe Li during the interview, we also met with him a second time to show our progress and see if there was anything that was missing. We also did some research about algorithm visualization as a learning tool to get other requirements.

-Pygame

We found a book about the library and as a popular game maker there was a lot of other information, tutorials, and examples to be found online.

-The Vigenere cipher

We talked to Prof. Joe Li as well as researched about this particular cipher.

(b) Communication is difficult during a pandemic where all communication needs to be through a computer screen and not in person. We worked on keeping each other updated as we worked on and completed tasks. We asked for help when necessary and met via zoom and saw each other face to face at least twice a week. We also kept meeting notes (see below) and an active spreadsheet of assigned tasks and other project requirements that were constantly updated and could be referred to as needed.

(c) Any tasks that were part of the critical path (see the Gantt chart below) needed to be completed in an orderly and timely fashion. If there were delays in these necessary components we worked together to ensure the delay was as minimal as possible and we could still complete the system ontime.

(d) As students we all had other classwork as well as jobs and other demands of our time. Unexpected emergencies could also limit the amount of time a particular team member might have to work on the system as well. We worked around this by being flexible with what was assigned to whom and worked together to assist each other when due dates were coming up.

2.

With all unexpected risks there is no one correct way to plan for and prevent them. They can all be worked around with flexibility and understanding by the team as a whole. Some emergencies are unavoidable and happen too fast to predict, others can be seen coming and can be mitigated. Not everyone can have reliable setups but we all know there are computers available at school and we kept our documentation, notes, code, and other shared materials on various servers accessible by everyone (Discord, GitHub, Google Drive). While some personal progress might be lost the project as a whole would not be.

Meeting Notes

Last Updated: 5/31/2021

These will be the meeting notes for our meetings, and the unique topics that pertain to those meetings. Joshua will lead the meetings but we will take turns on taking notes during meetings. We will start with Max, and rotate down the group members alphabetically by last name. Josh is exempt from notes in order to focus on leading meetings.

The notes will follow the general format of:

- Date, attendance, time
- Topics to discuss and whether or not they were, indicated by a check mark
- General notes (bugs, additions, ideas, thoughts)
- Individual progress (what was worked on, what is next, what is holding you back)

Meeting Notes for 5/28/21 at 3:00

Attendance: all
3:00 - 3:50

Topics to discuss
_v_ Progress and next steps

Notes:

- Change color of pause to red when activated, color of play to green when activated
- Add one button for pause/play
- Add indicator for what speed you are on
- Split button into visualization and testing (if time permits)
- Error message on screen
- Format buttons correctly
- Limit input (prevent overlap with letters and table)
- Fix issue with screen being black when changing screen size while animation is paused

Max - Not too much progress since last meeting. Next: menus set up (all buttons between menus) (use menu, …), Holding Back: Nothing of note

Josh - Integrated table into the main program. Added pacing functionality (pause/play, step forward/back, speed up/down, restart). Next: Help with other tasks. Holding back: No

Yuyao - Not too much progress since last meeting. Next: Print current message, keyword, and result to screen and implement highlighting specific letters. Possibly help with other tasks. Holding Back: No

Meghan - Not too much progress since last meeting. Got some stuff typed up. Next: get it into the scene manager. Holding back:

Meeting Notes for 5/26/21 at 3:45 - Meeting with Prof. Joe Li

Attendance: all + Professor Joe Li
3:50 - 5:00

Topics to discuss
_√_ Talk about prototype
_√_ Get feedback from professor Li
_√_ Progress and next steps

Notes:
To make better:

- indicate which line is ciphertext, which is key (Label what is what)
- more descriptions, which color is for what
- Describe which process is currently being shown
- One more thing to suggest: Vernam cipher (1 time pad? 1 time pass?) (if time permits), if done in time can possibly show in next class for applied cryptography,

Max - work on next: Work on some more menus and integration between the scenes

Josh - update pacing, using values from encrypt/decrypt to dynamically highlight correct values

Yuyao - Animation, cipher/table integration

Meghan - Info write up and formatting, project plan paperwork

Meeting Notes for 5/25/21 at 12:00 - Group Meeting

Attendance: all
11:45 - 1:00

Topics to discuss
_√_ Check up on everyone's progress

Notes:
Josh - worked on pacing and visualization highlighting. Next: work on integration

Meghan - buttons for animation scenes, work on pausing game

Yuyao- Made letter grid, looked at highlighting stuff, Next: focus on animation, not sure what next task is, focus on integration

Max - Worked on changing between menus, and

Meeting Notes for 5/21/21 at 3:00 - Group Meeting

Attendance: Max Hopkins, Meghan Riehl, Yuyao Zhuge

Topics to discuss
_v_ Check up on everyone's progress

Notes:

- Don't use a slider for speed, use buttons.
- Speed up/speed down buttons will display a number on the screen showing your speed.
- Need step forward and step back as well.
- Main menu will have input/start buttons, it will load the visualization menu.
- Josh could help Meghan with her next task over the weekend.
- Possibly CC Hornof to the prototype meeting.
- Need to have a prototype done by 5/24, some slack time for a meeting on Wednesday.
- Send an email to Joe Li by Monday.

Josh: Worked on: Created Cipher.py with basic encrypt/decrypt functionality. Started basic research about pygame to help with understanding of each module. Next: update Cipher.py? Maybe help implement other modules? Holding Back: Not sure what I need to do next, waiting for progress on other parts to see how Cipher.py might need to be modified, or if there are other parts of the project that I need to help with.

Meghan:worked on: Pseudo code, learning pygame. Next task: add mouse logic/buttons for the speed up/speed down buttons. Holding back: TIME.

Yuyao: Worked on table/grid. Next task: Update table to include letters and highlighting. Holding back: Time.

Max:worked on: Setup test menu/learn pygame. Next task: Setup the main menu to have inputs and load the next screens. Holding back: Other assignments/time.

Meeting Notes for 5/18/21 at 12:00 - Group Meeting

All in attendance
11:45 - 12:20:

Topics to discuss
_√_ pygame yes or no? Yes

_√_ Talk about individual progress

- What are your responsibilities on the project?
- What have you done in the last few days?
- What is your next task?
- What, if anything, is holding you up from accomplishing that task?

_√_ Discuss system building plan (What you'll build and how you'll build it)

Notes:
-Max's research: important to control pacing, being able to rewind, hypertext.
-If not using a flask server: how will we handle user inputs? PyGame has options to build an interactive menu.
-How could we best deliver the final product to Prof. Joe Li? Ask him
-Find and try games built by pyGame to test/learn how it works from a user standpoint
-What information needs to be sent from the algorithm to the visualization? Have a separate animation code that doesn't necessarily talk to the algorithm?
-Divide visualization into: animation, time control, user menu
-Different colors for key, plain text, encrypted highlight
-Make only one prototype so there's more to present to Prof. Joe Li (Target- 5/24)

- Takes in user input ★MH
- Provides proper encryption/decryption ★JF
- Can highlight table (hopefully text too) ★ YZ
- Some form of pacing (stepwise or controlled continuous) ★ MR

Yuyao: Worked on: Looked at other visualization tools, leaning toward pyGame as the final decision. Considered other ways to use different models. Found other sources, sent questions to others about how to use pyGame. Contacted creator of the other software for Vigenere cipher. Next: Communicate with contacts and creating a table in pyGame, learning how to do the animation. Holding back: learning pyGame.

Meghan: Worked on: Reading more about pyGame, how to use it and thinking about how to use specific functions for what we need. Considered memoization code for saving steps. Next: learn pyGame and how to translate encryption/decryption steps so pyGame can animate them. Holding back: time, working on other classes, learning pyGame.

Max: Worked on: looking into alternatives to flask server, created Github Repository. Programs that can compile the pygame files into an executable file we can send out. Found more citations for design specifications, with quotes. Next: Work on making a decision about which visualization tool to use. Holding back: learning pyGame.

Josh: Worked on: Researched best ways to implement encryption and decryption in python. Next: Get a working model of the encryption/decryption code. Holding back: time, working on other classes.

<u>Meeting Notes for 5/14/21 at 3:00 - Meeting with Prof. Hornof</u>

Additional Persons: Anthony Hornof
All in attendance
3:00 - 4:20

Topics to discuss
_√_ Organize meeting document
__ Review timeline plan
_v_ Discuss each individual member’s progress
- What are your responsibilities on the project?
- What have you done in the last few days?
- What is your next task?
- What, if anything, is holding you up from accomplishing that task?

__ Discuss interactions with Professor Li
__ Discuss system building plan (What you’ll build and how you’ll build it)

Notes:
Yuyao: responsible for collecting information about how to do the visualization part. Just got started and found some helpful resources that we can use by using python, especially modules called matlib, pygame, dash, barchartrace. Next task: look more into it. Try to see if can plot table format based on libraries (evaluating libraries for ease of use/practical). Holding up: technical stuff: knowing how to plot table and how to do step by step animation. Suggestions: Meghan’s book, save state for each step, pattern

Meghan: responsible for looking up visualization tools/techniques to show the vigenere cipher. Read through how pygame works (pygame book), best way to get interactivity from users (so far), meet with Yuyao next to discuss findings. May experience time constraints for the next task.

Max: responsible for findinding/reading research papers and getting the flask server set up with input fields. Has been working on SRS, SDS, got one paper quoted with a citation. Next, getting another citation or two then setting up a server with plain text and key input fields. Expressed being able to get a lot done over the weekend.

Josh: Responsible for researching articles to find more design specifications, needs to get started on cipher functionality (not visualization). Look into cryptanalysis if time permits. Hasn’t had much time the last couple days, but has been able to look at a couple articles so far. Look into integrating cipher functionality with other modules nicely. Time constraints due to other classes.

Suggestions for leading meetings:
- Ask for specific answers
    - Be persistent
    - Rephrase questions
- Clarify Understandings
    - Did that help?
    - Did that answer your question?
- Lead the Discussion (with leading questions)

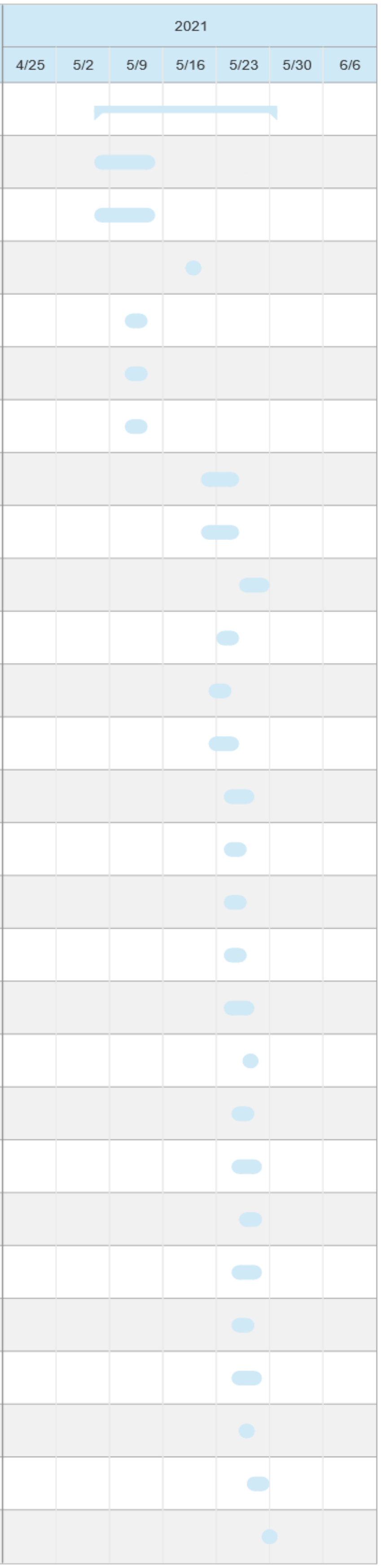
Action Items:
__ Joshua will find software pattern

Hard to make progress if tasks are too big. True for all projects. Specific tasks

please continue like this
figure out when to meet - “nice to meet in zoom with video” - “helps with group cohesion”
Come up with agenda items
when you look at the papers… looking for ideas for specific things to add to system/design based on what does/doesn't work.
Flask - do you need a flask server? Focusing on the visualization aspect. Go to the stakeholder: maybe he doesn't want a flask server?

Timeline and Assignment Spreadsheet

	Task	Assigned To	Start	End	Dur
	Vigenere Cipher Visualization Tool Project	JF / MH / MR / YZ	5/7/21	5/30/21	24
1	Cipher: Basic Encryption	JF	5/7/21	5/14/21	8
2	Cipher: Basic Decryption	JF	5/7/21	5/14/21	8
3	Cipher: Testing Encryption/Decryption	JF	5/19/21	5/20/21	2
4	Research: Visualization	JF / MH	5/11/21	5/13/21	3
5	Research: Visualization Tools	MR / YZ	5/11/21	5/13/21	3
6	Research: Existing relevant Software	JF / MH	5/11/21	5/13/21	3
7	Menu & Buttons: Start Menu	MH	5/21/21	5/25/21	5
8	Menu & Buttons: Main Input Menu	MH	5/21/21	5/25/21	5
9	Menu & Buttons: Information	MR	5/26/21	5/29/21	4
10	Menu & Buttons: User Input	MH	5/23/21	5/25/21	3
11	Menu & Buttons: Animation	MR	5/22/21	5/24/21	3
12	Game Functionality: Table Display	YZ / JF	5/22/21	5/25/21	4
13	Game Functionality: Text Display	YZ / JF	5/24/21	5/27/21	4
14	Game Functionality: Stepping	YZ / JF	5/24/21	5/26/21	3
15	Game Functionality: Play/Pause	YZ / JF	5/24/21	5/26/21	3
16	Game Functionality: Restart	YZ / JF	5/24/21	5/26/21	3
17	Game Functionality: Speed control	YZ / JF	5/24/21	5/27/21	4
18	**Milestone: Prototype**	Everyone	5/27/21	5/27/21	1
19	Animation: Ciphers with table & Highlights	YZ / JF	5/25/21	5/27/21	3
20	Animation: Text Highlights with table highlights	YZ / JF	5/25/21	5/28/21	4
21	Animation: Control button functionality	YZ / JF	5/26/21	5/28/21	3
22	Integration: Integrate main input menu and visualization scene	MH / JF	5/25/21	5/28/21	4
23	Integration: Integrate main input menu and cipher functionality	MH / JF	5/25/21	5/27/21	3
24	Integration: Integrate Table and Pacing modules	JF / YZ	5/25/21	5/28/21	4
25	Integration: Integrate all Info & Menu scenes	MH / MR	5/26/21	5/27/21	2
26	Integration: confirm navigable	MH	5/27/21	5/29/21	3
27	Testing	Everyone	5/29/21	5/30/21	2



Project 2 Timeline

Group: **Group 3**

Team Members (and initials): **Joshua Fawcett (JF), Max Hopkins (MH), Meghan Riehl (MR), Yuyao Zhuge (YZ)**

Date this was last updated: **5/31/21**

Last updated by (whom): **Everyone**

key:

Begin By	Started
Due	Slack Time
Done-MR	Overdue
Special	

* If a task is started on the begin by date, change only the cell color to: 🟩
* If a task was completed on the due date, change only the cell color to: 🟩
* A task completed within slack time is not considered overdue
* A special cell is something not covered by the other options, delays, changes, etc. Please provide more information in the cell.
* When a task is completed another team member will initial the cell confirming completion (EV = Everyone)

Grouping	Task or Milestone	Assigned to	5/5	5/6	5/7	5/8	5/9	5/10	5/11	5/12	5/13	5/14	5/15	5/16	5/17	5/18	5/19	5/20	5/21	5/22	5/23	5/24	5/25	5/26	5/27	5/28	5/29	5/30	5/31
Project Planning					1hr x 4			50min x 4	55min x 3	2.5hr x 4		1.5hr x 4				35min x 4			20min x 3				1:15 x 4	1:10 x 4		50min x 4			
	Group Meetings	Everyone			3:00 PM			1:30	12:00 PM	4:30		3:00 PM				12:00 PM			3:00 PM				12:00 PM	3:50 PM		3:00 PM			12:00 PM
	SRS	Everyone				Begin By		Started		Initial	Revisions														Final				Done-EV
	SDS	Everyone				Begin By			Started	Initial	Revisions														Final				Done-EV
	Project Plan	Everyone				Begin By			Started	Initial	Revisions														Final				Done-EV
	Programer Documentation	Meghan/ Josh									Begin By														Due				Done-MH
	User Documentation	Yuyao/ Meghan									Begin By														Due				Done-JH
	Read Me	Josh/ Max									Begin By														Due				Done-YZ
	Installation Guide	Max/ Meghan									Begin By														Due				Done-EV
	Presentation Power Point	Everyone																							Begin By				Due
Cipher Functions																													
	Basic Encryption	Josh												Due		Done (1hr													
	Basic Decryption	Josh												Due		Done (1hr													
	Testing Encryption/Decryption	Josh														Due	Started	Done-MH											
Research																													
	Visualization	Josh/ Max															Due												
	Read/ Summarize Papers	Josh/ Max							Started		Done-MR						Due												
	Citations	Josh/ Max							Started		Done-MR						Due												
	Visualization Tools	Meghan/ Yuyao																											
	PyGame/Dash	Meghan/ Yuyao														Done-JH													
	Integration with user interface	Meghan/ Yuyao														Done-MH													
	Download package/ website application	Meghan/ Yuyao														Done-MH													
	Existing Software research	Josh/ Max							Started		Done-MR						Due												
Game functions																													
	Menus/ buttons																												
	Start Menu	Max																								Done-JH			
	Main Input Menu	Max																	Started				Done-YZ						
	Information	Meghan																						Started			Done-JH	Due	
	User Input	Max																			Started		Done-JH						
	Animation	Meghan																	Done-JH										
	Animation																												
	Table	Yuyao/ Josh																		Started			Done-MH						
	Stepping functionality	Yuyao/ Josh																				Started			Due-MH				
	Play/pause functionality	Yuyao/ Josh																				Started		Done-MR					
	Restart functionality	Yuyao/ Josh																				Started		Done-MR					
	Speed control	Yuyao/ Josh																				Started		Done-MR					
Prototype		Everyone																				Due		Done-MR					
Integration																													
	Integration																												
	Integrate main input menu and visualization scene	Max/Josh																	Started				Done-YZ						
	Integrate main input menu and cipher functionality	Max/Josh																	Started				Done-YZ						
	Integrate Table and Pacing modules	Josh/Yuyao																						Started		Done-MH			
	Integrate All Info/Menu Scenes	Max/Meghan																					Started			Done-JH			
	Confirm navigable	Max																	Due							Done-JH			
	Animation																												
	Ciphers with table/ hightlights	Yuyao/Josh																				Due	Started		Done-MH				
	Text Hightlights with table hightlights	Yuyao/Josh																				Due	Started			Done-MH			
	Control button functionality	Yuyao/Josh																				Due		Started	Done-MR				

Software Requirements Specification
Vigenere Cipher Visualization Tool

Table of Contents

1. Revision History

Date	Author	Description
5/11/21	JF, MH	Document created
5/30/21	JF, MH	Document revised for updated system requirements
5/31/21	JF, MH, MR, YZ	Final revisions for submission

2. Concept of Operations

2.1 Overview

The proposed system is a Vigenere cipher visualization tool. It will allow users to encrypt and decrypt using the Vigenere cipher and see how the cipher works during each step. The goal of this software is to provide an adequate teaching/learning tool for Professor Joe Li to use in his cryptography class to help students learn how to use the Vigenere cipher. We will try to accomplish this goal by deriving requirements for the system from existing Algorithm Visualization research and implement the features of the system based on these requirements.

Problem Statement: We need to design a system to allow visualization for the Vigenere cipher. The system will be able to encrypt a plain text using the vigenere cipher and decrypt an encrypted text using the vigenere cipher. Each step of the encryption and the decryption process will be visualized to produce an understanding of how the cipher works in order for Professor Joe Li to teach the cipher to his students.

2.2 Current system or situation

Currently there are tools out there that are able to effectively encrypt/decrypt using the Vigenere Cipher. However, there aren't any effective tools for visualizing the steps of the Vigenere Cipher that are satisfactory to our stakeholders (Professor Joe Li). Being able to effectively visualize the steps of the cipher would allow instructors to give students a deeper understanding of how the cipher works.

2.3 Justification for a new system

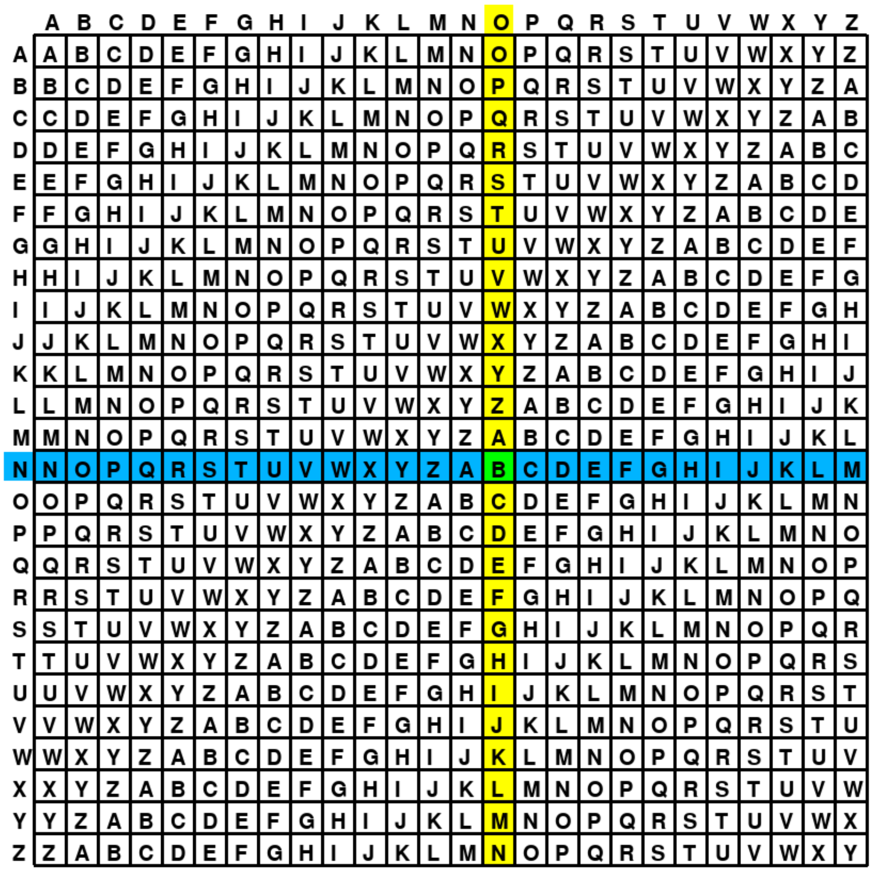
As described in the previous section, there are no satisfactory visualization tools for the Vigenere cipher. Creating a new visualization tool for this cipher will allow a greater understanding of the underlying steps for the cipher and how it works. This could be used by instructors to display to students in order to facilitate this deeper understanding.

2.4 Operational features of the proposed system

The proposed cipher visualization tool is designed to effectively display each of the steps that occur in the encryption/decryption process for the Vigenere cipher. One of the easiest ways to encrypt/decrypt using the Vigenere cipher is to use something called the Vigenere table (which can be seen in Figure 2.4.1). The encryption process uses the plaintext letters provided by the user as indices into the columns of the Vigenere table and uses the keyword letters provided by the user as indices into the rows of the table. The intersection between the specified row and column gives the resulting encrypted letter. Decryption follows a similar process.

According to Shaffer et al. in ***Algorithm Visualization: The State of the Field***, "AVs are an effective tool when used to actively engage students' attention" (Shaffer, 7). Therefore, in order to engage the students' attention, our system will graphically display the Vigenere table on the user's screen, and will highlight the correct row and column for each step in the encryption/decryption process. Figure 2.4.1 provides an example of what a single step in the visualization process might look like for encryption. In this example, the current plaintext letter is "O", the current keyword letter is "N", and the encrypted letter "B" is found at the intersection between the highlighted row and column.

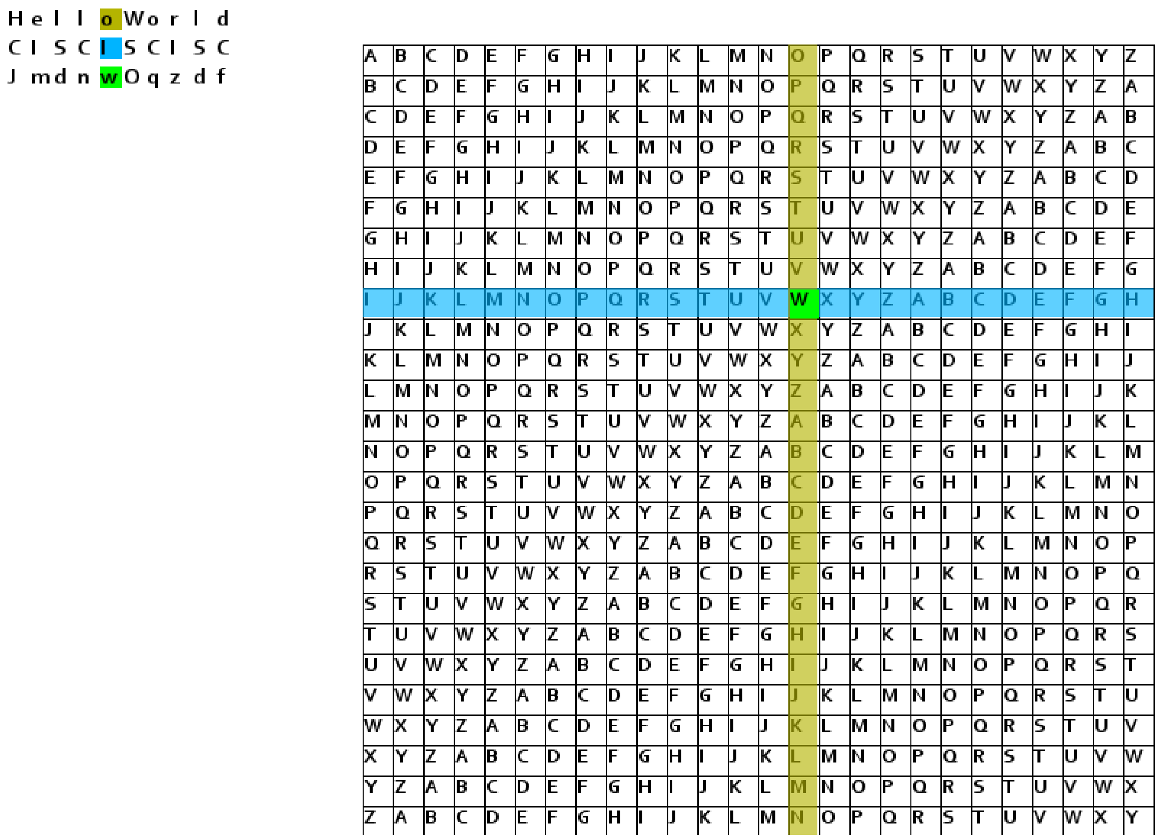
Figure 2.4.1: Sample Vigenere Table with Highlights



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

In addition to highlighting the Vigenere table on the user's screen, the system should also display and highlight the plaintext, keyword, and resulting text in order to provide a visual description of what is happening in the current step. An example of this can be seen from a simple prototype in figure 2.4.2, where the current plaintext letter "O" is highlighted with yellow, the current keyword letter “I” is highlighted with blue, and the current encrypted letter “W” is highlighted with green:

Figure 2.4.2: Screenshot of Visualization Prototype



H e l l o W o r l d
C I S C I S C I S C
J m d n w O q z d f

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

One of the most important operational features of the proposed system is user control of the visualization pace. According to Saraiya et al. “the ability to control the pace of the visualization, versus watching an animation, had the single greatest impact on AV effectiveness of the factors studied” (Clifford, 6). The proposed system will include a pacing feature, which will allow the user to pause, play, slow down, speed up, step forward, step backward, and reset the animation.

2.5 User classes

User class: Instructors. Any cryptography instructors will use this system in order to teach their students about the process in which the Vigenere cipher encrypts and decrypts text.

User class: Students. Any student that is interested in learning about the Vigenere cipher will use this system to get a better understanding of the steps needed to encrypt/decrypt using the Vigenere cipher.

2.6 Modes of Operation

Main mode of operation: Visualizing the encryption/decryption process. This mode allows users to enter information such as plaintext/ciphertext and a keyword and then use the visualization tool to step through the process of encrypting and decrypting the text that the user input.

2.7 Operational Scenarios

Use Case: Visualize the encryption process
Brief Description: This use case describes how a user would go about using the proposed system to visualize the process in which plaintext is encrypted into ciphertext.
Actors: A user of the system
Preconditions:

1. The user must have access to the system
2. The user must have a plaintext that they wish to encrypt and a keyword that they would like to use to encrypt with

Steps to Complete the Task:

1. The user opens the application
2. The user navigates to the visualization tool and enters a plaintext message and a keyword in the appropriate input fields
3. The user clicks on the 'encrypt' button
4. The encryption visualization page will load and the animation will start playing
5. The user can use any of the control buttons (pause/play, step forward/step backward, slow down/speed up, and restart) to step through the visualization at their own pace

Postconditions:

1. The user has an encrypted message and an understanding of how that message was encrypted.

Use Case: Visualize the decryption process
Brief Description: This use case describes how a user would go about using the proposed system to visualize the process in which ciphertext is decrypted into plaintext.
Actors: A user of the system
Preconditions:

1. The user must have access to the system
2. The user must have previously encrypted text that they wish to decrypt and a keyword that was used to encrypt the text.

Steps to Complete the Task:

1. The user opens the application

2. The user navigates to the visualization tool and enters a ciphertext message and a keyword in the appropriate input fields
3. The user clicks on the 'decrypt' button
4. The decryption visualization page will load and the animation will start playing
5. The user can use any of the control buttons (pause/play, step forward/step backward, slow down/speed up, and restart) to step through the visualization at their own pace

Postconditions:

1. The user has a decrypted message and an understanding of how that message was decrypted.

3. Specific Requirements

This system is designed to allow teachers/professors to pedagogically visualize the steps and results of encrypting and decrypting using the Vigenere cipher. Some strategies that we use to increase the effectiveness of this system as a teaching tool:

- Controlling the pace at which the visualization runs
  - *"They found that the ability to control the pace of the visualization, versus watching an animation, had the single greatest impact on AV effectiveness of the factors studied" (Clifford, 6).*
- Being able to start, stop, and step through the visualization
  - *"The visualization exploration task is the crux of the visualization process, for it is the task in which humans presumably reap the benefits of SV. In this task, the visualization explorer manipulates the visualization's user interface to explore the visualization. In the seminal work on SV user interfaces, Brown (1988a) suggests four kinds of view exploration to be supported:*
    - *visualization execution control, by which visualization explorers start, stop, and step through the visualization, and adjust the visualization speed;*
    - *..." (Hundhausen, 6).*
- Allowing users to provide input to the algorithm
  - *"...students should be able to provide input to the algorithm being visualized. This becomes particularly critical for determining best and worst-case behavior of algorithms" (Rößling, 1).*
- Rewind capabilities
  - *"To be truly effective for algorithm understanding, the users of the AV system must be able to rewind the algorithm's execution"(Rößling, 2).*
- Hypertext explanations of the display in the form of colored rectangles over the letters of interest
  - *"The system should support the integration of explanatory hypertext to help the student understand the mapping of the algorithm's abstractions to the AV system's display of those abstractions"(Rößling, 2).*

The specific requirements for this system will be split up into 3 categories: must have, should have, and could have.

Must Have:
- Able to encrypt text using the Vigenere cipher
- Able to decrypt using the Vigenere cipher
- Graphically display the steps (using the Vigenere table and related plaintext, keyword, and result) for encrypting and decrypting the cipher
    - Use hypertext to let the user know what is being highlighted
- Allow users to enter the message they would like to encrypt based on a particular keyword
- Allow users to enter the message they would like to decrypt based on a particular keyword

Should Have:
- Users should be able to control the pacing of the step visualization.
- Rewind capabilities
- Start, stop, and step through the algorithm

Could Have:
- Page describing how the Vigenere cipher works
- A page that allows users to test their knowledge by letting them encrypt a plaintext message on their own using the Vigenere cipher and check their answer.
    - *"Laboratory studies show that taking a test on studied material promotes subsequent learning and retention of that material on a final test (termed the testing effect)." (McDaniel)*

3.1 External Interfaces (Inputs and Outputs)

User inputs:

1. Plaintext input
2. To input a plaintext to be encrypted by the system using the vigenere cipher
3. User will input the plaintext
4. Plaintext must be letters in the english alphabet.
5. String input

----

1. Encrypted input
2. An encrypted message that can be decrypted by the vigenere cipher with the correct key (the key will also be input to decrypt the message)

3. User will input the encrypted message
4. Does not have to be a valid encrypted message, but must have a valid key. In other words, it does not have to be decrypted to a readable message, but it must be characters from the english alphabet.
5. String input

----

1. Pace of visualization
2. Two buttons (speed up and speed down) that will control how fast each step in the visualization plays
3. User will be able to change the speed with these buttons, will be set to a default speed otherwise
4. Values will be an integer ranging from 1-10, with 1 being the slowest animation speed, and 10 being the max speed the visualization will play at
5. Integer input

System outputs:

1. Encrypted message
2. Display the encrypted text after applying the Vigenere cipher
3. Output to the UI module
4. Output should be the correct text after applying the vigenere cipher
5. String output

----

1. Decrypted message
2. Display the decrypted text after applying the Vigenere cipher
3. Output to the UI module
4. Output should be the correct original plaintext message before encryption
5. String output

----

1. Vigenere visualization
2. The Vigenere cipher visualization steps showing how the Vigenere cipher works
3. Output to the Visualization module
4. Output should adhere to algorithm visualization research
5. Image outputs

3.2 Functions

1. We will need validity checks on encryption/decryption input in order to ensure that the characters entered are within the english alphabet

2. The sequence of operations will have two main interactions:
    a. The encrypt/decrypt inputs will be checked for validity before being sent to the Cipher Module to calculate the result.
    b. Button presses will be sent to the Scene Manager module where they will be processed in different ways, depending on the type of button that was pressed:
        i. The pause/play button will change the current state of the animation from playing to paused, and vice versa.
        ii. The speed up button will check to make sure that the pace is below 10 before increasing the speed of the animation.
        iii. The speed down button will check to make sure that the pace is above 1 before decreasing the speed of the animation
        iv. The step forward button will verify that the animation is currently paused before advancing the animation forward by a single step.
        v. The step backward button will verify that the animation is currently paused before displaying the previous step in the animation.
        vi. The restart button will pause the animation and then will start it over from the very first step
3. The abnormal situations:
    a. If there is no input message or there are characters not in the english alphabet for the encrypt/decrypt inputs, then the visualization module will not be activated and the system will prompt the user to please enter a valid input.
    b. If the user entered a plaintext message without a keyword (or a keyword without a plaintext message) then the visualization module will not be able to encrypt/decrypt and will prompt the user to enter a plaintext/keyword.
    c. The user clicks on speed up/slow down when the animation is running at its maximum/minimum value, respectively. In this case the system will ignore the user input to prevent an error.
    d. The user clicks on the step forward/step backward buttons while the animation is currently playing. In this case the system will ignore the user input to prevent unexpected behavior.
4. The three user inputs and three system outputs correspond 1-to-1 to the relationships between the inputs and outputs:
    a. Encrypt input → encrypt output
        i. The encryption input will send the plaintext message and keyword input by the user to the encryption output, which uses the Vigenere cipher to encrypt
    b. Decrypt input → decrypt output
        i. The decryption input will send the encrypted message and a key to the decryption output, which uses the Vigenere cipher to decrypt

c. User button press input → Display output
    i. The user will click on certain buttons depending on what they want the visualization tool to do. Each of the button presses will affect the screen that is being displayed to the user in different ways. Examples of ways that the display can be changed are:
        1. Changing the current scene being displayed (for example, clicking on the 'visualization tool' button will take the user from the main menu scene to the visualization tool scene where they can input their message and key for encryption/decryption).
        2. Changing the state of the animation (for example, going from a paused state to playing the animation, modifying the pace at which the animation plays, etc.)

3.3 Usability Requirements

The usability requirements for the system will be guided by two things: simplicity, and effective algorithm visualization techniques. The user interface will be composed such that it is obvious where to input text for the vigenere cipher to encrypt/decrypt. We aim for this to be as simple and intuitive as possible. The visualization technique for the Vigenere cipher will also be guided by existing algorithm/cipher visualization research in order to maximize internalization/learning of the algorithm, including components that aid the effectiveness of the visualization. A list of researched techniques to increase learning includes:

- Controlling the pace at which the visualization runs
- Being able to start, stop, and step through the visualization
- Allowing users to provide input to the algorithm
- Rewind capabilities
- Hypertext explanations of the display in the form of colored rectangles over the letters of interest

3.4 Performance Requirements

Encryption/Decryption should complete in 1 second.
The visualization module should be ready to output visualizations within 10 seconds 95% of the time.

3.5 Software System Attributes

This system has three attributes that have the most importance in the development of a high quality system: Correctness, Reliability, and Usability.

Correctness: The extent to which a program satisfies its specifications and fulfills the user's mission objectives. This is important for any system, but especially for this one so that students are not being misguided by the wrong encryption/decryption of a text. Extensive testing of the encrypt/decrypt functions will be deployed to make sure that the cipher algorithm is correct and working as intended.

Reliability: The extent to which a program can be expected to perform its intended function with required precision. This is important for the same reasons as above, if there is a situation where the system does not get the right answer then this would mislead students who are trying to learn the cipher. Extensive testing and studying of the vigenere cipher will be deployed to ensure reliability of the system.

Usability: The effort required to learn, operate, prepare input, and interpret output of a program. Arguably the most important of the three attributes, making the UI simple enough and intuitive enough for the user allows them to focus more on learning the cipher itself rather than how to use the system. Ensuring usability will require research into algorithm visualization methods and prototyping with Joe Li to test for intuitiveness of the system.

4. References

Shaffer, Clifford A., et al. “Algorithm Visualization.” *ACM Transactions on Computing Education*, vol. 10, no. 3, 2010, pp. 1–22., doi:10.1145/1821996.1821997.

Rößling, Guido, and Thomas L. Naps. “A Testbed for Pedagogical Requirements in Algorithm Visualizations.” *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education - ITiCSE'02*, 2002, doi:10.1145/544414.544446.

HUNDHAUSEN, CHRISTOPHER D., et al. “A Meta-Study of Algorithm Visualization Effectiveness.” *Journal of Visual Languages & Computing*, vol. 13, no. 3, 2002, pp. 259–290., doi:10.1006/jvlc.2002.0237.

Mark A. McDaniel, Janis L. Anderson, Mary H. Derbish & Nova Morrisette (2007) Testing the testing effect in the classroom, European Journal of Cognitive Psychology, 19:4-5, 494-513, DOI: 10.1080/09541440701326154

Software Design Specifications

Vigenere Cipher Visualization Tool

Table of Contents

1. Revision History

Date	Authors	Description
05/12/2021	JF, MH, YZ	Document Created
05/30/2021	JF, MH, YZ	Document Modified. Text and diagrams changed.
05/31/2021	JF, MH, MR, YZ	Final revisions for submission

2. System overview

We will have a system that operates as a native application, and will have three main components:

- The user interface
- The cipher module
- The scene manager module

The user interface is made up of two smaller components: the main display module and the visualization module. These two components provide the user interface with the functionality of being able to collect user input and display visualization output. The user interface sends input data to the cipher module and the scene manager module.

The cipher module will provide all functionality for correctly calculating the encryption/decryption results based on the user's plaintext input. The cipher module sends its results to the scene manager module to be processed and displayed.

The scene manager is in charge of managing/changing the current scene that is being displayed as well as enforcing the pace of the visualization. It receives input from the user interface in the form of button press data from the user and input from the cipher module in the form of encryption/decryption results. With this input, it tells the user interface what to display and when to display it.

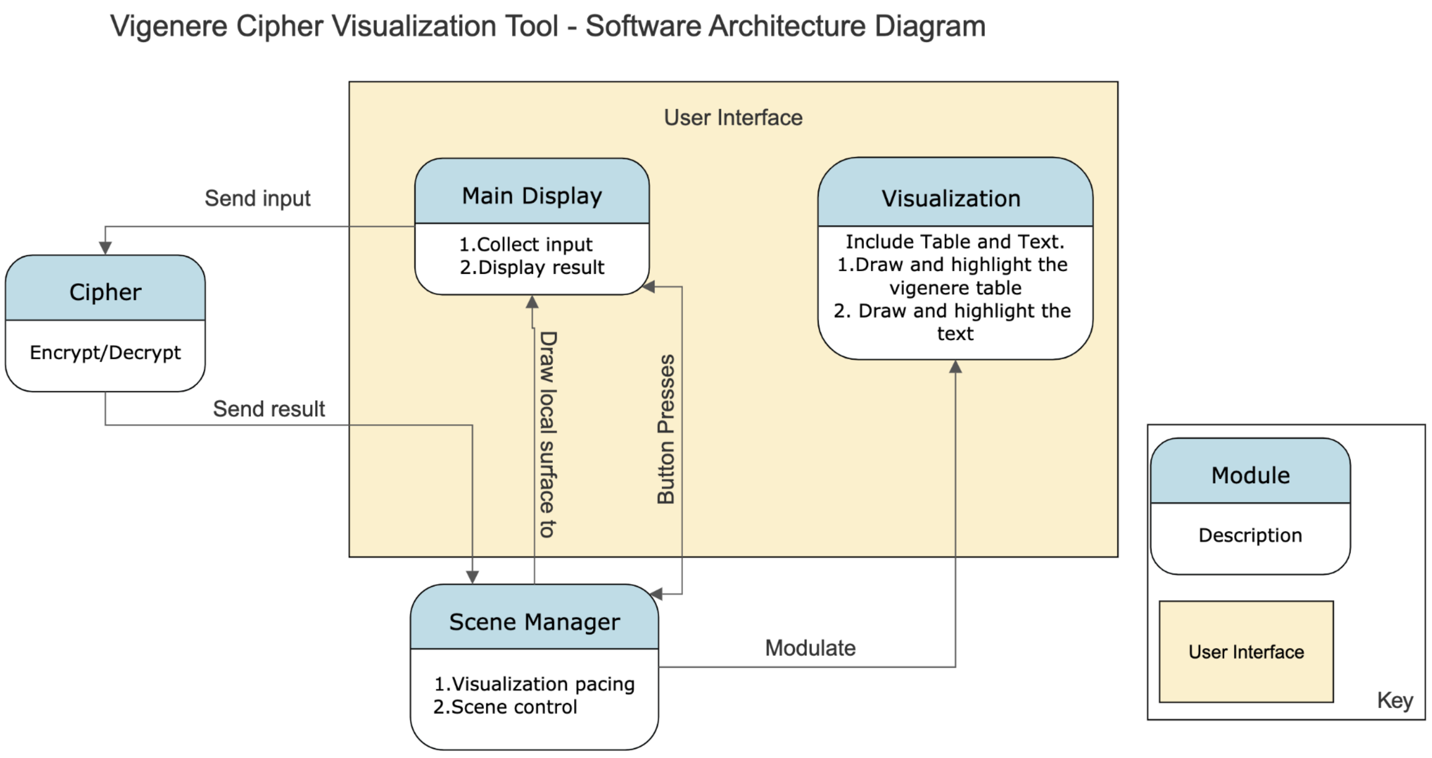
List of technology intended to be used
The project will use these technologies:

- Python
- pygame

3. Software architecture

3.1. The set of components

- Main Display
- Text Encryption/Decryption
- Cipher Visualization
- Scene Manager



3.2. The functionality provided by (or assigned to) each component

1. The first component of the system is the main display. The main display will be in charge of handling all interactions with the user via the system. The interactions that the user interface will manage are:
    a. reading user inputs for encryption/decryption information
    b. displaying the visualization outputs to the user.

2. The second component of the system is the cipher module. This component is in charge of generating an encrypted/decrypted text given a message and keyword by the user, along with all of the steps that were needed to complete the process of encryption/decryption.

3. The third component is the cipher visualization module. The visualization component will handle all of the functionality needed in order to generate sequential visual displays of the encryption/decryption process, including generating the table, highlighting the rows and columns in the table, and highlighting the respective letters of the message, key, and result

4. The last component of the proposed system is the Scene Manager component. The Scene Manger module is in charge of switching the scenes that the user sees depending on what menu buttons they push, and also making sure that each of the visualizations are displayed to the user at the appropriate rate and controlling the pace of the visualization.

3.3. Which modules interact with which other modules

- Cipher Module
- Main Display Module
- Visualization Module
- Scene Manager Module

- The Main Display Module will invoke the Cipher Module with the appropriate message and key input.
- The Visualization Module will be called and looped by the Scene Manager Module to obtain the animation effect.
- The Scene Manager Module takes button presses for pacing control from the Main Display.

4. Software Modules

4.1. Main Display Module

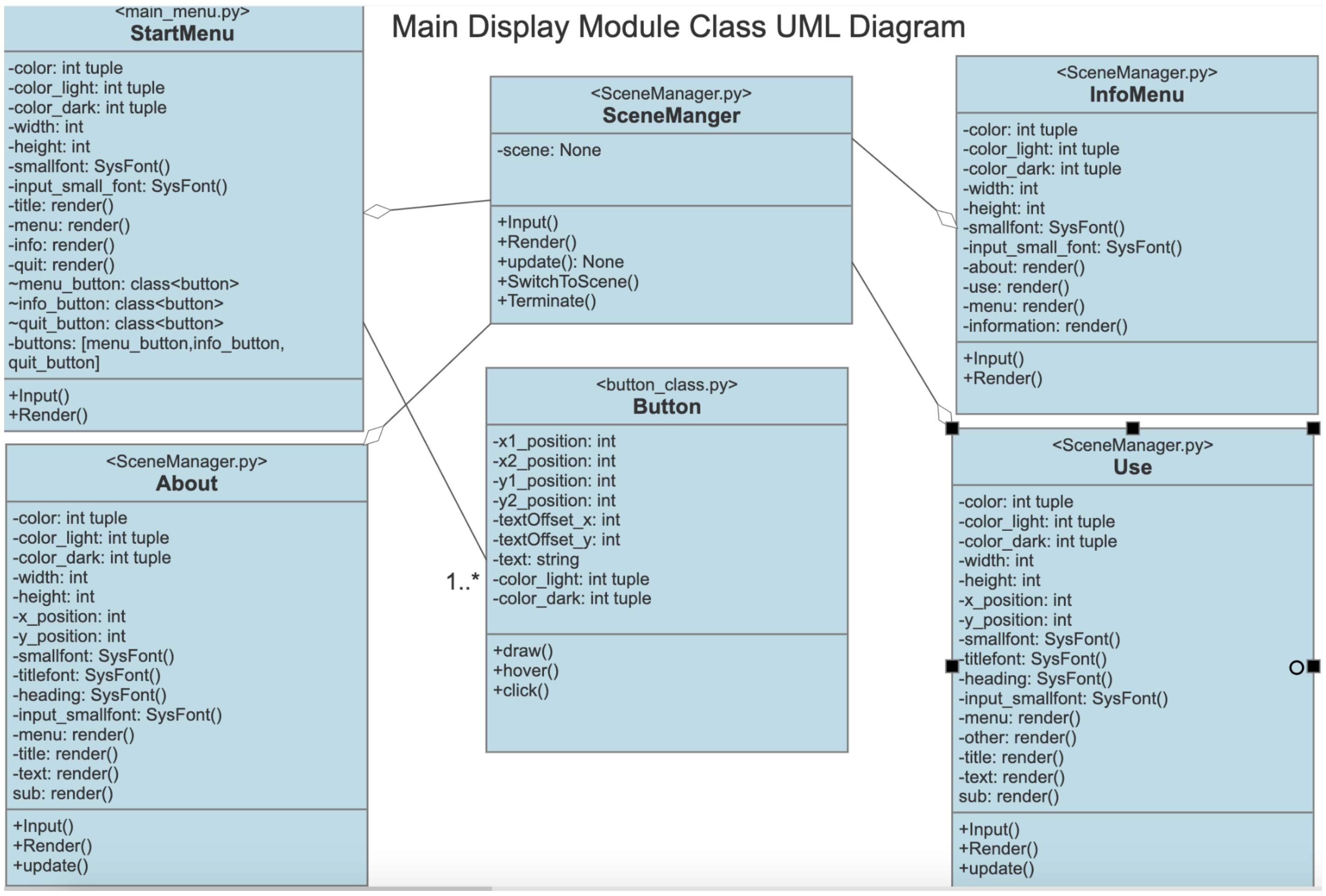
The module's role and primary function.

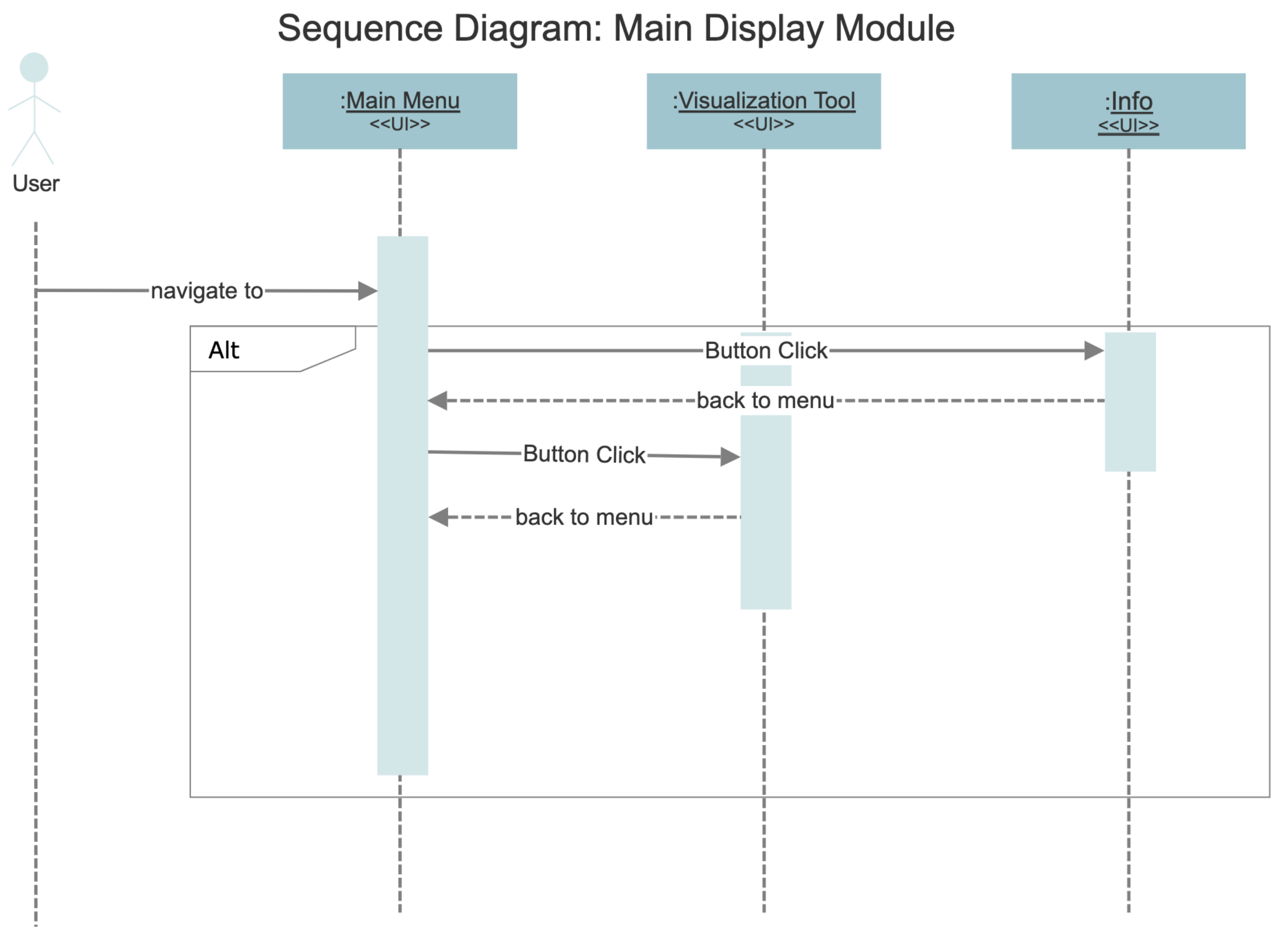
The Main display will interact with the user in order to gather the information needed to start the visualization process. This information consists of the plaintext/ciphertext that the user wishes to encrypt/decrypt, the keyword that they want to use, and the pace that the user wishes to visualize at. It displays all the menus that the user can navigate to.

Interface Specification

The main display will send the user's key and message input to the cipher module. Any scene that is displayed through the main game loop is the main display, i.e. all classes defined in the SceneManager file.

A Static and Dynamic Model





Design Rationale

The Main display module will be necessary in order to allow the user to interact with the system in various ways. The proposed system needs to be able to read user input and display output to the user, making this an essential component of the system. The main display was designed with the intent of handling all user interaction within a single component of the system in an attempt to create strong cohesion.

4.2. Cipher Module

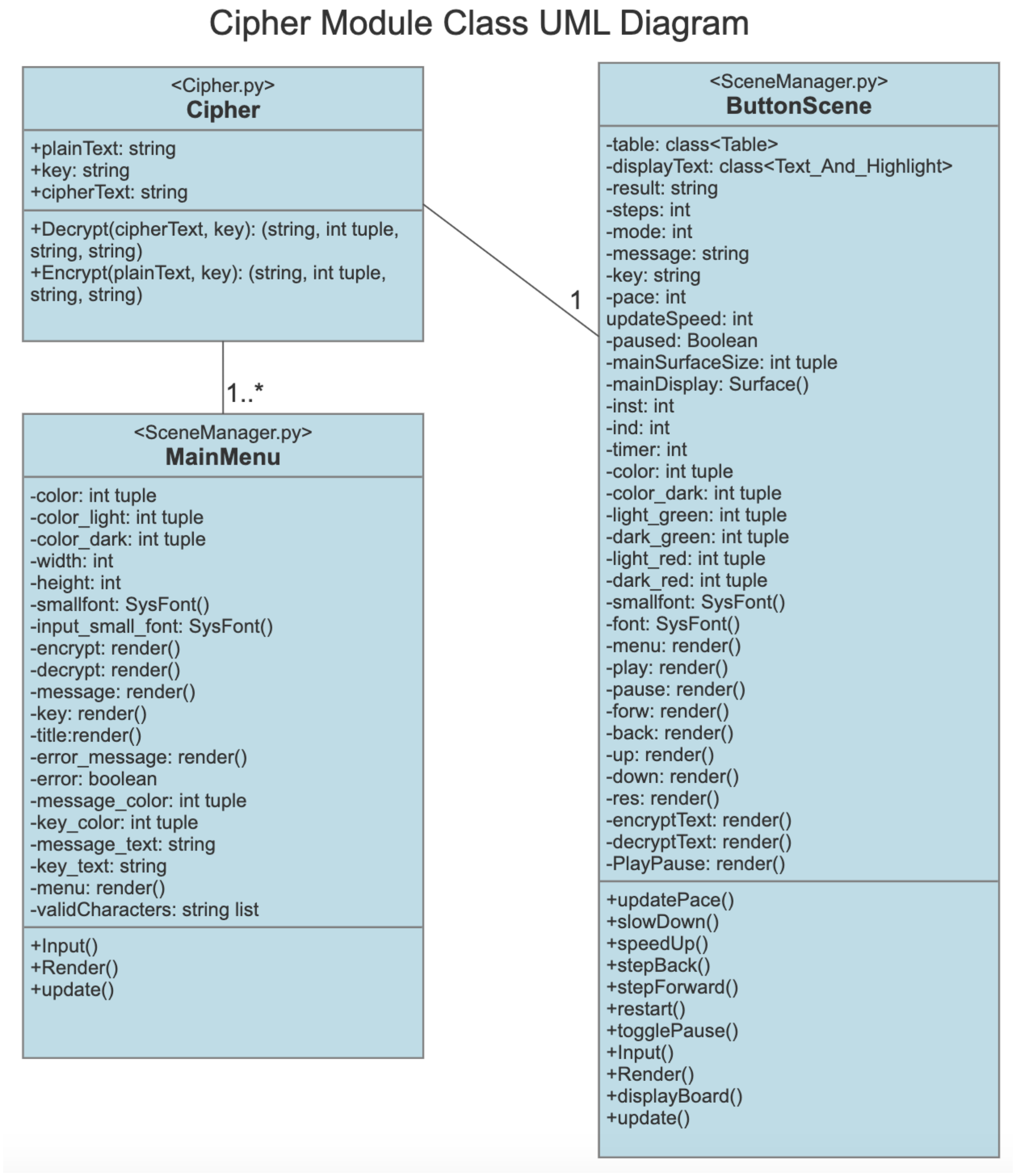
The module's role and primary function.

This module will take input from the user and, depending on which functionality the user wishes to use, will encrypt/decrypt a plaintext using the Vigener cipher.

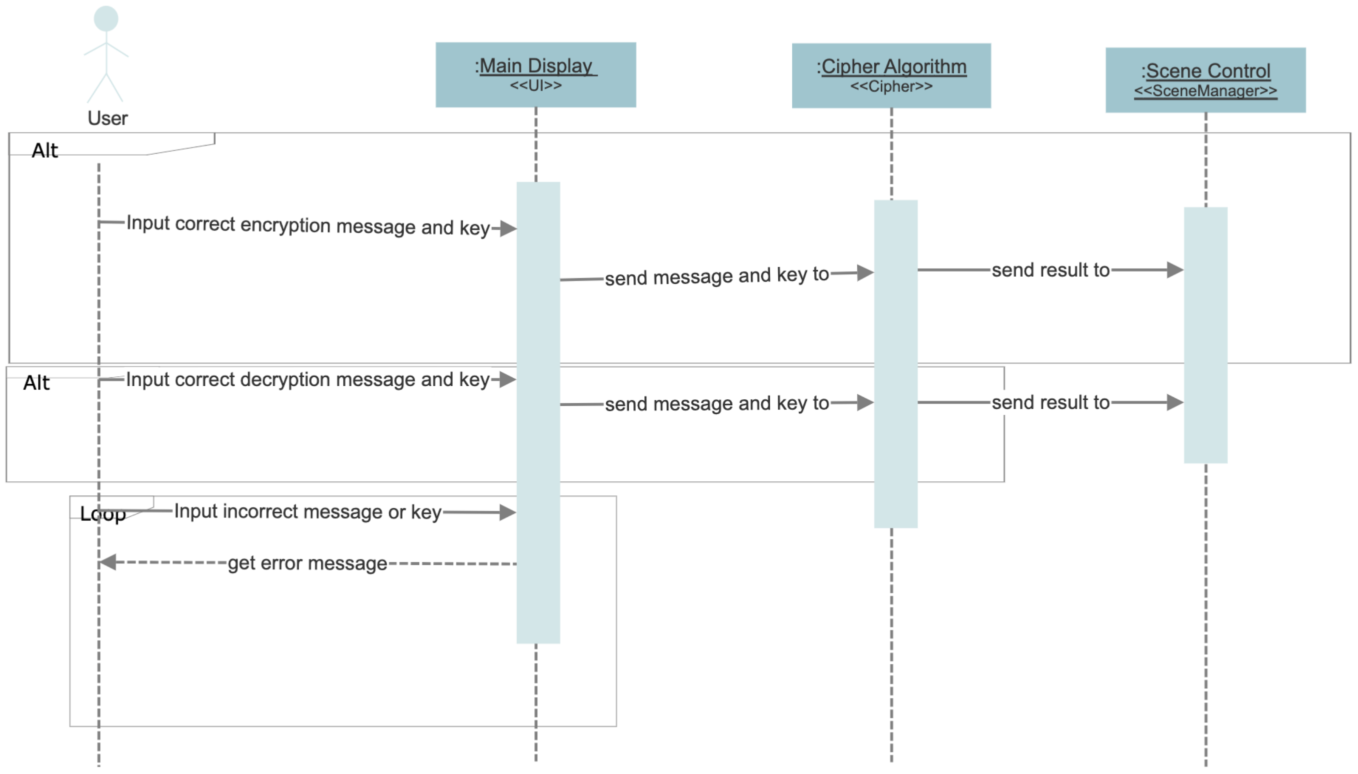
Interface Specification

The cipher module will receive input from the user interface. It should receive 3 parameters: Message to encrypt/decrypt, the key to encrypt/decrypt, and the mode (encryption/decryption). The result of the Vigenere cipher according to the parameters given will then be sent to the scene manager module.

A Static and Dynamic Model



Sequence Diagram: Cipher Module



Design Rationale

This component will be responsible for making sure the output of the Vigenere cipher adheres to the correctness and reliability attributes of the software. Having a module that handles the correct input/output for the Vigenere cipher is of utmost importance to upholding these attributes. It also simplifies the code, and integration with the other modules will be as simple as receiving the correct input and sending the expected output.

4.3. Visualization Module

The module's role and primary function.

This module will be responsible for drawing and highlighting the vigenere table and the corresponding plaintext message, keyword, and result. The functionality of the visualization module is split up into two separate classes: The table class and the displayText class. The table class manages the drawing/highlighting of the table. The displayText class manages the drawing/highlighting of the display text (plaintext, keyword, and result).

Interface Specification

The table class provides two public methods which are available to the scene manager in order to highlight the correct rows and columns. The first public method is used by the scene manager to draw highlights on the table for a single step in the encryption process. The second public method is used by the scene manager to draw highlights on the table for a single step in the decryption process. Both of these class methods take 3 arguments from the scene manager:

1. The plaintext letter to be highlighted
2. The keyword letter to be highlighted
3. The result letter to be highlighted

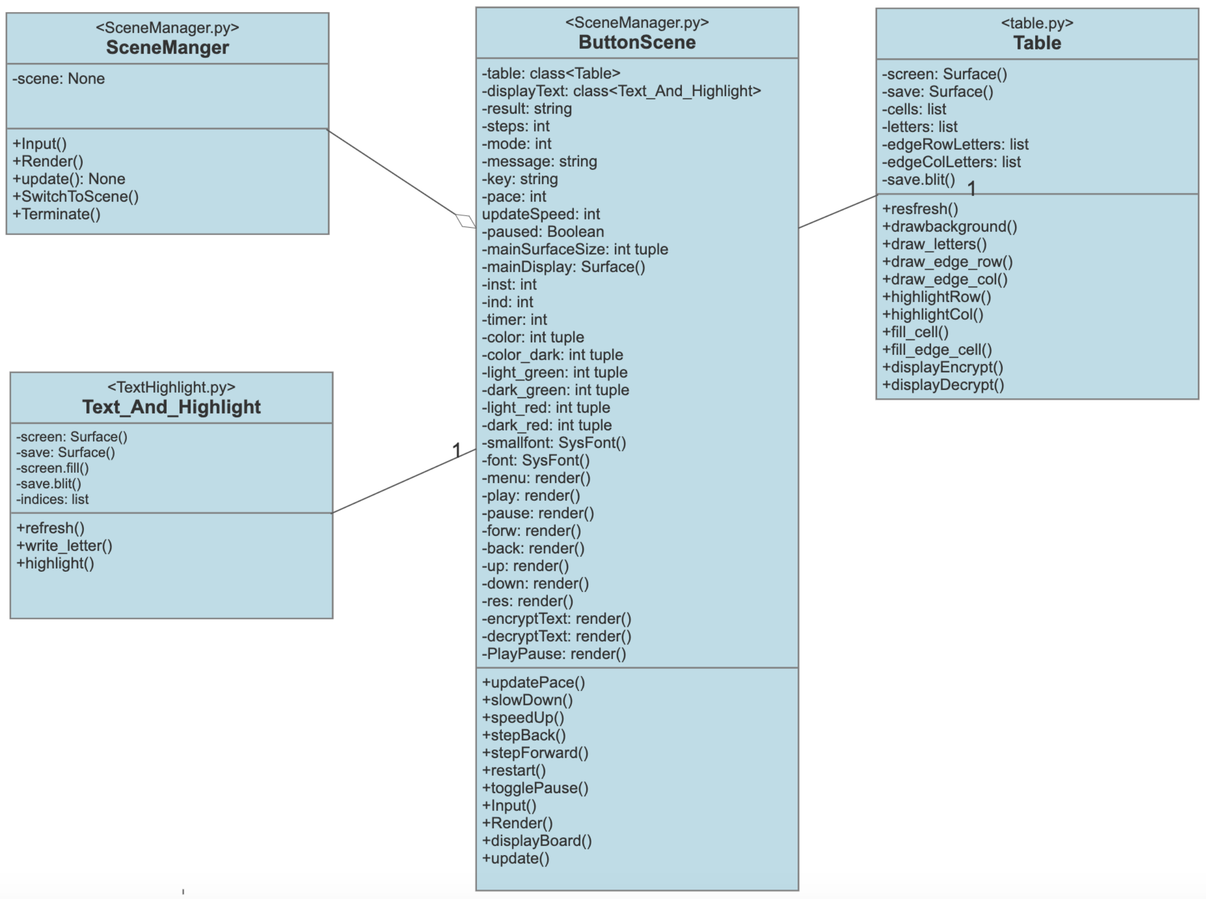
The displayText class provides one public method that is available to the scene manager in order to highlight the correct letters in the display message, keyword, and result texts. This class method takes 3 arguments from the scene manager:

1. The plaintext letter to be highlighted
2. The keyword letter to be highlighted
3. The result letter to be highlighted

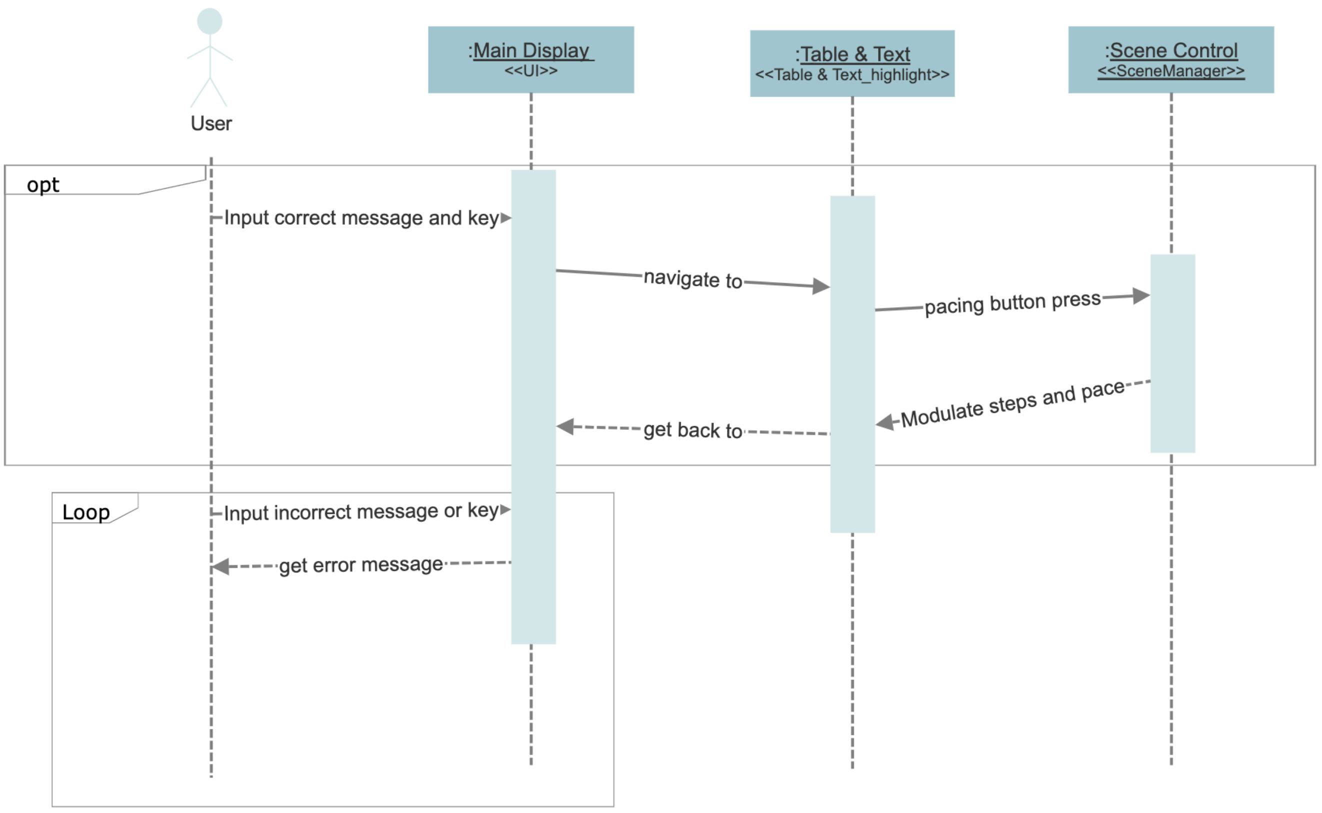
Once the visualization module has drawn all of the appropriate highlights for the current step in the visualization process (on its local drawing surface), it will copy the drawings over to the main display module through the SceneManager file for the user to see.

A Static and Dynamic Model

Visualization Module Class UML Diagram



Sequence Diagram: Visualization Module



Design Rationale

The main rationale behind designing the visualization module is to abstract the process of drawing and highlighting the vigenere table and display text out of the main display module to simplify the code. The way that this module was implemented allowed us to draw the table and the display text on their own local drawing surfaces, and then copy them over to the main display surface in the correct position through a relatively easy process. This allowed us to handle the positioning of the vigenere table on window resizes very easily. If the functionality for the visualization module was implemented in the main display module, it would complicate the code for the main display as it would have to handle drawing every single aspect of the table and display text during every single frame update in the game loop.

4.4 Scene Manager Module

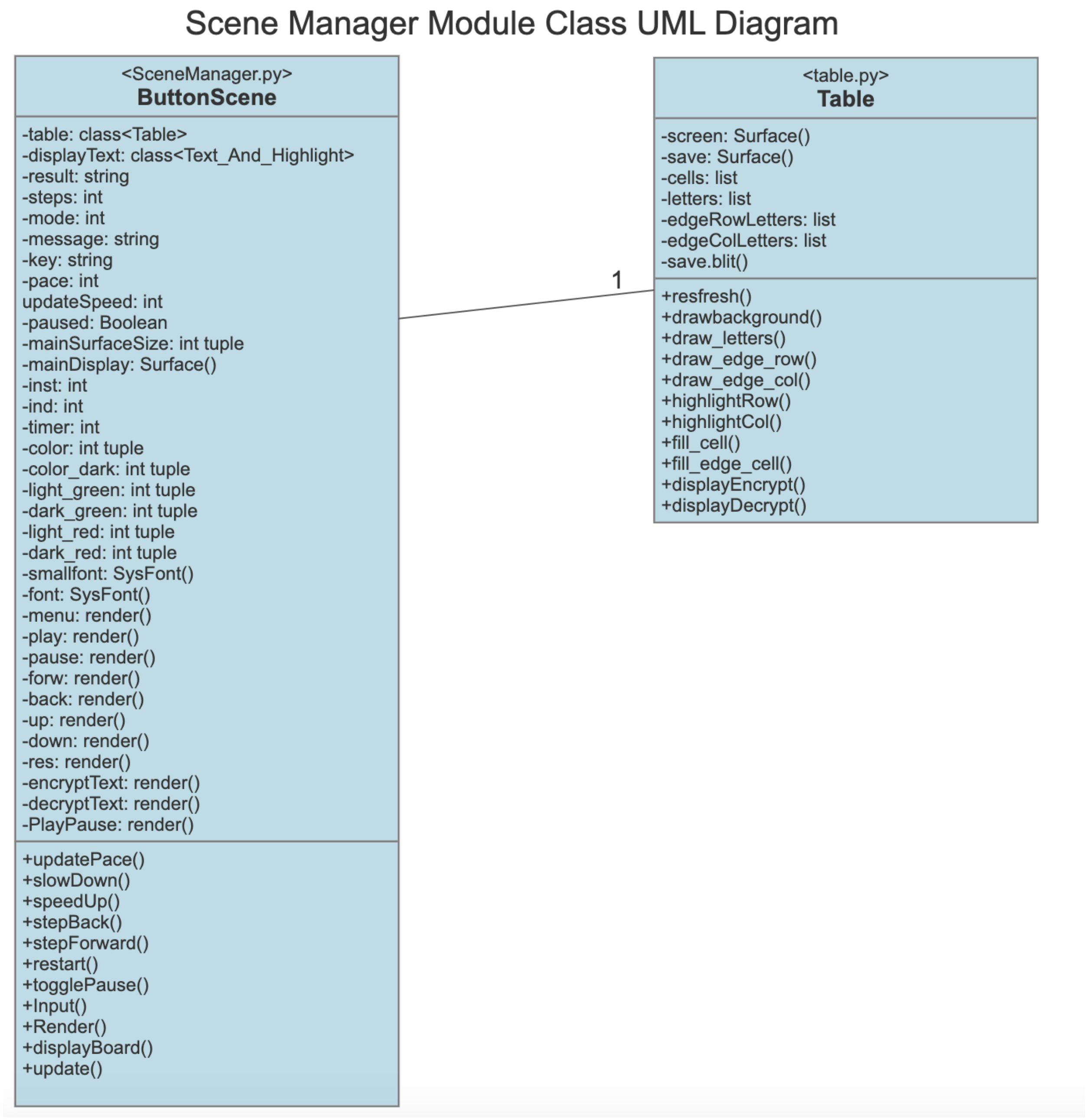
The module's role and primary function.

This module is in charge of switching the scenes that the user sees depending on what menu buttons they push, and also making sure that each of the visualizations are displayed to the user at the appropriate rate and controlling the pace of the animation.

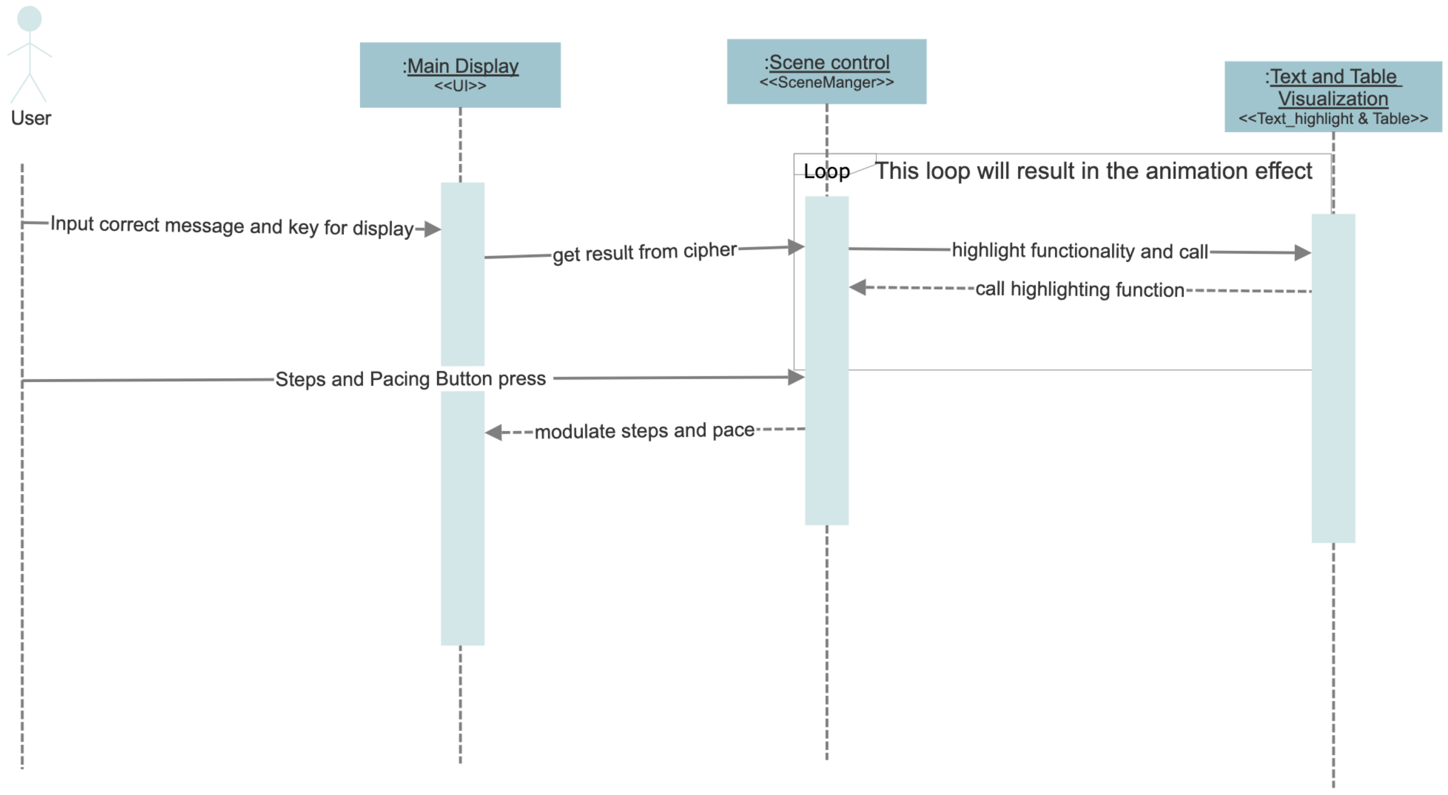
Interface Specification

The Scene Manager module switches the scene whenever the corresponding button for the scene is pushed. It also receives input in the Visualization screen in the form of Speed up, speed down, step forward, and step back which controls the pacing of the animation.

A Static and Dynamic Model



Sequence Diagram: Scene Manager Module


User
:Main Display
<<UI>>
:Scene control
<<SceneManger>>
:Text and Table Visualization
<<Text_highlight & Table>>
Loop
This loop will result in the animation effect
Input correct message and key for display
get result from cipher
highlight functionality and call
call highlighting function
Steps and Pacing Button press
modulate steps and pace

Design Rationale

In order to switch between scenes in pygame, each scene had to be defined as a class that the main game loop can call abstractly. For example, in the main game loop we call:

- active_scene.Input()
- active_scene.Render()
- active_scene.update()

with the active_scene being the scene currently displaying to the user. This allows us to define scenes as classes with these functions, and we can seamlessly switch scenes using this model by updating the active_scene variable with the new scene.

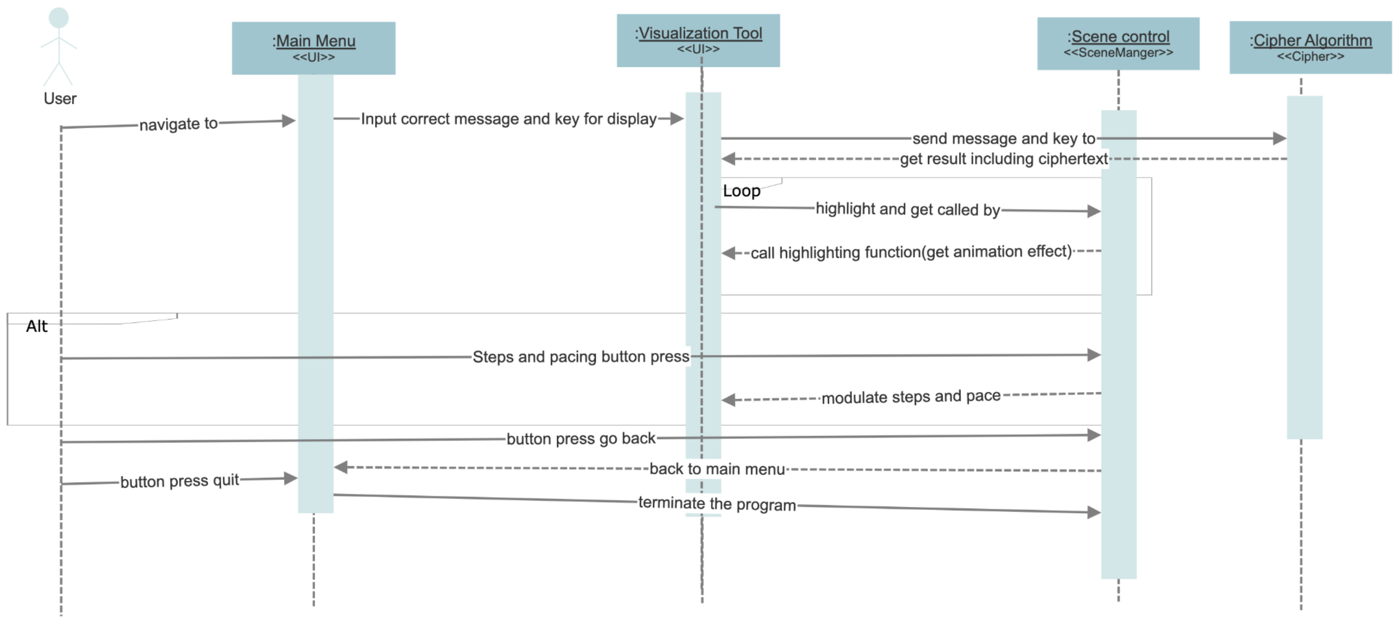
5. Dynamic Models of Operational Scenarios

The system is designed as a Vigenere cipher visualization tool. It allows users to encrypt and decrypt using the Vignere cipher and visualize how the cipher works during each step. In the meanwhile, users can also interact with the program by pressing buttons for controlling the pace of the animation. Here is an user case example:

1. The user is navigated to the main menu.
2. The user inputs message and key for encryption/decryption.
3. Then the visualization Tool collects the input and sends it to the cipher module.
4. The cipher algorithm sends the result to the visualization module for displaying the animation.
5. Scene manager will loop through table and text highlighting to obtain an animation display.
6. The user may choose to step forward/back and move faster/slower.
7. Then, the scene manager module will modulate the pace according to the user's button press.
8. The user presses back to the main menu.
9. The user quits the program.

The UML Sequence Diagram of Use Cases displays as following:

Sequence Diagram: Use cases



6. References

Simmons, Gustavus J.. "Vigenère cipher". Encyclopedia Britannica, 9 Aug. 2017, https://www.britannica.com/topic/Vigenere-cipher. Accessed 12 May 2021.