Group 8

Design Document

Starter Health

Bus Factor - Andrew

Description:

The bus factor refers to the level of risk a project faces based on the knowledge concentration within its team. It measures how reliant a project is on individual team members. A high bus factor is desirable as it indicates that the essential knowledge for a project is distributed among many different team members. A low bus factor means that the essential project knowledge lies with few people. If these people were to be sick or disappear, then the project would be in jeopardy. In our context, the bus factor will be measured by the number of people who do 50% of the work.

Data Needed:

The data needed to calculate the bus factor of a product would consist of contributor IDs, the number of commits, comments, messages, etc associated with that ID, and the overall number of interactions within the project.

Toggleable Features:

The main toggleable feature of this chart would be filtering by type of contribution. You could choose between commits, comments, messages, or any other type of contribution. This is an important feature as commits are not the only way that users can contribute to a project.

Chart:

The bus factor would be best visualized using a pie chart. The bus factor would be displayed as a number above the pie chart while the chart itself is split into that number of slices displaying each user’s contribution percentage

Release Frequency - Joe

Description:

Release Frequency is a metric that determines the timeliness of a project's releases. Generally, more releases could mean often breaking code, or constant updates of software. This metric is great in determining the health of a project over the course of its lifetime. Delays in releases be detrimental to a project and its users, leading to security issues and insatiable versions.

Data Needed:

Release frequency is a function of time so the data needed would simply be a timestamp of a release. Additionally, for the toggleable features, other data needed could be a type of release.

Toggleable Features:

The main toggleable feature for this type of metric would be to filter by the type of release. Examples could include security patches, bug fixes, or feature releases. These features would drastically change the chart represented as software could go long periods of time without feature releases but release bug fixes every other week.
The user should also have the ability to change the view window of the graph.

Chart:

The chart associated with this metric would be a simple line chart, possibly with different points depending on the type of release highlighted. Averages and other stats like quarterly release frequency could be derived from the data.

Change Request Closure Ratio - Chris

Description: Change Request Closure Ratio measures the ratio between the total number of open change requests during a time period versus the total number of change requests closed in that same period. Having a swift decision about a request leads to benefits such as whether the project has enough maintainers, encouraging the closure of change requests that will not be merged, and monitoring long closure times.

Data Needed: This metric would need a time period and the number of closed change requests divided by the number of open pull requests during the same time period.

Toggleable Features: Some features include looking at different time periods and the types of change request/close. Options for the change requests could be whether they are bug fixes or new releases. For close, we can look at if the request was accepted or rejected.

Chart: A line graph would be the visualization plotted on time and number of pull requests. There would be two lines for the number of total pull requests and the number of closed requests.

Time to First Response - Kai

Description:

Time to first response is the time that passes after an activity is first created by somebody (e.g. a pull request, an issue report) before someone else gives feedback. This is important to measuring project health, as projects with lengthy times to first response may make developers feel alienated and ignored. On the other hand, a quick average time to first response is a sign of a healthy project, since development is active and changes can be pushed up and resolved quickly with less delays. For large projects that may take a long time to implement, this metric may have large variations over time as more contributors join or leave the project.

Data Needed:

Not much data is needed to calculate a project's average time to first response, since the metric is relatively simple. An individual activity's time to first response is calculated by the time the activity is first submitted minus the time it takes for a response to be made. With enough data points, we can average these values and get an accurate value of the average time a project's developer would wait before getting a response.

Toggleable Features:

One possible toggleable feature would include a feature that filters out automated responses from bots, as those may inaccurately drive the time to first response down. Another feature would be to sort the time to first response by the type of activity: for example, a pull request may be expected to have a larger time to first response than an issue report, since careful code review may be required for a pull request, while an acknowledgement would suffice for an issue report.

Chart:

Time to first response can be visualized via a simple bar chart that has each bar representing the average time to first response for a certain block time (e.g. a week, a month). Blocking by larger periods of time would give a more accurate impression as it would average out inconsistencies such as weekends and holidays that will have less people working. For this visualization, the x axis would represent time and the y axis would represent the amount of days required for an activity to be responded to.