

# A guide to our sprint process,

# Trello and reference for best practices

# Why Trello

Having previously used FogBugz, we were struggling with clarity and unable to see exactly where we were when faced with an excessive workload stretching more than 12 months into the future. To be fair, FogBugz was never designed for this, and is a struggle to make it fit an agile process with sprints.

Inspiration for using Trello came from a [blog post](https://community.uservoice.com/blog/trello-google-docs-product-management/) by the (then) CEO of Uservoice, a well-known platform for gathering feedback on web applications from its customers. They had some of the same problems that we had, and posted their Trello success story online.

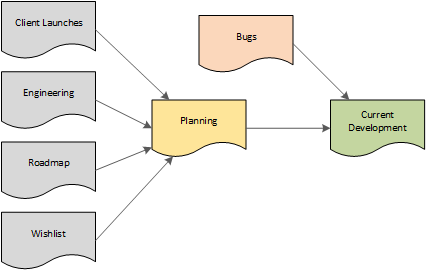
# Trello Basics

To get a basic knowledge of what Trello boards and cards are, see the [Getting Started guide](http://help.trello.com/customer/portal/articles/891656-getting-started-with-trello) in the [help section](http://help.trello.com/).

# The Boards

We have implemented a similar model to Uservoice, with some customisation to fit our company and the way we work.

Here is a visual representation of the boards we use:



**Bugs** is for, erm, bugs!

**Client Launches** is where details of clients who will be launching on TT first originate

**Current Development** is work to be undertaken by developers in the current sprint (two week period). It also contains development work being QA’d

**Engineering** is for any ideas from developers on “behind the scenes” stuff which will help products perform better, or make them easier to maintain

**Planning** should contain work for upcoming sprints, organised in planning meetings between the Technical Director and the Ops Director, with work taken from the Roadmap, Engineering and Bugs boards

**Roadmap** is the future vision for all products, with lists for each of the upcoming quarters (normal year quarters, not financial quarters)

**Wishlist** is for new ideas from people in the business, or from clients

# The Process

## Creating a new card

Here’s a list of where cards should be added depending on the scenario:

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Board** | **List** |
| A brand new bug not seen before, so not in the bugs board or current dev board | Bugs | Inbox |
| A recurrence of a bug you’ve seen before which had already been fixed and deployed | Bugs | Inbox |
| A critical issue (e.g. nobody can log in to TT) – add details to card and notify the Technical Director (or if they’re not available, any of the dev team) by either speaking to them, telephoning them or Lync’ing them | Bugs | Inbox |
| A new idea for any of our products | Wishlist | Product Ideas |
| A request for bespoke functionality from a Client \* | Client Launches | Month |
| An idea from a developer for how to improve the codebase and/or architecture of our products | Engineering | To Do |
| \* = and discuss in your kick-off meeting |  |  |

## Adding information to a card

Different cards require different information. E.g. a bug needs a lot of specific information, but a new idea can be very “big picture” as the UX and speccing will be done on that before it reaches the developers.

General rules are:

* Be concise and precise – short and sweet but with all the information needed, no room for ambiguous words/phrases, unknown acronyms, pleasantries, chit chat etc
* If there’s a deadline date, add a Due Date to the card
* If a card needs attention from a particular user, assign the card to that user
* When adding a bug, ensure all the details from the [bug card template](https://trello.com/c/hba19FrY/1-clientname-short-description-of-the-problem-blank) are included. Create a copy of that card, modify it and move it to the Bugs board Inbox list, at least until you learn by heart all the details that need to be included
* If a card involves multiple tasks, use the checklist feature so the developers can tick them off as they complete them

## Planning

Sprints run as two week phases. On the first day of a sprint, the Technical and Operations teams will meet to discuss and agree priorities for the next sprint.

They will review the following boards and agree on work to be done in the next sprint:

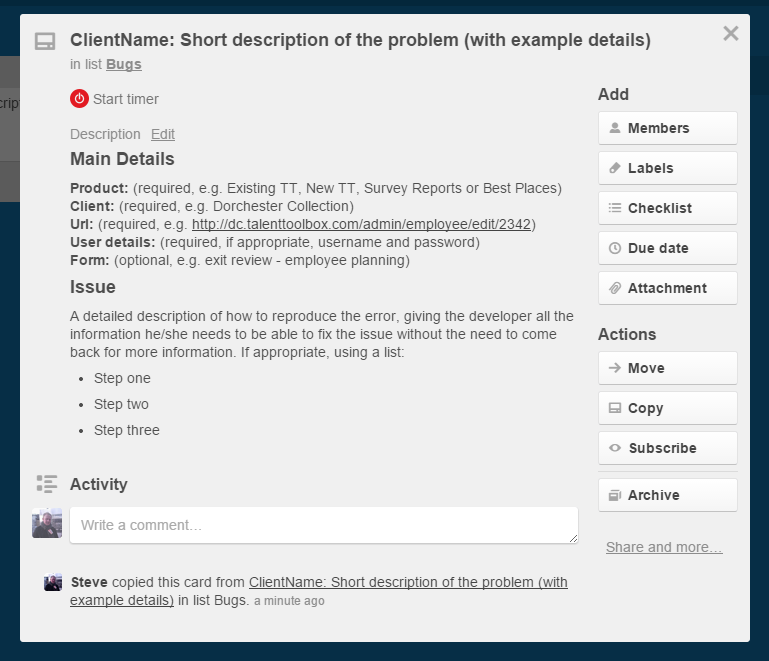
* Current development – in case anything from this sprint isn’t going to be finished
* Planning – anything planned previously for the next sprint
* Bugs
* Engineering
* Product Roadmap

At the end of this meeting, the “Ready” list in the Planning board should be full with work that can reasonably be expected to be completed by developers in the next sprint, including time (with slack) for the dedicated bug fixer for each week.

If any changes to the roadmap are made as part of this meeting, the Technical Director will update the company with details on the change.

## New Bugs

New bugs should be added directly to the “Bugs” board “Inbox” list. And unless the bug is ultra simple, use the bug template card from the “Card Templates” board, to give the developer all the information they need. If information is missing, the developer will move the card into the “Needs Input” list and the card will not be actioned until information is received.



Priority should be assigned (as per the labels section further below in this document), and any notes added that will help us decide a due date. For any bugs that need to be fixed within the next two days, @ mention the dedicated bug fixer, who will action the bug asap, add the pink label to highlight this is a bug card, and take the card into the “Next Up (Bugs)” list in the “Current Development” board.

## Build

The Current Development board is the board which developers work from throughout the active sprint. At the start of the sprint, they get together to discuss what needs to be done and divide the work up. At the end of the sprint, the developers (and anybody else in the company) can sit in the sprint review meeting to see progress, where developers can demo the work they’ve done in the sprint.

Do not add new cards straight into the Current Development board, because that would effectively be moving the goalposts for the developers for the current sprint, and could result in a difficult re-planning process.

Developers pick up cards to take on from the “Next Up” lists in the Current Development board, and move them into the “In Progress” list once begun (with work being done on a topic branch in the version control system), and should assign themselves to the card if not done so already.

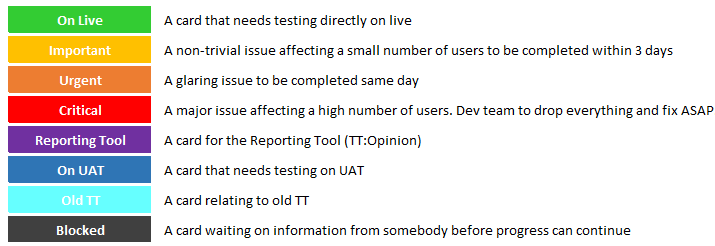
If a developer requires more information before he/she can begin on a card, the card should be moved back into the “Next Up” list, assigned to the user who can give the information required, and a comment should be added @ mentioning that user with information on the extra details that are required.

When a card from the Current Development board is completed by a developer, it should be moved into the “Queued for QA“ list. If a code change has been made, that change needs to be tested on UAT, and the “On UAT” label applied to the card. The developer will also need to merge his/her topic branch into the “uat” branch. If the completed card was a data change or anything that needs to be done directly to live only, add the “On Live” label. This allows the tester to quickly see where to test without having to trawl through the comments.

Developers should remember that the “uat” branch is expected to be ready to go into production if it passes QA, so incomplete work not ready for production should not be merged into the “uat” branch until ready.

## Labels - for priority and test locations

Make sure these are used so it’s easy to see what’s going on at a glance



## Quality Assurance

Testers should monitor the “Ready to QA” list one or more times per day, and test cards completed by the dev team on the UAT addresses (or live, if the issue/fix is for the production environment only). Once sufficiently tested, if the card is verified

by the tester to be complete, the card should be moved by the tester either:

* The “Passed QA” list if tested on UAT
* The “Live” list for the current sprint if the card was tested directly on live

If the card isn’t complete, or if the work has introduced a new issue in the same area, the tester should move the card back into the relevant “Next Up” list (the “Next Up (Bugs)” list if it’s a bug, otherwise the “Next Up (New Work)” list) and @ mention the developers who completed the work. It’s then up to the developer to look at the issue detailed by the tester. If the issue was related to the work done for the card, the developer should fix and put back for QA through the normal process. If it’s a completely separate issue, the developer should create a new card in the Bugs board “Inbox” list.

## Deployment

Deployments can be performed by developers when one or more cards are ready to go into production, and the “Ready to QA” list is empty. Developers will need to be aware of what they’re deploying (in case it involves deploying to more than one environment, e.g. Talent Toolbox and Survey Reporting), and move the deployed cards into the “Live (sprint ending {last date of sprint})” list.

It’s the responsibility of the deploying developer to do basic checks once the deployment has been done, to ensure the deployment has gone smoothly.

This list will be reviewed in the sprint review session to see what went successfully into production.

## Sprint Review

On the final day of the sprint, the Technical and Operations teams meet again to review the state of the Current Development board.

* Review how the past sprint went
  + Teams to challenge each other to find a solution if avoidable problems have occurred during the sprint
* Discuss the impact of any work not completed
* Developers can demo any relevant changes to the Operations teams to help ensure they know all about the latest developments in TT

# Trello Best Practices

* Keep cards concise. Pleasantries are nice and all, but there’s no need for “Hi”s and “Thanks” in cards – developers know you appreciate them!
* Make sure important details are kept in the description field, rather than the comments. If an important piece of information is added once a card has already been started, it’s best to update the description and @ mention users already involved (if any) explaining that more info has been added to the description.
* Use the @ symbol to mention another user when a card needs somebody else’s attention. This can be done in the description or in comments, and will give the other user a notification.
* Don’t put deadlines in descriptions/comments. Use the Due Date field, then it’s easy to see without going into the card, and reminder notifications/emails get sent close to the due date.
* Use clear generic terms that aren’t open to interpretation
* For bugs:
  + Ideally, copy and customise the bug template card so no information is left out (see [here](https://trello.com/c/hba19FrY/1-clientname-short-description-of-the-problem-blank)). Make sure you move it to the Bugs board Inbox list too.
  + Make it clear which client & which type of form so time isn’t wasted investigating a review form when the issue is on the induction form
  + Include the steps to reproduce unless it's really obvious. Follow them back yourself to make sure they make sense.
  + Provide username where possible - saves developers having to hunt the database for the one to use
  + If it relates to a particular question in a review form, which stage, page and question?
  + Add annotated screenshots where the words may not be sufficient or clear
  + Before marking a bug as client-specific, check if it's client-specific or if it impacts all clients (a global bug)
* Re-read the card before saving. Make sure it makes sense and it's simple and clear
* Avoid client-specific initialisations/acronyms or ones that may confuse - think, would a new developer know what a MYR KPO is?
* If the \*exact same thing\* has been done before, give clear examples as to who for or link to an old card
* Don't "recycle" cards (reopen a completed card) just because it's the same/very similar as a new one. It takes longer to determine what needs to be done (as you end up reading through history) and skews the figures for resolved cards - if you think an old card will have useful information in it, link to it instead.
* Developers should add version control branch details to cards