Assignment 12: Final Prototype

Brett Bloethner CSCI E-34

Links

Prototype demo part 1: https://youtu.be/jSRC2zPE1x8

Prototype demo part 2: https://youtu.be/73KPZ1-Siiw

Proto io Prototype: https://pr.to/BJLLCM/

Prototype Implementation

How I went about creating the prototype for my final project

Invision App & Sketch v. Proto.io

My final project prototype was the first real prototype I've ever created so I was in pretty new territory. I had originally hoped to use a combination of Sketch and Invision app to create my prototype but quickly ran into problems. Invision app didn't seem to have some pretty important functionality that my prototype needed to really demonstrate the user experience. Specifically, Invision app didn't have great support drag and drop and variables. I started looking for other options and set my eyes on Sketch's new prototyping feature set but quickly found Sketch's prototype capabilities were behind even Invision app. After some more searching and quite a bit of online forum reading, I came across Proto.io. Proto io seemed to have all of the advanced drag and drop and variable features my prototype needed, so I chose to move from making my prototype in invision app and sketch, to making it entirely in Proto.io.

Basic Implementation

Proto.io provided a robust feature set of transition interaction elements which allowed me to keep away from photoshop and graphic editors in place for creating shapes and interaction features and basically directing that X gets shown when Z shape is clicked. This took a while to get used to but after a few hours I was able to hit the ground running. I believe a pretty decent porotype can be created without even scratching the surface of Proto io's interaction feature set. This style of prototyping allowed me to get the prototype completed pretty quickly. Rather than spending time back and forth in multiple apps locally and in the cloud, I was able to focus my full attention on the Proto io app, it's functionality and how it can get my sketches into prototype form. The implementation I did was basically what was in my wireframe sketch set with the exception of a few discoveries made during user interviewing. Combined with the aesthetic plan, I had a pretty solid set of specification to which I was able to build my prototype.

Challenges

Deep linking proved the be a pretty big challenge for the prototype since most prototyping software seemed to be designed to suit only having one entry point. To get around this I had to split the demo video into two videos and the prototypes start page is different for each video.

After doing that I was able to simulate linking directly to a mockup from an agile project management tool.

Proto io didn't have any file drop zone capabilities, so I had to basically describe what would have happened at that point in the demo. Proto io also had a funny bug where hidden elements would still cover other elements' clickable areas. Because of this, only some parts of some of the buttons are actually clickable since they're covered by other hidden elements like the hidden assign a developer or designer dropdown. This could have been a drawback of Proto io but was probably more likely due to the fact that I'm a novice with Proto io and don't yet know how to navigate around this problem.

I also found it pretty difficult to stay within the 8 minutes asked for for the demo video. The most difficult part, however, was trying to simulate the user tagging auto suggest dropdown. I actually found this to maybe be impossible in Proto io and couldn't get it into the prototype unfortunately. What was supposed to happen was that when the "@" was typed a dropdown should have appeared, offering a list of users you could tag. I couldn't for the life of me figure out how to even get anything close to that behavior going in Proto io. I added a tag just by typing in the demo and briefly described what would happen due to the tag.

Video Transcript

Hey I'm Brett and this is my final presentation of my prototype for David Platts class, User Experience Engineering. The primary user for my web application is the typical web developer who works on a small to mid-sized team, like most developers do, and works side by side with a designer or a team of designers. My goal is that this application will help facilitate good communication between developers and designers in order to get them working more efficiently together. My idea came out of my own experiences as a software engineer but was refined with tedious amounts or research, analysis, and interviewing. Lets get started…

There are three main user flows we'll demonstrate in the prototype. Because of the nature of the app two of the user flows have a different entry point into the application. The demo is separated into two clips due to this.

Here we are on the login screen. I want you to imagine that I'm a designer who has just completed a mockup and now I want to upload it for developers to look at and comment on.

I login, and this is my first time logging in so I'm greeted with a telemetry tracking prompt which I'll accept.

Now I'm on the home screen, it's really only designers and managers, or anyone who's uploading and managing mockups, who will ever need to view this page. Most other users will deep link straight into the mockup from outside links.

I want to upload my new mockup, which I've created locally and saved as a jpg on my desktop. Ill drag and drop it into this here file dropzone.

Now I'm taken to the mockup viewer page. There are a few things we can do here. First let's assign a developer and a designer to this mockup that way they'll get notified when changes happen or comments are made. I click the assign a developer drop down, I'll assign this one to Jim Bo. Next, I'll assign myself as the designer.

The mockups almost ready to go, I'm just missing one thing, I know we don't have a preferences page. I jumped the gun in designing that, so I'll leave a little info annotation letting Jim Bo know that link should be omitted.

There we go. Now Jim Bo and everyone else who's interested needs a way to access this mockup. I'll copy the mockups short link to my clipboard and paste it in our team's agile software. For this demo I've created an empty waffle io project. We'll just copy and paste the link in the card so Jim Bo can view the mockup and the notes whenever he needs. He'll also be able to leave annotations on the mockup and once he does, I'll receive a notification of that he did.

But wait, to get notifications I need to configure my Slack and notifications settings. To do that ill click on the setting icon in the top right corner of the web app.

I'm greeted with a settings page that shows the integrations tab. Sign in with Slack, that's what I'm looking for. I'll click that link and follow the Slack authentication callback prompts, once that's completed, I can click this button to quickly get to my notification's configurations. I'll turn all of my notifications on for Slack since I just ignore most of my Slack messages anyway. There, now I'll get notified when anything involving me is changed in the app.

Part II

For part two of the demo were going to look at how Jim Bo might go about creating a blocking annotation. For this let's imagine I'm Jim Bo and I started writing code for the feature that Robert Bobs put in the first mockup we saw. About part way through writing the code for this feature I noticed we don't have an "I'm feeling lucky" feature to go with the button on the mockup. So I need to let Robert know that this feature can't be completed until I know the story behind that.

Let's start by going to our agile workflow app Waffle io. This is where the short link to the mockup was posted. We'll click this link and be taken straight to the mockup.

Now ill click the new blocking annotation icon and it creates a new blocking annotation on the screen. Ill add a comment and press save, I'm also going to tag Robert in this comment so he knows its directed at him.

Instantly Robert Bobs will get a Slack message notification saying that I've filed a blocking annotation and now Bob can go back to the app and respond to the annotation accordingly.