UXPin

The Practical Handbook to Rapid Lo-Fi Prototyping

Tips, Methods, and Exercises

UXPin

The Practical Handbook to Rapid Lo-Fi Prototyping

Tips, Methods, and Exercises

Index

Marek Bowers is a UX and Product Designer. He studied Mechanical Engineering at UCLA and went on to design cooling systems for commercial aircraft. However, Marek found that his true passion was for creating web and mobile applications, leading him to become a specialist in interaction design and usability. In the past decade, his clients have included Activision, Johnson & Johnson, Dell, Lexus, Experian, and dozens of other high-profile companies.

Jerry Cao is a content strategist at UXPin where he gets to put his overly active imagination to paper every day. In a past life, he developed content strategies for clients at Brafton and worked in traditional advertising at DDB San Francisco. In his spare time he enjoys playing electric guitar, watching foreign horror films, and expanding his knowledge of random facts.
Follow me on Twitter

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.

When Prototyping Is a Life or Death Scenario

In September 2008, outdoor outfitter Eddie Bauer worked with two world-class, high-altitude mountaineers – Ed Viesturs and Peter Whittaker–to develop and test the first prototype of the Katabatic Tent. A rugged expedition tent, **six prototypes** were iteratively tested on the world's most grueling peaks and terrain, including Mt. Rainier, Aconcagua, Everest and Antarctica.

Photo credit: ilkerender

Each prototype was rigorously tested in the field by Viesturs, Whittaker and their team of guides, and then refined and improved based on their feedback.

As Viesturs and Whittaker put it, “Nothing goes to market until the guys who will be using it are happy with the product.” Eddie Bauer’s stance was the same: “If we’re going to sell it, it has to work.” Hence, after **three years** of prototyping and countless cold nights, Eddie Bauer finally had an MVP of the Katabatic Tent in 2012.

Now, you might be asking yourself, what does this story have to do with digital UX design? A lot, actually. Think about it… if Eddie Bauer had simply gotten approval to sell V1 of the Katabatic Tent from a room full of executives based on 2D and 3D renderings that looked pretty cool, *without developing iterative prototypes tested by actual users*, what could have gone wrong?

- **Best case scenario**: Some weekend warriors have a chilly night car-camping in Yosemite.
- **Worst case scenario**: The product fails to protect its inhabitants at Camp 4 on Everest, which is *26,000 ft above sea level* (aka the “Death Zone”) where, from an emergency support perspective, is basically on the moon.

Designing software isn’t typically a life or death situation. However, the implications of a product failing as it is providing a critical need is the same.

Photo credit: Lloyd Smith

Static designs might *show* how a product looks and works, but a prototype actually lets someone *experience* the design. As a general note, low fidelity prototypes are visually and functionally simple. They lack the visual detail (colors, typefaces, etc) and rich interactions of the final design, but they embody the core content and functionality. In a digital environment, they are more like "clickable wireframes" which allow a user to experience the flow between pages.

Designs are meant to be used, not looked at, so let's take a look at one of the most powerful ways to quickly create functional designs for more informed stakeholders and happier users.

Prototyping Your MVP

Now, how do you ensure “functional” features are actually functional to your key personas? How do you ensure that you’re going to market with your “guide team’s” buy-in?

The answer is… prototype… prototype…. prototype… and keep prototyping, iteratively, until you reach an MVP (minimum viable product).

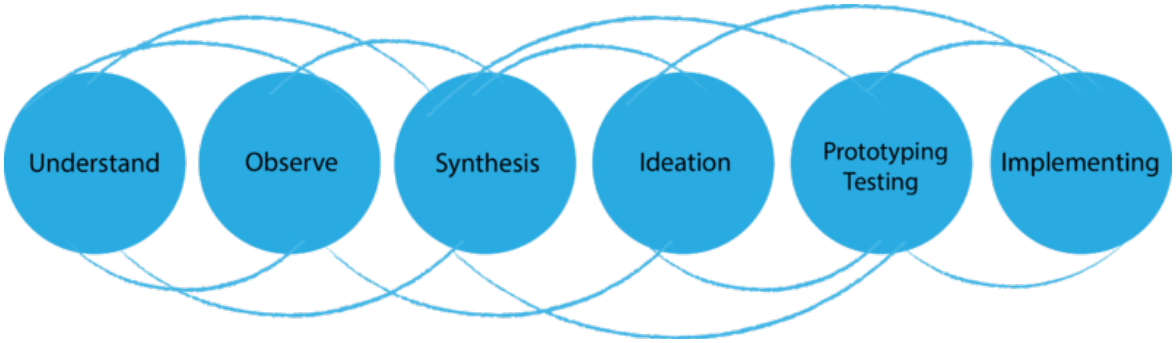


Photo credit: Wikimedia

This is called rapid prototyping, which Smashing Magazine defines as the process of quickly mocking up the future state of a system, be it a website or application, and validating it with a broader team of users, stakeholders, developers and designers.

In an article published on UXmatters, Robert Reimann (Lead Interaction Designer at Sonos, Inc.; Past-President, Interaction Design Association (IxDA)) was quoted as saying:

> *I think, without question, that rapid prototyping, when properly employed, can make user experiences better. Getting prototypes in front of colleagues, stakeholders, and target or existing users is a great way to get quick feedback addressing your design direction, business needs, customer needs, and usability.*

While Reimann does admit that feedback from rapid prototypes may not yield the type of results that in-depth, in-context usability testing uncovers, he follows up by stating:

> *[Rapid prototyping] can certainly identify gross interaction, navigation, and presentation issues with a design, as well as areas to watch more closely in follow-on testing.*

Photo credit: Unsplash

Using tools that are widely available, you can easily turn your low-fidelity wireframes into low-fidelity prototypes and engage in the rapid prototyping loop of building, reviewing and refining. The goal of your low-fidelity prototype is to accurately simulate your final product's actual interactions for a group of target users, which is very similar to what Eddie Bauer did with the "First Ascent" team.

The Power of Low Fidelity Prototyping

What do you think Viesturs and Whittaker asked themselves and their team of guides when testing the Katabatic Tent prototype?

We can only speculate. But the following questions seem reasonable:

- How easy was it to set up the product and how long did it take to set up?
- What was the experience of actually using the product to meet our immediate needs?
- Did the product fail in its usability? Where did it fail?
- Is there room for improvement and, if so, where?

These questions not only apply to testing the Katabatic Tent prototype, but also apply to your low-fidelity prototype. By putting your prototype in the hands of *actual users*, you can gain critical insight into how the product is performing (or underperforming) against its UX value proposition.

A Practitioner's Prototyping Tale

Here's a real story from a UX Designer working for a Fortune 500 tech company that's attempting to break into the healthcare space:

> *At [my company], we've always been great at consumer tech. But now we're trying to break into the healthcare space. One of my first projects at [this company] was to build a brand new HIPAA-regulated portal for physicians, nurses and technicians to easily view radiology images in the cloud. My focus group wasn't responding to my wireframes. They didn't get it because they didn't understand the interactions between screens. I decided to build a low-fidelity prototype using my wireframes so that the group could see how clicking a link here would take me to a screen there, or how selecting a drop-down option on this screen would show me an option on that screen. After showing my focus group the prototype, they got it and I received valuable product feedback.*

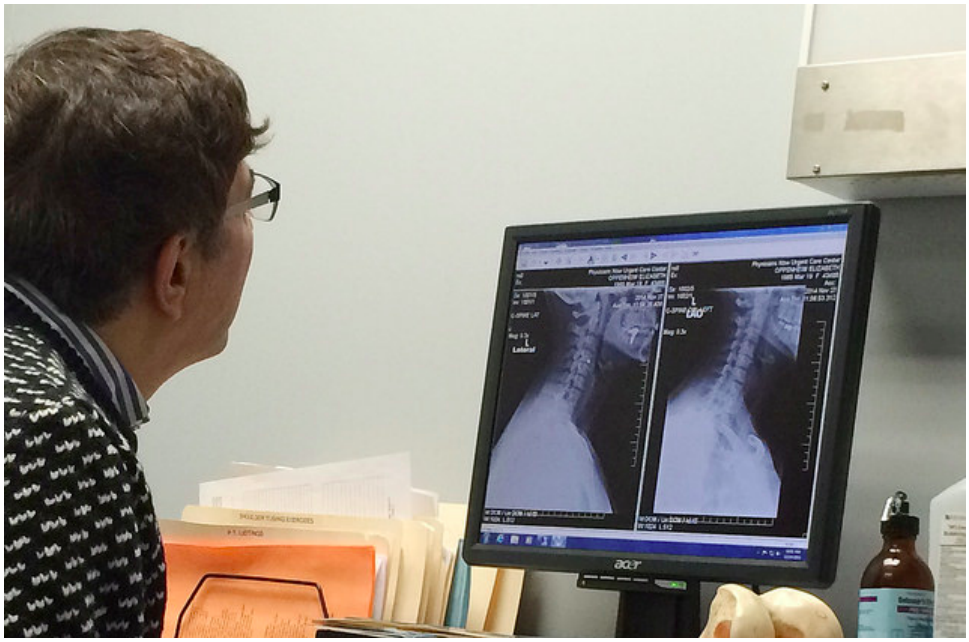


Photo credit: woodleywonderworks

But that wasn't the final step for this particular UX Designer.

He told me that once the project team saw the low-fidelity prototype in action, they were able to make insightful observations that impacted the future design. Because the low-fidelity prototypes were so fast and easy to create, the UX Designer was able to crank out a bunch of them with new features and enhancements to test simultaneously. Each prototype yielded valuable usability data from his group's testing.

Using a Low-Fidelity Prototype to your Advantage

Let's briefly review some of the key advantages to low-fidelity prototyping:

- **Get quicker and potentially more feedback:** When you give test users a beautiful visual design, the first thing they may assess are the colors, typography, imagery and general visual "feel" of the design. With a low-fidelity prototype, your testers can jump right into the nitty gritty of the functionality, without being distracted by what makes the design pretty.

- **A/B testing can be done faster/easier/cheaper:** When you can crank out 10 low-fidelity prototypes in the time it takes to create 2 high-fidelity prototypes, you can see that testing feature and content variations can be done much faster in low-fidelity. You can pivot much quicker.

- **Focus on flow:** Low-fidelity prototypes allow you to link up multiple wireframes to create flows. In this way, you can test the effectiveness of the order of things, rather than just the elements that the user sees on-screen. You can validate that sequences of interactions / actions make sense for users.

User Flows: The Heart of Every Prototype

If someone asked you to define the word “flow” or describe an example of one, what would you say? Would you immediately think about flow as it relates to user experience or interaction design? Maybe… but, maybe not.

You may think about flow in terms of water. For example, how snowmelt flows into waterfalls and streams, which in turn, flow into reservoirs, lakes and the ocean.

Photo credit: Marek Bowers

You may also think about flow in terms of air. Did you know that a golf ball has dimples to turbulate the flow of air around it (which reduces drag), causing the ball to fly farther than a smooth ball?

You can also measure the success of a flow (or lack thereof) in terms of efficiency. Just look at NASA and Boeing who collaborated on solving a flow problem this past April in a wind tunnel.

Photo credit: Steve Calcott

The teams outfitted a Boeing 757 tail with special technology that could potentially make flight more efficient. The impact of this? Lower fuel burn that might save airlines millions of dollars, which in turn, could save frequent flyers hundreds, possibly even thousands, of dollars on plane tickets. Sounds good to us!

How Flows Relate to UX Design

So what do all these notions of "flow" have in common?

- **Flow depicts movement**: movement through water, through air, through websites, through apps, etc.

- **Flow is variable**: Flow variability can cause a golf ball to fly farther (or fall short); a plane to cruise faster (or have more drag); and an online shopper to checkout with ease (or abandon her cart).

Long story short, *the concept of flow is important*. The success of your website or app is often contingent upon how well your modeled flows meet the needs of your targeted users (personas), as well as the needs of your business.

Because lo-fi prototypes lack visual detail, the user flows are the heart of your prototype. Lo-fi prototypes help you focus on creating the smoothest flows for users to accomplish their goals.

Building a Flow with Users in Mind

When you build a user flow, what's the first thing you should think about?
It might be obvious… your users!

For example, if you are designing a business intelligence tool that allows users to create reports and share them, you will want to have at least two flows: one flow for the Data Consumer (i.e. the user receiving/reviewing the reports) and an entirely separate flow for the data analyst (i.e. the user building/sharing the reports).

Photo credit: Gary Barber

Before you start creating your prototype flows, you should clearly understand your personas' motivations and needs. Ask yourself, what drives my personas and what are they trying to accomplish?

Once you've created your persona, you can better grasp user goals. Map out both sets of goals so you know what your prototype needs to accomplish.

For templates and helpful tips on creating personas, check out the chapter "Analyzing Users Before Diving Into Design" in our free ebook *The Guide to UX Design Process & Documentation*.

Considering Goals & Entry Points

If you are designing a website user experience, another good practice before creating your flows is to determine and map out where your users are coming from. Based on Morgan Brown's advice in *Stop*

Designing Pages and Start Designing Flows, we'd recommend you consider the following sources for your web or mobile prototype:

- Direct traffic
- Organic search
- Paid advertising
- Social media
- Referral sites
- Email

Different entry points define different user behavior. Take a look at the difference in the below hypothetical scenarios for someone looking to buy a smartphone on Amazon.

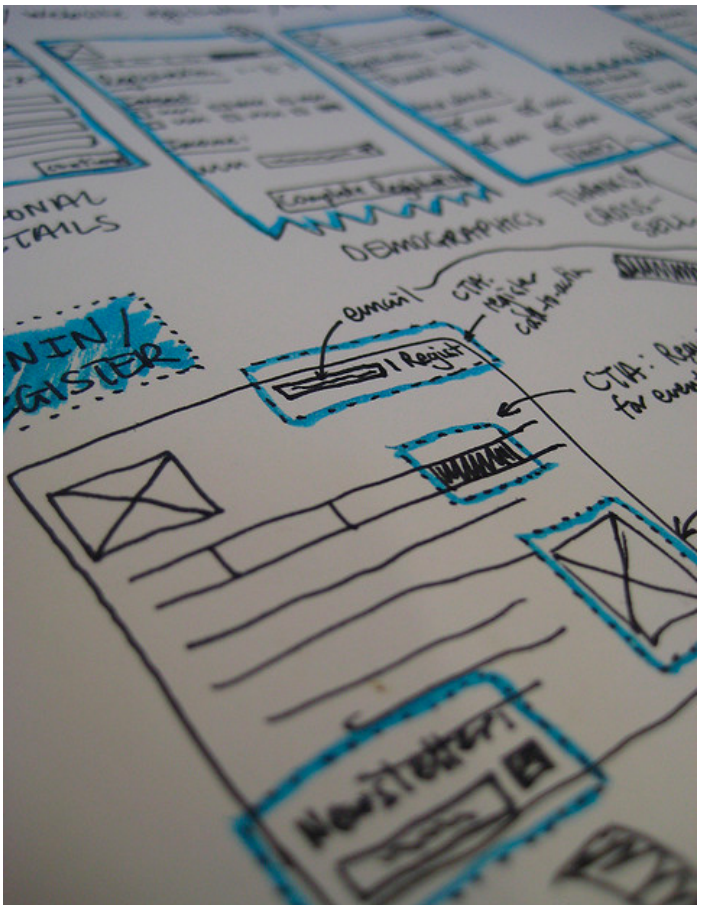


Photo credit: Rob Enslin

Organic Search Visitor:

1. Searches for reviews of iPhone
2. Enters Amazon.com.
3. Uses search bar to find iPhone
4. Browses more iPhone reviews
5. Uses search bar to find Samsung Galaxy
6. Browses Galaxy reviews
7. Returns to original iPhone Amazon vendor
8. Buys iPhone

Direct Visitor:

1. Enters Amazon.com
2. Uses search bar to find iPhone
3. Buys iPhone

Now, we're not saying the comparative shopping experience is that simple (or that the behavior between direct and organic visitors is always black-and-white). We are saying, however, that you must map out these different flows in order to deliver a comprehensively smooth experience.

To view some sample flows based on these entry points, check out the article Build it with the User in Mind: How to Design User Flow. The author, Peep Laja, has created three different user flows, where each flow originates from a different entry point and aligns with specific user and business objective.

Creating a Flow Outline

At this point, you should know:

- Which users/personas you will be designing your flows for
- What user and business objectives need to be accomplished
- Where your users are coming from (i.e. entry points)

Now you can think about what happens before and after a user is on a particular page. Like we first described in *Interaction Design Best Practices*, you can link up your pages and create as many flows as you need.

One quick way that you can begin to begin exploring different page flows is by creating a simple flow outline. Before sketching or prototyping, a written outline helps you explore the most important part of your app or website – the content. Building flows around content gives you a much more accurate assessment of the total number of pages required for the user experience.

Here are a couple techniques for outlining your flow.

1. **Writing-first Approach to Outlining a Flow**

   You can use the writing-first approach, which Jessica Downey writes about in her article Jumpstarting Your App Conception Without Sketching UI. This outlining method helps flesh out ideas and build a "common understanding" of each page of your app or site.

Let's create one for, say, a banking app. The scenario: someone wants to turn on auto deposit. Note in the outline below, content in [brackets] represents action buttons/links.

Step 1: Would you like to set up auto deposit?
[Set auto-deposit]

Step 2: Select Deposit Frequency
[Once per month][Twice per month]
[Every other week][Every week]

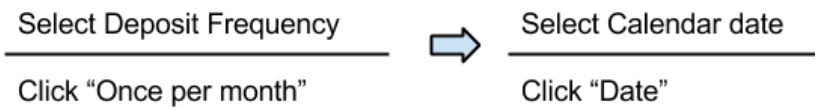
Step 3: Deposit Once per Month
[Select calendar day]

Step 4: Set Amount
Display amount field
[Set auto-deposit]

2. Shorthand Approach to Outlining a Flow

You can also try a shorthand approach used by Ryan Singer at Basecamp. Ryan's approach treats flows as ongoing conversations.

For our banking app example above, we can create a shorthand for Steps 2 and 3 that looks something like this:



To see how Singer demonstrates shorthand for Basecamp and how he can illustrates complex flows with this outlining process, check out A Shorthand for Designing UI Flows.

3. Sketching and Prototyping a Flow

Now we're ready to create a low-fidelity sketch for each page in our flow outline and/or flow shorthand. The sketches bring your ideas to life with more detail around layout and structure. Once you have created your sketches, a simple low-fidelity prototype can help you test those ideas with users.

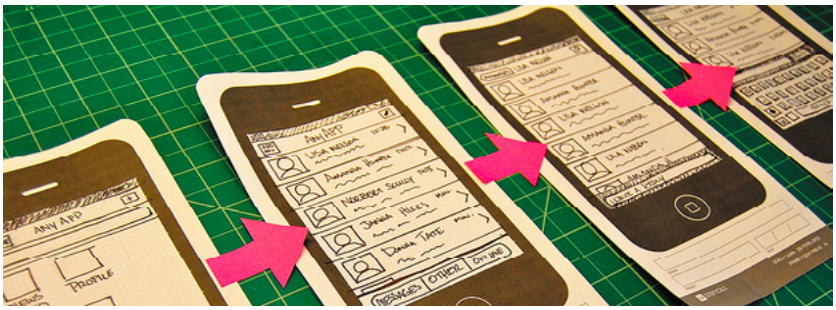


Photo credit: Rosenfeld Media

You can sketch out user flows in a variety of ways, as demonstrated in these examples of user flows and wire flows from Wireframes Magazine. Before committing to any particular flow, however, create a simple prototype to validate that its alignment with your user and business objectives. It doesn't have to be anything fancy – your prototype can be done on paper so you can start understanding how users flow between content and actions.

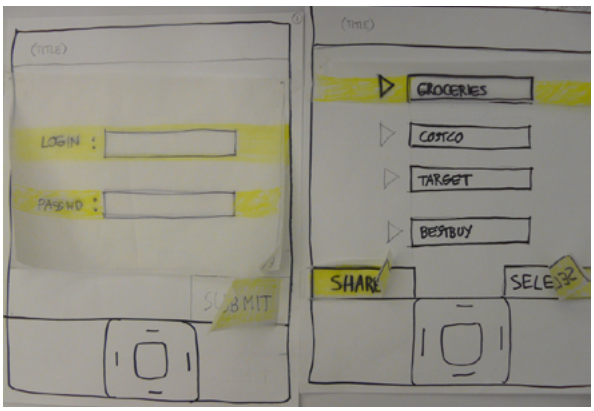


Photo credit: Rodolphe Courtier

From there, you could continue iterating the sketches on paper and cut them out for a paper prototype, or move to a digital prototyping tool like UXPin.

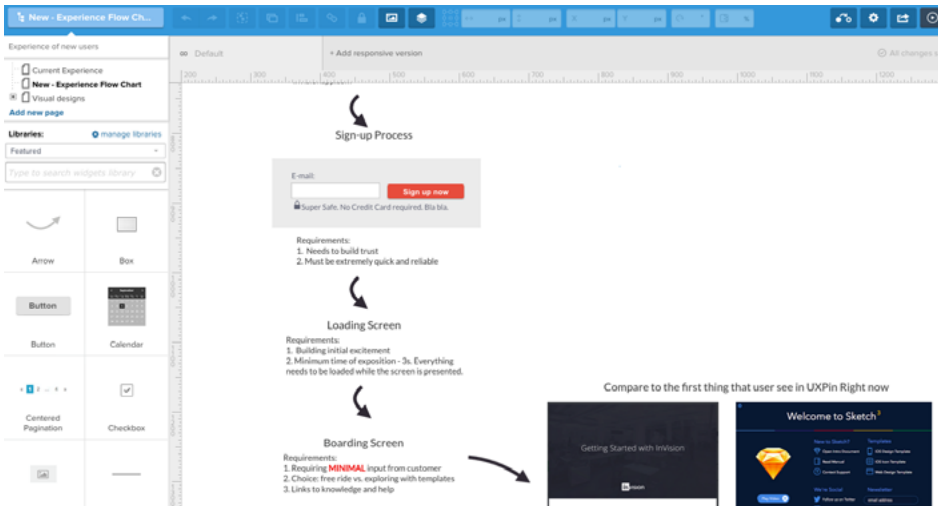


Photo credit: UXPin

4. Real Examples of Flows and their Teardowns

Now that we've shown you the process for creating your own user flows and turning them into sketches and low-fidelity prototypes, we're going to leave you with some interesting examples of real UX flows for *user onboarding*.

User onboarding is a great scenario in which it requires particular skill to balance user needs and business needs. The user wants to dive straight into the app with as little learning curve as possible. The business obviously also wants the user to dive in, but they also want to gently nudge users into upgrading their plans.

You certainly need to master the art of persuasive design in order to create flows that educate users while helping them discover the benefits of upgrades.

For some great analysis, we highly recommend checking out User Onboarding, a site dedicated to UX teardowns for popular website and application onboarding flows.

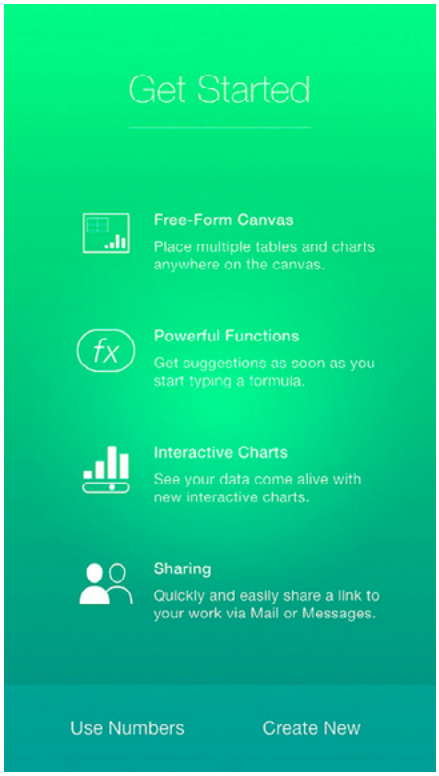


Photo credit: Gustavo da Cunha Pimenta

Lo-Fi Rapid Prototyping Methods

Once you have a rough idea of user flows, how do you bring them to life? Luckily, there's quite a few options for even the simplest of prototypes.

We'll describe the pros and cons of paper prototypes, presentation software, specialized prototyping tools, and prototyping in code.

Photo credit: Samuel Mann

Paper Prototypes

In the age of modern technology, it can sometimes be refreshing getting back to the tools we used in kindergarten. Paper, scissors, glue, coloring utensils – what we once used to make unicorn collages can

now be co-opted for creating useful yet quick prototypes. Let's explore the benefits, disadvantages, and methods for paper prototyping.

1. **Advantages**

   The benefits of paper prototyping are quite straightforward: they're cheap, fast to make, and easy to collaborate with. Here's a few reasons why:

   - **Low budget** – Paper prototypes can be built with supplies already in your office. All you need is paper, pencil or pen, scissors, and some creativity. If you want to get fancier, grab some index cards or Post-Its.
   - **Rapid iteration** – Which would you rather throw away, 2-hours worth of coding or a sketch that took 20 minutes? Due to the highly dynamic nature of the prototyping phase, you're always going to create some waste, so sometimes it doesn't make sense to spend a lot of time on any one design.
   - **Fun collaboration** – It's hard not to bond when a group of people are given art supplies and asked to create. The casual nature of paper invites more participation and feedback, which you should encourage as much as possible during early product stages.

   Moreover, a quick session of paper prototyping can help brainstorm solutions if you're currently stuck with your digital prototype. Prototyping with paper or software don't have to be exclusive – they can complement each other well if you play on each's strengths.

Photo credit: Converting Offline to Online Prototypes

2. Disadvantages

However, paper prototyping isn't for everyone. Jake Knapps, the designer behind Google Hangouts, claims that it is a "waste of time." While we think it has its time and place, these are some of its glaring disadvantages Knapps pointed out that we feel are very valid (along with some of our own thoughts).

- **Lack of direction** – While paper prototyping can be useful for individual processes, you need to explain the context. Otherwise, you might get feedback based on your efforts, not based on your actual product.

- **Inaccurate reactions** – Research should be based on your users' reactions, as those come naturally without thought. Because paper prototypes require the user to imagine what the final product will be like, you're getting feedback on a deliverable instead of reactions to something that resembles the product.

- **Can be slower than prototyping tools** – Given the wide selection of prototyping tools out there like UXPin, Invision, Omnigraffle, and others, you can get prototyping quite quickly (and

less messier) with the added benefit of collaborative features. It all depends on your preferences.

Photo credit: Samuel Mann

In our experience, paper prototyping can be an excellent way of exploring ideas (as long as you know its limits).

In fact, our CEO even wrote a piece for UXMag, where he explained how you get the most out of paper prototyping when you treat it more as an informal concepting exercise. Draw out sketches for the sake of exploring your own ideas, then run a quick hallway usability test with 3-5 people. Afterwards, you can move to a digital platform for wireframing and higher-fidelity prototyping.

To learn the steps of paper prototyping, we highly recommend checking out Justin Mifsud's post on Usability Geek.

Presentation Software: Powerpoint & Keynote

For starters, there's the traditional PowerPoint, a reliable business staple used for presentations for over two decades. If you're looking for a more modern alternative, Keynote is rising in popularity. Interestingly enough, Google Ventures mentioned Keynote as a secret weapon for their "design sprint" initiative.

Let's take a look at the pros and cons so you can make an educated decision.

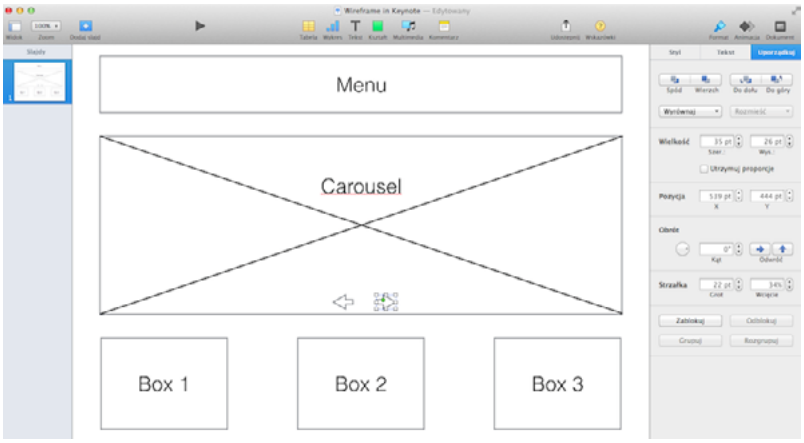


Photo credit: 4 Digital Wireframing Weapons

1. **Advantages**

   Almost everyone has used presentation software before, so they're a quick way to start a simple prototype.

   - **Familiarity** – You know the basics, and it's not that hard to learn more advanced features like animations, slide transitions, and linking slides for interactions.
   - **Basic element libraries** – Thanks to simple wireframing libraries like Keynotopia, you can quickly create low-fidelity wireframes

and then link them together for a clickable prototype. You can also use master templates, and reuse slides or parts of slides as needed.

- **Natural linear flow** – The slideshow nature of these tools takes you through a sequential user flow, which forces you to think about the experience aside from visuals. For more advanced users, you can link slides in complex ways that go outside the linear progression. Most wireframing and prototyping apps can still be clunky for visualizing user flows, but UXPin, Flinto, and Invision do a great job.

2. **Disadvantages**

   Like we described in *The Guide to Wireframing*, once you start playing around with advanced user flows and interactions, you've basically hit the limit of presentation software.

   - **Non-stock element libraries** – It's not easy finding the right element libraries (if they exist at all). Unlike dedicated prototyping tools, presentation libraries aren't updated as frequently and their quality usually isn't as good.

   - **Limited collaboration** – Most presentation software doesn't offer any collaboration (except for Google Presentation). The tradeoff though is that collaborative presentation software lacks interactivity, graphics manipulation, shapes, text, and color options that make them worthwhile for prototyping. If you want to collaborate without compromise, stick to a prototyping tool.

- **Limited flow charting & user flows** – As we discussed, you can communicate advanced user flows since you can link slides together for user flows that aren't purely linear. But it's not easy to do and the sitemaps aren't linked to the prototypes in a way that Axure or UXPin can do.
- **Limited interactivity** – Resourceful users can get pretty far if they use all the features in Keynote or Powerpoint. But once you think about how easy it is to add basic interactions with prototyping tools, and the sheer breadth of options available in the combinations of elements, content, views, and animations, it might just be easier to switch over to something specialized. You may not need them for the lo-fi prototype right away, but you'll definitely want them as you iterate to a hi-fi prototype.

If you'd like to learn more, Keynotopia has some basic prototyping tutorials for Powerpoint and Keynote.

Coded Prototypes

Furthering the discussion from the previous chapter, one of the biggest questions designers have about prototyping is whether or not to use code.

This uncertainty stems from some designers' lack of comfort with coding: they either don't know how to do it, or don't like doing it. When faced with the more fun and intuitive method of using a prototyping tool or even sketching by hand, writing code can feel tedious.

1. **Advantages**

   Today there are more reasons than ever to start coding early, as explained in a UX Booth article by Andy Fitzgerald, Senior UX Architect at Deloitte Digital. The "I design it, you build it" waterfall mentality taken by designers in the past is becoming outdated as technology advances in large strides and collaboration becomes mandatory.

There are a few distinct advantages of prototyping roughly in code, mostly owing to the fact that you're starting the design in something that resembles the final form. Some advantages include:

- **Platform agnostic** – HTML prototypes work on any operating system, and nobody needs outside software to use it.
- **Modularity** – HTML is component-based, which can help with productivity.
- **Low cost (aside from time)** – There's many free HTML text editors, but you'll need to spend some time learning the language before it's helpful.
- **Technical foundation for the product** – Provided you're creating production-ready code (and not just throwaway for the sake of a quick prototype), you can end up saving time in development.

Coded prototypes can be built in a variety of ways like HTML (or even Python), depending on your preferences. Ash Maurya, Founder and CEO of Spark59 and design speaker, suggests using Ruby on Rails because of the ease in which he can set placeholders for each page and link them together for navigation. However, the most popular code choice for prototyping will likely still be HTML.

2. **Disadvantages**

   Of course, the real consideration in deciding whether or not to use code in your prototype is your skill level.

- **Learning curve** – Not all designers have the ability to code, so don't overextend yourself unless you're technically confident.
- **Slower and less generative** – Diving straight into code may inhibit creativity. Ask yourself how many interactions and page flows you can create with 30 minutes in a prototyping tool versus a code like HTML or Javascript.

Prototyping Software & Apps

Eager to dive straight into a program that's an actual representation of your idea?

Photo credit: 4 Digital Wireframing Weapons

The beauty of prototyping software and apps is that it's specifically designed for this purpose, so they provide the perfect balance between functionality, learning curve, and ease-of-use. Both beginner and veteran designers use specialized tools like the ones below – beginners for the ease-of-use, and veterans for the controls crafted to their particular needs.

These tools vary in their capabilities, with some being better attuned to certain situations than others, so it's best to find the one best suited to your needs. To start on your search, you can check out tools like UXPin, Invision, Axure, or Omnigraffle.

1. **Advantages**

   As described in *The Guide to Wireframing*, these tools have an advantage in that they are built specifically for wireframing and prototyping. Once you learn the basic features, you may find it even faster to prototype with these versus traditional methods like paper prototyping.

   - **Speed** – From speaking to our own customers, we've found that power users can work in specialized tools even faster than paper prototyping because they can create, copy, and produce advanced interactions with just a few mouse clicks.

   - **Element libraries** – While tools like Invision work great for quick clickable prototypes that link together multiple screens with simple interactions (like click and hover), other tools like JustInMind, JustProto, and UXPin come with built-in element libraries (and let you create your own for repeated use). Element libraries make it quicker and easier to prototype in lo-fi since you focus more on putting the right puzzle pieces together instead of designing each puzzle piece.

   - **Advanced flowcharting & user flows** – Flow and functionality are the most important aspects of prototyping, and most tools come with these features built in. Most tools allow you to gen-

erate sitemaps as you create new screens for prototyping, and let you see these screens laid side-by-side so you can navigate them.

- **Built-in collaboration** – First, make sure the tool you select has basic commenting and resolving comments capabilities. Secondly, the tool needs to be able to allow collaborative editing and sharing of prototypes (as links). Finally, revision history and cloud storage simplifies your workflow by making it device agnostic. While UXPin and Invision are the most robust, JustInMind, JustProto, Flinto, and Marvel also have some level of collaboration.

- **Streamlined presentation** – This can mean exporting to PDF, a built-in presentation mode, or exporting to a web or mobile app for a real prototype experience. Most tools will export to PDF, while only a few like Mockflow export to presentation software, and only a few like UXPin let you export the file as an HTML-based prototype. Some tools like UXPin and Invision also have presentation and screen-sharing capabilities.

2. **Disadvantages**

   All prototyping tools require a little bit of time to become familiar, but they can be well worth the effort. Nonetheless, let's make sure you consider all aspects before deciding.

   - **Lack of familiarity** – Like anything in life, if you've never used it, you'll need to learn it.

- **Limited fidelity & functionality** – With the exception of tools like UXPin and Axure, most prototyping tools are low fidelity and/or low functionality. For example, Invision is used mostly for quick clickable prototypes, so it's limited to two interactions (hover and click) and you can't create anything in the app (only stitch screens together). If you only prototype in low fidelity, then a simple tool might be just fine. But if you're going to iterate, make sure you pick a tool that supports a fuller spectrum.

If you'd like to learn more about prototyping methods, processes, and tools for a wide range of fidelity and functionality, check out the free *Ultimate Guide to Prototyping*.

Lo-Fi Digital Prototyping Processes

There's no green light that will magically blink when it's time to start prototyping. How and when to prototype is the subject of much debate in the design world (as you can see by the comments in this piece), and various differing theories and strategies have emerged. As discussed in *The Guide to UX Design Process & Documentation*, the traditional linear process looks something like this:

1. **Sketching** – Brainstorm by drawing quick and dirty 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.

However, 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.

Below we'll look at some new variations specifically for lo-fi digital prototypes, and explain their advantages and disadvantages.

1. **Digital Wireframe -> Digital Lo-Fi Prototype**

   Some designers prefer to get to 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."

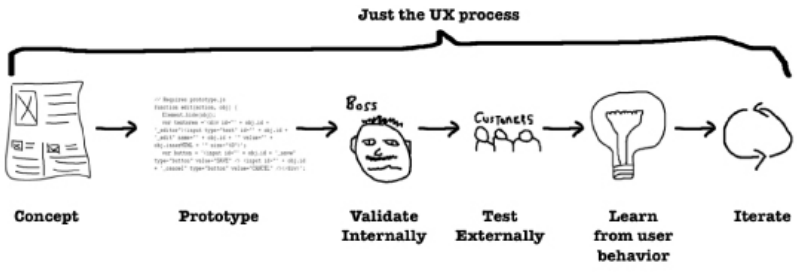


Photo credit: Lean UX: Getting Out of the Deliverables Business

The advantages of a Lean UX process are easy enough to understand:

- **Faster** – Skipping and consolidating phases will get you to the end product faster, but possibly at the cost of quality.

- **More efficient** – The nature of the method is to avoid waste, so the work done will be only 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 lo-fi prototype. This can be done as simply as adding a few animations and basic interactivity to your prototype, testing with a minimum of 5 users, and iterating as needed.

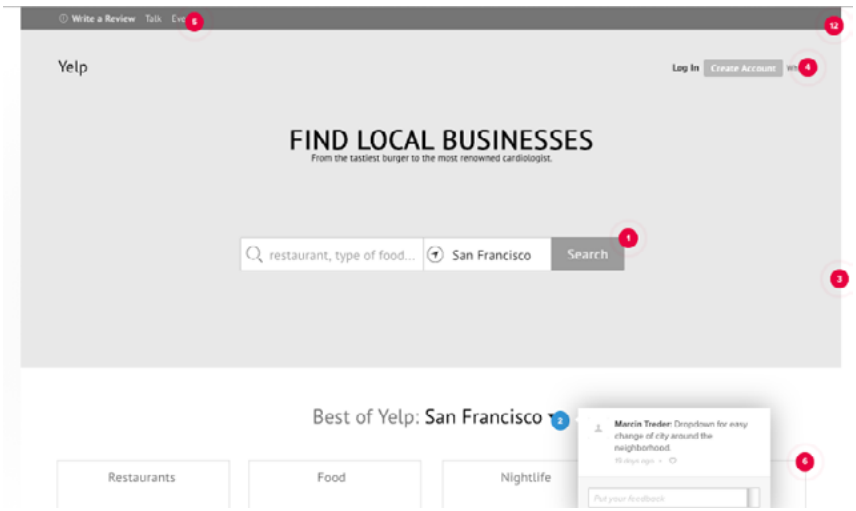


Photo credit: Low-Fidelity Yelp Design

This is very similar to our early process to redesigning Yelp's website as part of a design exercise. We adapted the process to suit our 3-office project by first drawing wireframes collaboratively in UXPin.

From there, we added a few basic interactions (like transitioning to the search screen), plus commented on the features we had yet to incorporate. You can see that the low-fidelity Yelp prototype was far from complete, but far more telling than just a wireframe. Most importantly, we built our prototype quickly and let the design serve as the documentation.

The lo-fi digital prototype is a great tool for testing designs quickly. In this case, the prototype featured just enough functionality and it didn't require too much additional time (since we just added some interactions to our wireframe). After you analyze the results from testing the lo-fi prototype, the next iteration would be a hi-fi prototype (more visual details) followed by code.

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

2. **Sketch -> Coded Prototype**

Introducing code early into the design process has a lot of benefits, namely in having a more solid foundation when beginning development, and in cutting down the amount of needed revisions. But it's not so much a question of *should* you code with the prototype, but *can* you.

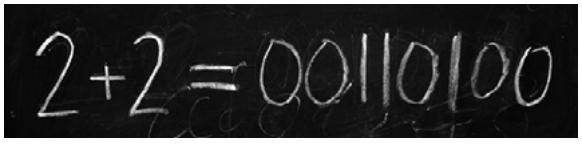


Photo credit: Businessweek

As **Andrew Fitzgerald, Senior UX Architect for Deloitte Digital**, points out in a post for *UX Booth*, 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 mobile and tablet). This lets you first explore ideas on a whiteboard or on paper where they're easy to alter.

Regardless of whether you choose to add coding to your prototype, make sure you get the developers involved in the process.You don't want the first time your developers seeing the prototype to be when you email it with a long list of notes and instructions.

A Rapid Lo-Fi Prototyping Lesson

UXPin offers some excellent tools for building low-fidelity prototypes, including responsive breakpoints that can help you simulate site behavior in different screen resolutions, setting up interactions between UI elements and pages, creating multiple element states, and more.

In this hands-on example, we'll create a multi-state prototype to simulate the interactions a user has with a tabbed container. Each tab contains a different value proposition, which is a combination of feature benefits and imagery.

If you haven't already, go ahead and create a free UXPin account so you can follow along. The live preview of the project is available here, so feel free to open the project for quick reference.

1. **Create low-fidelity wireframe**

   Using UXPin's wireframing tools, we created a low-fidelity wireframe with the following elements:

   - Boxes
   - Text Elements
   - Icons
   - Image Placeholder
   - Button

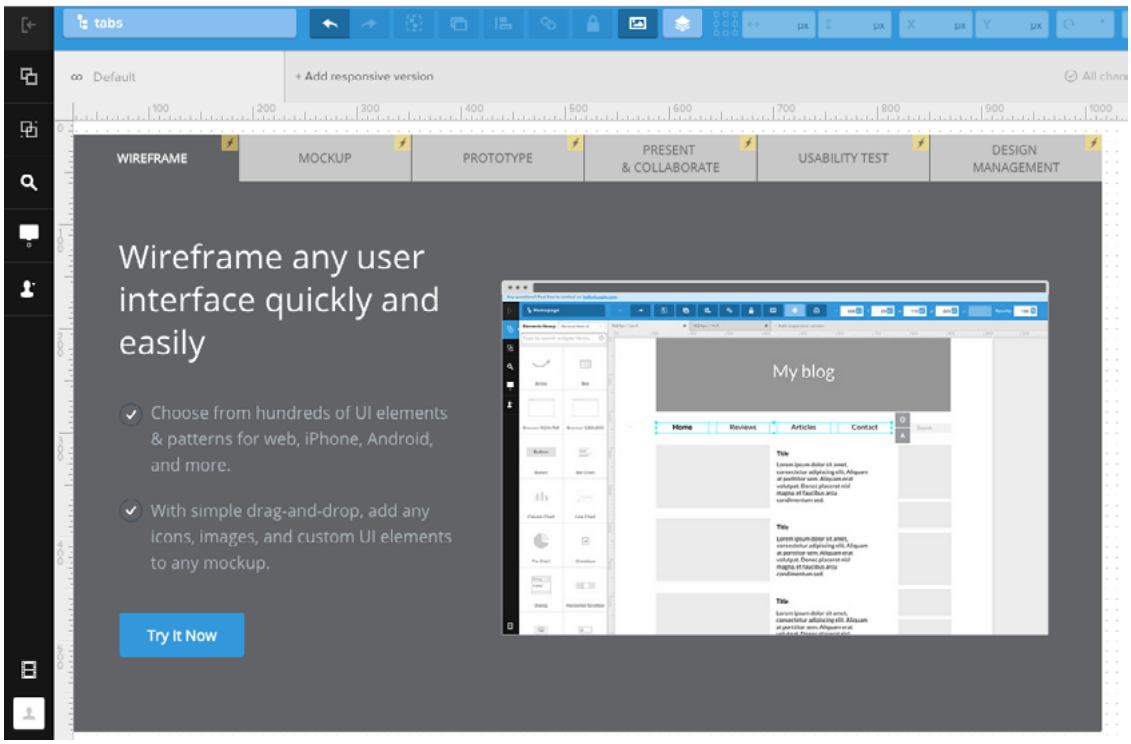


Photo credit: UXPin

2. **Create multiple states for the tab group**

   To create a prototype that allows us to interact with the tabs on-screen, we need to create multiple states for the tab group. Below, we selected all 6 tabs, right-clicked the selected elements and choose **Grouping ▸ Multistate**.

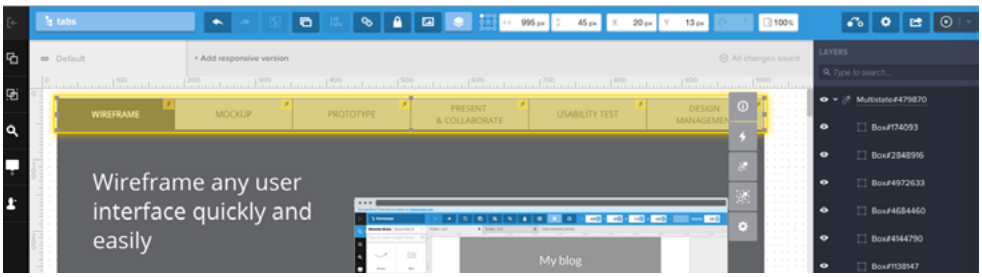


Photo credit: UXPin

Then, we selected the newly formed multistate tab group and clicked the States icon . Here, you will see six states shown: each state represents one of the tabs being selected (and the others deselected).

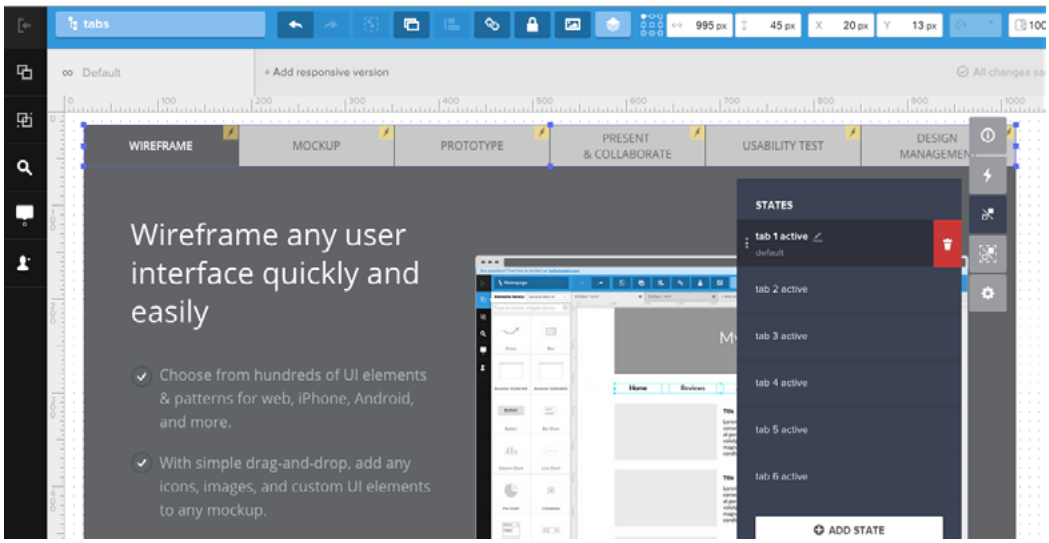


Photo credit: UXPin

The state of the second tab when activated is shown below.

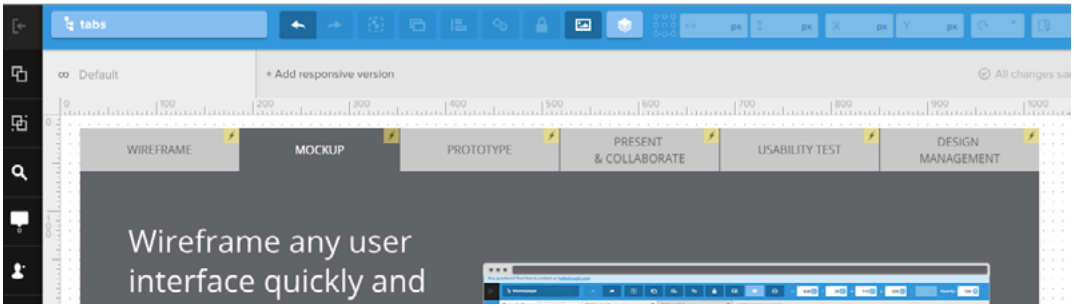


Photo credit: UXPin

We proceeded to do this for all 6 tabs.

3. **Create multiple states for the content inside the container (content and image)**

   Following the same process as we did for the tab group, we selected all the content inside the container, right-clicked the selected elements, and choose **Grouping ▸ Multistate**.

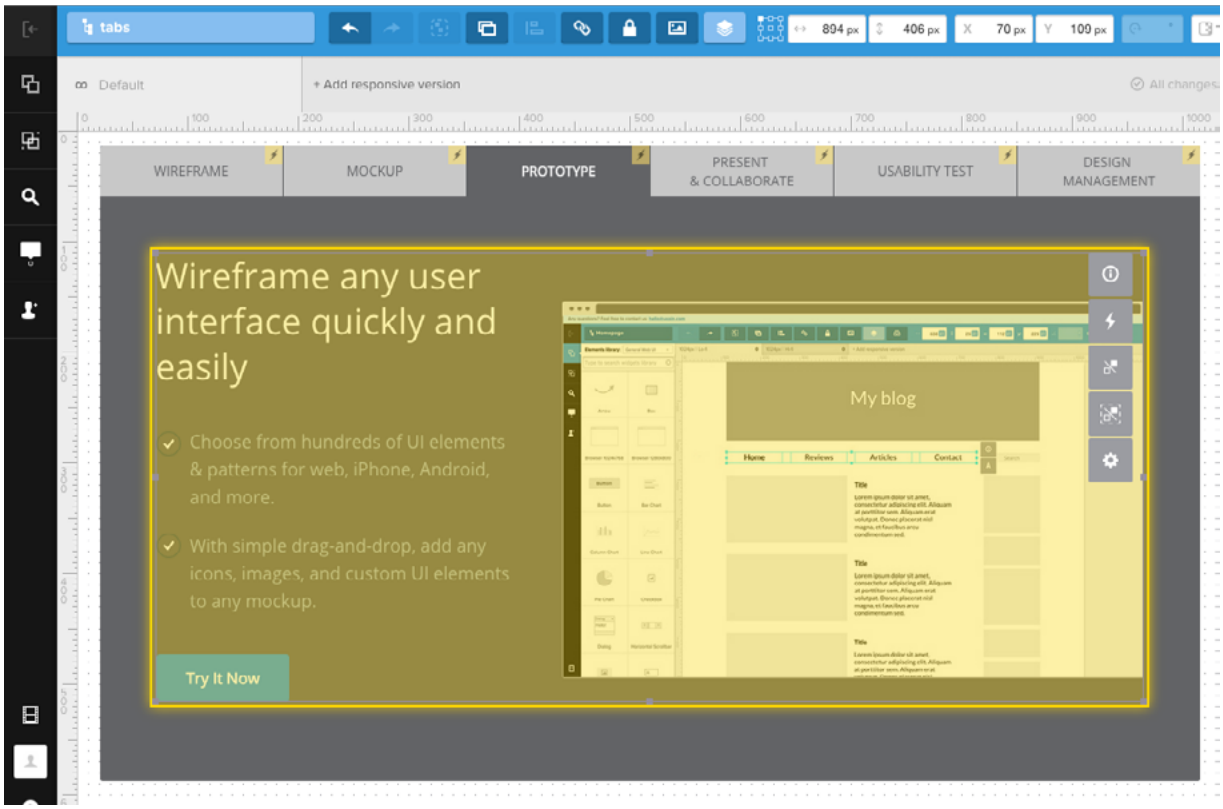


Photo credit: UXPin

Then, we selected the newly formed multistate content group and clicked the States icon . Again, you will see six states shown: each state represents the content changing when a different tab is selected.

We proceeded to do this for all 6 tabs / content areas.

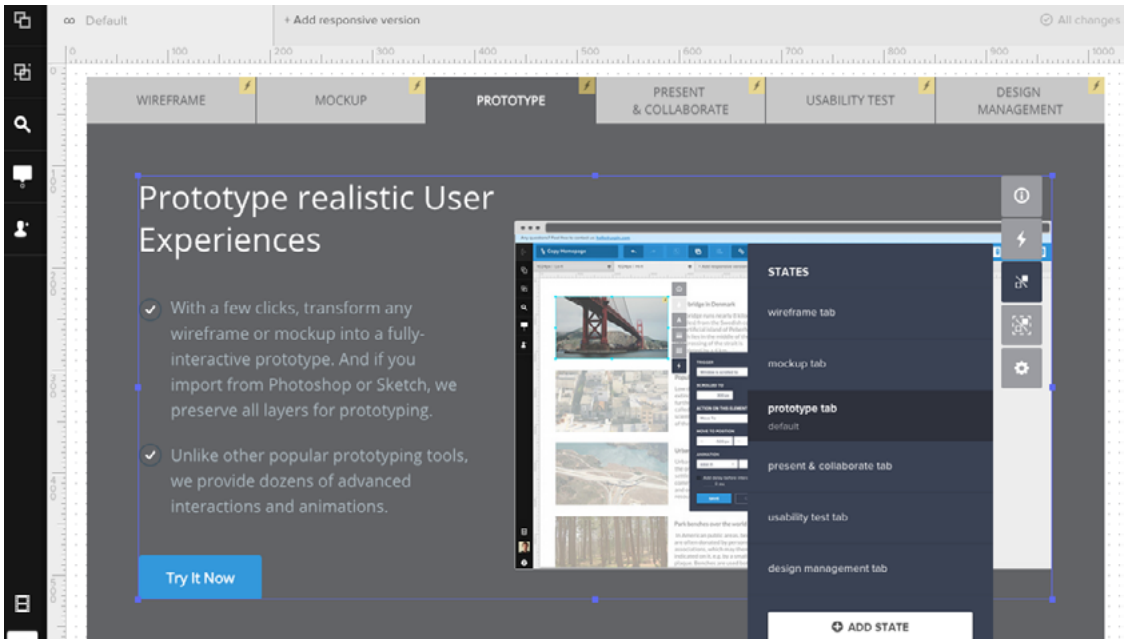


Photo credit: UXPin

4. **Add the interactions of clicking tabs.**

   Lastly, we added the interactions of clicking the tabs by selecting a tab and clicking Interactions ⚡. Each tab has two interactions:

   - On **Click** action, **set the state of the tab** to the selected state.
   - On **Click** action, **set the state of the content container** to the content matching the tab.

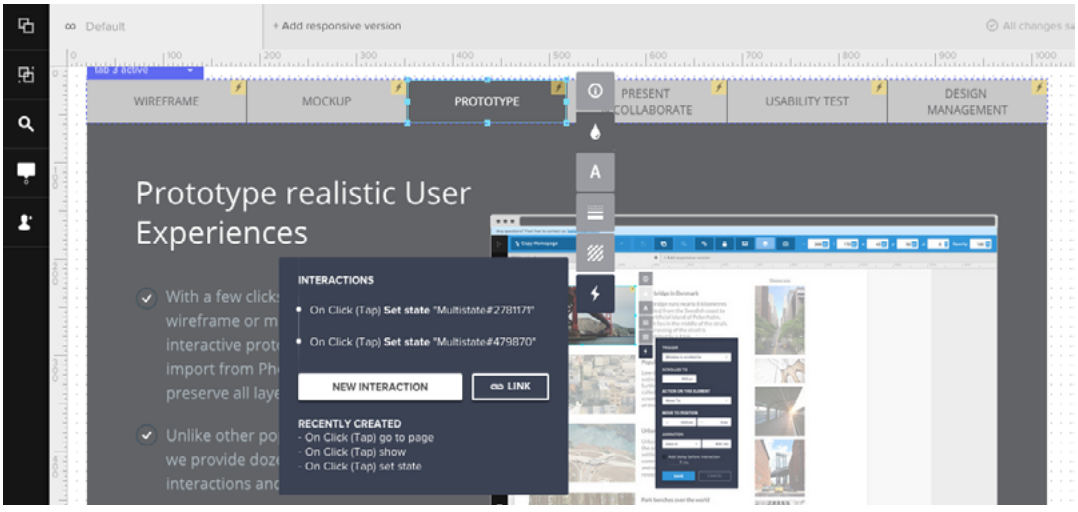


Photo credit: UXPin

Not that we're done with that, we can preview our clickable, low-fidelity prototype by selecting the **Simulate Design** button in the upper right-hand corner of the screen.

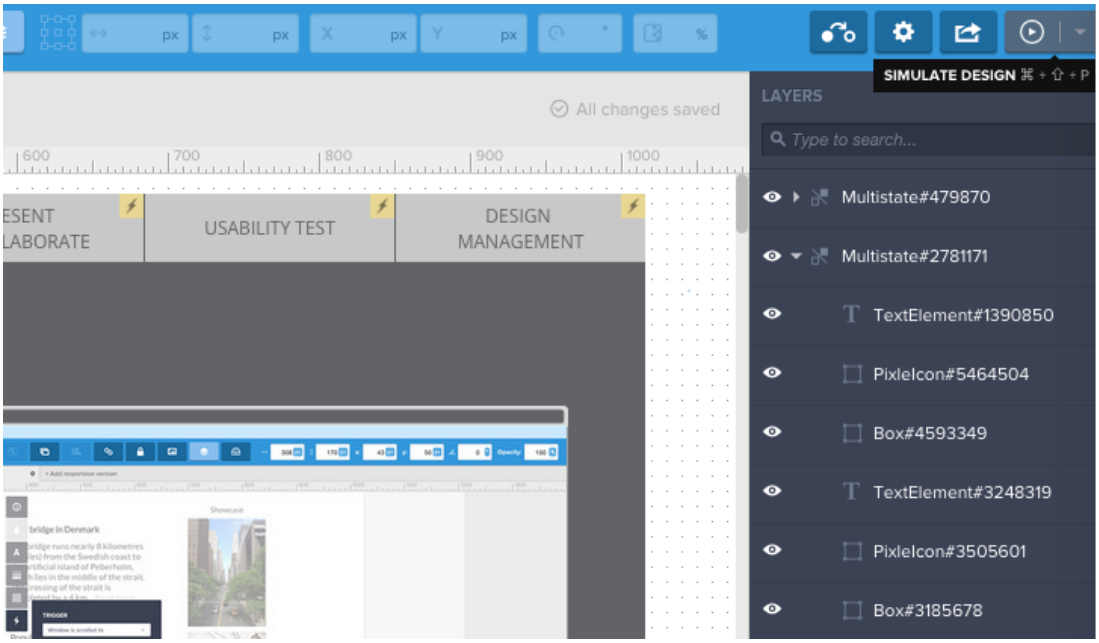


Photo credit: UXPin

And wa-lah! Our new prototype! Check out the live preview to play with the functionality we just created.

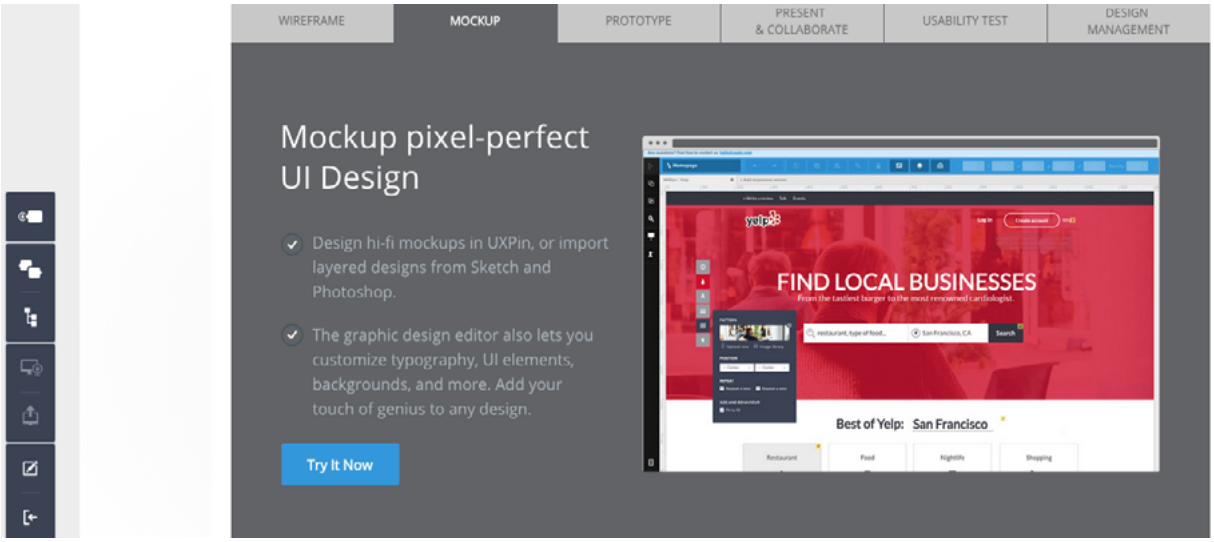


Photo credit: UXPin

In this guide, we've covered *why* we create rapid low-fidelity prototypes and *how*.

Now that we have our low-fi prototype (i.e. interactive wireframe), we can iterate, iterate, iterate on our prototype to put our best digital MVP out there, much like the Eddie Bauer "First Ascent" team did to achieve their Katabatic Tent product win.