Scientist Pre-Interview Notes 2

Team 5 Pattern Pandas

Scientist Information

Leo Ureel - Assistant Professor, Computer Science
Email: ureel@mtu.edu
Office Phone: (906) 487-1816
Office: Rekhi Hall 209

Meeting Date and Time

Date: January. 24th, 2023
Time: 4pm EDT

Meeting Location

Location: Library 112 (in person)

Team Members and Interviewing Roles

Zane Smalley (technical lead) - Host
Ame Tian - take notes
Isaac Elenbaas
Maritza Gonzalez
Mya Davis
Noah Riske (product owner)- Unable to attend (class)
Quentin Ross

Intro: Overview of our Initial Design

Conclusions:

- No home/navigation page, landing page contains the main purpose of the application (similar to unit conversion sites or YouTube downloaders)
- explainshell.com is a great resource to take inspiration from
- https://regex-generator.olafneumann.org/
    - Cool ideas (step 2 in particular, their implementation is poor but could be built on)
    - The idea of stages of creating a pattern instead of one big one is good
- More complicated features like lookaheads/lookbehinds are rarely necessary
    - We can be pretty free with abstracting away generation and still should be able to generate a matching expression
    - Grouping is nearly entirely unnecessary, we may implement some sort of shorthand for groups of patterns if we investigate the block format

- Help page with documentation or hints?
    - Click drop down/scroll down for step by step of how the regular expression was produced.

Design Outline:

- Multiple stages outlined below
    - Single-screen, ideally non-scrolling with back/forward stage buttons
        - Stages 2 and 3 may be on one screen, or scrolling moves/organizes elements and transitions between the two

1. Inputting test cases and specifying matching/non-matching
2. Block generation
    - Highlight sections, suggested capture groups
    - Blocks can be combined into aliases for groups of blocks
    - Some default blocks such as quantifiers, any amount of whitespace, limited and infinite loops, any alphanumeric character
        - Character class / individual character selection menu is crucial - graphically including or excluding classes or characters
3. Block assembly
    - User assembles blocks into full regular expression
    - Given test cases are shown below assembly, what text is causing them to incorrectly match or not match is highlighted with arrows to the matching offending block or space between blocks
    - Stretch goal / may not be helpful: Site can display generated matching and nonmatching generated test case examples

Questions

- Do we need a login page or should it be open to the internet?
    - Yes, ISO
- Should we limit the user input language for code snippet to just one language? Some set of languages? Or try to keep it more general and possibly reevaluate?
    - Can use things like language grammar specifications as improvements or requirements, but should be things that can be easily extended to / written for other languages
- Discuss user interface general layout and how expressions will be constructed
    - How do we want users to submit their “bad” code snippets?
    - Is generation largely automatic or manual?
        - Manual, moving forward with block idea
    - How many/what kinds of categorization systems?
- What do we want to display and show to users?
    - Regular expressions
        - If tweakable (would need to edit blocks, etc.) yes, otherwise just submit to database

    - Test cases?
    - Advice?
- How do we want to display the regex to users?
    - Only the scratch like building blocks and highlighting parts of code and explaining it (plus upload to database), or an end text output?
        - See “What do we want to display” section above

NOTES

- Possibly getting grammar for languages. Might not be able to get one for matlab because matlab is a proprietary property.
- It would be easy to convert if we have a grammar. It’s ok if we focus on one language as long as it is expandable for other languages. Do what we think is the best.
- Stretch goal: would be helpful for users to understand what is going wrong with the goal, should probably stay as a stretch goal since it could be tricky to implement
- We need to put the site behind a login, so we protect the app from database attacks. A simple mtu login will do since it is using the mtu database