HARVARD EXTENSION SCHOOL

Term Project Proposal

Brett Bloethner

9/18/2018

Homework 1

CSCI E-34

The Users

This web app/project targets two types of users. The first user is the typical software engineer. I think this persona is described very well in the book we were required to read before this course started called, “The Inmates are Running the Asylum.” This user is typically not very well versed in user experience design although they’ll often find themselves making snap decision regarding the design and user experience of the product their developing. In addition to that, this user tends to sway more towards being a little arrogant and lacking empathy.

The other type is an experienced UX designer who needs to better relay designs to developers through a more effective means for delivering design requirements and translating designs into design requirements for developers. This user is well versed in tech, personas, and user stories and understands that good UX design isn’t based on colors and shapes but on metrics, empathy, and an understanding of the user, something they wish their engineers had more of.

These users often times work very close together. However, the relationship between the two types of users usually isn’t optimal. It’s typical for there to be a large communication barrier between designers and developers due to a few reasons, the biggest of which is simply a different way of thinking. Designers see problems in a more user centered way while developers see things in a system engineering way and these two ways of thinking have been said to be fundamentally at odds with each ether.

Business Problem

The business problem is simple. Things cost more money when they take longer to complete and as long as developers and designers have trouble communicating then things will take longer to complete. Then, they will ultimately cost more money. Solving the problem is much more complex. Having worked in software engineering for a handful of years now, I’ve been able to observe the communication problems first hand. Sometimes I’ve even been the root of these problems myself. One of the biggest showstoppers I’ve noticed was in translating requirements from design lingo to developer jargon, which almost always fails the first time around. Additionally, once it’s discovered that the requirements for a task aren’t up to par, then communication issues arise such as scheduling, arguments, and simply running into difficulties figuring out what each other is traying to say. Compound this by adding it every task your development team has to do and you’ll see that it adds up quick and results in a great deal of wasted time and wasted money.

The Pleasurable State

The pleasurable state is open communication. It’s easy to shoot a message over Slack and hang back for a while, taking a half hour break waiting for a design question to get answered, but this isn’t efficient. Often times people dread having to setup face to face video chats or meetings with coworkers to solve a simple problem and it’s understandable because these meetings shouldn’t happen in the first place. Instead, the user should have a medium they can use at work to communicate specific issues with their designer and get an answer back without having to deal with any of the difficulty of describing, via text,

what's an issue that lives in an inherently visual realm. So, the pleasurable state is better and quicker communication as well as an added boost of confidence knowing that you can open a painless dialog to get your questions answered. Added efficiency and ultimately cost savings could be claimed as a pleasurable state for managers of teams using this tool as well.

Characteristics of a Good Solution

Visual: Designs are visual, so naturally, problems with designs are best communicated using the same medium. In my personal experience and in the experience of developers I've discussed this with, when problems arise with designs and the problem solving escalates to a meeting of some sort, the mockup is always present. Always. So, the mockup should be present from square one when issues arise, saving both designers and developers time by no longer having back and forth regarding what element or part of the design they're referring to.

Alert: Promptness becomes an issue when you need to reach out for help getting a problem solved. The longer it takes for the other person to see your cries for help, the less you're getting done. A good solution would do it's best to make sure problems are communicated in real time, using which ever medium is best suited to get in touch with the other user.

Collaborative: Ultimately, a good solution optimizes communication among the development team and the design team. This makes the solution a collaborative solution and as one, it should do it's best to help create a more cohesive and efficient team by enabling them to more effectively solve problems together to the best and quickest resolution.

Characteristics of a Pleasurable State

A pleasurable state with this application would be that developers and designer enjoy their jobs just a tad bit better. Efficiency is a characteristic of an aspect of the pleasurable state that may appeal better to leadership. But when it comes down to it, developers and designers should feel like they're running into less friction in their daily tasks and dealings with their developer or designer counterparts. I think that's a big win for developers and designers alike, and that may prove to be the most obvious characteristic of a pleasurable state for this application.

User Population

This web application is designed to fit into the agile workflow of any development team. Many teams simply upload images of mockups or add links to images of mockups in their respective virtual Kanban board style project management applications. This application would fit right in there, acting as a host for those mockups and providing a communication tool on top of said mockups. That being said, this application could be used by anyone on the development team that has any interest in the team's project management web app. If the person is already in the team's project management web app on the daily, then it's likely they'll be in this app on the daily as well. Those peoples job roles aren't really limited to any particular domain. However, I anticipate the most common users would be developers and designers, with managers and product managers trailing pretty far behind. The application is a team/workspace based collaborative app, so everyone on the app is in a team and involved with their coworkers also in that team, otherwise the app wouldn't provide very much value.

Economics

This app serves to increase the efficiency of a development team, so management of that team has a vested interest in how the app benefits their team and workers. Because of that, I see this being a tool that management ultimately decides to push down on their employees. However, "management" in a software development team could also mean many things. It could be a team lead who chooses to use the product and may have a small budget or simply says they need this subscription. It could also be a VP of engineering who decides he wants every development group in the company to give this new tool a shot. It's a subscription-based web app that will ultimately be sold to someone who in some way has power of their development team's workflow. More effective translation of design requirements as well as quicker time to resolution of problems between design and development will be the core benefit that will lead to more efficiency and costs savings for the company using the product.

Competitors

There are a few competitors in this niche designer/developer communication space. However, many of the competitors seem to have much more full featured applications that are geared more towards designer/client relationships. One example of this is https://slickplan.com, which goes well beyond simply sharing mockups, with its sitemap and diagram builders' tools, just to name a few. Mockvault.com is also another potential competitor although, it doesn't seem to have much of a focus at all on efficient and effective communication about the mockups as opposed to just getting them "approved" by clients. It looks like Balsamiq also has a feature to share mockups called myBalsamiq. However, this tool is also geared more toward the designer/client relationship rather than the designer/developer relationship.

Advantages

Stickiness: If the application gains widespread usage among a team, it could soon become a vital part of how the team functions since it's designed to fit into their current workflow. This would make the team pretty reluctant to leave the platform for any reason.

Notifications: Letting users know when another user needs their input is vital in making sure problems are solved efficiently and effectively. Integrations with communications platforms, starting with Slack, can help make sure every user is in the loop and help make the application fit even deeper into the team's stack of tools they use daily for development.

Designed for Conversation: Testing and constant iteration on the design of the app can ensure that the app is optimized for communication dealing with design, mockups, and requirements. Many project management apps use simple time-based commenting on each task/card/story. Multiple comment threads in this app help to ensure that in app communication is efficient, on topic, and easy to consume.

Alternatives

There are some mockup and prototype tools that can be used as alternatives simply because they allow for commenting of some sort. However, many of these apps are no more suitable for this use case than google docs is because it also can house a picture and comments. Many alternatives would also prove to be too feature rich of a design application and could very well lack an adequate workspace/team implementation that's vital to effective collaboration in any team-centric web app.

Essence of the Project

The essence of this project is effortless communication and to even have the opportunity to prove itself it also has to have an essence of being able to fit snugly into any development team's workflow. If the application proves to be too burdensome for the development team then they simply won't use it. That being said, the application needs to be able to fit right in the flow the teams already use every day and it needs to work to add features to their workflow rather than friction. If the application can do this, only then will it have the opportunity to prove its usefulness via it's feature set. Initial integration is also important. Interfacing with communication apps such as Slack will prove vital in delivering value for the application via notifications and keeping users in the loop. All in all, it needs to be easy to integrate into the team's workflow, dead easy to setup for each users integrations (Slack, email, etc…), and it needs to be able to show its worth when creating design/development requirements and when problems arise between developers and designers.

Hardware and Software

The app is a standard desktop web application with user authentication and as such would require the same things you would expect from an app like facebook or twitter. A modern computer, keyboard, mouse, internet connection, internet browser, and email account are required to use this app/project.

Prototyping and Development Environment

Prototyping would be done via a rapid prototyping application like balsamiq or invision app. Well validated designs would then be delivered to engineers who would decide on how to architect the supporting API in a scalable way while the frontend single page application would be built to the specs decided on between the API/backend and frontend developers. The actual development environment could be a simple NodeJS/Express API and an Angular 5 front end single page application. To launch, this sort of application could be easily buildable and maintainable by a small team of full stack JavaScript engineers.