Design QA

The Importance of Design QA in Product Design

Great customer experiences don't happen by accident. In product design, the customer experience encompasses everything that the product team does; development, design, DevOps, and QA — everyone's role impacts the customer experience but especially the design and user experience.

Why is this important?
Consistency is one of the main tenets of good product design. Over time as a product is designed and developed inconsistencies inevitably crop up. Over time this can add up and turn into "design debt."

Design Debt *affects the integrity of the user experience. This is what happens when a bunch of incremental changes collect over time and yield a disjointed, inconsistent, and patched-together experience.*

Design QA can be a challenge
There are other fundamental flaws that can make Design QA a challenge:

- Teams or companies do not understand or value design enough to create a process that fosters good design outcomes → "the feature works"
- People don't see the difference between a design and a poorly coded version of it → "looks good enough to me"
- The focus of teams is on speed and feature delivery — and visual integrity is easier to cut than coding → "we don't have time for it"

Speed vs. quality
As a product team works together, they sometimes run the risk of getting into feature delivery mode and shipping. Teams can lose sight of the bigger picture and attention to detail while trying to close as many tickets as possible before the sprint ends. As a team races to the end of the sprint and increases velocity, this may create a scenario where the integrity of the design implementation can fall to the wayside as a "time-saving" measure.

This is where Design QA comes in!

What is Design QA?
Design QA (QA = Quality Assurance) is merely a step in the process between development and testing. It's a chance for the designer to:

- Review the coded version of the UI prior to testing
- Work with the developer(s) to make updates to the UI in code

Design QA as part of the workflow

Our standard workflow might look like some version shown below. If your team works in any sort of sprint and moves a ticket from one part of the development lifecycle to another, our (design) work may be in any one of these categories at any given time.

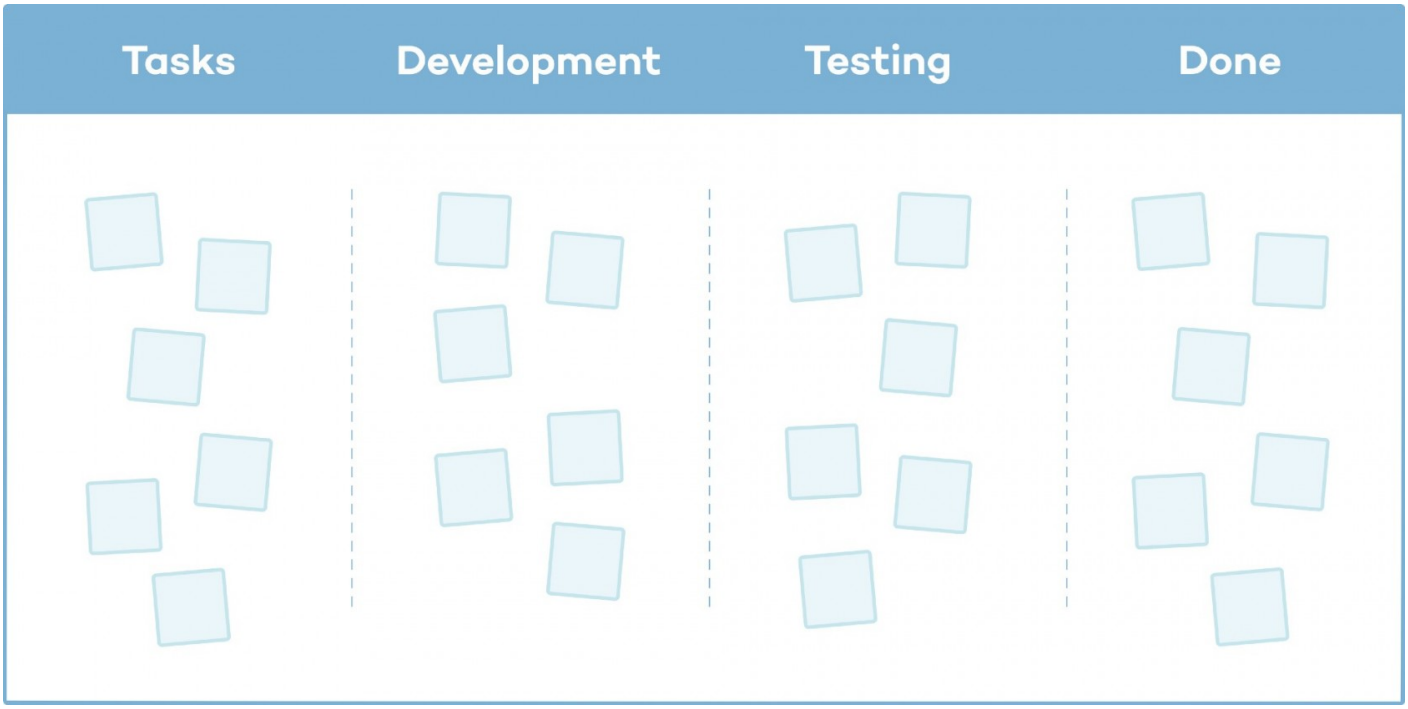


Fig: Normal Workflow

In a workflow like this, how do we ensure the integrity of design? When a ticket is done in development, it's usually up to the developer working on the ticket — or the product manager to move it to testing. Sure, teams can get into the habit of trying to do some version of Design QA without having a step in the process for it, but that breaks down quickly; people forget or decide for themselves that the design implementation is good.

The more important question is: if design implementation is essential, why don't we just have a step for it?

By making Design QA a deliberate step in the process, it can't be skipped. It's also recognition that design implementation is an important part of the process that the team values. When we account for Design QA in the process, the workflow mentioned above, now looks more like this.

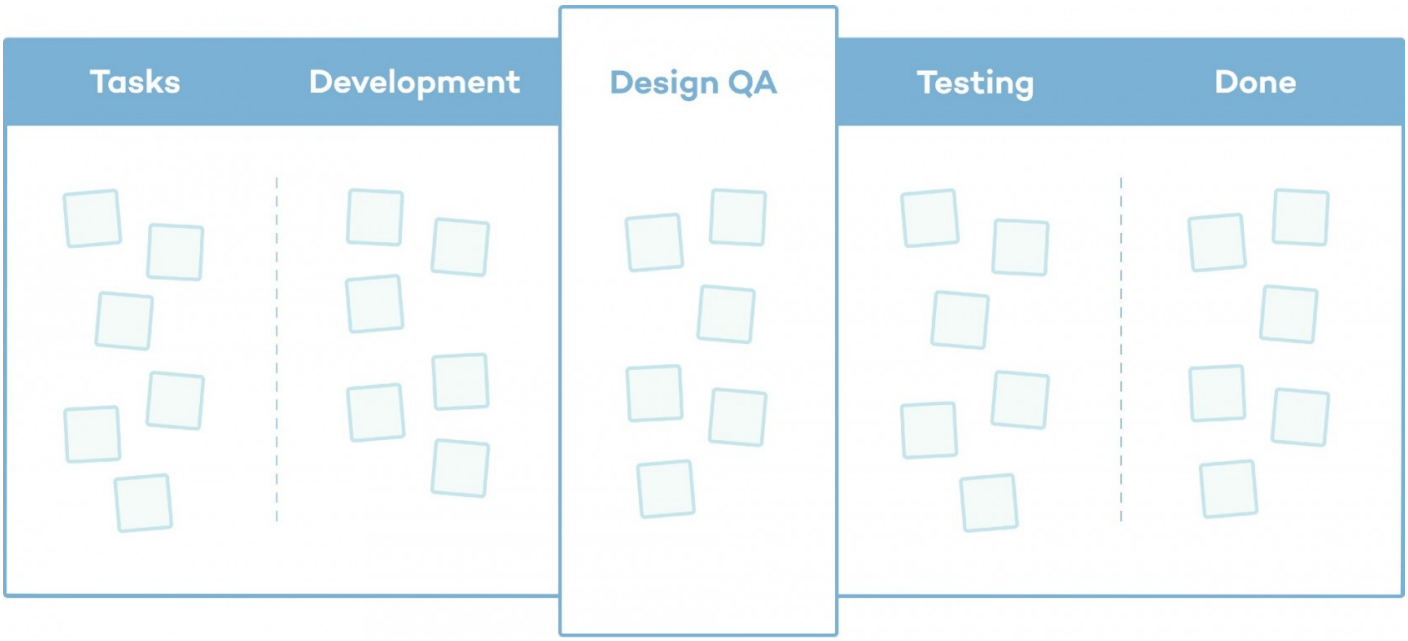


Fig: Inclusion of Design QA in our Normal workflow

Include developers in the design process
Just as it helps for designers to be involved when designs are being coded, it’s also key to include developers in the design process.

To include developers in the design process, you can do things like:

- Discuss the requirements of a feature before beginning design to suss out technical details that may impact design decision
- Sketch out initial design solutions together
- Share designs with developers throughout the process to get feedback

Most of the challenges we face in product design and development can be solved with mutual respect, communication and honesty.

Happy testing!!

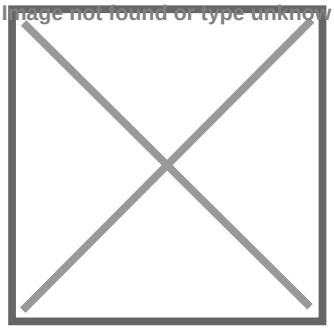
Design QA Checklist

Basics of Design Fundamentals

Balance:

Balance can be symmetrical (with items of equal weight on either side of a centerline) or asymmetrical) (with items of different weights laid out in relation to a line that may or may not be centered.))

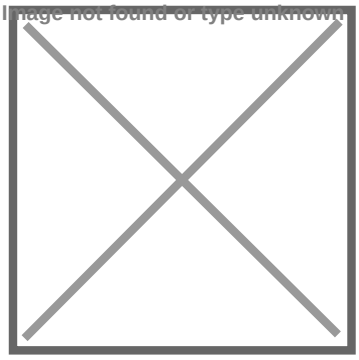
Image not found or type unknow



Emphasis:

Emphasis causes certain parts of a design to stand out compared to other elements. Conversely, it can also be used to minimize how many elements stand out.

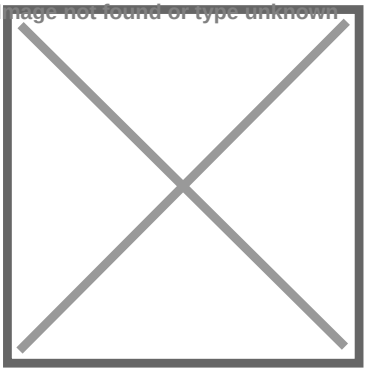
Image not found or type unknown



Hierarchy:

Hierarchy refers to the importance of elements within a design. The most important elements should appear to be the most important, and vice versa.

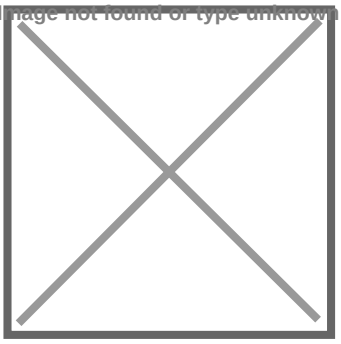
Image not found or type unknown



Repetition:

Repetition reinforces an idea or perception. It can be done via things like using the same format for header, reusing the same colors, image, or similar choices.

Image not found or type unknown



Things to look while checking mocks (Checklist):

S.N	Checklist	Remarks
1.	Need to have a good understanding of the project flow and follow the flow within the mock.	
2.	Have a basic knowledge of web and mobile elements.	
3.	Looks and feel of the mocks. (Branding and theme of the application) (Eye pleasing)	
4.	Always try to reduce the clicks/taps.	
5.	Design should be minimalist and simple.	
6.	Reduce overload of information on a single page.	
7.	Check the size(pixels) of each element. (Literally every elements of the page)	

8.	Check for Typos!! (Check the grammar errors and spellings) Prioritize wording of the app's user.(US-eng, UK-eng, etc)	
9.	Design should use a minimum number of colors and fonts.	
10.	Icons used should be relevant and the size and weight of the icons should be consistent.	
11.	Check for the Symmetry and Alignment of elements.	
12.	Spacing/padding/margin should be checked.	
13.	Alerts of system generated messages should have similar design and Alerts must be informative. (Red- Danger, Green- Ok, Blue- Informative)	
14.	Level of shadow, highlight, opacity, Contrast, Animations should be checked.	
15.	Prototype should have relevant data displayed.	
16.	Check the flow and links of the prototype.	
17.	Buttons/Input fields/drop downs/ other elements should be the same/similar throughout the application.	
18.	Check for the texts. (Title Case/ uppercase/ lowercase/Bold/italics etc)	
19.	If the application uses a custom loader, try using similar loaders in the overall application.	
20.	In case of long text it might require truncating.	
21.	Think of the empty cases when there is no data on the application. Try including proper message/info if the page is empty.	
22.	Try being dumb customer of your own application.	