CSC207 Progress Report (Phase 0)

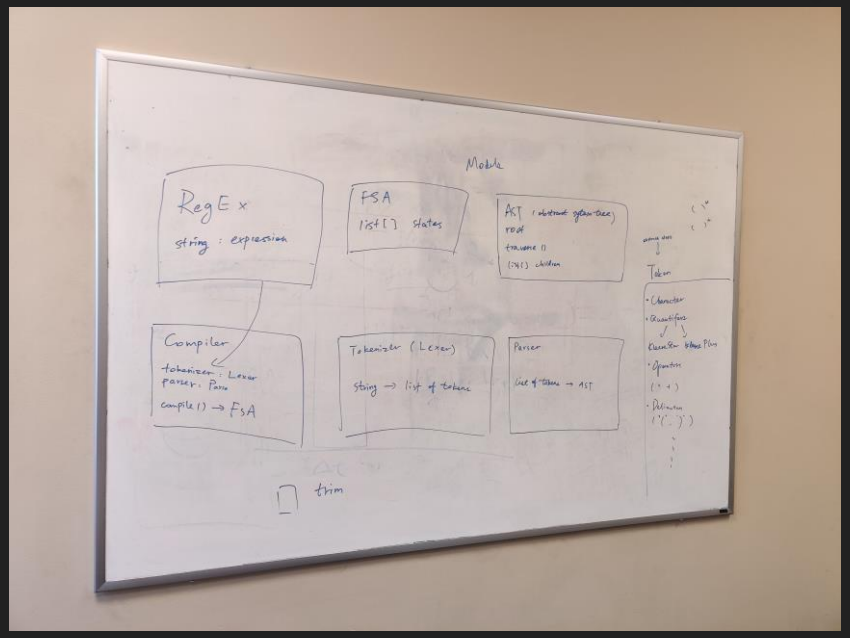
Group-030 **uwutever**

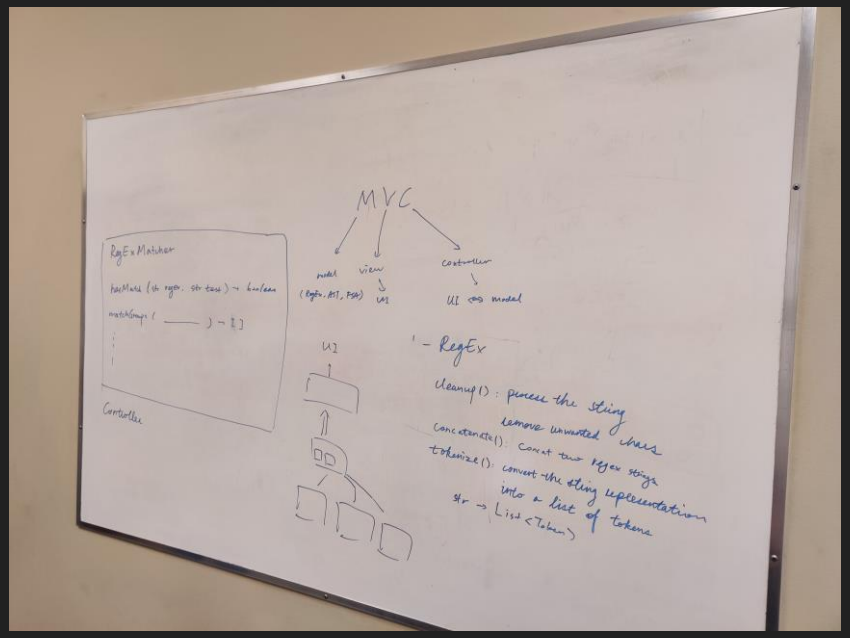
Kevin Gao, Hanrui Fan, Arkaprava Choudhury, Brian Ho, Franklin Yeung, Letian Cheng

What we've been doing

During phase 0 of the project, we had one in-person meeting and multiple online meetings to discuss how to proceed with our project, here's a progress report of what we've accomplished so far over the past few weeks, including the specification, CRC model, scenario walkthrough, and skeleton program.

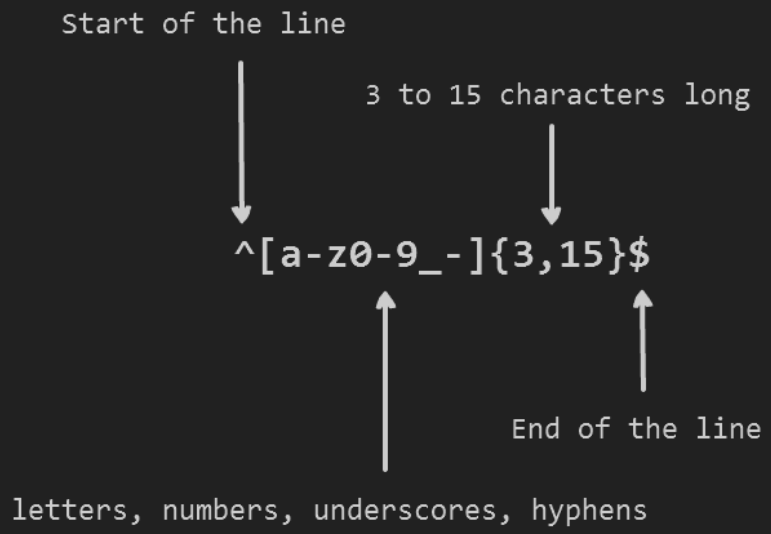
In addition, we also have some open questions we would like help with.





Project Domain

We've decided to make our project a regular expression toolkit, with features implemented that will facilitate the understanding of them. The target user for this piece of software would be a student or enthusiast who has just started learning about regular expressions at an elementary level.



RegEx

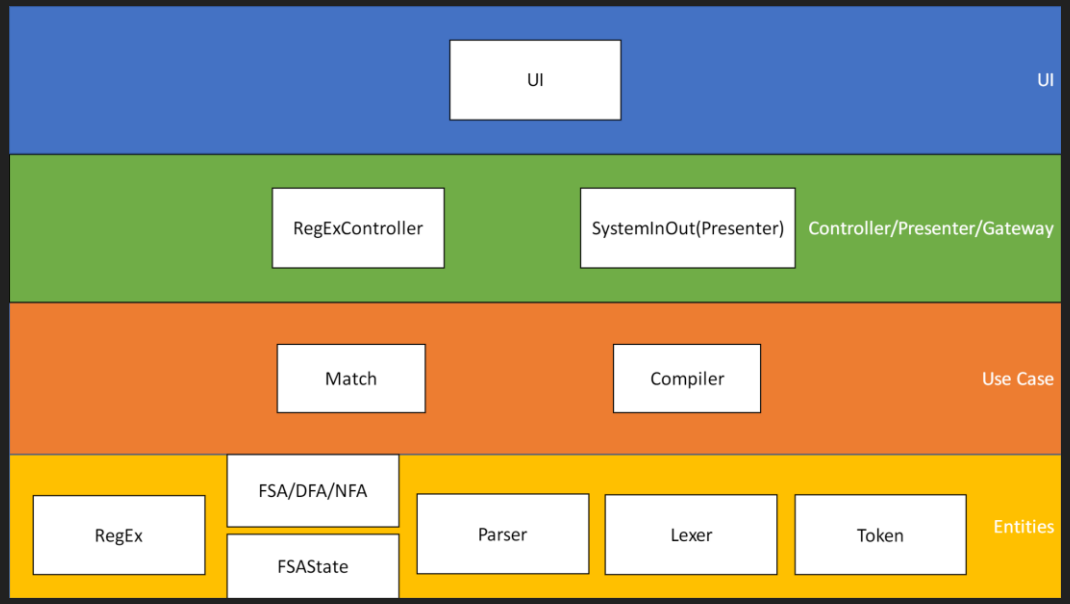
Specification

We plan to implement two main features in our program:

- It should be able to perform matching between a given regular expression and a given text string, i.e., determine whether a given string is in the language denoted by a regular expression.
- It should be able to visualize a Finite State Automaton which accepts the same language as the regular expression that is given. The user should be able to view the states and transitions of the FSA. The FSA will, by default, be an NFA, and the user will have an option to construct the corresponding DFA too.

CRC Model

We've developed a rudimentary CRC model contain all the necessary entity, use case, and controller classes. Below is a clean architecture graph of the model.



The detailed model can be found on the group repository.

(We will continue to adjust the model as needed in the future.)

Scenario Walk-Through

- The user has a valid regex and wants to know whether a trial text consists of matches with the regex (and seeks to find a way to visualize the regex, in the completed program);
- Through a keyboard input, user inputs a regular expression and a text for matching on the `interface`; for example, consider the regular expression `hallo`, and the text `yahallo, this is uwutever`;
- A `RegEx` object is created by the program;
- `Compiler` uses `Parser` and `Lexer` to create the corresponding `FSA` for the input `RegEx ;`
- `RegExController` calls `match` to use `FSA` to match with the given text.
- The result of the matching is displayed on the `interface`. (In the skeleton program, this is a CLI, which will be replaced with a visual interface in the competed program).

Scenario Walk-Through

- User could also request to visualize the FSA through commands input
- The FSA is converted into a displayable style
- The visual is outputted to the UI through presenter

Skeleton Program

Open Questions

- What method of input should we use for the regex?
- Are there any good UI frames we could use?
- Should we use only NFA instead of constructing DFA?
- For our project domain, is it more suitable to develop a mobile app, desktop app or web app?
- Should we move FSA to use case group?

Individual Responsibilities

- Kevin – Phase 0: Project Idea, CRC Model, Backend; Future: Backend(, Frontend)
- Hanrui – Phase 0: CRC Model (+Cards); Future: Frontend
- Arkaprava – Phase 0: Progress Report, Unit Tests; Future: Backend
- Franklin – Phase 0: Progress Report (+Slides), General Edits; Future: Backend(, Frontend)
- Brian – Phase 0: CRC Model, Specification, Walk-Through; Future: Backend
- Letian – Phase 0: CRC Model, Progress Report, Arrangement; Future: Frontend