Milestone 2 Progress Evaluation

Project Title: Improved Visualization for Formal Languages

Group Members: Chris Pinto-Font (cpintofont2021@my.fit.edu), Vincent Borrelli (vborrelli2022@my.fit.edu), Andrew Bastien (abastien2021@my.fit.edu), Keegan McNear (kmcnear2022@my.fit.edu)

Faculty Advisor: Dr. Luginbuhl (dluginbuhl@fit.edu)

Client: Dr. Luginbuhl

Milestone #2 Progress:

Task	Completion %	Chris	Vincent	Andrew	Keegan	To do
Have running stable version of the computer application	100%	20%	20%	30%	30%	Further expand application functionalities in working program
Have a working basic version of the the DFA graphing process	50%	10%	10%	15%	15%	Need to produce a better, more visual form of the graphing process beyond its current textbased internal state.
Implement a comprehensive onboard “read me” file for current	70%	20%	20%	15%	15%	Need to further expand onboard “read me” file as things are better

application features						fleshed out and expanded.
Implement internal logic to check DFA completeness and string validity	80%	15%`	15%	25%	25%	Further refine internal logic and checking systems

Milestone #2 Discussion (Task Details):

- Have running stable version of the computer application
    - We successfully built an initial stable version of the application to serve as a foundation for upcoming enhancements and feature integration, providing a reliable baseline for future testing and optimization. The most basic aspects of the gui are present and will be fleshed out and attached to further logic and interactive elements which are currently being written detached front the front end of the GUI.
- Have a working basic version of the the DFA graphing process
    - A preliminary DFA graphing process has been implemented to a limited degree in a text based format, enabling users to visualize deterministic finite automata structures using the currently built node structures which will soon be attached to the GUI. This early version includes basic state creation and connection features, establishing the groundwork for future improvements such as animation, labeling, and interactivity for future milestones.
- Implement a comprehensive onboard “read me” file for current
    - An early version of the onboard “Read Me” file has been developed to provide users with essential setup and usage instructions for the program as it currently exists. This documentation explains the purpose of each feature we have constructed at this time, the basic operational workflow, and troubleshooting information for the current version of the program. It will continue to evolve as more features are introduced, though its existence in the current GUI is to remain a constant through further updates.
- Implement internal logic to check DFA completeness and string validity
    - We have begun implementing internal logic to verify DFA completeness and validate input strings through a text based incarnation of the DFA logic system that will be further attached to the larger visual program. The system checks whether each state has defined transitions for all symbols in the alphabet and

evaluates whether input strings are accepted or rejected based on transition rules. Additional improvements to enhance performance and accuracy are planned for future milestones.

Milestone #2 Discussion (Team Contribution)

- **Chris:** Chris played an important role in developing the early stages of the graphical user interface present in the stable base application, focusing on establishing a tone for the front-end framework going forward. He assisted in coding some of the early logic models present and researching connecting the underlying logical systems to the eventual interactive GUI, ensuring the design would accommodate future interactivity. Chris also contributed to writing the "Read Me" file by exploring the current logic present in the text based back end programs. Additionally, he participated in testing and debugging efforts to ensure the application's stability as new components were integrated both for the basic text based program and base application.
- **Vincent:** Vincent continued to contribute to the documentation and bug testing for backend logic of the current project. He helped consult and refine the early text-based DFA graphing system, ensuring that node creation and transition connections operated correctly in accordance to DFA logic within the current limited framework. He also consulted on establishing the internal logic that checks DFA completeness and string validity, testing early versions of the input validation functions. Beyond programming assistance, Vincent oversaw updates to the "Read Me" documentation during its .
- **Andrew:** Assisted in the coding of some of the core files used in the development of our text based incarnation of the project, assisted in helping flesh out organization elements of the code and maintaining our code base's upkeep as well as unifying some ideas and aspects of the current code base. He too assisted in the coding of the current running interactive program and bug testing for it.
- **Keegan:** Keegan led development efforts for most of the code present in the current programs, exploring options for node usage and internal logic systems for DFA creation. He was responsible for building several of the code files that will be integral to our program going forward and did further research into the sort of libraries and techniques he found necessary for us to implement for the code going forward as to fully render our program as intended. He too assisted in bug testing of said aforementioned program.

Plan for Milestone #3:

Task	Chris	Vincent	Andrew	Keegan
Implement interactive canvas space for graphing	Bug Fixer/Code Contributor and designer	Bug Fixer/Code Contributor and designer	Co-Lead coder and development head	Co-Lead coder and development head
Implement basic animations in graphing space	Bug Fixer/Code Contributor and researcher	Bug Fixer/Code Contributor and researcher	Co-Lead coder and development head	Co-Lead coder and development head
Tie text based program version to visual version	Bug Fixer/Code Contributor and researcher	Bug Fixer/Code Contributor and researcher	Co-Lead coder and development head	Co-Lead coder and development head
Implement basic DFA reduction functionality	Logic writer, DFA logic consultant, and bug tester.	Logic writer, DFA logic consultant, and bug tester.	Co-Lead code side implementor	Co-Lead code side implementor
Updated and Accessible Program Side “Read Me” File	Co-Writer	Co-Writer	Code Side implementation	Code Side implementation

Milestone #3 Plan Discussion (Task Details):

- Implement interactive canvas space for graphing
    - Building upon the existing GUI framework, our goal for this milestone is to fully implement an interactive canvas space where users can visually construct DFAs through direct manipulation of graphical elements. Users will be able to click, drag, and connect nodes and transitions in real time, allowing for an intuitive and educational visualization experience. This feature will serve as the central visual workspace of the program, forming the bridge between user interaction and the underlying DFA logic system.
- Implement basic animations in graphing space
    - To enhance user engagement and comprehension, we plan to integrate basic animations within the graphing space. These animations will visually depict state transitions during DFA traversal and string evaluation, showing users exactly how

an input string is processed through the automaton. The animations will also be used to demonstrate the step-by-step creation of a DFA, supporting the program's teaching-oriented design and making abstract concepts more accessible and interactive.

- Tie text based program version to visual version
    - One of the primary objectives for this milestone is to merge the existing text-based DFA logic system with the developing visual interface. This integration will allow the program's backend logic—currently operating independently in text form—to synchronize with user actions within the canvas. As a result, DFAs created via textual input will automatically generate corresponding visual representations, while modifications in the graphical interface will dynamically update the underlying data structures, ensuring consistency between both modes of operation.
- Implement basic DFA reduction functionality
    - We plan to introduce an initial version of the DFA reduction feature, enabling users to simplify DFAs by merging equivalent states and removing redundancies. This will serve as the first step toward a more comprehensive DFA optimization system in later milestones (along with an animated component). The functionality will be implemented in a way that can be demonstrated visually, allowing users to observe how the reduction process affects the structure of their automaton in real time.
- Updated and Accessible Program Side "Read Me" File
    - The onboard "Read Me" file will continue to evolve alongside new feature additions. For this milestone, our focus will be on updating it to include documentation for the newly implemented canvas, animation, and reduction systems. This file will remain directly accessible within the program's interface, allowing users to learn about new functionalities as they are added. As before, Keegan and Andrew will focus on coding and integrating these features, while Vincent and Chris will be responsible for drafting, editing, and maintaining the accompanying documentation to ensure clear and accurate user guidance.

Date(s) of Meeting/Client Feedback:

August 27:

- Focus is on DFAs
- User should be able to create a DFA intuitively (we'd need to nail down what that meant)
- The system should be able to check for completeness (e.g., are there transitions missing?) or correctness (is it truly a DFA)
- Maybe the system should be able to complete given an incomplete DFA (e.g., fill in missing transitions – but how?)
- Eventually it would be nice to format/prettify the DFA graphs (align nodes, improve readability)
- Be able to designate initial and final states and maybe even designate a state as a dead state and let the system do its magic
- Multiple symbols on a single transition between arcs in a way that is natural and readable
- Animate processing of strings on a DFA – step-by-step, play, pause, etc.
- Minimization
- The vision is, given a set of strings that are accepted and not accepted, to let the system create a minimal DFA (in terms of states) that accepts/rejects as specified

September 10:

- Involve in the program is lambda transitions.
- The key things needed is when given strings it would create the DFA, when given strings it would have to determine whether it accepts or rejects and being able to create these states whenever

October 24th:

- Discussed update on milestone two progress
- Expressed current work on internal program logic
- Spoke about current in-program node connecting format
- Expressed interest in further implementation of and use of visual canvas type graphing form.
- Spoke to us about our current challenges regarding the state of the project and its code going forward.

Date(s) of Meeting/Advisor Feedback:

- ***Same as above***

Faculty Advisor Signature: _____ Date: October 27th, 2025