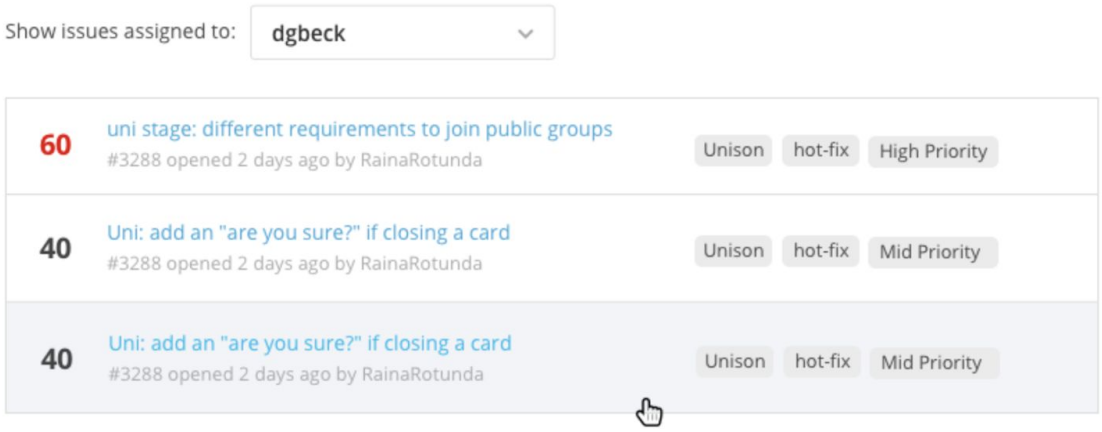
Rotunda SOFTWARE

GitHub Issue Tracker Project

The goal of this project is to provide a means for developers at a given organization to quickly determine the order in which they should work on their assigned GitHub issues.

We will implement a simple web application that lists GitHub issues in a given repository ordered by a priority score. Developers will work on the issues with the highest priority score first, which will appear at the top of the list.

The application will be a single web page that looks like this:



Issues are to be fetched using the GitHub API. Each row represents a GitHub issue and contains the following information about the issue:

- Title
- Number
- Relative date created
- Opener
- Label(s)
- Score

The list of issues should be sorted in descending order by issue **Score**, a calculated value described below, and clicking on any item in the list should take you to the issue's page on github.com in a new tab.

As shown in the mockup, we'll want a control that filters the list based on the GitHub user to which the issues are assigned. The GitHub users available in this filter should also be fetched

through the GitHub API and match those in the organization that owns the repository - in our case, the people in the Rotunda organization. By default, no filter should be applied and all issues should be shown, but a “who” query parameter appended to the web page's url should change the default filter to a particular user. Using this "who" parameter, each developer in the organization will be able to bookmark the issue tracker pre-filtered for their own GitHub user.

Issue Score

The way the score of an issue is calculated should be flexible so that different organizations can customize the calculation for their needs, but we would also like you to implement default logic that works with our way of prioritizing bug issues at Rotunda.

At Rotunda, we use five labels to prioritize bug issues, each with an associated weight. Each issue should in theory have exactly one of the following five labels. (However, the corner cases of issues incorrectly being labeled with more than one of these four labels or none of them should also be considered and handled.)

Critical Priority: Weight is 1000
Very High Priority: Weight is 500
High Priority: Weight is 50
Mid Priority: Weight is 15
Low Priority: Weight is 7

The **Score** of an issue is the weight of its priority label times the floor of the number of working days (i.e. Monday through Friday) since its creation.

For example, one issue assigned as Mid Priority will have 30 as its score if it is created on a Friday at 12:00 PM, and we are looking at the issues web-view on the following Tuesday at 2:00 PM. (Saturday and Sunday are not counted when calculating the issue’s score.)

Scores that are over 100 and considered “overdue”, and should be shown in red (like the score of the first issue in the mockup above).

Deliverable

The deliverable for this project is a GitHub repository (kind of funny, right?) that contains the application source and a README.md that describes the steps to build and run the application. The application should be written in modern JavaScript, and be compatible with node.js v10+.

Please keep track of the time spent on this project. This project was scoped to take around 10-15hours to complete (never more than 20hs).

Finally, please implement this application with attention to code structure and readability.