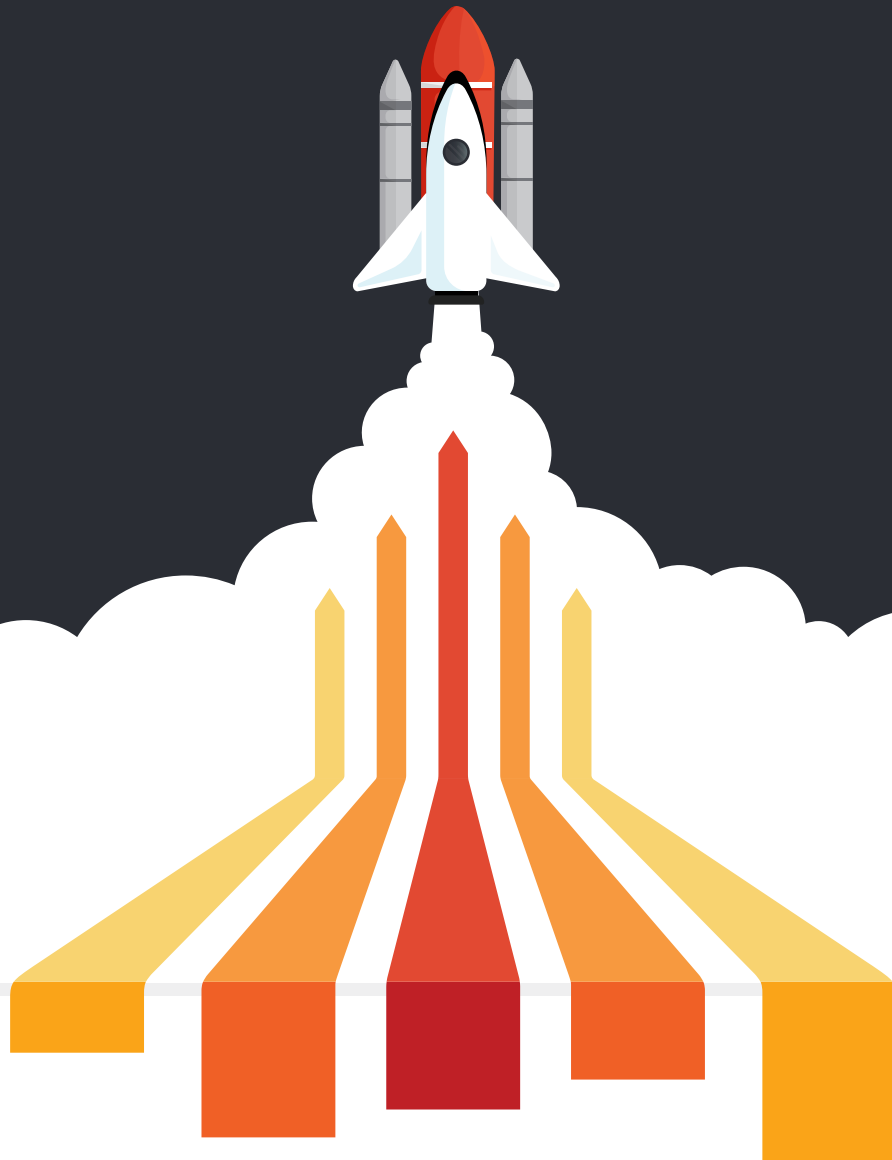


IMPLEMENTING FEEDBACK SCORING IN YOUR BETA PROGRAM



centercode

TABLE OF CONTENTS

INTRODUCTION	1
The Power of Feedback Scoring	1
THE SCORING PROCESS	2
Feedback Weight	2
Impact Score	2
Popularity Score	2
TURNING ON FEEDBACK SCORING	3
IMPLEMENTING FEEDBACK WEIGHT	4
Configuring Feedback Weight	4
Example Severity Weights	4
Using Feedback Weight	5
Filters and Views for Feedback Weight	5
Best Practices for Feedback Weight	6
IMPLEMENTING POPULARITY SCORE	7
Configuring Popularity Score	7
Default Popularity Scoring	7
Using Popularity Score	8
Filters and Views for Popularity Score	8
Best Practices for Popularity Score	9
IMPACT SCORE	10
Using Impact Score	10
Filters and Views for Impact Score	11
Best Practices for Impact Score	12
USE CASES	13
Weekly Reports	13
Trending Feedback	13
Identifying Problem Areas	13
CONCLUSION	14

INTRODUCTION

Prioritizing feedback during a beta test is one of the most important and difficult parts of being a beta manager. A successful beta test means receiving lots of great feedback. This is especially true when feedback types like bug reports and feature requests are made "public", allowing other testers to add their experiences to the original feedback. It doesn't take much for an engaged tester team to generate an overwhelming amount of feedback, but the more feedback that's collected during a beta test, the more labor intensive it becomes to review and organize it. Without being able to effectively review large quantities of tester feedback in real time, beta managers have relied on educated guesses and gut feelings to decide which bug reports and feature requests could have the biggest impact on their product.

To combat this, our platform has powerful Feedback Scoring features that fully automate the process of prioritizing **ongoing feedback**, such as bug reports and feature requests. These features look at a variety of aspects of the feedback (like severity, category, and frequency) and use an algorithm to calculate the impact resolving or implementing the feedback would have on your product. With a few clicks you can turn on these powerful features, which will allow you to make quick, informed decisions about what to do with the feedback from your beta program.

The Power of Feedback Scoring

Being able to instantly assess the impact of your feedback has many advantages, but a few stand out:

- **Prioritize Feedback in Real Time**

One of the most labor-intensive parts of feedback management is figuring out which bugs and feature requests need to be addressed first. Instead of spending hours assessing each bug report, feature request, and discussion thread yourself, Feedback Scoring automatically allows the most important feedback to rise to the top.

- **Generate Reports with Ease**

You likely have a variety of stakeholders that are interested in the progress of your beta tests. Instead of using Excel to prepare complicated, data-heavy materials on the status of your test, Feedback Scoring allows you to create easy-to-understand reports within minutes.



IS THIS RESOURCE FOR YOU?

*This guide is intended for Centercode users on **Impact or Enterprise Edition** that are responsible for configuring projects within their company's Centercode portal. This document will help you understand the value and purpose of Feedback Scoring, and walk you through the steps to implement scoring in your Centercode projects. It will also provide proven best practices our beta managers use to reap the rewards of prioritized feedback.*

THE SCORING PROCESS

When it comes to scoring your feedback, you need to take two different pieces into consideration. The first is the inherent elements of the feedback, such as severity and category. This is called **Feedback Weight**. The second piece is the popularity of the feedback. How many testers are experiencing or discussing the feedback? This is the feedback's **Popularity Score**. These two elements combine to become the feedback's **Impact Score**. The higher the Impact Score, the more important the feedback could be to the overall success of the product.

$$\text{IMPACT SCORE} = \text{Feedback Weight} * \text{Popularity Score}$$

Feedback Weight

To determine the Feedback Weight of a bug report, feature request, or other type of ongoing feedback, the system assigns weights to different aspects of your feedback based on settings you configure within your implementation. The system uses a weight of 1.0 as the baseline. This number can then be adjusted up or down based on the importance you assign to a particular attribute of the feedback.

For example, a critical bug is more valuable than a cosmetic one, so you would give a bug with a critical severity a weight of 2.5 and a cosmetic bug a lower weight of 0.5. By combining different weights, the most significant feedback becomes easy to pick out.

The Centercode platform allows you to assign weights to each single-choice element of your feedback form, such as severity, feature, feedback type, or any other element that you feel is relevant to the weight of the issue or idea.

Popularity Score

In addition to looking at the innate aspects of a piece of feedback, Feedback Scoring also takes into consideration the popularity of your feedback. Our system combines the following factors, called **Occurrences**, when calculating Popularity Score for a piece of feedback:

- **Duplicates** - How many times was the same issue submitted by different testers?
- **Predictive Matches** - Did a user select their submitted feedback as a match to a previously submitted piece of feedback?
- **Votes** - How many testers indicated that they had the same issue or opinion as the submitter?
- **Comments** - How many of the testers contributed to the discussion?
- **Viewers** - How many testers looked at the feedback?

These types of Occurrences are each given a different weight in the system, which combine to give the overall popularity of the bug report, feature request, or discussion thread.

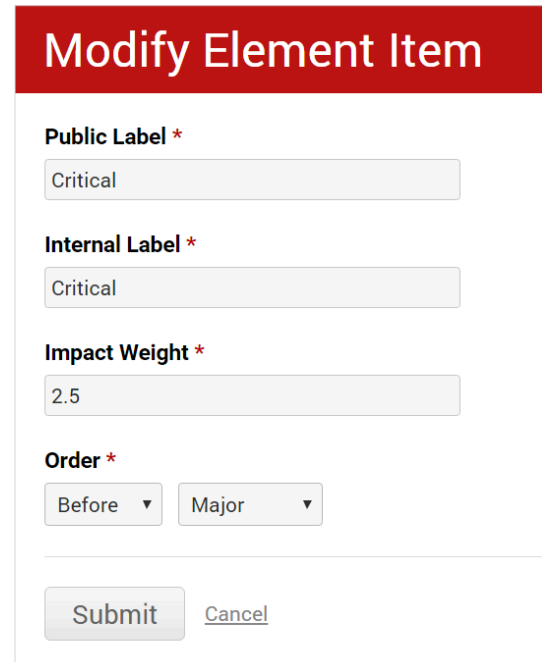
Impact Score

Our system uses an algorithm that combines Feedback Weight and Popularity Score into a single Impact Score. The feedback with the highest Impact Score will have the biggest effect on your product. This will help you make sense of the pool of information that comes from your beta test, and determine where to focus your team's limited resources in order to have the largest positive impact on your product before launch.

TURNING ON FEEDBACK SCORING

Now that we've given you an overview of the scoring process, let's dive into exactly how you set up this system in your Centercode implementation. Feedback Scoring is not turned on automatically in your Centercode portal. To enable Feedback Scoring you need to do three simple things (we give step-by-step directions later):

- 1 Enable Feedback Weight by checking the "Use this Element for Impact Analysis" on the form elements you want to carry a weight, such as severity and category.
- 2 Modify the Feedback Weights for your feedback form elements (see screenshot to the right). All weights will default to the baseline of 1. You'll need to vary the weights for them to start affecting the score of the feedback.
- 3 Make sure your feedback can be listed as public (either by default or after your review). Popularity Score will automatically be calculated for all public feedback. Because private feedback has no views, comments, or predictive occurrences, it cannot have a Popularity Score.



Modify Element Item

Public Label *
Critical

Internal Label *
Critical

Impact Weight *
2.5

Order *
Before ▼ Major ▼

Submit Cancel

Once these elements are in place, the system will automatically calculate the Impact Score for your feedback. Next we'll look at these processes in more detail.

IMPLEMENTING FEEDBACK WEIGHT

To start using Feedback Scoring, you need to configure the Feedback Weights for each element item on your forms to reflect their relative importance.

Configuring Feedback Weight

As an administrator, you can choose the weight associated with each single-choice **Element Item**, assigning greater or lesser weight to each choice depending upon its importance. By default each choice will have a weight of 1.0, so it's important to adjust each of these before going live with your Project (unless of course, you don't want a certain choice to affect the Feedback Weight). Follow the steps below to set up Feedback Weight for an Element Item. *Note: Only single choice elements may be configured to have weight.*

Example Severity Weights

Every company will have slightly different weights for their feedback. These are the severity weights we use for our tests, but we recommend considering all the different elements you're going to ascribe weight to before settling on your weights.

Modify Form Element

Basic Options

Title *

Severity

List Items

Public Label	Internal Label	Weight	Default
Critical	-	2.5	<input type="radio"/>
Major	-	1.2	<input type="radio"/>
Minor	-	1	<input type="radio"/>
Cosmetic	-	0.5	<input type="radio"/>
N/A	-	1	<input type="radio"/>



ADDING / CHANGING WEIGHT OF A FEEDBACK ELEMENT

Watch this video to see these steps in the Centercode platform.

- Click the **Project Tools** link in the left hand menu.
- Click the **Feedback Types** button in the main body area.
- Hover over the Feedback Type you wish to modify and click the **Form** icon.
- Click the **Modify** link below the single-choice element you wish to weigh.
- Under the Advanced Options section, enable **Use this Element for Impact Analysis**.
- Allow the interface to refresh, adding a Weight column to the List Items area.
- Click the Public Label cell to edit an Item, adding a positive decimal value. (Note: your weight should be between 0.1 and 2.5; 1.0 will not influence total weight.)
- Repeat for each Item on the Element.

Using Feedback Weight

When calculating the total weight of a particular piece of feedback, all form elements with weight scores are multiplied together, resulting in your total Feedback Weight for that individual issue or idea. Consider the following example:

FEEDBACK WEIGHT TABLE

Bug ID	Severity	Category	Feedback Weight (*)
BUG-001	Major (1.0)	Installation (1.5)	1.5
BUG-002	Critical (2.0)	Hardware (2.0)	4.0
BUG-003	Major (1.0)	Software (1.0)	1.0
BUG-004	Cosmetic (0.25)	Documentation (0.5)	0.125

In this simplified example, we have applied weight to the severity and category elements of the bug form. Note how critical issues are weighted by a factor of 2.0 over a cosmetic issue that carries a factor of 0.25. The same holds true for category. Hardware issues are of highest priority with a weight of 2.0 while documentation issues only maintain a factor of 0.5. We can see that the issue with the highest overall Feedback Weight is BUG-002, which is a critical hardware issue (a worst case example based on only these values). Our lightest issue is BUG-004, which is a cosmetic documentation issue.

Filters and Views for Feedback Weight

Feedback Weight is built into the Data Engine, allowing you to use it in Feedback Lists, Macros, E-mail Templates, and Reports; basically anywhere filters and views are used. Using Feedback Weight as a Filter item, you're able to generate Feedback Lists relative to a certain value or range (less or greater than #). As a View item, you can sort via this column to see the most or least significant feedback tickets within the current filter.



USING WEIGHT IN FILTERS AND VIEWS

Watch this video to see how to use Feedback Weight in filters and views.

Wherever Filters or Views are used (Feedback, Reports, Dynamic Tags, etc.):

- 1 Select the **Feedback Type** Data Set
- 2 Navigate to **Score** and select **Weight**

Best Practices for Feedback Weight

For any of your feedback forms, be wary of the values associated with your Feedback Weights as they're multiplicative. You'll want to ensure that they scale appropriately from element to element. An errant 0.01 or 5.0 can disrupt the outcome of your weights to an extreme. We recommend using weights between 0.1 and 2.5. Remember, a Feedback Weight of zero will cancel out all of your other scores for that piece of feedback.

Use this multiplicative nature to your benefit -- critical blocking issues should be front and center; pending or back-burnered tickets should be out of sight.

For private or non-collaborative tests, you'll want to leverage the Feedback Scoring functionality associated with weight, however, keep in mind your beta program wouldn't be able to support the collaborative nature of full Impact Scoring. While weight-only numbers will likely be much "flatter" than full Impact Score values (due to finer-tuned calculations), it still allows your feedback form to help guide you in the right direction in regards to prioritizing your new feedback submissions.

Also keep in mind that non-standard questions can support weight as well. For example, the standard fields for Feedback Weight may be Severity, Priority, Feature / Category, etc. but don't forget about other high-value elements like:

- **Feedback Type** - put an emphasis on one type of feedback (bugs) over another (features)
- **Internal Priority** - control priority and weight from "behind the curtain"
- **Status** - escalate or de-escalate the weight based on the lifecycle of the ticket
- **Show-Stopper** - immediately push blocking issues to the top of your priority list
- **Reproduction** - account for reproducibility/frequency in your weighted list



ADDITIONAL NOTES ON FEEDBACK WEIGHT

- *While there is no hard limit to the number of elements that you can use to score weight, trying to use too many elements may be difficult to balance. Find a reasonably modest combination that works well for you.*
- *Feedback Weight is not limited to only participant-driven data. Given Centercode's element level access control, fields like "Internal Priority" which are not available to participants can still be used to influence Feedback Weight.*
- *Feedback Weights are updated in real time across all your feedback (within a type). For example, if you decide to change the weight assigned to a particular answer within a form element, such as "Cannot Reproduce" within the Reproducibility form element, all of the existing Feedback Weights across this entire feedback type will be updated in real time to reflect your conversion.*

IMPLEMENTING POPULARITY SCORE

We are big advocates of cultivating Public Feedback. By setting feedback types like bug reports and suggestions to “public” (either by default or after your review), other beta testers can add their experiences to a single piece of feedback. The platform uses the number of Occurrences (Duplicates, Predictive Matches, Votes, Comments, and unique Viewers) of a bug report or piece of feedback to calculate a Popularity Score, which allows you to compare different pieces of feedback without having to count duplicates and comments yourself. As long as your feedback is public, it has a Popularity Score that you can leverage.

Configuring Popularity Score

Specific values can be set within each Feedback Type that will determine how much emphasis is placed on each type of Occurrence. See the default values in the screenshot below and follow these steps to change those values.

CHANGING POPULARITY SCORING

[Watch this video to see how to change Popularity Score Values in the Centercode platform.](#)

- 1 Click the **Project Tools** link in the left hand menu.
- 2 Click the **Feedback Types** button in the main body area.
- 3 Click the name cell of the desired Feedback Type.
- 4 Click the **Feedback Collaboration Settings** button.
- 5 Adjust the scores as you desire and hit **Submit**.

Default Popularity Scoring

Each Centercode implementation has default values set for each type of Occurrence. These defaults are designed to provide you with the most balanced initial settings possible. You’ll want a firm understanding of Popularity Score before you use the steps above to change any of these default values.

Feedback Collaboration Settings

Predictive/Duplicate Search Configuration

Duplicates *	5
Predictive Matches *	3
Votes *	2
Comments *	0.5
Viewers *	0.1
Predictive Text Match Factor *	50
Duplicate/Search Text Match Factor *	10

Using Popularity Score

Utilizing the values from the table on the previous page, we can see that each piece of feedback will start with a score of 5 (the value set for Duplicates will act as the starting value). From here, the feedback will gain a higher value depending on the frequency of the other Occurrence factors. The table below shows how Popularity Score is calculated and used with our four theoretical bug reports submitted in a project that allows tester collaboration on public feedback.

POPULARITY SCORE TABLE

Bug ID	Reports (5)	Predictive (3)	Votes (2)	Comments(0.5)	Viewers (0.1)	Popularity (+)
BUG-001	1 (5)	1 (3)	0 (0)	5 (2.5)	10 (1)	11
BUG-002	1 (5)	0 (0)	0(0)	2 (1)	2 (0.2)	6
BUG-003	1 (5)	1 (2)	1 (2)	3 (1.5)	6 (0.6)	12
BUG-004	7 (35)	20 (60)	28 (56)	20 (10)	60 (6)	167

Notice BUG-004, highlighted in the example above. One user submitted this bug, and 6 others reported the same issue. You've marked their submissions as duplicates, bringing our total reports of this bug to 7. Each report is valued at 5 points (see: Occurrence Value Table on page 9) so the total popularity score regarding "Reports" (7 x 5) equals 35. Predictive matches are valued at 3 points, so 20 matches equals a popularity score of 60. The remaining factors utilize the same formula, resulting in a total Popularity Score for BUG-004 of 167. Unlike Feedback Weights, Occurrences are added together rather than multiplied, so they don't scale as rapidly as Weights.

Filters and Views for Popularity Score

Similar to Feedback Weight, Popularity Score is built into your portal's existing Data Engine, allowing its use in Feedback Lists, Macros, E-mail Templates, and Reports; basically wherever filters/views are used. As a Filter item, you can generate lists relative to a certain value or range. As a View item, you can sort Popularity Score to see the most or least popular feedback within your filter.



POPULARITY SCORE IN FILTERS AND VIEWS

Watch this video to see how to use Popularity Score in filters and views.

Wherever Filters or Views are used (Feedback, Reports, Dynamic Tags, etc.):

- 1 Select the **Feedback Type** Data Set
- 2 Navigate to **Score** and select **Popularity Score**

Best Practices for Popularity Score

Popularity Score is particularly useful for exposing the frequency of a bug and providing insight to its priority “in the wild”. For example, Priority is usually an internal rating set by an administrator that may or may not be exposed to the end users. While an issue might technically fall into a more minor rating, a particularly prolific issue may warrant a higher priority rating. Popularity Score shows just how pervasive the issue is with your testers.

When used with feature requests or discussions, Popularity Scoring emphasizes the concept of votes. Voting is an extremely intuitive way for your testers to weigh in on potential features without having to articulate the details on their own. Additionally, leveraging collaborative functionality and voting encourages your testers to be more engaged in your test and to voice their opinions via comments. Quantifying these votes and comments via Popularity Score allows you to immediately see which suggestions are most requested by your target market, thus leading you to better decide which features to include in your final (or future) product.

For discussions you can see the “hot topics” or major questions that come up during the beta test. These could influence your support documentation or marketing messaging, depending on the nature of the discussion.



ADDITIONAL NOTES ON POPULARITY SCORE

- While Popularity Score may end in a decimal, it will always round down for simplicity.
- Popularity Scoring is used to suggest previously submitted feedback in both the tester-facing Predictive Match and admin-facing Duplicate Management systems. These results are a blend of text matches (with points associated for each string match) and the Popularity Score.
- You do not have to use all of the Occurrence Values in a Feedback type to obtain a Popularity Score. If a specific feature is disabled (such as voting or comments), that value will simply not be included in the total.
- Popularity Scores are updated in real time across all your feedback (within a type). For example, if you decide to change the value assigned to a particular occurrence, such as votes within bug reports, all of the existing Popularity Scores across this entire feedback type will be updated in real time to reflect your conversion.

IMPACT SCORE

Now that you have a Feedback Weight and Popularity Score for each piece of feedback, it's time to find out how impactful that feedback is to the success of your product. The Centercode system does this by assigning an Impact Score, which is generated by multiplying Feedback Weight by Popularity Score. The full algorithm for Impact Score is below:

$$\text{IMPACT SCORE} = \text{FLOOR} \left(\begin{aligned} & (\text{Viewers} * \text{VIEWER WEIGHT}) + \\ & (\text{Comments} * \text{COMMENT WEIGHT}) + \\ & (\text{Reports} * \text{REPORTS WEIGHT}) + \\ & (\text{Votes} * \text{VOTES WEIGHT}) + \\ & (\text{Predictive} * \text{PREDICTIVE WEIGHT}) \end{aligned} \right) * \text{Custom Element Item}^n \text{ WEIGHT}$$

Note that each of the bold values can be customized on a per Feedback Type basis; there is no hard limit to the number of custom element items used.

Using Impact Score

To illustrate Impact Score, revisit the examples we used for Feedback Weight and Popularity Score. To get the Impact Score for bugs 001 through 004, we use the equation mentioned above:

IMPACT SCORE RESULTS

Bug ID	Feedback Weight	Popularity	Feedback Impact (*)
BUG-001	1.5	11	16
BUG-002	4.0	6	24
BUG-003	1	12	12
BUG-004	0.125	167	20

Impact Score gives you the best idea of which issues are most impactful to your beta test and your product. For example, you may have a bug with a high Popularity Score but a low Feedback Weight, like BUG-004. After seeing the Popularity score of 167 you may assume that this bug is a major issue that deserves significant time, but when combined with the reductive weight, we can see its total Impact Score is 20, which is lower in the grand scheme of things when compared to BUG-002's Impact Score of 24. It's not often that your raw Popularity Scores or Weights will "trick" you, but Impact Score, being the final result of the variables, is the most accurate and directly valuable way to know which issues to focus on first.

Filters and Views for Impact Score

Once you've implemented your Feedback Weights (and optionally adjusted Popularity factors), you can access Impact Scores anywhere the Feedback Data Engine is available (Views, Filters, Reports, etc.). Just like Popularity Score and Weight above, filtering for Impact Score allows you to generate lists relative to certain Impact values or ranges (less or greater than #). As a View item, you can sort by Impact Score in order to see the most or least significant feedback issues within the current filter. Note that this is different from Feedback Weight which focuses on a singular piece of feedback — the combination of collaboration and weight lets you hone in on the topic of the feedback, boiling significantly more information into a single, intelligently prioritized source.



IMPACT SCORE IN FILTERS AND VIEWS

Watch this video to see how to use Impact Score in filters and views.

Wherever Filters or Views are used (Feedback, Reports, Dynamic Tags, etc.):

- 1 Select the **Feedback Type** Data Set
- 2 Navigate to **Score** and select **Impact Score**

Bug Report List (Refresh Now)

<div> All Feedback ▼ <input type="checkbox"/> Archived <input type="checkbox"/> Duplicates <Unsaved Filter> (10) ▼ New Feedback Scoring ▼ </div>							
<input type="checkbox"/>	ID	Title	Severity	Popularity Score	Weight	Impact Score	Feature
<input type="checkbox"/>	BUG-022	Online Documentation gives a 404	Critical	17	1.25	22	Documentation
<input type="checkbox"/>	BUG-021	Apps from the store won't install	Critical	9	2.5	22	Software
<input type="checkbox"/>	BUG-007	Crashing when Loading a Preset	Major	17	1.2	21	Software
<input type="checkbox"/>	BUG-017	The color keeps wiping off	Major	6	2.4	14	Hardware
<input type="checkbox"/>	BUG-016 ✓	Crashing Windows XP	Major	5	2.4	13	Hardware

Best Practices for Impact Score

It's important to understand that Impact Score isn't just an obscure value applied by the system. The two variables (Feedback Weight and Popularity Score) both come from the popular business practice of honing into what tasks are most important to accomplish in order to successfully meet your broader goals. While you will likely do some tweaking of their values (which will update the existing results in real-time), Impact Score is the most effective way to accurately convey what feedback is most important to the success of your product.

While integrating Impact Score into your day-to-day operations is highly valuable, Impact Score can offer the biggest payoff when used for external prioritization. Reporting on issues or suggestions at the top of your list is an ideal way to bring high-priority information to stakeholders with limited attention spans or time. Executives and engineers alike will appreciate the clear "Top #" lists you can provide based on Impact Score, and you won't have to prioritize them based on arbitrary numbers or gut feelings. In addition, since the Centercode platform is built to keep these high priority external recipients up to speed, the reporting functionality can easily generate these "Top #" lists and distribute them automatically in a variety of formats.

Once you have your Impact Scoring established, template it! Build or copy it into your **Master Template**. With a bit of regular housekeeping, it's easy to keep your configuration in good shape. This will allow all the new or upcoming Projects in your program to leverage the same advanced, informative functionality that we've covered in this document.



ADDITIONAL NOTES ON IMPACT SCORE

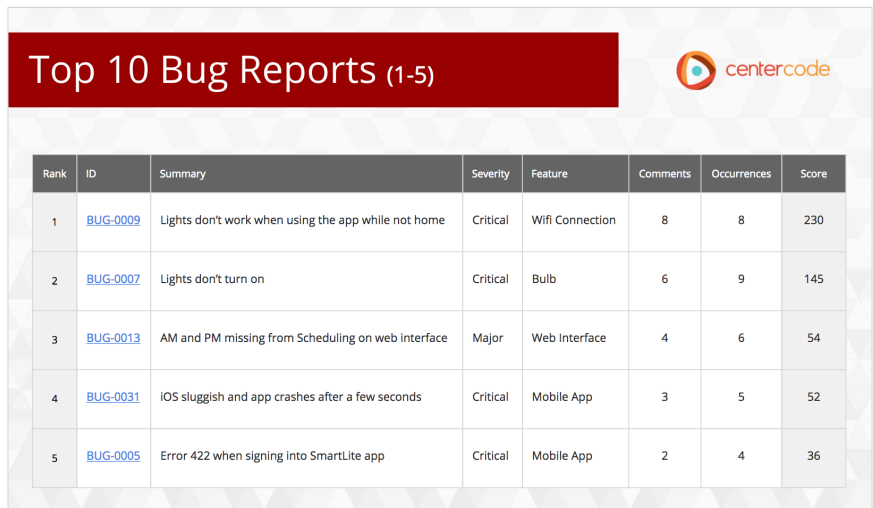
- *Like Popularity Score, Impact Score is rounded down for simplicity.*
- *Feedback Filters greatly enhance the functionality of scoring. For example, you can view the most impactful issues for a specific category, in a specific status, or recently submitted.*
- *Adding a Grouping to your view allows you to sort within group items. For example, using a Category custom field as a View Group will allow you to sort your Feedback in such a way that you can see the most impactful issues for each Category.*
- *While Impact Score is an excellent gauge, you should still review all Feedback if possible.*
- *The relative distance between any two Impact Scores can be very meaningful. For example, if the first issue scores 200, the second 190, and the third 40, it's clear that the first two issues are much more impactful than the third and following.*
- *Impact Scores are updated in real time across all your feedback (within a type). For example, if you decide to change around the values associated with Popularity Scoring or Feedback Weight for a particular answer within a form element, all of the existing Impact Scores across an entire Feedback Type will be updated in real time to reflect your conversion.*

USE CASES

Our Managed Betas Team has been using Feedback Scoring to deliver easy-to-understand, prioritized data in each and every one of our beta tests. Here are a few examples of how they're using Feedback Scoring to help our services clients.

Weekly Reports

Every week our beta managers send a Weekly Report to the stakeholders for their beta test. These reports provide a snapshot of the test's progress. A mainstay of the Weekly Reports is a list of the Top 10 Bugs and Top 10 Feature Requests from the test. Using Feedback Scoring, our team has up-to-the-minute lists of the most relevant feedback in the test, so they can easily pull the most impactful feedback for our clients.



Rank	ID	Summary	Severity	Feature	Comments	Occurrences	Score
1	BUG-0009	Lights don't work when using the app while not home	Critical	Wifi Connection	8	8	230
2	BUG-0007	Lights don't turn on	Critical	Bulb	6	9	145
3	BUG-0013	AM and PM missing from Scheduling on web interface	Major	Web Interface	4	6	54
4	BUG-0031	IOS sluggish and app crashes after a few seconds	Critical	Mobile App	3	5	52
5	BUG-0005	Error 422 when signing into SmartLite app	Critical	Mobile App	2	4	36

Trending Feedback

There can be a lot going on during your beta test. Impact Score gives you an easy way to see the feedback that's garnering a lot of attention and traction with your tester team. Trending discussions that are sparking a lot of debate could require someone from your team to step in and address points of contention or help users find a workaround to their issue.

Identifying Problem Areas

You can also use Impact Score to dive deeper into your feedback. If you organize your bugs and feature requests based on category or feature, you can combine that information with Impact Score to find the most problematic areas of your product. Maybe all of your high impact bugs are taking place during the installation process, while your primary features are working smoothly. Perhaps your top feature requests all revolve around your app UI. Impact Score can help you identify the parts of your product that need the most attention, along with the specific things you need to do to make them better.

CONCLUSION

The benefits of learning how to leverage Feedback Scoring in your beta program cannot be overstated. Adding **Feedback Weight**, **Popularity Score**, and **Impact Score** to your beta test will dramatically affect the value your beta test can bring to your product's development. This functionality will help you save valuable time during your beta test, and make informed decisions once your test is complete.

We hope that this resource has helped explain the concepts behind our Feedback Scoring features and how to put the right pieces in place to leverage this powerful tool in your next test. If you need more help getting Feedback Scoring in place or simply have other questions about our platform, [schedule a coaching session](#) with a member of our support team.

Learn more about how to use
FEEDBACK SCORING
in our platform

SCHEDULE A COACHING SESSION

