Use Case 5: Filter what is being searched for

Primary Actor

Searcher (user): A researcher or programmer interested in a GitHub repository

Stakeholders and interests:

The searcher - Needs accurate, fast results
The people the searcher is searching on behalf of (ex: their team)

Preconditions

The user has the URL of a Github repository of interest and a query in mind

Success Guarantee

Results are filtered based on the options (filters) the user selected

Main Success Scenario

1. The user inputs the URL of the desired GitHub repository

2. The user types in the query

3. The user selects filters to show results by (ex: show only results from commits the past week and/or from function names)

4. After clicking submit, the system finds all results for the query matching both the keyword and filters.

5. The system displays these results to the user

Extensions

4a. The repository doesn't exist: The system generates an error and displays it to the user

4b. No results were found:
1. The system generates an error and displays it to the user

2. The system suggests alternate search filters based on what it did find if any

Special Requirements

-Results must be returned quickly and there should be a variety of relevant filters

-Filters should not make the screen looked cluttered and difficult for mobile browser users

Technology and Data Variation List

3a. Filtering may be done in a few ways, but chiefly either by filtering as it searches or first finding all the textual matches, then filtering

3a. Filtering may be done in JavaScript, PHP, or other applicable languages

4a. Searching may use JavaScript, PHP, or other applicable languages.
4b. Searching may be done for a query string or keywords

Frequency of Occurrence

Whenever the uses decides to search

Use Case 6: Enable/disable suggested search terms based on history

Primary Actor

Searcher (user): A researcher or programmer interested in a GitHub repository

Stakeholders and interests:

Searcher - wants their search history to help them with future searches
People the searcher is searching on behalf of (ex: their team)

Preconditions:

The user has made searches and the system has stored the search terms and or filters

Success Guarantee:

The user sees suggested search terms based on their history or not if they disable the feature

Main Success Scenario

1. The user starts typing in the search box

2. Suggested search terms appear based on their search history as they type

3. User selects one of the suggested searches

4. The filters associated with that suggested search become selected on the UI

5. The system searches with the suggested search term(s) and its associated filters (if any) and display results

OR

I. The user selects an option to disable suggested search terms

II. The user starts typing in the search box

III. Suggested search terms do not appear

Extensions

2a. If the user types a totally new search query, no suggested searches should display

Special Requirements

-suggested search terms need to appear very quickly on the UI to be effective as the user types

-suggested search terms should not be take too much of the screen

Technology and Data Variations List

Ia. The option to enable or disable can be checkbox or toggle on the main page of the program or in a separate settings page- may depend on how much screen clutter there is after implementing the majority of the UI

Frequency of Occurrence:

Whenever the user is typing in the search box and there exist relevant search terms and the user has not disabled the feature

Primary Actor

Searcher (user): A researcher or programmer interested in a GitHub repository

Stakeholders and interests:

Searcher - wants useful ways to understand the information of results
People the searcher is searching on behalf of (ex: their team)

Preconditions:

The system has found relevant results for a search

Success Guarantee:

The user can choose to display or hide visualizations and can hover their mouse or click on visualizations for more useful information if any exists.

Main Success Scenario:

1. The user clicks to display visualization

2. The visualization is displayed

3. The user hovers their mouse over visualization and additional info is displayed

4. The user clicks on the on the additional info and the system shows where in the code or commit message the search criteria was found or other relevant info depending on the search and the type of visual

Extensions

2a. If no relevant results were found from the search, no visualization should appear and a message such as “No results found” should display instead

Special Requirements

-Visualizations should scale well on many different monitor sizes

Technology and data variation List:

2a. The visualization can be viewed in a new browser tab or window

3a. Additional info should be suitable to the visualization as well as the search, but specifics will be dependent on other implementation details (what filters there will be for example)

Frequency of Occurrence:

Anytime the user decides to search, but visualizations only appear when there are results returned by the system

Use Case 8: Enable/Disable sound

Primary Actor:

Searcher (user): A researcher or programmer interested in a GitHub repository

Stakeholders and interests:

Searcher - wants sound because it offers a unique type of feedback

Preconditions:

- Searching is working

- UI is behaving in expected ways

Success Guarantees:

When the user has selected the sound options he or she wants to hear, they should hear them when they perform the relevant action

Main Success Scenario:

1. The user goes to a settings page

2. The user selects the sounds he wants to be played

3. The user performs actions that cause the sounds to trigger based on his selection in step 2

Extensions:

1a. If the user finishes selecting settings, they can navigate to other pages of the app

3a. If the user searches with invalid input, rejection sound should play

3b. The user may conduct a search and hear “audibilization” of result data as an option

3c. If a sound option is disabled, sound does not play when the user performs the associated action

Special Requirements:

-Whatever implementation means used for sound should be very compatible with modern browsers

-Sounds should be consistent in overall loudness, with none being too loud or quite in comparison to one another

Technology and data variation list:

3a. For more reusability, there should be many types of sound files supported (ex: .mp3, .wav, .aif, .midi), however more file support should not be of greater value than efficient playback (ex: .mp3, .midi)

Frequency of Occurrence:

Whenever the user performs an action that is associated with a sound and they have that sound option selected