Assignment 9: Telemetry

Brett Bloethner CSCI E-34

User Personas and Stories Overview

User persona and stories (no points, but your assignment will not be accepted without them)

- My primary persona is a 30-year-old software developer named Michael Davis. Michael studied computer science at Stanford and works at Stratos Cloud where he's on a team with 3 other front-end developers who all work together to translate mockups into working code. Most of the mockups come from their design team lead by James Schlotzsky. Michael and his team are constantly having to go back and forth with James and his team regarding missing requirements or just (in their opinion) plain downright poor design. Michael wants to improve communication between his team and the design team and minimize this back and forth to ultimately get more done faster. Being able to communicate better through design assets seems like a good start and Michael is willing to give that a shot with hopes that it may eliminate the long design related threads that make their way into the teams Kanban board software while also minimizing impromptu meetings between his team and the design team. Michael realizes to power of telemetry and metrics but he's cautious to implement it in haste and risk compromising sensitive user data in the process.

- **Michaels user stories…**
    - 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.
    - 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.
    - 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.
    - 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.

- My secondary persona is a 46-year-old user experience designer names James Schlotzsky. James is on the receiving end of Michaels constant complaining about designs and design requirements. This happens on every project James has worked on though so he's gotten used to it. Regardless, James heard about the application from Michael and his team and is willing to have his group use it as long as it improves communication between everyone and gets the developers to stop bugging him and his design team. James is skeptical about whether the app will really help but he appreciates the effort Michael and his team are putting forward to improve the relationship between the two groups, so he'll give a shot. The most important thing to James is that he and his team are able to communicate the requirements better or at

least answer questions quicker and more conveniently, that way their department can meet the tight deadlines set by leadership.

- **James' user stories…**
    - 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.
    - 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
    - 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 Population

What is the approximate size of your user population?

I think if the application were to be successful then it would have a fairly large user population in the thousands to tens of thousands of users. However, it's important to note that the user population would be segmented into groups or teams due to the collaborative nature of the application. This may not be a large consideration for many other applications but I think it's especially important for this app since the content a typical user works with is likely to be content created by one of their teammates and this design could affect which metrics are important.

Content Creators

Content creators would be those who are responsible for uploading mockups to the web application and hashing out the original mockup notes, info annotations, and assignees. Any user could be a content creator since often times teams may meet together and while meeting any one user may take the lead, uploading mockups on the team's behalf. However, I anticipate that the content creators on any given team will likely be comprised of designers and project managers, not developers. This is simply because it's the designer and project managers that spec out any task's requirements and acceptance criteria while the developers ask questions about it and execute it.

Other Users

Users that aren't content creators I expect to be primarily developers. Although the same tasks (commenting, creating annotations) that non-content creators do often could also be expected from users that are content creators since there are no users that I expect would not participate in annotating or replying to a comment on a mockup if their input is valuable.

I anticipate the population of an average team of developers, designers and project managers to be around ten and no greater than twenty users. All of these groups/teams, combined together, would create that entire user population of thousands to tens of thousands of users. The range of thousands to tens of thousands of users reflects user population metrics for

applications I've seen with similar use cases like project management applications such as Waffle.io.

Frequency and Duration

What is your expected frequency and duration of user sessions, approximately?

This application is designed to fit into the daily workflow of designers, developers and project managers. As a result, I'm pretty confident that the average user would have multiple sessions per day in the application. The type of user they are can greatly affect the duration of the session, however. For example, if the user is a content creator, then they may have sessions of longer durations as they upload mockups, create initial annotations and notes and assign responsible parties to the new mockup. I think the two most typical sessions we would see are long sessions with high inactivity where the user has a mockup open simply for reference while they build the feature (this is common for developers), and short sessions where a user enters the app only briefly and only to add an annotation or reply on a comment thread in an already existing annotation. Although the session durations may vary, I think it's safe to say that the vast majority of users would have multiple session per-day no matter their role on the team.

Telemetry Permission

Describe your strategy for getting users to grant permission for telemetry. What percentage do you think will sign up? In what ways might this population skew from the centroid, and how will you attempt to detect and control that situation?

One of the trickiest aspects of telemetry permission as it relates to this application is tracking telemetry metrics where the collaboration occurs. It's possible that tracking metrics incorrectly could result in one telemetry granting users' metrics suggesting the behavior of the user they were collaborating with who may not have granted us access to their telemetry data. To help solve this I think it's important to make sure the user knows exactly what is and isn't being tracked.

I think the best way to ask for telemetry permission would be to ask first thing when they enter the app for the first time. Having a pop-up modal where the user can accept or reject our telemetry consent ask could make it pretty easy to also add some vital information that may make the user feel safer about sharing their information with us. I think transparency is key here and that the majority (more than 50%) of users would grant us access as long as were transparent with when and how their data is collected and how its anonymized. In this modal, I think it would be beneficial to outline our privacy and data collection policy in bullets as well as a link to the full-length agreements in addition to a short video describing how we treat telemetry data and why our users should trust and feel safe with our data collection policies. If our signup metrics were to stray array from the centroid I think it would because users may be hesitant to agree to telemetry data collection when the data they're working with is property of their employer which in many cases I imagine it would be. I think to help quell this concern some users might have, we need to effectively communicate that telemetry data

doesn't include any sensitive information related to the mockups and that it's comprised entirely of event metrics that include no personally identifiable information since these metrics would be stored separately from the user database and other app related databases.

One potentially interesting aspect of granting telemetry permission with a collaborative application is that special attention needs to be paid to fact that telemetry data of one user could suggest the behavior of another user that may not have agreed to provide telemetry data. I think an easy way to solve this problem would be to organize the telemetry data in such a way that all of it describes behavior related to the mockup. The event data would show that user X replied to a comment X on mockup X rather than user X responded to user Zs comment on mockup X. This could make the data more difficult to analyze but I think it could effectively ensure that a non-consenting user's telemetry information can't be implied from another user's telemetry metrics.

Analytic Questions

Describe the top 5 analytic questions that you want to answer. Provide the answer that you expect, and explain under what circumstances your guess would be right and wrong.

- **Who is uploading and creating mockups?**

  I assume that it will be mostly designers uploading mockups. Naturally, this makes sense because designer are the ones who make the mockups. However, the app isn't a tool only for designers, it's a tool for the whole team. I would be curious to find out if it could be project managers or even developers (rotating as scrum masters) uploading the mockups. If we see that designers are uploading the mockups and filling in the initial notes then I think it's safe to say that the assumption was correct. If the analytics show that mockups aren't usually uploaded by the designer, or they're uploaded by the designer but the designer doesn't fill out the initial notes and assignees then I think that's a good indicator that the app has room for improvement to better fit into any development team's workflow.

- **Who is creating blocking annotations for mockups?**

  I would expect that developers are making blocking annotations almost exclusively. This comes from the idea that a mockup or task goes to the developer then the developer finds a problem with it and declares it a blocker that needs to be resolved with some outside input. This assumption can be affirmed if we see annotations being created by developers. If we see that blocking annotations are often being created by designers and project managers then that might require some more user interviewing since I cannot think of a sensible reason why that could occur given the development teams I've work with and talked to before and workflows I've seen or experienced in the industry.

- **Who is creating info annotations?**

Info annotations could be created by basically any user for any reason. My assumption is that these will be created by designers to point out specific key details about the mockup. However, I could just as well find out that info annotations are more often created by project managers or developers than designers. This is also a simple event measurement where we simply see who is creating an info annotation and whether or not they're a developer, designer, or a project manager type user. I think the results of this metric could be pretty revealing as to the exact use case info annotations are serving and if we get some insight there, we could better tailor info annotations to its specific user type and related use case rather than having it be simply a note and comment thread attached to a point on the mockup.

- **How often are mockup images updated/changed?**

  I'm anticipating that mockup images won't be updated very often. If a mockup image is reuploaded then that must mean that the mockup was so incorrect in the first place, that it had to be revisited entirely and uploaded with the changes to make sense at all and that annotations weren't enough to resolve the problem the mockup had. I think this would be a very rare occurrence since I don't think a designer would upload the mockup until they're pretty confident that the mockup is good to go as is. I think if its discovered that the mockups are being changed constantly then we may have to revisit how we treat the mockups images and see if there may be an API to integrate the mockup software directly with the app rather than a simple image upload.

- **What is the designer to developer ratio on each team?**

  I expect that the designer to developer ratio would reflect what I've seen on most other development teams where each team consists of primarily developers and one or two designers as well as one manager or project manager. This should be easy to review with the ability to look at team sizes and composition. If we found that there were far more designers on average than expected then that would raise some questions about future UX considerations since there was no expectation that designer type users would make up that large of a portion of a team.

Data Points

For the previous section, where would you place your measurement points in your program to find out what you want to know?

- **Finding out who is uploading and creating mockups**
  Figuring out who is uploading and creating mockups should be fairly easy. For this we don't need any personally identifiable information since we don't need to know the identity of who's uploading the mockup, we just need to know what type of user they are. At signup or team creation, we would probably have to have the user declare whether they are a designer, developer, or a project manager. Then, we fire metrics

calls to Segment/Mixpanel for both the events of when a mockup is created and of when an image of the mockup is uploaded. In the event calls we will also include the user type that fired off this event. This should be sufficient to tell us if designers are the primary users responsible for creating mockups initially as well as uploading the mockup images.

- **Finding out who is creating blocking annotations on mockups**
  Figuring out who is creating blocking annotations on mockups would have a similar implementation compared to figuring out who is uploading mockups. The only difference is that this time the metrics event call would be fired once a blocking annotation is created and the user type (developer, designer, project manager) would be included with the expectation that the user type will almost always be of type developer.

- **Finding out who is creating info annotations**
  Figuring out who is creating info annotations should be as easy as figuring out who's creating a blocking annotation. The functionality and purpose of each annotation differs significantly but the creation is nearly identical. So, the implementation would also be nearly identical with the only difference being that the metric event is fired once an info annotation is created rather than once a blocking annotation is created.

- **Finding out how often mockup images are updated/changed**
  Collecting data regarding how often a mockup is updated or changed should be pretty simple but making sense of it may be a bit more difficult. We can't simply look at one data point and make a conclusion from that, we need to look at a mockup's telemetry data points over time, which is outlined later in this paper. For collecting the metric, we simply fire an event once any user initially uploads a mockup and then another every time that mockups image is changed/reuploaded.

- **Figuring out the designer to developer ratio on each team**
  For data collection related to the developer to designer ratio we can simply look at the team statistics and see how many developers are on the team and how many designers are on the team once the team is created. On team creation we could easily send an event with some simple team information including the number of developers as well as the number designers and project managers invited to the team. To follow that, it would also be good to fire a similar event once the developer, designer or project manager count changes that way we can keep up to date metrics rather than only having the number from when the team was originally created.

Detecting Changes and Historical Data

How will you detect changes in user behavior over time? (For example, beginners do [this], but as they gain more experience, they start doing [that] instead.)

Tracking historical changes in this application would be done by sending an age of account or time of signup parameter along with the metrics calls as an implied experience indicator. There is no specific behavior that could indicate how experienced a user is with the app other than the age of their account, with the exception of similar measures such as the user's total number of sessions or total duration of sessions. To measure historical trends, we will pull up the metric we're interested in, such as number of annotations created, then sort the metrics by the experience indicator to see if we can observe any correlation. The application is pretty simple, so I wouldn't expect that beginners would be doing something much differently than experienced users and I think if realizing some functionality of the app requires more expert knowledge of the app then maybe we should revisit the UX to simplify that feature. Retrieving these metrics and looking at the historical data can help to reveal those problems though, allowing us to go and simply the UX/UI for all users.

Telemetry Providers

Of the commercial telemetry providers that you can find, which do you think would be best for your application and why? Or if you insist on rolling your own framework [suggestion: don't], explain why, and where you would start doing that?

I have some experience implementing telemetry tools with web applications so I think I would go with a setup that I've found the be pretty flexible. I would feed all of the telemetry data points to the Segment Customer Data Infrastructure web app. This tool essentially collects the data points/events from the client or server and relays it to other metrics services. I would then use Mixpanel and integrate that tool with Segment so that it ingests all of the events routed through Segment. With Mixpanel and some diligent configuration in the Mixpanel app, we should be able to easily make sense of all the telemetry metrics coming from the browser. It may be important to note that many of these events can be sent on the server side or via the client's browser. I'm pretty sure I would just implement all of the telemetry metrics to be fired from the client's browser, I've found that method to be just as effective while also being easier to implement. Doing things this way should make implementation easier and should make the apps telemetry strategy far more scalable.

Example

Produce an imaginary data set from your measurements. Apply it to your analytical questions. Explain why you think you were right about the ones you got right, and wrong about the ones you got wrong. Explain how you would modify your program in response.

- **Who is uploading and creating mockups?**
    - *Query Result: {teamUploadsByUserType: { designer: 28, developer:9, projectManager: 20 }};*
    - The data represented above shows that designers and project managers are uploading mockups the most but what's interesting is that it shows that developers are also uploading mockups much more than initially anticipated. We know that developers are not designers, so it's unlikely that they would be creating mockups of their own. A more likely scenario is that their team's

workflow better facilities the developer uploading the mockup, maybe because of how many teams conduct their planning and scrubbing meetings. What these results show us is that we don't entirely understand the varied project workflows used across teams in the industry and that we need to go and talk to more users to figure out exactly how they get their tasks hashed out in planning. I think a good modification for the app for this new discovery might be some sort of mode that helps facilitate uploading of mockups while in meetings like planning meetings or scrubbing meetings, maybe some sort of "planning" mode.

- **Who is creating blocking annotations for mockups?**
    - *Query Result: {teamBlockingAnnotationsByUserType: { designer: 1, developer: 30, projectManger: 5}};*
    - This data helps to affirm our original beliefs about blocking annotations. Typically, a developer is the person getting blocked by inadequate requirements so seeing developers creating blocking annotations almost exclusively is no surprise. The fact that designers and project managers created any blocking annotations at all though is interesting and worth keeping an eye on. It's possible that a designer or project manager could have made a blocking annotation on a developer's behalf for convenience but if we see those numbers increase or stay around the same level, I think it might be worth asking if these project managers and designers are repurposing blocking annotations for something other than the developers' use case.

- **Who is creating info annotations?**
    - *Query Result: {teamInfoAnnotationsByUserType: { designer: 14, developer: 12, projectManager: 8}};*
    - This is interesting, info annotation creation seems to be split nearly equally across all user types. I originally expected them to be created mostly by designers but it looks like developers and project managers like to tag info on the mockups themselves too. I think this is a good candidate for a historical comparison, it could be useful to know of most of these info annotations are created at the same time right after the mockup is uploaded or if they're created randomly over the life of the mockup. Regardless, I think seeing this metric tells us that info annotations need to be focused on more since every user type sees value in it. Maybe a more prominent info annotation CTA could provide a more enjoyable UX for users wanting to create info annotations.

- **How often are mockup images updated/changed?**
    - *Query Result: {mockupAvgs: avgImgUploadCount: 2.7, ….}*
    - This metric is somewhat concerning. I had anticipated the number would be somewhere between 1 and 1.5 but what this number tells us is that most mockups see more than two mockup/image reuploads and the app wasn't originally designed for this purpose. To help with this I think it's important to refactor some of the UX in such a way that it allows users to more easily re-

upload mockup images to update their mockups. Integrations with box, dropbox, or cloud mockup tools could also be useful in removing the friction behind re-uploading which is important now that we know it's a much more common task than we thought it would be.

- **What is the designer to developer ratio on each team?**
    - *Query Result: {teamAvgs: composition: { developers: 6.3, designers: 1.4, projectManagers: .3, totalTeammateCount: 9}} …};*
    - These results are what was expected but are still a bit revealing of some information we didn't know before. There are quite a few more developers than designers with roughly 6 developers on each team and one or two designers on each team. What was unexpected though is that most teams either don't have project managers on their team in the app or the don't have project managers at all. This could be an indicator that any future development toward features geared toward a more project manager user type could be useless and that we may even just be able to remove that selection from the prompt where a user selects their role on the team since it seems like it's rarely ever used.