Pre-interview Notes 2

Scientist/Client:Pre-interview Notes 2

- Leo Ureel – Assistant Professor of Computer Science

Meeting Date and Time:

- January 24, Wednesday, 1/24/2024 at 4:20 pm

Meeting Location or Media:

- Rekhi Hall 217

List of Team Members and Interview Roles:

Guaranteed to make it

- Host: Pantaree Prathongkham
- Note-taker 1: Connor Ward
- Note-taker 2: Johnathan Oestringer
- Question Asker 1: Pantaree Prathongkham
- Question Asker 2: Nick Zimanski

Schedule Conflicts (Attempt to make it)

- Question Asker: Chris Torrey (Class ends at 4:15)
- Question Asker: Devon Gosnick (Class ends at 4:15)
- Question Asker: Noah Yacoub (Class)
- Question Asker:

List of Questions in Expected Order:

Parser Formatting

- Will we be expected to create a parser for user-submitted code, or can we use an already existing parser generator (yacc/bison/lemon)?
    - If the former, what languages will we be supporting?

Storing User Submissions

- Should user submissions be stored in a database after feedback is returned?
    - If so, how should those snippets be stored?
- Why are we storing the user's code?
    - How do we expect to use this stored code?
- Should we store the user submitted code and critiques in a separate text file, or just link critiques to line numbers on the page? Should we have the option to output a txt or PDF file with the code and critiques listed out?
- How should we give the user's critiqued code back to them?
    - Give a unique link for each submission and have the user store the link themselves? Or have it linked to a user's account, session, cookies, or ip address?
- When (not necessarily if) should we empty the saved code critiques?
    - Should we wipe a submission X days after it's been critiqued?
    - How can/should we prevent spam from filling the db?

Gauging a Student's Growth Over Time

- Should we be creating a user login authentication system in order to keep track of user's growth over time?
    - How would we create/measure statistical growth within the user's growth? (graph?, past history list?)

Security

- How can we prevent db injection, or spam? We were thinking making a unique link (potentially with time constraints) per user instead of full user authentication system with usernames and passwords
- How can we prevent a user from accessing other user's submissions, critiques, etc.? This implementation changes with how we keep track of a user's submission (by ip, cookies, …)