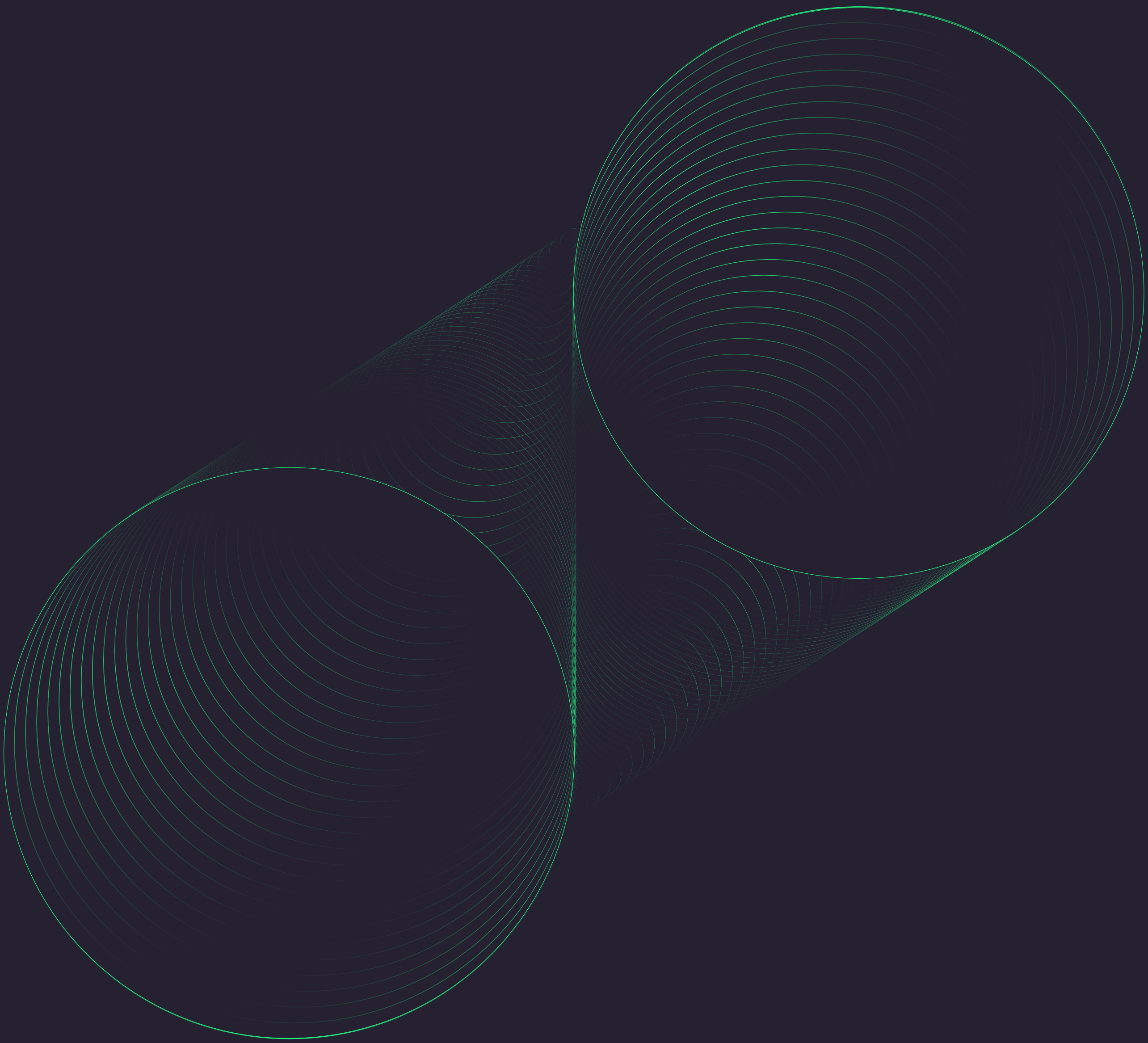
UXPin

The Guide to Prototyping Process & Fidelity



UXPin

The Guide to Prototyping Process & Fidelity

Index

Authors

Ben Gremillion is a Content Designer at UXPin specializing in responsive design. Previously, he was a Design Writer at ZURB. He has earned an Adobe Certification and knows CSS, HTML, regex, PHP, MySQL, and other impressive-sounding acronyms. He also builds and maintains a CMS for webcomic artists, and participates in bi-annual NaNoWriMo challenges.

With a passion for writing and an interest in everything anything related to design or technology, Matt Ellis found freelance writing best suited his skills and allowed him to be paid for his curiosity. Having worked with various design and tech companies in the past, he feels quite at home at UXPin as the go-to writer, researcher, and editor. When he's not writing, Matt loves to travel, another byproduct of curiosity.

Co-founder and head of product, Kamil previously worked as a UX/UI Designer at Grupa Nokaut. He studied software engineering in university, but design and psychology have always been his greatest passions.
Follow me on Twitter @ziebak

A Practical Introduction to UX Prototyping

Prototyping is the cornerstone of the design process, requiring a blend of creativity and practicality. We create and test the features that are most appropriate. Finally, our vision comes to life.

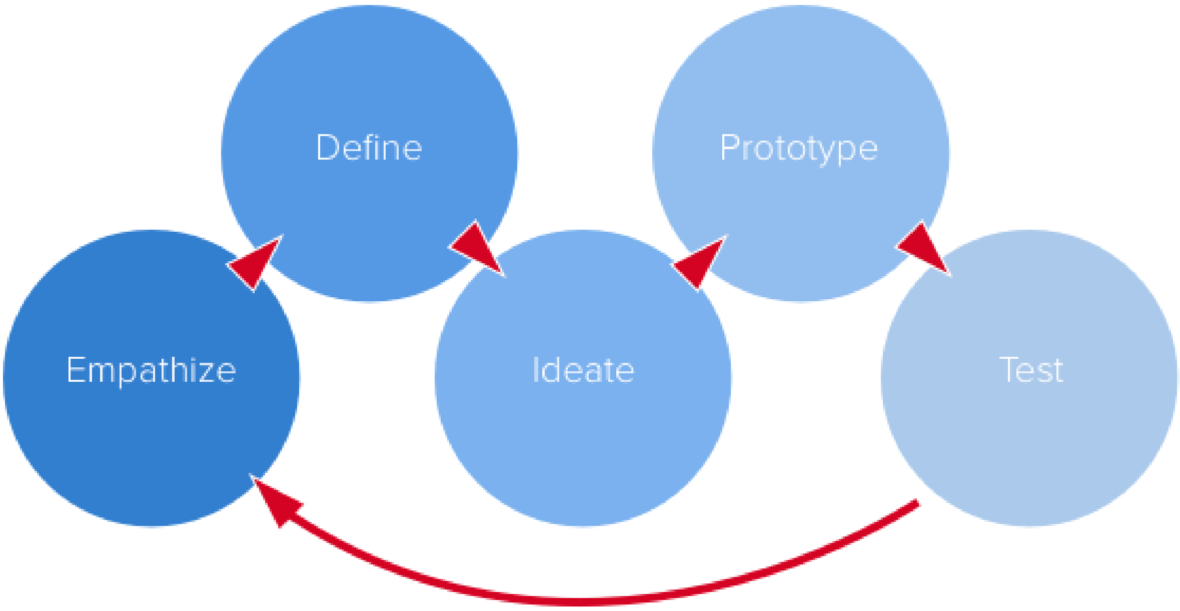


Photo credit: Adapted from Naked Design Labs

As shown in the above illustration, prototyping gives you something functional to test with users so you can iterate accurately. But while prototyping is an important technique, designers can choose from a variety of styles, methods, and processes.

The fidelity of your prototype helps determine the appropriate process.

When we talk about fidelity, we're referring to the functionality and visual detail. How close is your prototype to the final product? Low fidelity prototypes (like those created on paper) translate rough design concepts into something tangible and testable. High fidelity prototypes, on the other hand, are usually later iterations created in prototyping tools to help refine interactions.

Multiple factors affect the level of fidelity to build into a prototype, including:

- Client & company budget in terms of time and money
- Company's resources and specializations (e.g. simpler prototypes can be more feasible for startups)
- Tech savviness and creativity of users (less savvy users might have difficulty imagining a low-fidelity prototype as the final product)
- Potential conflicts of interest (e.g. if the person prototyping will also help in coding the product)
- Responsive design (lower fidelity prototypes aren't as helpful for visualizing how a design adapts to different devices)

Designers can also choose to build their prototype from wireframes midway in the design process, from more detailed mockups later on, or jump right into prototyping from the very start.

As described in the free *Ultimate Guide to Prototyping*, there is no "best" way to prototype, only the right tool and process for the job at hand.

In this guide, we'll explore the spectrum of prototyping processes, fidelities, and tools so you can find the best fit for your project. We've written it to be as practical as possible, so if you find it helpful, feel free to share.

Sincerely,

Jerry Cao
(co-written with Matt Ellis & Kamil Zieba)

When to Start Prototyping: 3 Points of Convergence

There's no green light that will magically blink when it's time to start prototyping. *When* to prototype is the subject of just as much debate as *how* to prototype. The traditional linear process looks something like this:

1. **Sketching** – Brainstorm by drawing quick and rough sketches on paper.

2. **Wireframing** – Start laying out the skeletal framework with boxes and rough shapes.

3. **Mockups** – Inject detail into wireframes with colors, typographies, photos, and other visual design elements.

4. **Prototyping** – Add interactivity to mockups by stitching screens together for basic prototypes or adding animations/interactions for advanced prototypes.

5. **Development** – Code in language of choice to turn the prototype into the final product.

But with the popularization of new ideas such as Lean UX and rapid prototyping, plus the school of thought that wants to get into coding as quickly as possible, this traditional sequential method is becoming outdated.

Let's look at some new variations, and explore their advantages and disadvantages.

Sketches/Wireframing & Prototyping

Some designers prefer to start prototyping right away, and for good reason.

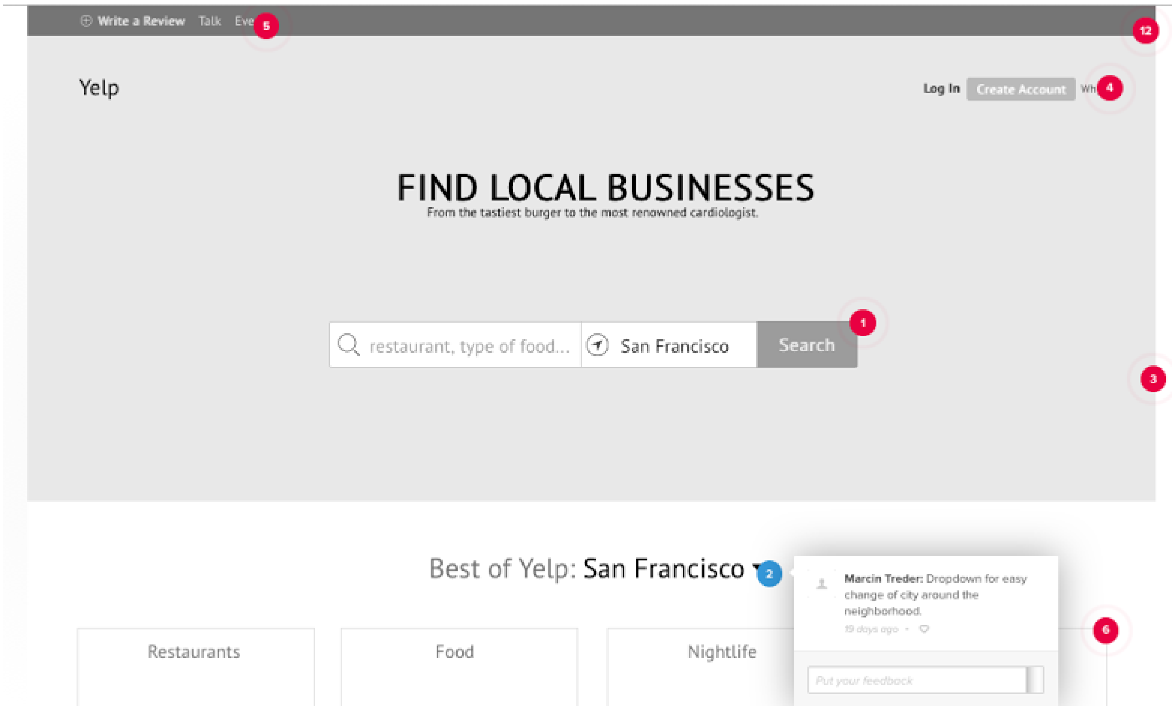
This is a core concept of Lean UX, devised by Jeff Gothelf, Director of UX at Neo Innovation Labs (you can hear him describe it in this thorough webinar). Lean UX emphasizes prototypes as "the fastest way between you and your code."

The Lean UX process offers three simple advantages over other methods:

- **Speed:** Skipping and consolidating phases will get you to the end product faster, but possibly at the cost of quality.
- **Efficiency:** The nature of the method is to avoid waste, so the only work done will be the essentials – no time spent on "busy work."

- **Experience, not deliverables:** Part of "trimming the fat" is minimizing documentation. Teams communicate better and faster, improving collaboration in designing the experience. State the design vision, then iterate with that in mind.

One of the core processes of Lean UX is going straight from a detailed sketch or wireframe into a prototype. You can do this as simply as adding a few animations and basic interactivity to your prototype (if you're using the right tool), testing with a minimum of 5 users, and iterating as needed.



Marcin Treder: Dropdown for easy change of city around the neighborhood.

Low fidelity prototype created during a Yelp redesign exercise. *(View large version)*

In our Yelp redesign exercise, our goal was to increase frequency and time on site for occasional Yelp users. We started with some rough sketches to explore concepts. Once we narrowed it down to a single concept, we created a wireframe. As we created the wireframe, we also created functionality by adding a few interactions.

You can see above that the lo-fi prototype created in UXPin was far from complete, but we were already able to arrive at the following structural decisions:

- Distinguish the search field as the primary visual element through generous white space
- As a secondary visual element, "Best Of" content is presented in a dropdown menu for fast access
- All categories are presented in a card format for ease of selection

While we don't yet know exactly how the design will look, we certainly have a clear idea of how it will function. Not bad for roughly

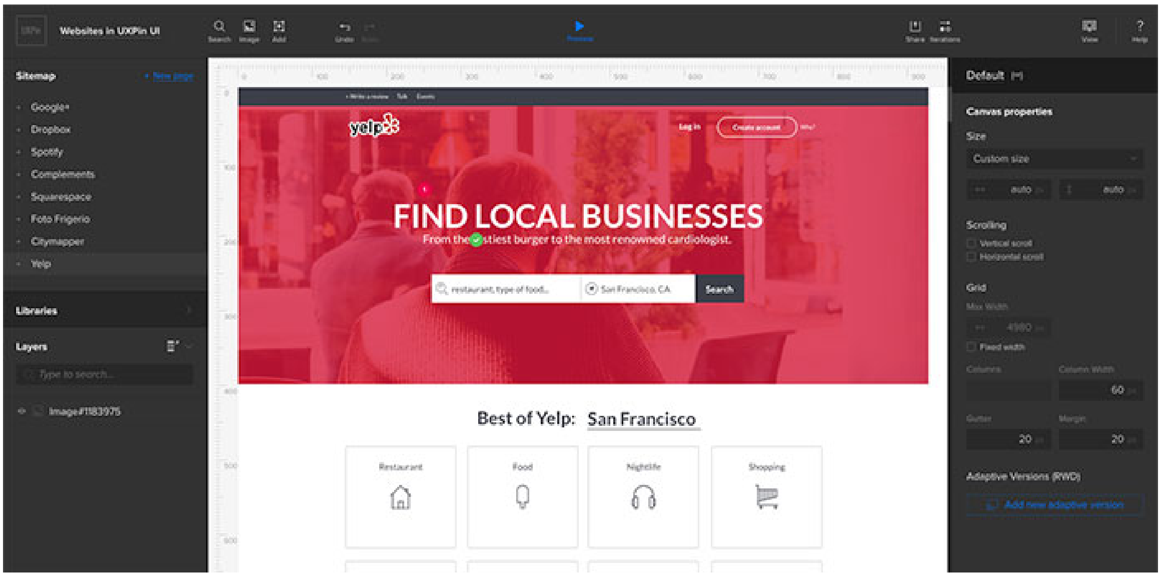
If you'd like to know more about Lean UX and rapid prototyping, UX Matters features a great discussion thread about it in which 9 industry experts weigh in with their opinions.

Mockups & Prototypes

Mockups are better-dressed wireframes.

Neither require functionality, but mockups give a better idea of what the final product will look like, and at times suggest how it will function. You could build a hi-fi mockup in a prototyping tool (if it includes enough visual libraries), but you could also create them directly in Photoshop/Sketch.

The main difference between building a prototype based on a mockup instead of a wireframe is that mockups automatically provide the baseline design for a mid- to high-fidelity prototype.



High fidelity prototype created during a Yelp redesign exercise. (View large version)

Because wireframes are more concerned with the broad strokes of design (such as overall layout), they usually lead to low-fidelity prototypes. You'll definitely want to consider this since, while low-fidelity prototypes are great for quick collaboration and exploration, high-fidelity prototypes can be better for product definition and estimates.

When redesigning an existing website, creating prototypes from a mockup makes a lot of sense. Since you already have the high-fidelity visual assets, you don't necessarily need to restart with a wireframe (unless you are doing a drastic rehaul).

In the second half of our Yelp redesign exercise, we created a new iteration in UXPin for the hi-fi prototype. We then created custom UI patterns and icons, saved them as "Smart Elements" for easy reuse, and then started layering them over the old lo-fi prototype. As a finishing touch, we added basic interactions like page transitions and dropdown animations.

The result was a hi-fi prototype that looked and worked much like the end product – perfect for another round of usability testing.

Coding & Prototyping

Introducing code early into the design process produces a lot of benefits, namely a more solid foundation when beginning development (which can mean less revisions in code). But it's not so much a question of whether or not you *should* code with the prototype, but whether or not you *can*.

```
type="text/javascript">window.NREUM||(NREUM={});NREUM.info={"beacon":"bam.nr-data.net","errorBeacon":"bam.nr-
","licenseKey":"0464f6fce4","applicationID":"11785576","transactionName":"dlpfQkJaXg9cS0wyXlpURURRGmcbSVANJ0RbW0UZd3BmQw==","queueTime":0,"applicationTime":5,"age
type="text/javascript">window.NREUM||(NREUM={}),__nr_require=function(t,e,n){function r(n){if(!e[n]){var o=e[n]={exports:{}};t[n][0].call(o.exports,function(e){va
orts}if("function"==typeof __nr_require)return __nr_require;for(var o=0;o<n.length;o++)r(n[o]);return r}({QJf3ax:[function(t,e){function n(){}function r(t){funct
r,o);for(var i=e(o),a=f(n),c=a.length,u=0;c>u;u++)a[u].apply(i,r);return i}function u(t,e){p[t]=f(t).concat(e)}function f(t){return p[t]||[]}function s(){return n
n{on:u,emit:c,create:s,listeners:f,context:e,_events:p}}function o(){return new n}var i="nr@context",a=t("gos");e.exports=r()},{gos:"7eSDFh"}],ee:[function(t,e){e
(){r(t,[(new Date).getTime()].concat(i(arguments)))}}var r=t("handle"),o=t(1),i=t(2);"undefined"==typeof window.newrelic&&(newrelic=window.NREUM);var a=
eViewName","addPageAction","setCustomAttribute","finished","addToTrace","inlineHit","noticeError"];o(a,function(t,e){window.NREUM[e]=n("api-"+e)}),e.exports=windo
ts=t("7eSDFh")},{}],"7eSDFh":[function(t,e){function n(t,e,n){if(r.call(t,e))return t[e];var o=n();if(Object.defineProperty&&Object.keys)try{return Object.defineE
}var r=Object.prototype.hasOwnProperty;e.exports=n},{}],D5DuLP:[function(t,e){function n(t,e,n){return r.listeners(t).length?r.emit(t,e,n):void(r.q&&(r.q[t]||(r.c
).create();e.exports=n,n.ee=r,r.q={}},{ee:"QJf3ax"}],handle:[function(t,e){e.exports=t("D5DuLP")},{}],XL7HBI:[function(t,e){function n(t){var e=typeof t;return!t
r++})}var r=1,o="nr@id",i=t("gos");e.exports=n},{gos:"7eSDFh"}],id:[function(t,e){e.exports=t("XL7HBI")},{}],G9z0Bl:[function(t,e){function n(){if(!v++){var t=l.i
&&t.licenseKey&&t.applicationID&&e){c(p,function(e,n){t[e]||(t[e]=n)});var n="https"===s.split(":")[0]||t.sslForHttp;l.proto=n?"https://":"http://",a("mark",["on
teElement("script");r.src=l.proto+t.agent,e.parentNode.insertBefore(r,e)}}}function r(){"complete"===f.readyState&&o()}function o(){a("mark",["domContent",i()])}
dle"),c=t(1),u=window,f=u.document;t(2);var s=(""+location).split("?")[0],p={beacon:"bam.nr-data.net",errorBeacon:"bam.nr-data.net",agent:"js-agent.newrelic.com/r
js"},d=window.XMLHttpRequest&&XMLHttpRequest.prototype&&XMLHttpRequest.prototype.addEventListener&&!/CriOS/.test(navigator.userAgent),l=e.exports={offset:i(),orig
entListener("DOMContentLoaded",o,!1),u.addEventListener("load",n,!1)):(f.attachEvent("onreadystatechange",r),u.attachEvent("onload",n)),a("mark",["firstbyte",i()
ts=t("G9z0Bl")},{}],12:[function(t,e){function n(t,e){var n=[],o="",i=0;for(o in t)r.call(t,o)&&(n[i]=e(o,t[o]),i+=1);return n}var r=Object.prototype.hasOwnProper
ndefined"==typeof n&&(n=t?t.length:0);for(var r=-1,o=n-e||0,i=Array(0>o?0:o);++r<o;)i[r]=t[e+r];return i}e.exports=n},{}]},{},["G9z0Bl"]);</script><meta content="
="og:title" /><meta content="https://www.uxpin.com/images/uxpinapp-fb.jpg" property="og:image" /><meta content="image/jpg" property="og:image:type" /><meta conter
="og:image:height" /><meta content="width=device-width,initial-scale=1.0" name="viewport" /><title>UXPin: UX Design & Wireframing Tools As Beautiful As Your Work<
erience Design Tools. That&#39;s why UXPin is handcrafted to be robust &amp; elegant - oh, and it&#39;s FREE to try!" name="description" /><link href="/images/fav
mages/favicon.ico" rel="shortcut icon" type="image/x-icon" /><link href="https://www.uxpin.com/" rel="canonical" /><link href="/stylesheets/colorbox.css?14563900
```

Coding your prototype is a good way of reducing waste. Before starting, always ask yourself how many ideas you can generate in code versus on paper or in a prototyping tool.

As Andrew Fitzgerald, Senior UX Architect for Deloitte Digital, points out, most designers have a "complicated" relationship with code.

When it comes to coding prototypes, it helps to start with a sketch and then dive straight into HTML or another language of choice (just make sure you check your work on smartphones and tablets). This lets you explore ideas on a whiteboard or on paper (where they're easy to alter) so you aren't trapped with mediocre concepts just because they're cemented in code.

Regardless of whether you choose to add coding to your prototype, involve your developers each step of the way. You don't want the first time your developers to see the prototype to be in an email with a long list of notes.

How to Prototype: The Rapid Prototyping Process

Rapid prototyping is less a separate process and more a filter for efficiency.

Designers consider it a core principle of Lean UX, but it also applies to any of the prototyping processes we previously described. In rapid prototyping, you revise quickly based on feedback and shift to multiple prototyping approaches based on the requirements.

We can break down the process into three stages:

1. **Create your information architecture and user flows**
   Your information architecture (usually represented in a site map) lays out the breadth and depth of the content. You create the the blueprint for the entire experience by laying out all your screens.

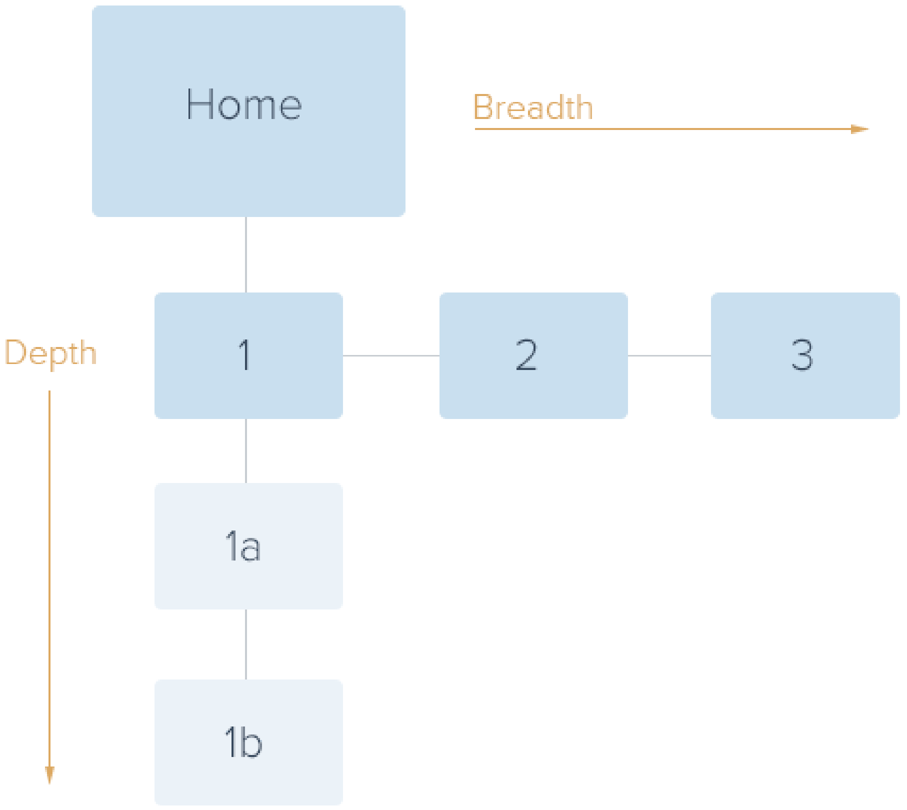
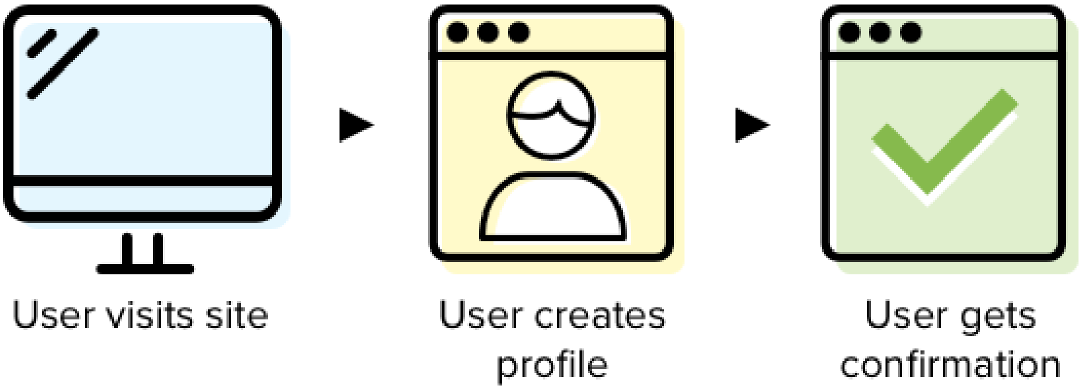


Photo credit: Adapted from UX Movement's site map chart

Once you've built the IA, you can dive one level deeper with user flows. User flows show common paths for moving between different the screens in your information architecture.



2. Scope the prototype

Your first prototype should focus on just the core user flows. Focus on the 20% of functions that deliver 80% of your product's value.

If you were creating a mobile banking app, the first prototype must build out the flows for primary functions such as:

- Logging in to the app
- Reviewing monthly statements
- Paying off monthly credit card balance
- Redeeming rewards points
- Disputing payment claims

3. Iterate, Test, & Repeat

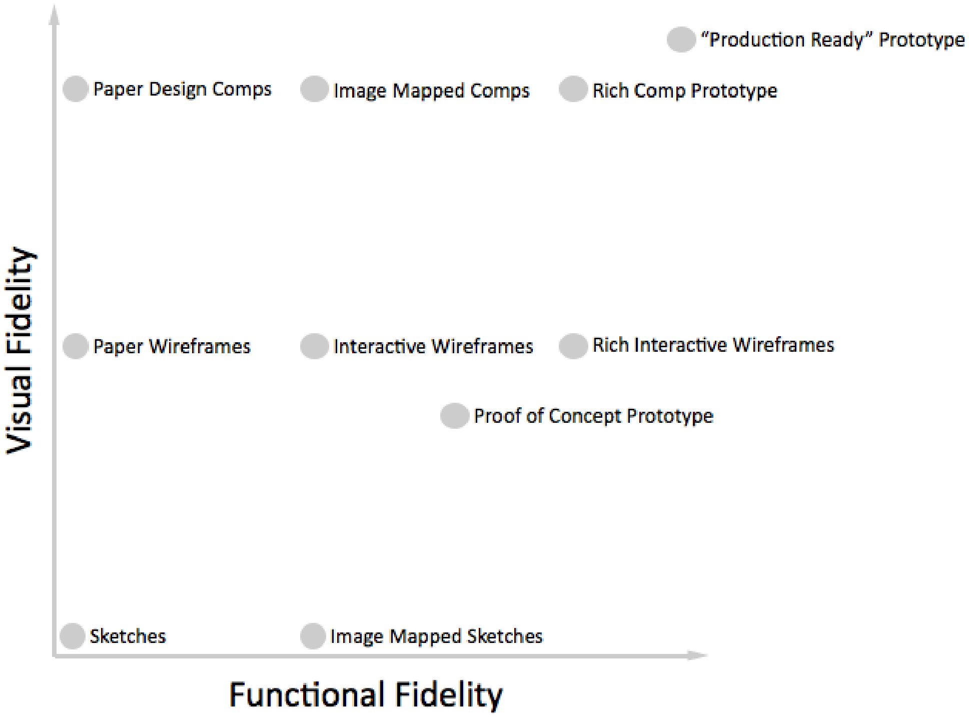
Start with breadth, then dive into depth.

You'll usually begin with a "horizontal prototype" that covers all your main screens (like a home screen and category screens).Your next iteration will dive deeper into user flows for each screen (like updating account information in the settings page, selecting different rewards programs on the home page, etc).

Make sure you test major iterations with at least 5 users.

What to Prototype: 4 Ways to Combine Fidelity & Functionality

Most prototypes fall into four categories, depending on how you combine high- or low-fidelity visuals and functionality. All of these types are useful – but at different times for different needs.



A quick landscape of prototyping fidelities.

Photo credit: Fred Beecher, Boxes & Arrows, used with permission.

As Bill Buxton once said, and Fred Beecher (Director of UX at The Nerdery) illustrates in his excellent article: there's no such thing as "high" or "low" fidelity – only the right fidelity.

Let's explore the different combinations he mentioned here so you can select the right fidelity.

Low Visual Detail + Low Functionality

These prototypes are usually made from paper, look rough, and require someone to act as a "human computer" to move from screen to screen.

Popular low visual/low functional prototypes include:

- Paper prototypes
- Paper sketches
- Digital static wireframes

The main advantage here, at the cost of both visuals and functionality, is speed. These prototypes are made quickly, and can be thrown away and replaced just as quickly. They're great for horizontal brainstorming in which you explore a wide range of concepts.

You can build this type of prototype over and over until some fundamental questions about UX concepts, feature completeness, and high-level page flows are answered. If you're following the rapid prototyping process, this will be one of the earliest prototypes you produce.

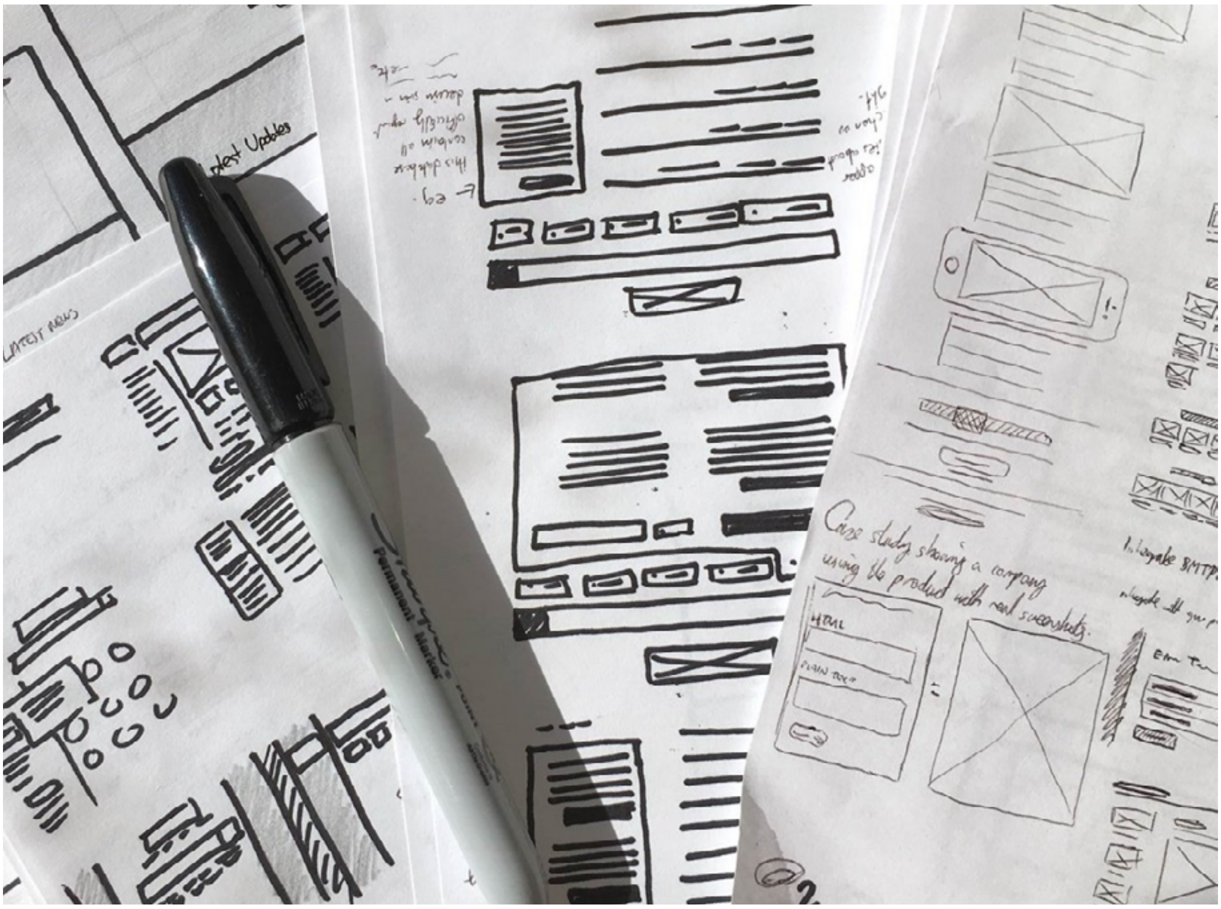

Latest Updates
Latest News
Case study showing a company using the product with real screenshots
HTML
Permanent Marker

Photo credit: Fairhead Creative

You'll be able to answer questions such as:

- Did we offer enough features to meet our user's needs?
- Does the overall content deliver content at the right pace?

Understand, however, that when you test this prototype, you won't be able to answer questions regarding interactivity such as:

- How well does the design adapt to other devices?
- Do the animations feel smooth, or are they jarring?

- Does the interface provide the appropriate feedback to the user?
- Can users scroll smoothly (either vertically or horizontally) between sections of content?

The beauty of this stage, however, is that you can explore multiple page layouts, page flows, and basic functionalities with very little friction and time.

For example, when redesigning our own app, our design team spent 4 hours paper prototyping the Preview mode. Even though all of the work had to be recreated in a digital tool, our designers were able to review current usability issues and walk out with new positioning of elements. In that sense, paper prototyping was an excellent exercise in restructuring the layout.

For paper prototypes, grab some paper, scissors, pens, and keep it lightweight and conceptual.

1. **Use printer paper and cheap pencils/pens.**
   The form affects your creative freedom. If you use a Montblac and a Moleskin, you might subconsciously restrict your thinking since you don't want to draw something "ugly".

2. **Start by loosening up.**
   Maybe you need to take a sheet of paper and scribble all over it. Or maybe you need to just start sketching out thoughts and ideas in your head as rapidly as you possibly can. Whatever works for

you, start by loosening up. It'll make your lines more confident, and your sketches stronger.

3. **Prototype mobile-first.**
   Because of the limited space on a mobile viewport, you'll be forced to prioritize content. When you prototype mobile-first, you create a 100% experience that you can scale up for other viewports.

4. **One sketch per screen.**
   No matter how big or small they are, draw a separate sketch for each screen.

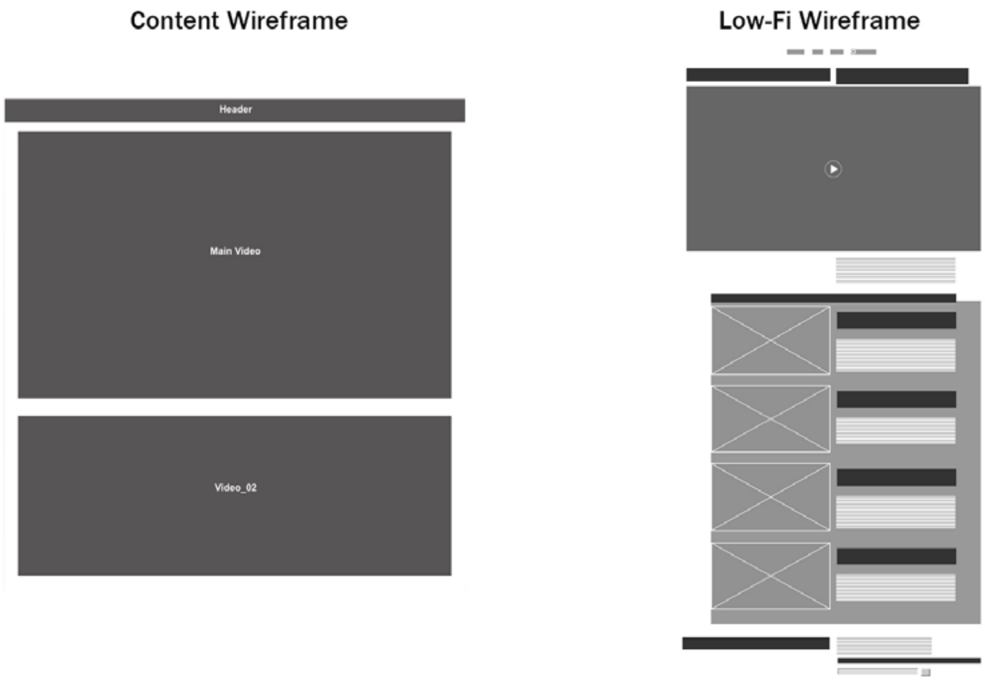
5. **Iterate as the ideas happen.**
   Don't question your ideas as they come – just let them out. You can question them all later. Remember, sometimes great ideas can come from a little detail within a terrible idea. Let them out, nobody's judging you.

Aside from paper prototyping, you could also create a low visual/low functional wireframe in a digital tool. For prototyping purposes, we recommend selecting a tool that also allows you to add interactions later (otherwise your low visual/low functional wireframe is a dead-end document).

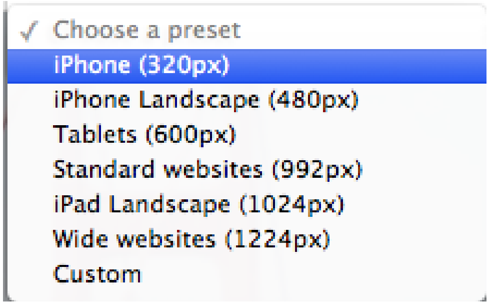
UXPin Pro Tip

In the earliest stages of prototyping, consider creating a "content wireframe." Instead of diving straight into the boxes and arrows, you're first carving out space for broad categories of content.

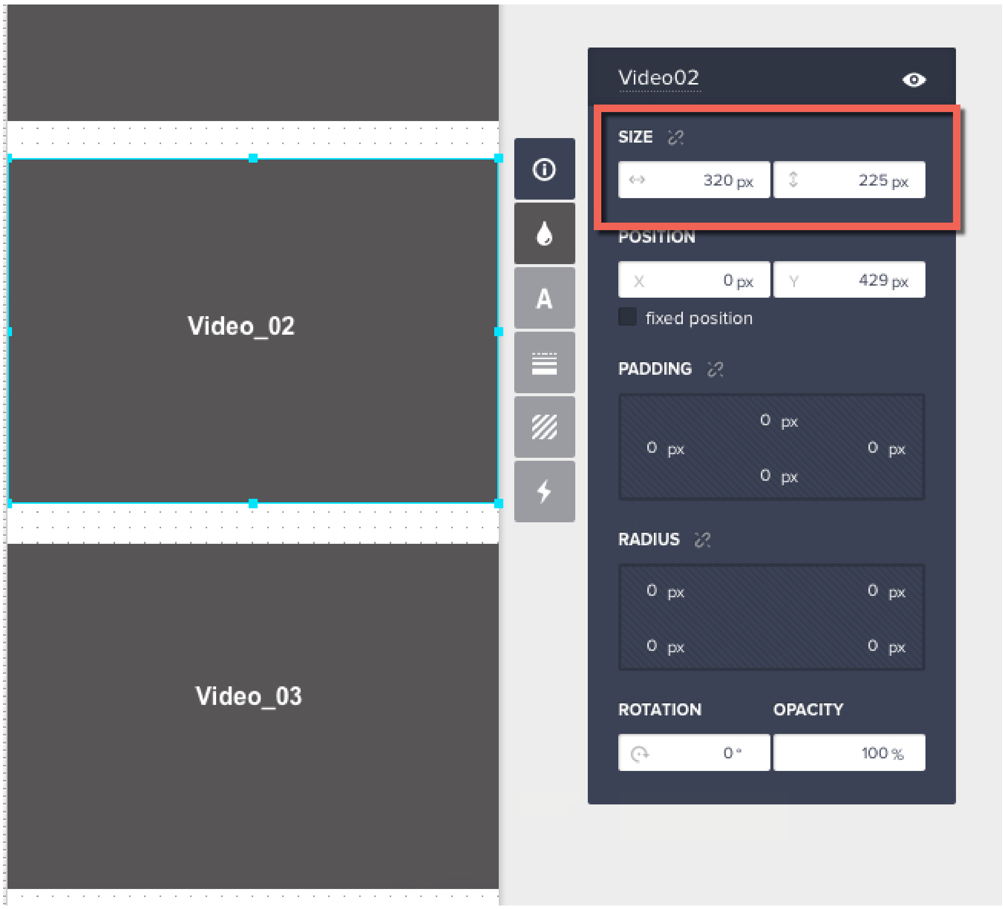
Not only do you focus on the most design element (the content), but the approach is also responsive-friendly.



If you're using UXPin, you could start mobile-first by creating a 320px breakpoint for your prototype.



From there, simply add box elements sized appropriately for the content. For example, as you can see below, we set the dimensions of 320px wide and 225 px high for the smaller videos (Video 2, Video 3).



Video02

SIZE
320 px
225 px

POSITION
X 0 px
Y 429 px
□ fixed position

PADDING
0 px
0 px
0 px
0 px

RADIUS
0 px
0 px
0 px
0 px

ROTATION 0°
OPACITY 100 %

Once you can start to see the whole content hierarchy for your design, you can then create a second project for your low visual detail/high functionality prototype.

For your second prototype, you'll be able to design faster since you've already answered your main structural questions in the content wireframe.

Low Visual Detail + High Functionality

Considered low- to medium-fidelity, these prototypes are basically "interactive wireframes."

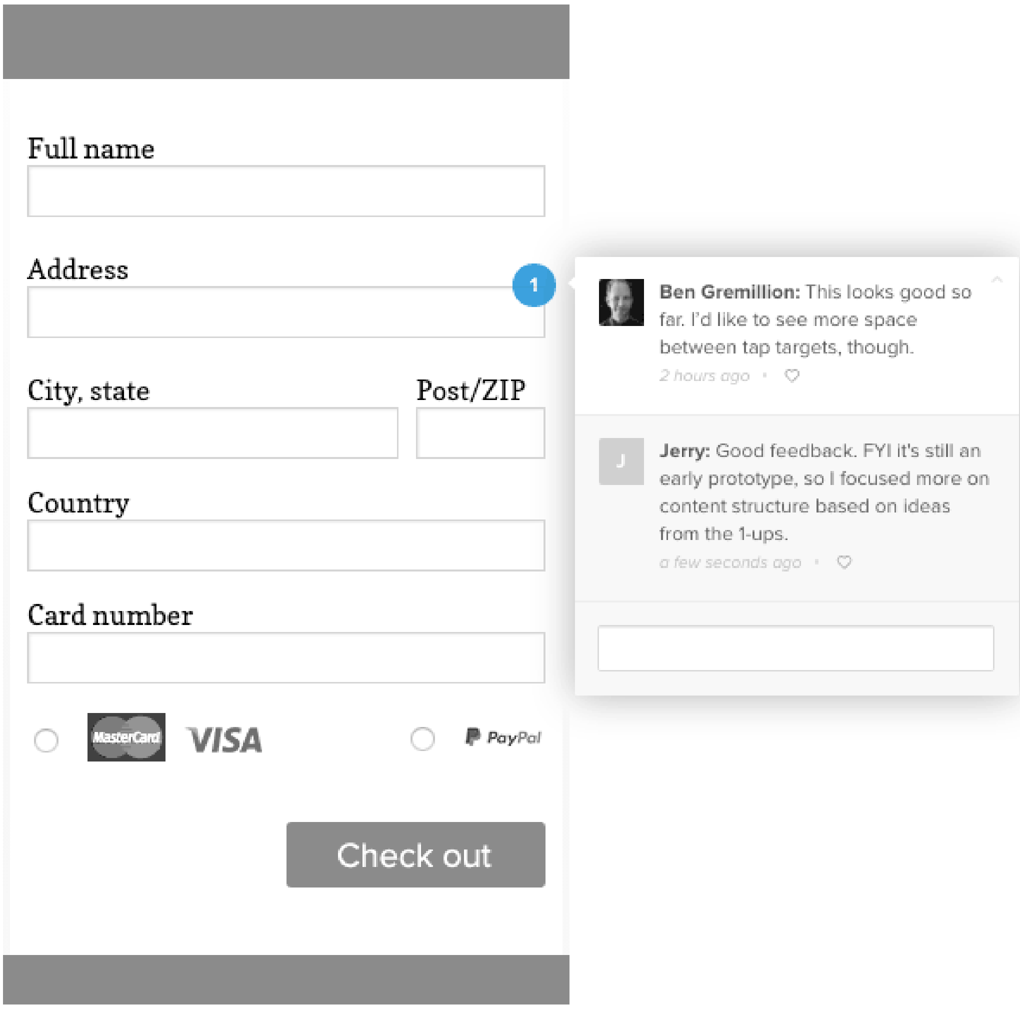
If you started with a paper prototype, you'll need to rebuild the prototype in a specialized tool that supports a high range of interactions. If you're technically adept, you can also build a low visual/ high functional prototype in code.

Popular low visual/low functional prototypes include:

- Interactive wireframes
- Coded prototypes

As Beecher explained in his article, low visual/high functional prototypes help with:

- Testing usability
- Gaining validation from stakeholders
- Supplementing documentation for developers
- Running remote usability tests (the higher functionality reduces risk of confusion, and thus the need for someone to moderate in-person)



Ben Gremillion: This looks good so far. I'd like to see more space between tap targets, though.
2 hours ago

Jerry: Good feedback. FYI it's still an early prototype, so I focused more on content structure based on ideas from the 1-ups.
a few seconds ago

Low Visual/ High Functional Prototype

If you're on a budget, these prototypes are very useful for usability testing thanks to the more powerful functionality. A lower level of functionality will confuse users, while a higher level of visuals can be a waste of time since you haven't validated your concepts.

In our experience, the low visual/high functional prototype gives you the best insight for the time involved.

They're rough enough to encourage honest feedback, but features enough functionality to test user flows. Plus, it's much easier to test than a low visual/low functional paper prototype since it actually works on it's own (no need to shuffle a bunch of pages around).

If you're following a lean design process, these prototypes will suffice. You can explore and test the interactions early on, saving the visual design for later (either in Photoshop/Sketch or straight in code).

High Visual Detail + Low Functionality

If you're building a visually rich site, you might jump to this step after your low visual/low functional prototype. This stage focuses more on testing visual design details rather than core interaction models.

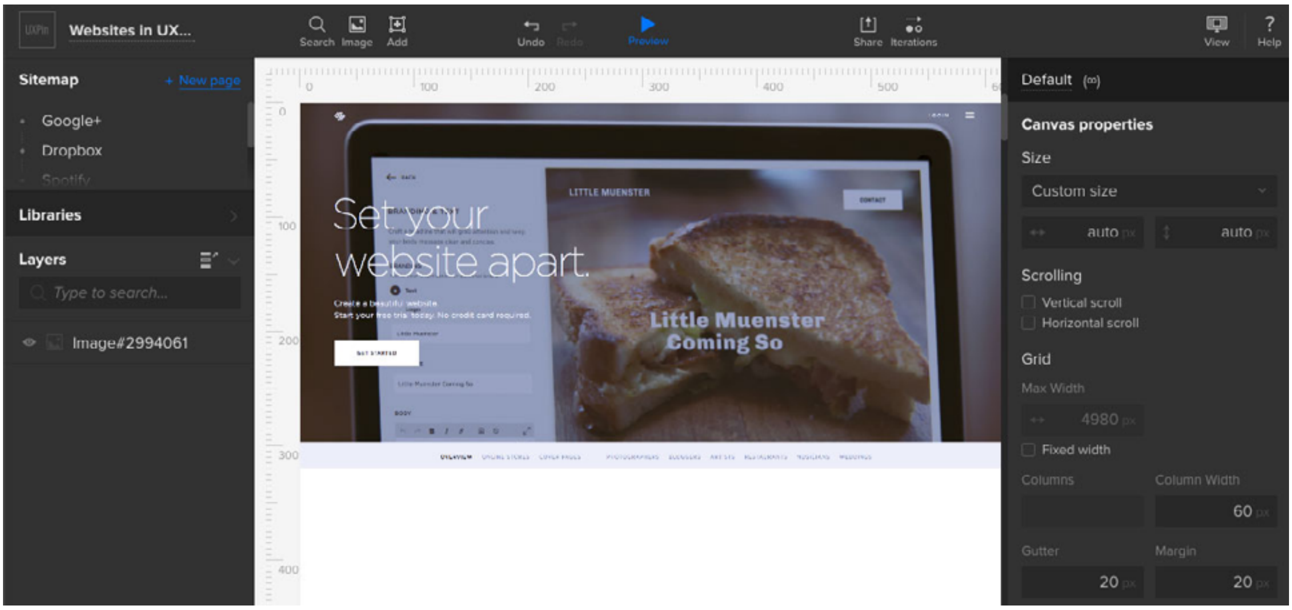
As such, we do not recommend creating high visual/lo-fi prototypes unless you've validated your assumptions through previous testing. Otherwise, you'll be investing in visual design when the information architecture or content structure is broken.

Popular low visual/low functional prototypes include:

- Digital prototypes with basic interactions
- Coded prototypes with basic interactions

Think about these prototypes as a "prettier" paper prototype – limited in functionality, but much more visually rich.

If you started on paper, you could create the visual design in Photoshop/Sketch and then drag and drop your files into UXPin to add interactions to each layer. If you've already created a lo-fi prototype in a specialized tool, you'll need to iterate on top of your work.



Size: Custom size
auto px, auto px
Scrolling
□ Vertical scroll
□ Horizontal scroll
Grid
Max Width: 4980 px
□ Fixed width
Columns
Column Width: 60 px
Gutter: 20 px
Margin: 20 px

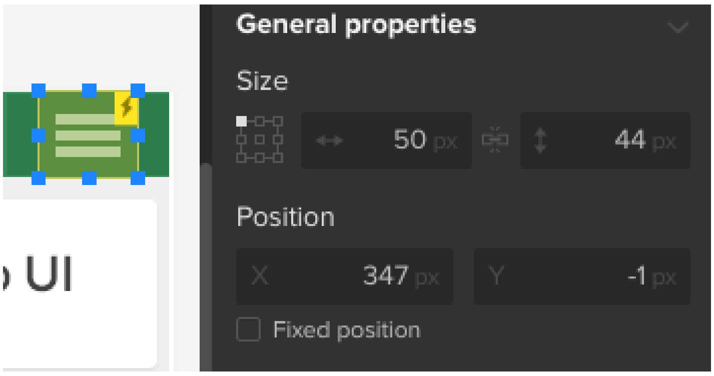
High Visual/Low Functional Prototype of the Squarespace homepage

Think of this prototype as as a "mockup +".

It serves all the purposes of a mockup – finalizing visual decisions, presenting to clients, etc. – but with the bonus of some limited interactivity. In some tools, you'll be "stitching" together screens through the creation of linked hotspots. As users click on the hotspot, the next page will load. Instead of the "human computer" required of paper prototypes, the prototyping app does all the work.

UXPin Pro Tip

When adding hotspots to mobile prototypes, remember the rule of designing for fat fingers. Your interface buttons (and any hotspots on the prototype) should be at least 40 x 40 points for the sake of tappability.



General properties

Size: 50 px × 44 px

Position: X 347 px, Y -1 px

□ Fixed position

High Visual Detail + High Functionality

The hi-fi prototype falls just one step below the finished product.

High visual/ high functional prototypes are helpful if you're building a product with complex functionality, intricate visual design, or dozens of microinteractions.

For example, Carbonmade is a web app that allows users to create custom portfolio sites. The app is full of microinteractions (like hover animations), rich color palettes, and visual elements that are core to the product experience. While you could test the basic usability with a low-visual/ high-functional prototype, you can't gauge the overall enjoyability of the full experience.

On a related note, high visual/high functional prototypes also help you test the effectiveness of new visual languages or new branding. Usability testing will reveal if any colors or typefaces wreak havoc on a previously usable design.

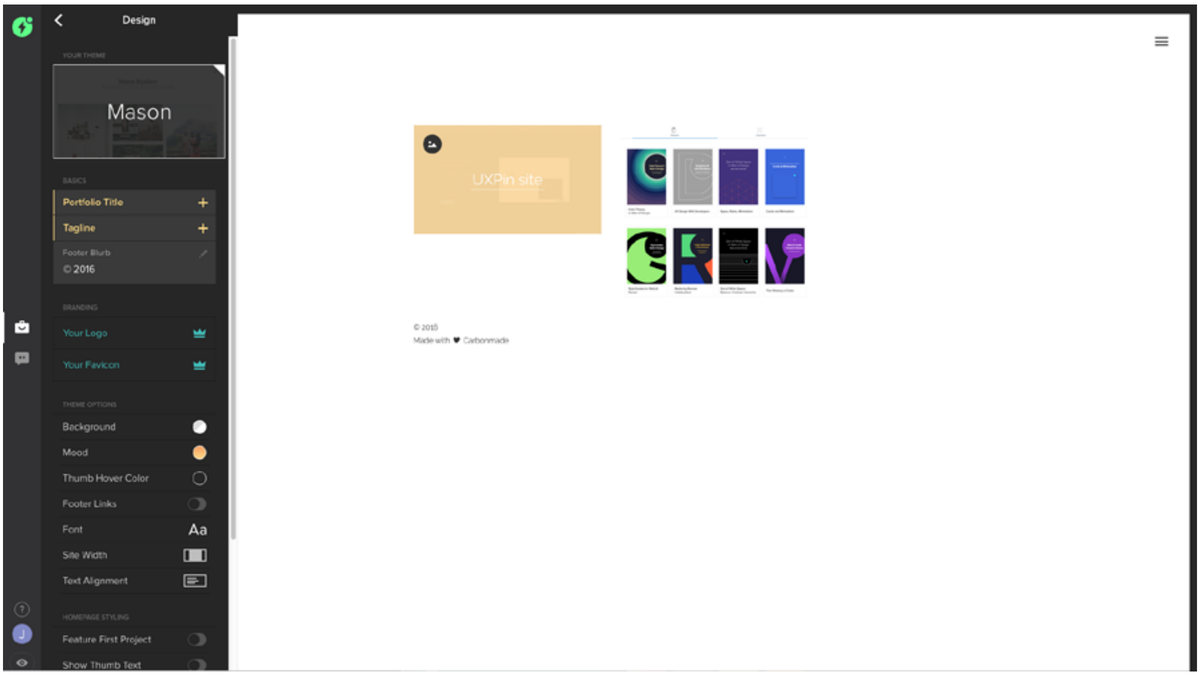
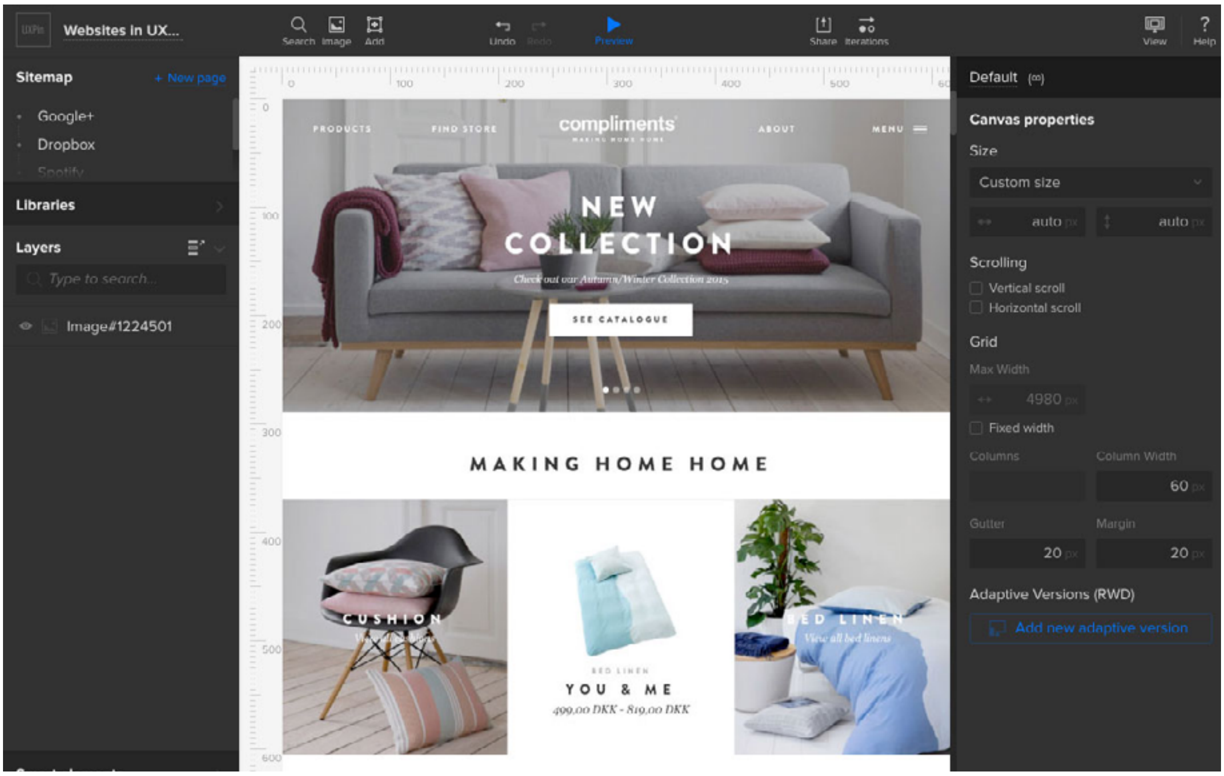


Photo credit: Carbonmade

High visual/high functional prototypes generally include:

- Digital prototypes with advanced interactions
- Coded prototypes with advanced interactions

Either of the previous prototypes can be iterated into a high visual/ high functional prototype. You can create high visual/ high functional prototypes with visual design tools and prototyping tools, or build it right in code if you are technically adept.



Size: Custom size
auto px auto px
Scrolling
□ Vertical scroll
□ Horizontal scroll
Grid
Max Width 4980 px
□ Fixed width
Columns
Column Width 60 px
Gutter 20 px
Margin 20 px
Adaptive Versions (RWD)

High Visual/ High Functional Prototype of the Compliments site recreated in UXPin.

Since they resemble the final product, these prototypes are also helpful for testing with users who aren't very tech savvy, or for communicating design needs with outsourced programmers against the specs (less chance of misinterpretation).

Because coding is always the next step after a high-fidelity prototype is finalized, these prototypes act as highly realistic "living" technical specs.

As Jared Spool suggests, high visual/high functional prototypes are most appropriate for convergent design (when you're narrowing down a list of ideas).

If you’re still exploring broad questions like “How should we present primary and secondary content?,” then you’re better off doing low visual/low functional prototyping since you can easily create 10–20 sketches, slap them on a wall, and get some nice perspective.

UXPin Pro Tips

Keep your interactions simple to start with. In our experience, they’re likely to change early in the design process, so don’t spend too much time developing them early on.

If you’re using UXPin, name your elements as you make them interactive so you can track which is doing what.

Can it be coded? Consult with your developers or, if you’re the one writing the code, make sure you know how to build interactions technically as you dream them up. Don’t be afraid to push your boundaries – but do your homework first.

Build pages before you link them together to make sure you know what needs linking first.

Conclusion

Prototyping enjoys a wide versatility in terms of process and purpose. Even before you get into the types of prototypes, the initial decisions (rapid prototyping vs. higher quality, basic vs. detailed visuals, coding vs. no coding) will all lead to different outcomes in the design process.

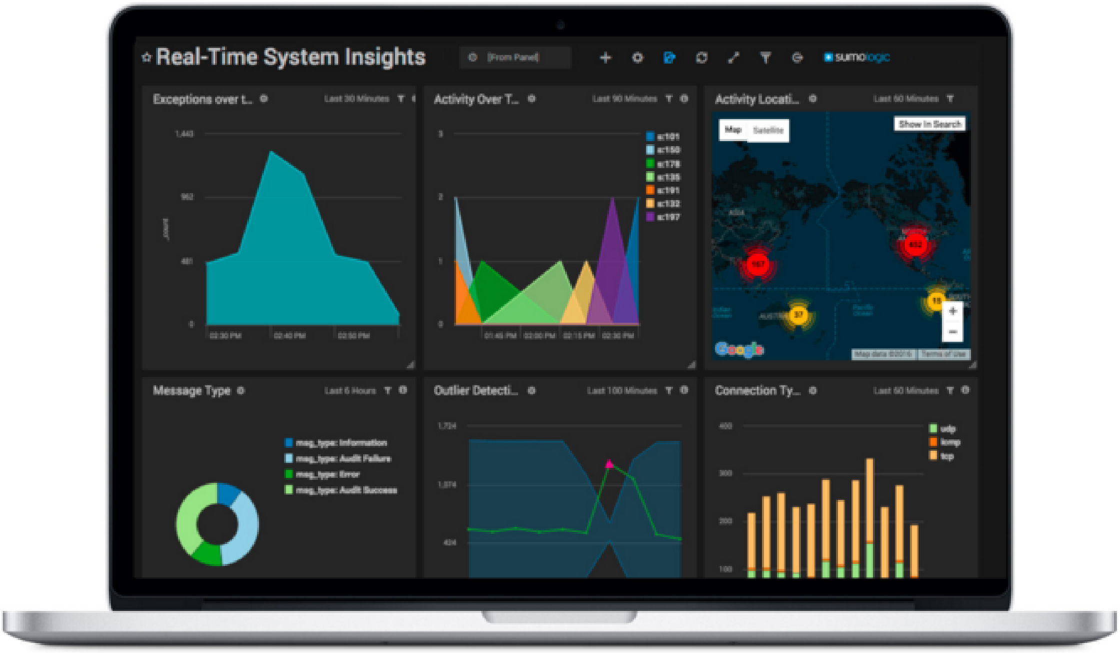
In closing, let's reiterate Bill Buxton's timeless advice: it's not so much about choosing *high-fidelity* or *low-fidelity*, as it is about choosing the *right* fidelity.

Scaling UX Process Case Study

Speeding Up Design Reviews By 300%

The Challenge

Based in the Bay Area with 250+ employees and $161 million in venture capital funding, Sumo Logic serves some of the top enterprises in the world. The company's analytics platform visualizes more than 100 petabytes of data per day, helping businesses harness the power of machine data to streamline operations.



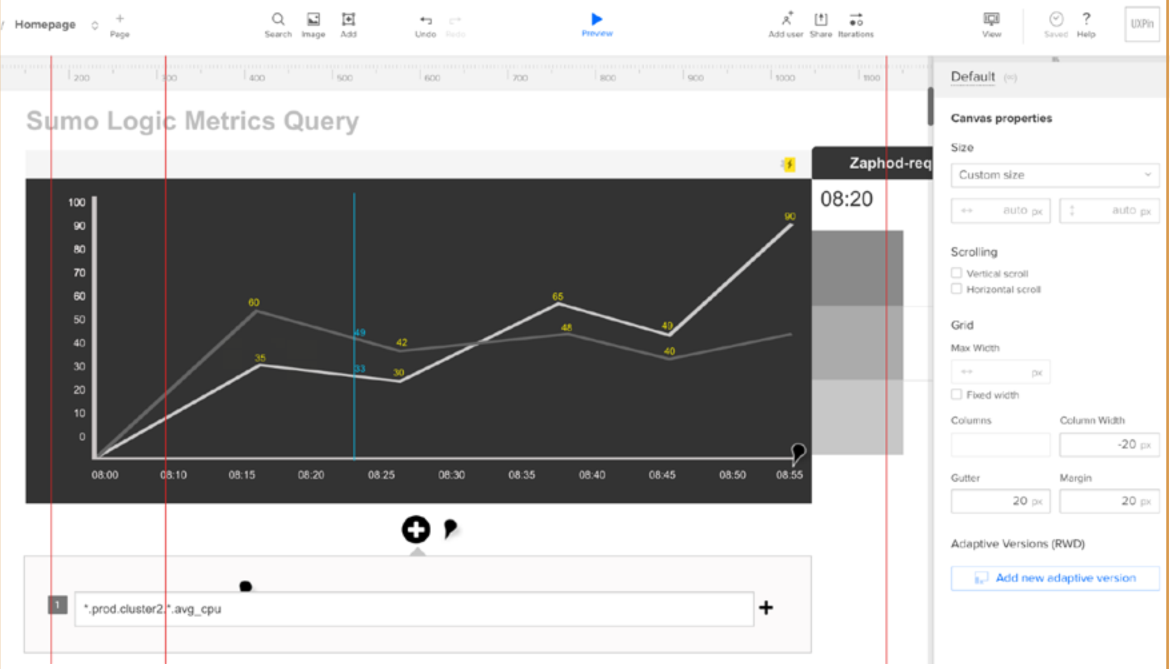
To make the data actionable for customers, Sumo Logic invested heavily in its own internal UX team to bring a "consumer-grade" experience to the enterprise. In 2015, they hired their first UX team comprised of design leaders, interaction designers, visual designers, and UX architects.

The company had been using Axure for wireframing but Design Director Daniel Castro quickly found that the solution did not allow for easy design modification and did not encourage collaboration.

Tired of sharing PDFs back and forth, the company needed a collaborative UX platform that could scale along with their teams and processes.

The Solution

For the first six months, Design Director Daniel Castro and his team spent much of their time holding happy hours and offering show-and-tells of great UX design. The next challenge was to create a collaborative environment where UX designers could work closely with both technical and non-technical staff to revolutionize the Sumo Logic experience.



Sumo Logic Metrics Query

1 *.prod.cluster2.*.avg_cpu

Default

Canvas properties

Size: Custom size

↔ auto px ↕ auto px

Scrolling

□ Vertical scroll

□ Horizontal scroll

Grid

Max Width ↔ px

□ Fixed width

Columns

Column Width -20 px

Gutter 20 px

Margin 20 px

Adaptive Versions (RWD)

Add new adaptive version

Sumo Logic prototype created in UXPin for their Unified Logs Metrics product

In a culture already used to collaborative tools such as Slack, this slow process of emailing thoughts on static designs was stifling. Castro had begun using UXPin at his previous job, and knew it would offer his Sumo Logic team the collaboration tools they needed.

> *"We are constantly collaborating with engineering and product managers and it used to take a significant amount of time to work together going back and forth," Castro said. "UXPin allows us to easily show the flow and main components of our projects. We can share a link and everyone can communicate with our key stakeholders, expanding on each other's comments and allowing us to manage feedback contextually without redundancy. It's like a visual version of our thought process. We can even make comments on a pixel level. This has made our review process three times as fast."*

> *"UXPin has played a vital role in creating a design-oriented culture at Sumo Logic," Castro added. "The team is great to work with, and I'm excited to see what we can do next."*

The Results

- Design modification is **quick and simple** with UXPin, instead of the limiting modifications possible with Axure wireframing.

- Design reviews are **three times as fast and now contextual** using UXPin to collaborate instead of emailing static PDFs.

- UXPin is “like gold” when trying to get approval from stakeholders on projects, **halving the effort** needed to communicate with stakeholders.

Want UXPin to help your team? See our Enterprise features in action.