Assignment 3, User Stories

Brett Bloethner

For my user story assignment, I interviewed one of my co-workers who is a web developer and works closely with our web app designer. His name is AJ and he matches my projects primary persona. I made sure to let him know honest answers were the most valuable and to share his opinion and stories no matter how harsh they may sound and whether or not they're related to myself as his coworker.

Interview with a User

Q: Tell me about your development process at work.

- User experience engineer (designer on the team) mocks out wireframes first
- A agile "scrubbing" meeting occurs where leaders on the team meet to review each designs feasibility, acceptance criteria, functionality, and translation into agile cards, or tasks, then the cards or tasks get assigned to developers
    - UX designers designs the feature
    - Front end okays that it's feasible and gives a time estimate
    - Back end okays that it's feasible and gives a time estimate
- Once a developer finished a task the card goes into "needs review" where the designer looks at the finished feature and either confirms it fits the spec or sends it back to the developer.
- Some things may get sent back to the designer by the developer before the task even reaches the "needs review" stage. This is indicative to two things…
    - Failures to communicate requirements in project management app
    - Failures to identify requirements in scrubbing meeting

Follow Up: Can you tell me more about the process when you send an item back to the UX designer for refined requirements?

- Things needs to be refined and communicated better in the scrubbing meetings
- Sometimes the refinements can be formal but often times it results in just a conversation
- Its inefficient to try to refine these requirements over messaging apps due to the subject matters complexity and how it mostly relates to images
- Live meeting is better (in person or over video chat) but only because data can be relayed faster and mockups can be referred to better, otherwise the communication medium doesn't matter

Q: Do you see yourself as a designer?

- Forced to be a designer since the requirements come as mockups without color or very specific specs
- Tries to implement standard CSS style library in order to limit UX designer so it's easier for developer to fill in the gaps with their standardized designs and specs rather than getting creative on their own
- Uses VMWare Clarity UI CSS framework along with Bootstrap
- Tries to just make things aesthetically pleasing

Follow Up: Tell me more about your role as a developer designer in regards to your impromptu designs? When do you do those and what are the outcomes usually?

- The chances that the rest of the team likes the impromptu designs are 50/50
- Often times the developer need to revisit his design at least once more
- Developers impromptu design could have been avoided if better requirements were made in scrubbing meeting

- Developers go straight to code so the designs they make cost more time when they have to go back and revisit the code over and over because the team doesn't like the impromptu designs

Q: Tell me about one time when you had an especially difficult time regarding the UX Designer/Developer relationship?

- Developer had to build a feature which included "nested tables" in Angular. It was an unknown for him, he was unsure of how long it would take or how difficult it would be.
- Original timeline was 3 days but after working on the task for 2 weeks, developer decided to call and emergency meeting with the UX designer
- The backend forth designs refinement process took about one week and the newly agreed upon designs were completed a few days after that.
- Developer and designer primarily used video chat, Slack, and virtual Kanban board (waffle.io) to communicate although video chat was usually the most effective and provided the quickest results

Follow Up: Can you describe what your video chat session are like when you have to meet with your UX designer to refine or clarify things?

- Unnecessary work is saved when you just point at a picture and say "this right here is what I'm talking about"
- Much of the time in text only alternatives is spent just figuring out exactly what part exactly is being talked about before people involved in the discussion even get to talking about the actual problem.
    - This process/pattern often repeated in each new thread even if it's already been addressed in a similar thread
    - Takes time out of the day if they have to go back and read an entire thread to get back in the right mindset and figure out whats being talked about when they see a new message response days later

Q: What other UX related pain points do you have?

- Sometimes it takes too long to create the initial wireframe. Sometimes Ideas are presented and discussed before the wireframe is created
- Reworking items due to lack of acceptance criteria
    - If the card is in the back log with too few acceptance criteria, then he assumes the team wants him to fill in the empty design specs himself. His designs only get okay about half the time and he does them on the fly so he invests development time into it
- Wireframes are not detailed enough in general

Follow Up: When you envision an app or tool that can help you address these pain points and other obstacles we've discussed, what features and functionality do you see?

- Must include wireframes, wireframes/designs are essential to any discussion that bridges UX and development
- Must help the designers and developers open a quick dialog between each other and promote the responsiveness of both sides of the conversation
    - This could improve the time between messages as opposed to messaging platforms like Slack and email, resulting in getting things resolved faster
- Must be able to integrate into agile workflow so that it feels less like another tool and additional work
    - Deep linking from agile project management tools and virtual SCRUM/Kanban boards
    - Native integrations with project management tools used in development/design
    - Usefulness in original scrubbing/planning of a task/feature could be key to making the tool useful later in a tasks/features lifecycle

Q: Tell me more about the UX designer role on your team and the relationship you two have.

- The design is exclusively the decision of the UX designer, except for when he doesn't want it to be by leaving out details in the mockup and occasionally accepting developers impromptu design decisions
- Professional and respectful relationship
    - Developer pings designer if something is incorrect or unclear
    - Dialog is opened up and item is redesigned
- Things don't often get personal
- Some people can get personally offended by feedback or raising criticisms on their design opinions
- Sometimes designer can be stuck in their old ways but sometimes designer is also too dynamic and wants to try too many new things and introduce too many new patterns into the app, costing a lot of time

Follow Up: Developers and designers can sometimes be at odds with each other. Have you ever experienced this between yourself and your designer? And do you see the developer/designer relationship as one that often needs to be mediated?

- There have been a few occasions where designer pulled the trump card and said he's the designer so it's his decision
- Developer just goes along with it when designer says his work is law
- The need for mediation is rare but could be helpful since he's seen some heated debates arise on this team

Michaels Short User Stories

1. When I'm conducting the scrubbing meeting, I can easily assign a designer and developer to a mockup, that way each person knows who to contact with questions.
2. When I run into a blocking issue on a feature I'm working on, I can easily leave a message at a position on the mockup that will notify the designer of my issue, that way I don't have to setup a time to video chat with the designer with the mockup.
3. When I'm leaving a message on a mockup, I can easily search for and tag one of my teammates to notify them that they need to be brought into the conversation.

Spin off story: When I want to get notified of updates to a mockup, messages I'm tagged in, and messages of mine that have been responded to, I can configure an integration for slack or email so I can be attentive and not a blocker for me teammates.

James' Short User Stories (for 2nd persona and not being used in the rest of the project)

1. When I complete a mockup, I can easily upload it to the app and receive a shortlink to the mockup online, that way the mockup is easily accessible from our agile software.
2. When I need to iterate on a design that had an error, I can re-upload the mockup, updating the last mockup and notifying the parties involved of the change
3. When someone has a question about my design, I can get notified via slack with a message including the mockups short link, that way I can respond quickly to problems and get the developers back on track.

User stories

The high-level context is the same for each story. The context is set with AJ choosing to implement FireUX for his team, the user stories drill down into the ways in which AJs team uses the application.

Eight co-workers make up the team that AJ works on. AJ is on the UI development team, his title is “Software Engineer” but many others would call him a front-end developer. There are 3 other developers who work on the backend API as well as one UX designer and a manager who runs the business and marketing aspects of the product. This month is a special one for AJ. Usually AJ just takes cards out of the task backlog, completes them, and goes onto the next task, but this month AJ is completing his rotation as the SCRUM master. With this added bit of power AJ can take the opportunity to implement a few ideas he thinks may optimize the team’s workflow.

One of the downright largest problems AJ has noticed on his teams is how tasks constantly go back and forth between the designer, the backend, and the frontend developers after they get started. Most of AJs team attributes this to lack of substantial requirements and acceptance criteria in the tasks. What causes the lacking requirements is up for debate between those of his team members that think its a natural cause of going “fast” and those of his teammates that think its simply laziness. Either way, the added back and forth for nearly every task has been stressing AJ out ever since he started 2 years ago and he can tell it’s stressing out his other front-end developers too, so something needs to be done to fix it.

He’s tried simply telling people to add better requirements before card are worked on and that works for about a week before everyone goes back to their old ways. AJ needs to find something that fits in their process, something that the team can’t just forget to do or can’t just stop doing and go back to their ways instead. After researching for a bit, AJ found a web app called FireUX and it promised to fix this very problem. He signed up, created a workspace, and added all of his workers.

1. **Writing better requirements**

   It’s scrubbing meeting Thursday and it’s bitter sweet for AJ. On one hand AJ and the rest of his team get to avoid any real work for a few hours while they choose what to build for the next iteration, but on the other AJ has to face lacking requirements once again. Attempts to reprimand co-workers because of poorly thought out requirements are usually meet with “move it over into the backlog and ill do that later.” Later usually never comes and is instead replaced with long threads in the teams virtual Kanban board where the designer and developer try to parse out the actual requirements on the fly.

   AJ’s determined to do it differently this time. Instead of simply copy and pasting links to mockup jpegs, AJ’s making his team use FireUX to host mockup images and mockup and design related conversations. Everyone’s pretty reluctant to try yet another new tool, but AJs driving the scrubbing meetings from his laptop, so today he has the power force this new tool on everyone. A quick demo shows the designer how to upload mockups into FireUX to prepare them for being pasted into tasks on their virtual Kanban board then AJ gets started with his team’s agile app’s inbox section, where all the teams

feature drafts and ideas live. The team goes through each one, adding the mockup link to their agile cards and parsing out the acceptance criteria. AJ knows that each card needs a front man for the design and development aspect. Typically, not having these people assigned would result in someone sending a message in the team chat asking for help from anyone who may know the answer. Now, AJ and his team are going to figure out who knows the answers before the problem even occurs. With his team, AJ assigns a developer and a designer to each mockup. These two will be in communication whenever an issue arises or a change occurs on the mockup. AJ knows that having a specified person to ask when a blocker pops up will already save his team some time and this is something the team feels more confident about. Now there will be less cries for help to everyone in the group chat by one person trying to get an answer from anyone that might possibly know.

2. **Accessing Mockups from Agile Software**

   One of the biggest complaints AJ faced when pushing FireUX on his team was the difficulty of integrating yet another tool into their workflow. His team already used one tool for project management, one for chat, two for video chat, one for email, and one for creating mockups. AJ assured the rest of his team that if FireUX ends up slowing them down and causing more difficulty, then they'll just stop using it all together. AJs hoping this new tool will deliver and do it fast, that way he and his team can put this worry to rest.

   One of the biggest obstacles AJ and his team face against getting things done effectively and efficiently is getting their questions answered, especially blocking questions. Blockers, like any developer knows, are issues that simply stop development in its tracks until the issue is resolved. AJ would like to see his team work blocker free but AJ and his team know that blockers are inevitable and they see plenty of blockers every iteration.

   Another blocker pops up, this time its blocking AJ against getting his development tasks done. Typically, AJ would send an email or a message to designer about this blocker, since it's specific to the design implementation of the task he's working on, but he's had pretty bad experiences using that method. Often times there's a long back and forth conversation in a comment thread in Slack that eventually leads to a video chat or meeting and the problem isn't resolved for a few days sometimes. AJ remembers a feature he noticed in FireUX that allowed him to add blocking annotations to the mockups, alerting everyone involved that he had a question that was blocking him from getting his task done.

   AJ visits the associated mockup by clicking the direct link to it that was pasted in his tasks card in his team's agile project management software. Easy enough. He then continues to create a new blocker, adding details and pinpointing on the mockup where the problem is occurring. Instantly the designer involved gets a notification and visits the page to quickly identify and resolve the issue AJ had. This was a breath of fresh air

for AJ. In the matter of a few minutes, AJ got his question answered and could get back to work finishing his implementation of the design assigned to him. No more long comment threads trying to explain the problem, no more face to face meetings, no more frustration.

3. **Tagging and updating mockups**
   Software engineering often happens in teams or pairs. AJ's team is no exception. When AJ pair programs with his co-workers Mike and Steph, it's much easier to catch potential problems and write more performant code.

   Before AJ and his team had the FireUX app at their disposal, they used to have to keep everyone in sync via instant message, email, or telling them things in person. It became way too easy to miss a message or forget some important information that was relayed to you regarding a change in a design or a change in implementation. This frustrated AJ and his team and lead to wasted time. Doing incorrect work because someone missed a message wasn't a rare occurrence on AJs team, and that's never good for metrics and never good for morale.

   With FireUX, AJ and his team are easily kept in sync. AJ was working together with Mike on a special and complex feature. When he ran into an issue AJ, rather than messaging Mike simply mentioned his name in the app, Mike was notified and everyone was kept in sync. It was a breath of fresh air for AJ, Mike, and the designer. No longer did they have to worry about executing on the wrong acceptance criteria or possibly being left out of the loop about a recent change in the design. Everyone's in sync when anything happens and this helps make the team more efficient and effective.

4. **In the Loop**
   As the SCRUM master, one of AJs jobs is to make sure no one's blocked on anything. AJ knows this sort of stuff should usually be reported in the morning standup meeting. However, sometimes a teammate may be too busy to attend the standup and sometimes things in the standup meetings are forgotten just shortly after the standup ends. This has proven to be a problem for AJ a few times, as he noticed blockers spanning a few days without any resolution.

   This is frustrating for him, since he's supposed to keep these blockers in check and make sure everyone knows when something regarding their work is blocked or blocking someone else. Sometimes AJ has so much work and so little time that worrying about blockers takes the back seat to his writing and testing features.

   He needs to find another solution and he's hoping the FireUX app may be the answer. AJ sets up the app, and directs each of his colleagues to setup notifications in the application, that way everyone gets pinged when something involving them changes. Now blockers are automated. When someone adds a blocker, everyone with interest in the blocked feature knows immediately what the issue is and that's without AJ having to

lift a finger or even think or remember who the blockers related to or what the blocker is. AJ quickly finds his role as SCRUM master a little bit easier now that the team's blockers are no longer blocking him from getting his work done, and that's something AJ enjoys.