Team PSD Manual

Team PSD

2020-08-03

Table of Contents

# Team PSD Manual

Welcome to Team PSD’s manual for all things related to resources and guides! Click within chapters or search throughout the manual for key terms that you are curious about.

# TeamPSD Guiding Principles

## Open-Source, Transparent, Reproducible

We value an open-source, transparent & reproducible workflow.

* Most of our work is public-facing with the exception of any items that include Protected Health Information (PHI) or Personally Identifiable Information (PII). All of our public materials are free to download and use. We want our insights and resources to benefit the larger community.
* We use R, a free open-source coding language and a specific file naming convention to ensure all of our matierals are machine & human readable (all lower case, no spaces, with dates as yyyy\_mm\_dd i.e. teampsd\_guiding\_principles\_2020\_01\_01).
* Use email for any private discussions. Host any private files in password-protected locations or in folders behind the VA firewall. When in doubt, ask an HQ member or err on the side of caution.
* Make sure your work and accompanying documentation allows other team members or scientists in the field to reproduce and understand your work without further questions.

## High Visibility Work

Our work has high visibility.

* Keep in mind we work under the federal government of the United States for the Veterans Health Administration (VHA), the largest integrated healthcare system in the country. Everything we produce is a reflection of the VHA.
* We work with a wide range of nationally-distributed partners both internal and external to VA, including very important and high-level stakeholders. Double check the role and responsibility of people you are communicating with before you do.
* With these partners, we work in a participatory learning manner and iterate based on feedback from the field to ensure our work is responsive to ongoing changes.

## Team Time

Any time saved is team time.

* Ask questions early and often to prevent escalating issues down the road. Refer to existing resources (cheatsheets, checklists, etc.) as well for clarification.
* Double check all work before handing it off to the next team member to reduce rework.
* Think through dependencies across the team and partners and prioritize work based on the most recent information you have.
* Manage workflow asynchronously (via GitHub) and only schedule meetings when absolutely necessary.

## Communication

Use effective communication (across all types of communication including emails, GitHub, Lucid, etc.).

* Assume everyone you’re communicating with is smarter than you and cares more than you and is busier than you.
* Use clear and concise language, with formatting (bold, underline, bullets, etc) to emphasize the main decisions or issues making sure to include the “Who” “What” and “When”.
* Always include the full context and details necessary to make an informed decision (and make sure you are up to speed on the context and details before responding).
* Use complete sentences as much as possible and write in the active voice for clarity
* Often the team looks back to previous records of meetings, GitHub discussions, etc. to remember and track decisions that were already made or if they missed the meetings where an issue may have been discussed. As such, we always need to keep the most accurate record possible to reduce rework and provide clarity.
* Use emojis or humor (as appropriate) to help maintain a positive and collegial vibe. Turn on your webcam as much as possible for the face-to-face interaction.

## Active Listening

We use active listening skills to ensure understanding and accurate tracking.

* We work daily with team members and partners that are experts in their respective fields, and it is easy to lose track of a complex discussion.
* We’ve found that reflecting-back requests and decisions in your own words has been the best method to reduce miscommunication and to keep track of all of our decisions or issues accurately. If you ever need help while scribing, always “tap” someone else on the team for assistance.
* Ask for clarification and slow down if necessary.

### Active listening is a stance.

Taking the stance that misunderstanding is the norm and using skills appropriate to that reality.

#### 1. General skills: Reflecting content for efficiency and interpersonal rapport (i.e., avoiding rework and frustration)

* “When in doubt, check it out” - It’s the listener’s job to help the speaker understand what they are getting or what they missed.
* Let the speaker know when you’re falling behind. Stop them before the conversation exceeds your understanding.
* Use the same precise language/vocabulary, esp. Team PSD or *MTL* terms with specific meanings.

#### 2. “Chunking” components of what someone said (e.g., listen for the “and” when someone is speaking)

* Listen for the main points.
* This is a key skill to keep up with complex ideas in the moment (You don’t want to get bogged down on point 1 and miss point 2).

#### 3. Reflection of feeling - listen for the feeling words and reflect them exactly.

* Using the exact feeling word you heard is the safest way to ensure the person feels heard.
* Don’t reflect “frustration” when someone is expressing “disappointment” or “stress.”
* You can use these for non-verbals too.
* If you hear a feeling, it’s best to address it in your reflection.

#### 4. Synthesize two person’s ideas - or just two ideas (or more)

* e.g. “on the one hand, I hear Anthony saying…” and “on the other hand, I hear Stacey saying…”
* Synthesis brings the ideas together. It is not just active listening both ideas as separate ideas.

#### 5. Summarize - used when it’s time to wrap up, and move on, and a lot of ideas have been said.

* This is not the time to bring up new points. If you do have something to add, say it first.
* It’s best to end with a summary of the key points of consensus and the key take aways of what to do next.

#### 6. After you reflect, add to what the person is saying (a key to improv is “yes, and…” - don’t reject ideas, just try to understand first and then build on the ideas).

* Forward momentum comes from making sure you understand the person and add your own contribution.
* Listening is about being present to understand; you do not have to agree with what a person is saying.
* Before you even get to agreement/disagreement, make sure you have a shared understanding first.
* KEY IDEA: Really attend to make sure you understand. Do NOT think about your response when you should be listening.

#### Things to look out for when listening -

##### Know thyself:

You are the only one who knows whether it’s time to multi-task, focus, scribe/document as a form of listening, or whether typing would be a distraction.

##### Repetition cycle:

If the speaker you’re listening to keeps repeating something, your first move should be to assume that you’re missing something and ask what it is.

##### Your own feelings:

If you’re starting to feel frustrated, it’s a key sign that you should use active listening to get back on the same page.

# Team PSD 2.0 Monthly Process

As Team PSD continues to grow, our workflow alongside team values in order to better support the needs of our team, just like how we developed [*Modeling to Learn*](https://mtl.how) to scale nationally in the Veterans Health Administration.

The **participatory learning** principles we scaled in *Modeling to Learn* are:

* equitable access to resources
* mutual learning
* shared decision-making

**Team PSD 2.0** is about becoming more scalable due to continuous integration, deployment and documention.

* We aim for a completely free, accessible, transparent and reproducible workflow (aka **“open science”**).
* **Our mission** to improve how healthcare quality improvement decisions are made enlists diverse team members and partners.
* We want to **empower** effective contributions from all potential stakeholders.

The Team PSD 2.0 Monthly Process is based on the design thinking principles of **user experience**, which attemps to account for a **person’s needs**, **pain points**, **goals**, and **emotional experience** when using a **product/service** and/or going through a **process** and **system**.

**Team PSD 2.0 Process for monthly sprints/epics:**

**Week 1:** Gather user-centered hypotheses

**Week 2:** Clarify user assumptions w/ Minimum Viable Product (MVP) test

**Week 3:** Review results of user persona testing of your MVP Prototype (with concurrent video and retrospective verbal)

**Week 4:** Review user persona artifacts and second story perspectives mindfully and empathically to discover new understandings you might have missed or still need to learn

Our monthly **process for each monthly sprint/epic** is based on the **process for design thinking**:

## Week 1: Gather User Hypotheses

In week 1, we want to gather the user centered hypotheses of the user/user group that we want to help and design for either by meeting with them and/or referring to their user personas.

Hypotheses should consist of the user’s needs, pain points, and consider the background and context of the user.

*Week 1 follows the* ***first phase of the process for design thinking*** *which involves the ability to* ***empathize*** *with the user’s current state of experience.*

1. **Empathize** - To gain an empathic understanding of the need or problem you are trying to solve in terms of people, **processes** that set aside individual **assumptions** to get insight into the need.

* **What?** Details of what to do (what happened).
  + This is when we **analyze** observe and document details that relate to users.
* **How?** How the person does it (effort, etc.)
* **Why?** Motivations/purpose (test this out…)

**Exercises for assumptions** (remember: everything is a perspective):

1. **List assumptions**
2. **Ask**: How could this not be true?
3. **Ask**: What if we could do this twice as well in half the time?

**Example of a User Persona:**

## Week 2: Clarify User Assumptions w/ MVP test

In week 2, we want to narrow down the week 1 user centered hypotheses by clarifying assumptions based on those hypotheses.

Assumptions the designer has based on week 1 hypotheses should clarified to **define** only the most crucial and necessary needs and pain points that will be addressed **before prototyping a minimum viable product (MVP)**. This will prevent creating/designing for needs that **do not solve 80% of the user’s pain points with only 20% effort** being used in the development of the MVP and test and **quickly rule out MVPs** that do not respond to needs of the users in a **fast and cheap** manner.

*Week 2 follows the* ***2nd phase of the process for design thinking*** *which involves the ability to* ***define*** *with the user’s specific set of problems that needs to be immediately addressed.*

1. **Define the problem** - This is when we **synthesize** observations into holistic **point of view** (POV).

**Goal is to make linkages, so that we define the right problem to address.**

[Example POV: Busy, nationally distributed, cross-functional team of scientists has many partners, is about to hire more people, leads a national participatory system dynamics simulation learning program in, and is gaining increasing national and international interest in partnership HMW…]

2a. **A problem statement: Focuses on**

* **Specific peopleʼs needs** (not the technology or specs),
* **Value and insights for the project** (not the technical requirements), yet it is…
* …narrow enough to be **managed within our constraints**

**Exercises for definitions with how-why/why-how laddering** (a variant of **5 whys** principle):

1. Asking **“Why?”** Explores to understand **root causes** (abstract, more common across people)
2. …then **“How might we?”** to get to a **specific** problem/challenge we can solve (concrete)

**Part I - The HMW Brainstorm…HMW**

* use the efficiency of GitHub [good]
* remove the [bad] steep learning curve
* integrate with existing GH norms the best part [explore the opposite]
* remove reliance on training [question the assumption]
* go after adjectives [make it easy, instead of hard]
* use code instead of by hand [leverage unexpected resource]
* enable self-directed learning like “Googling” [analogy from need/context]
* attract help from Forio, MITRE, VA to solve this [shift POV against the challenge]
* get it up and running now [shift a status quo]
* divvy up chunks for each workgroup [break up POV]

**Part II**

1. “**Why** do we need to integrate our manuals, workflows and processes on GitHub to achieve Team PSD values? (if start with Why, phrase as a need and make it meaningful.”

Because…

1. “**How** do we use manuals, workflows and process on GitHub now?”

* [describe]
* **“Why?”**
* **“Why?”**

1. What what was most **surprising**?
2. What would we have **missed** if we hadnʼt asked why?

## Week 3: Review User Results of MVP

In week 3, we want to share our MVPs with the user to collect user feedback.

To collect feedback, users will be participating in 2 Think Aloud protocols: **Concurrent Screencast Video (no audio) and Retrospective Verbal.**

1. **A Concurrent Screencast Video Think Aloud** requires users to screencast themselves going through the MVP prototype, but without audio so they can focus soley on using the MVP.
2. **A Retrospective Verbal Think Aloud** occurs after the Concurrent Screencase Video Think Aloud in which users think about how they felt about the MVP, what went wrong, what went right, what was missing, etc and write it up afterwards.

These 2 Think Aloud protocols allows us to leverage both the pros and cons of a Concurrent and Retrospective Think Aloud in which:

* the Concurrent provides an undisturbed recording of the user going through the MVP.
* the Retrospective allows the users to verbalize their thoughts, emotions, and feelings about the MVP.

*Week 3 follows the* ***3rd and 4th phases of the process for design thinking*** *which involves the ability to* ***ideate*** *MVPs based on the needs and assumptions of the users and get user feedback on a* ***prototype*** *MVP that will solve 80% of the user’s pain points and needs with only 20% effort.*

1. **Ideate** - Expand the problem space by identifying and testing out elements that would circumvent problems.

* time-limit
* quantity over quality
* no distractions
* no bad ideas

1. **Prototype** - inexpensive, scaled down version with the **key features** to investigate the problem and solution. **Goal is to identify the best possible solution**.

* Solutions are investigated, accepted, improved, re-examined, rejected based on user-experiences.
* Should give much better sense sense of constraints and how users would behave, think, and feels when interaction with it.

### Create Screencast Video

#### Instructions

Make **user-persona Concurrent Think Aloud videos** testing Team PSD prototypes (aka MVPs: ‘minimum viable products’).

#### Context - *Why* this is critical for Team PSD 2.0 to scale?

* To scale Team PSD processes, we evaluate the **usability** and **user experience** of Team PSD prototypes.
  + **Passive Screen Video Observation**
    - Usability (quantitative MVP tests)
    - This test is the **concurrent** think aloud protocol to detect problems with MVP assumptions.
    - Prototype developers gain re-susable insights into the users (through the library).
    - Users don’t have to think about producing these scalable artifacts (just upload).
    - These files are named with this convention “date\_skill\_training\_user” e.g., “2020\_05\_13\_git\_ch1\_busy\_pi”
  + **Verbalization**
    - User-experience (qualitative MVP tests)
    - This test is the **retrospective** think aloud protocol to detect problems with MVP assumptions.
  + **Learning task performance.**
    - We need **distributed Team PSD users** to *learn at scale.*
    - All prototypes are teaching our learners a skill they need to contribute to the team.
    - As we upskill, we continuously integrate prototypes, which scales learning *and* improvement across Team PSD.

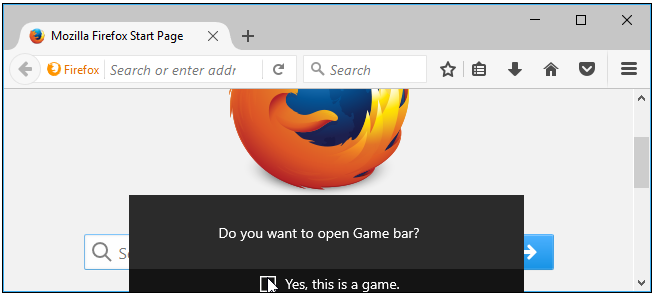
#### Record Screen (No Audio)

#### PC User (VA or Non-VA)

#### Non-VA, Windows 10 users only: Record screen using Game DVR

The Game DVR feature can capture any application’s window.

1. **Press Windows + G** in any application on Windows 10.
2. **Look for the Game Bar** that will appear.
3. **Select “Yes, this is a game”** even if the application isn’t a game.



**Reminder:** *Protect your privacy* and only include information that is relevant for Team PSD prototyping.

#### VA or Non-VA: Record screen using PowerPoint



Powerpoint can has a built-in ability to screencast your desktop and input the video onto a slide in blank powerpoint presentation.

You can save the video file separately afterwards.

Follow the guide [**here**](https://support.microsoft.com/en-us/office/record-your-screen-in-powerpoint-0b4c3f65-534c-4cf1-9c59-402b6e9d79d0#OfficeVersion=Newer_versions).

1. Follow **steps 1-6** under the header: **Record from your computer screen and embed the recording in PowerPoint.**
2. **VA Machine User**: Your PowerPoint Version on your VA machine will be able to screencast with the instructions in the guide above.
3. **Non-VA Machine User**: Be sure to check your PowerPoint versions as this only works in *PowerPoint for Microsoft 365, PowerPoint 2019, PowerPoint 2016, PowerPoint 2013.*

**Reminder:** *Protect your privacy* and only include information that is relevant for Team PSD prototyping.

#### Mac User (Usually Non-VA)

#### Record any window on your Mac.

1. **Press Shift-Command-5**. The *Onscreen Controls* will appear.
2. Once the *Onscreen Controls* appear, click on one of the buttons with the record circle in the bottom right (4th icon from the left - **Record Entire Screen** or 5th icon from the left - **Record Selected Portion**)
3. **Click the down arrow** under *Options*, located next to the record button.
4. Under *Microphone*, **select none** to turn off audio recording.
5. Under *Options* in the *Onscreen Controls,* **adjust the recording window,** setup a *Timer*, and where to *Save To* your recording. 
6. In the *Menu Bar*, **click the Stop Recording button** or **press Command-Control-Esc (Escape)** to stop recording.



**Reminder:** *Protect your privacy* and only include information that is relevant for Team PSD prototyping.

**Reference:** van den Haak, Maaike, De Jon, Menno, Schellens, & Peter Jan. (2003). Retrospective vs. concurrent think-aloud protocols: Testing the usability of an online library catalogue. *Behavior & Information Technology, 22.* <https://doi.org/10.1080/0044929031000>

### Upload Screencast Video

#### Instructions

**Post user persona Concurrent Think Aloud videos** testing Team PSD prototypes (aka MVPs: ‘minimum viable products’) to **YouTube.**

#### 1. Create a Google Account

If you don’t already have a gmail/Google account, **select this link** to create a Google account.

[https://accounts.google.com/SignUpWithoutGmail](https://accounts.google.com/signup/v2/webcreateaccount?flowName=GlifWebSignIn&flowEntry=SignUp&nogm=true)

* A Gmail is not required to sign up.
* This email is only used to login and will not be public.
* Uploaded user persona videos will be posted under the Brand Account **LZPhD**.

#### 2. Share Email with Brand Account Manager

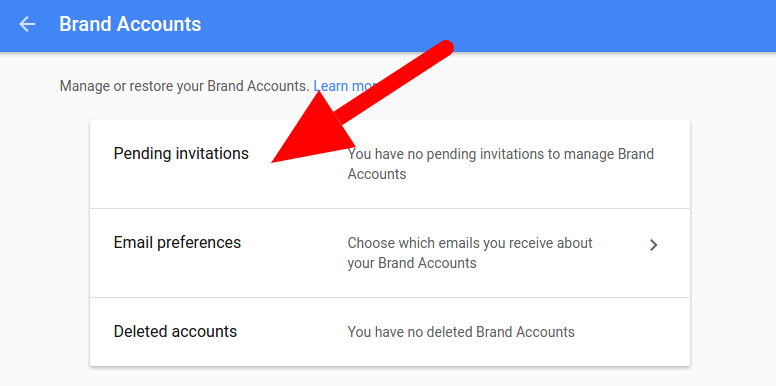
* **Share the email** associated with your Google account with a **brand account manager** to process an invitation.
* A **brand account manager** is any manager that already has access to the YouTube channel.
* **Communicate** through Microsoft Teams.

#### 3. Manage YouTube Invitation

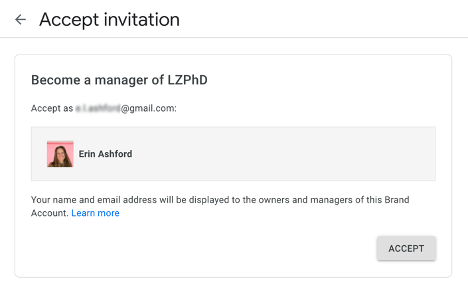
* **Login** to your Google account
* **Select** this link

<https://myaccount.google.com/brandaccounts>

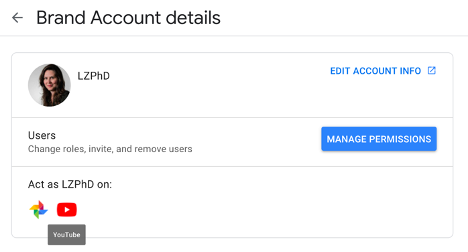
* **Click** Pending invitations.



* **Select** LZPhD.
* **Accept invitation**.



* **Scroll down**: “Act as LZPhD on”:
* **Select** the YouTube red icon.



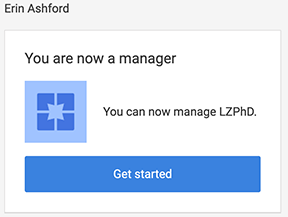
* **Navigate** to the inbox associated with your Google account.
* You will see a confirmation in your inbox that reads, “You are now a manager of LZPhD”.



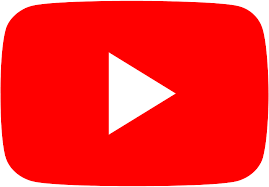
* **Select** the new email.
* **You are now a manager** Congratulations!

#### 4. Upload User Persona Video

* **Select** Get started.



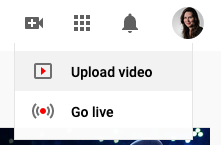
* **Select** the YouTube red icon.



* Your YouTube dashboard will open up.
* **Navigate to the top right** and **select** your circular profile pic.



* From the drop-down menu, **select switch accounts**.
* **Select** LZPhD.
* On the top right, **Select** the video camera + icon (Create).
* **Select** Upload video.



* Find the user persona video file you’d like to upload.

#### 5. Convention for Video Details During Upload

* Use this naming convention **“date\_skill\_training\_user”** ex: **“2020\_05\_13\_git\_ch1\_busy\_pi”**
* A description is not necessary at this time.
* No playlist.

**Audience**

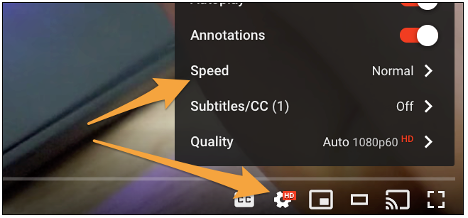
* **Select, Yes, it is made for kids**. This will remove any advertisements.
* **Click Next** advance to next screen
* **Click Next**, no video elements at this time.

**Visibility**

* **Select Unlisted** only people with the link can see it.
* **Click Save** and your video will continue processing.
* You can safely **logout of YouTube** when your **screencast is fully uploaded.**

#### 6. Speed Up or Slow Down YouTube Viewing

* While viewing your user persona video in YouTube, **click** the gear icon located on the bottom right.
* **Select** the “Speed” option.



* **Choose** a range of speeds.

## Week 4: Review Second Story Perspectives

In week 4, your **prototype and MVPs should be able to be fully incorporated into Team PSD’s workflow** as we **do NOT** want to spend time developing solutions that do not solve 80% of our team’s needs and utilizing only 20% effort in order to fail fast and fail cheap by quickly ruling out assumptions and solutions that do not work.

*Week 4 follows the* ***last phase of the process for design thinking*** *which involves the ability to* ***review tests*** *of your MVP and* ***reflect*** *on those tests that can help you even better mindfully and empathize with users, ultimately to further improve your skills for month’s design process.*

1. **Test** - Review tests your prototype with the user personas in question & look for 2nd story perspectives.

* Does the prototype address the personas needs and pain points?
* What could we have missed or ignored from the user?
* What worked well in the month and what did not?

### Reflect Second Stories

**Use these set of question to help think through Second Story Perspective Questions to reflect on the design process from the month:**

1. Do you see any places where moving forward independently could have been more aligned with the team or integrated better when you were done?
2. Could a dependency or root assumption have been caught any sooner for the benefit of the team?
3. Have you discovered a new empathic understanding of the situation from another users’ POV (point of view)?

# MS Teams

TeamPSD and the VA are both using [Microsoft Teams](https://www.microsoft.com/en-us/microsoft-365/microsoft-teams/download-app) as their main communication platform.

TeamPSD is currently using the **Free Version of Microsoft Teams** so the entire team can get used to it as the VA begins to roll it out system-wide between mid-July to early August.

## Quick Start Guide

To review the basics of Microsoft Teams, check out their [Quick Start Guide](https://www.chorus.co/media/2989/quickstart-guide-to-teams-chorus.pdf).

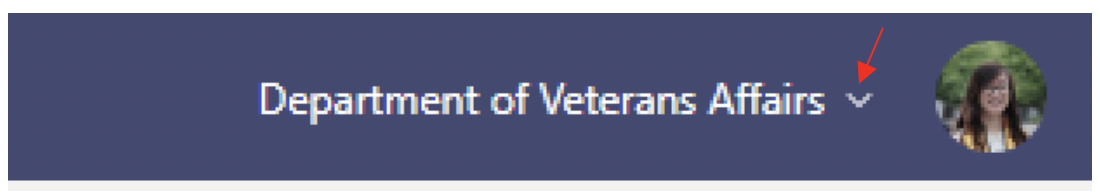
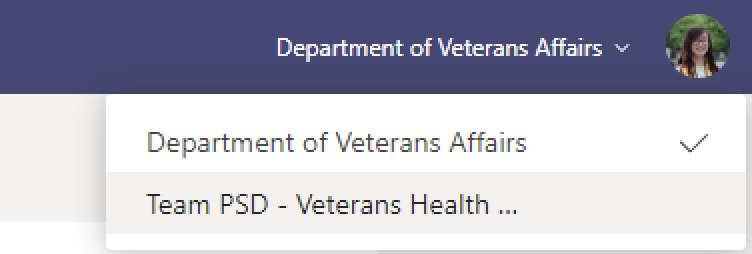
This guide will help you learn the basic user interface functions of Microsoft Teams (**Free version or VA**) such as:

* Signing In
* Manage your profile settings
* Choose/browse channels
* Start a meeting or call
* Start a new conversation thread
* Reply within a message/conversation thread
* @ Mention someone
* Start a 1:1 or small group chat
* Add/Share/Work with Files
* and more!

## Additional Tips/Functions

### Switch Organizations

To switch out of an organization:

1. Click on the **organization you are currently in**, located next your profile picture & name. 
2. To switch out of the organization, click on the **organization you want to switch into**. 

### Meet in a Channel

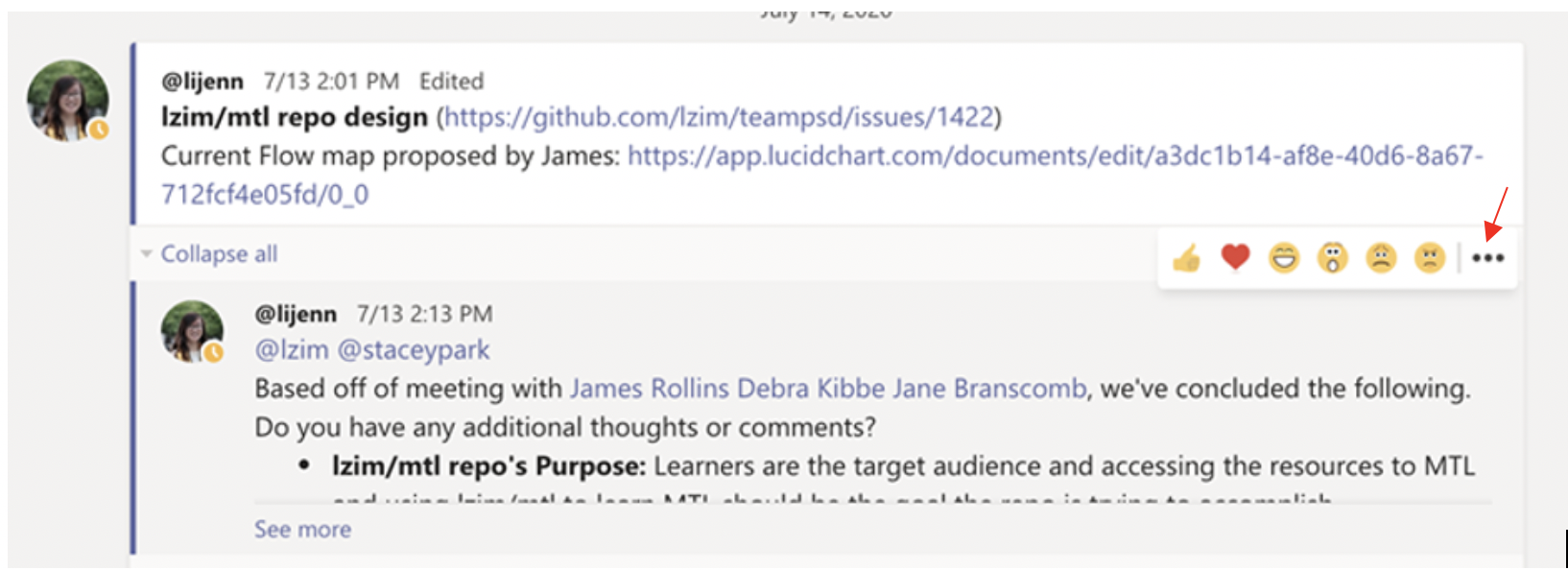
To know where a meeting call is being held in a channel:

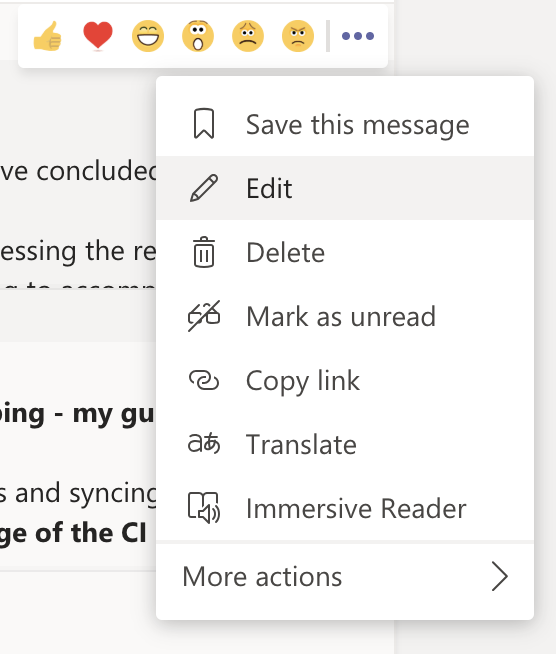
1. Look to see the **purple video icon** next to a **channel’s name**.

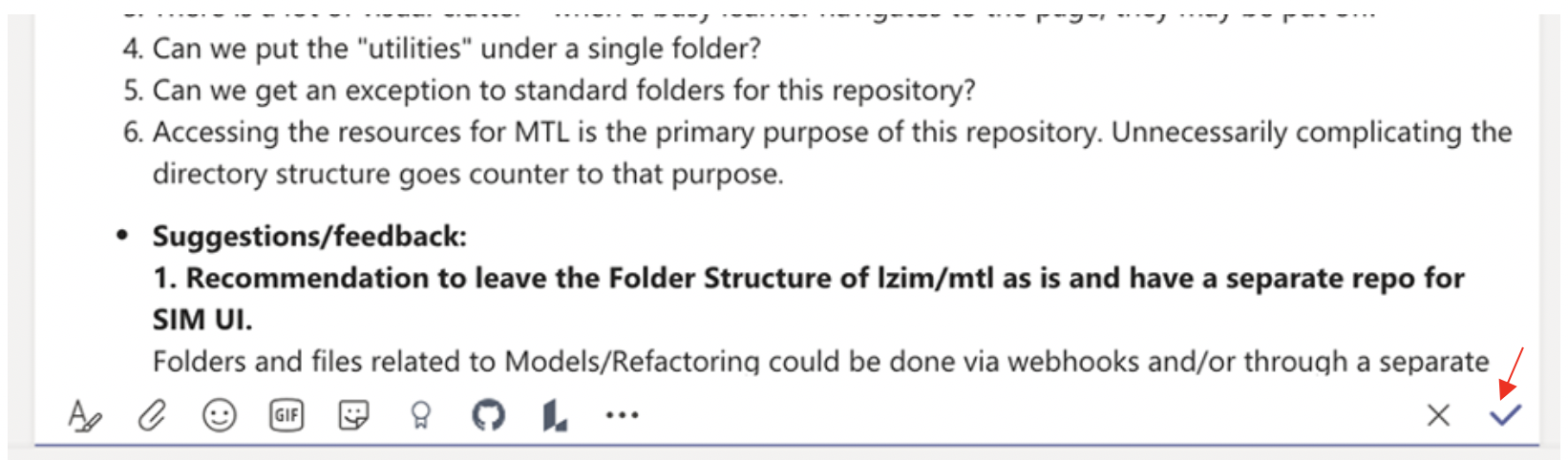


### Edit Message/Reply

To edit a message:

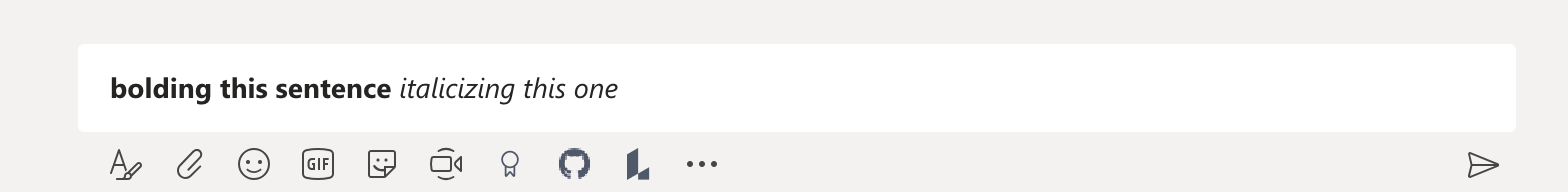
1. Drag your cursor over to an existing comment and **click on the 3 dots** next to the 6 “react” emojis. 
2. Select the **pencil icon Edit button**.



1. After finishing your edit, select the **check mark button** to publish or the **“x button”** to exit out of the draft. 

### Use Markdown

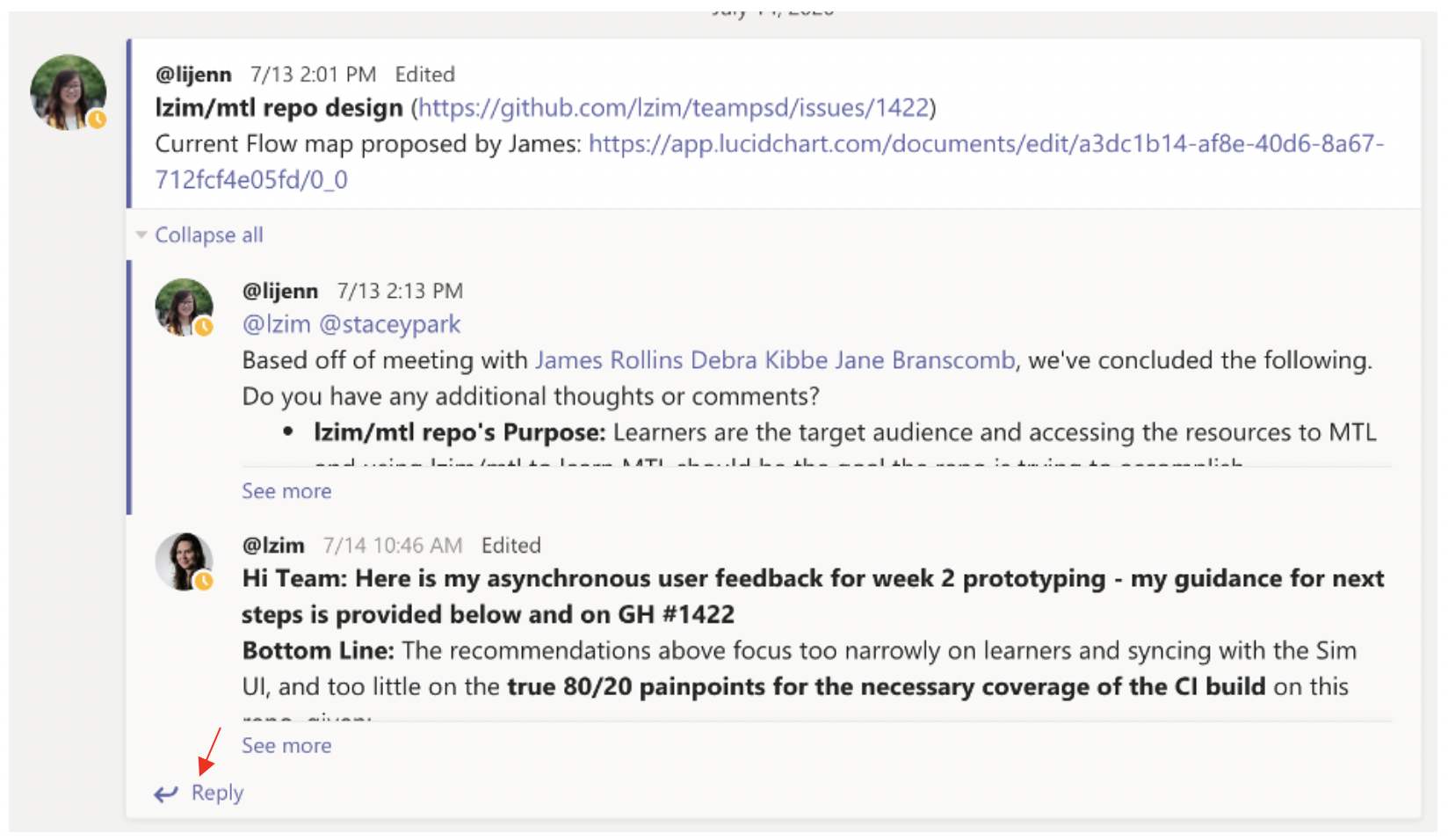
Teams is Markdown friendly!

1. Try using **basic markdown** such as bolding or italicizing text in a comment/reply. 

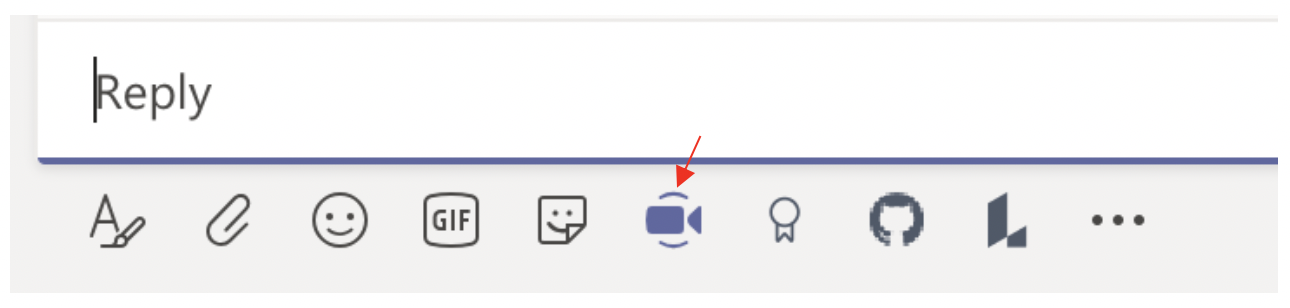
### Start a Call in a Thread

To start a meeting call within the same conversation thread to keep context:

1. In the conversation thread, click on the **reply button**.



1. Click on the **video icon button** to begin a call.



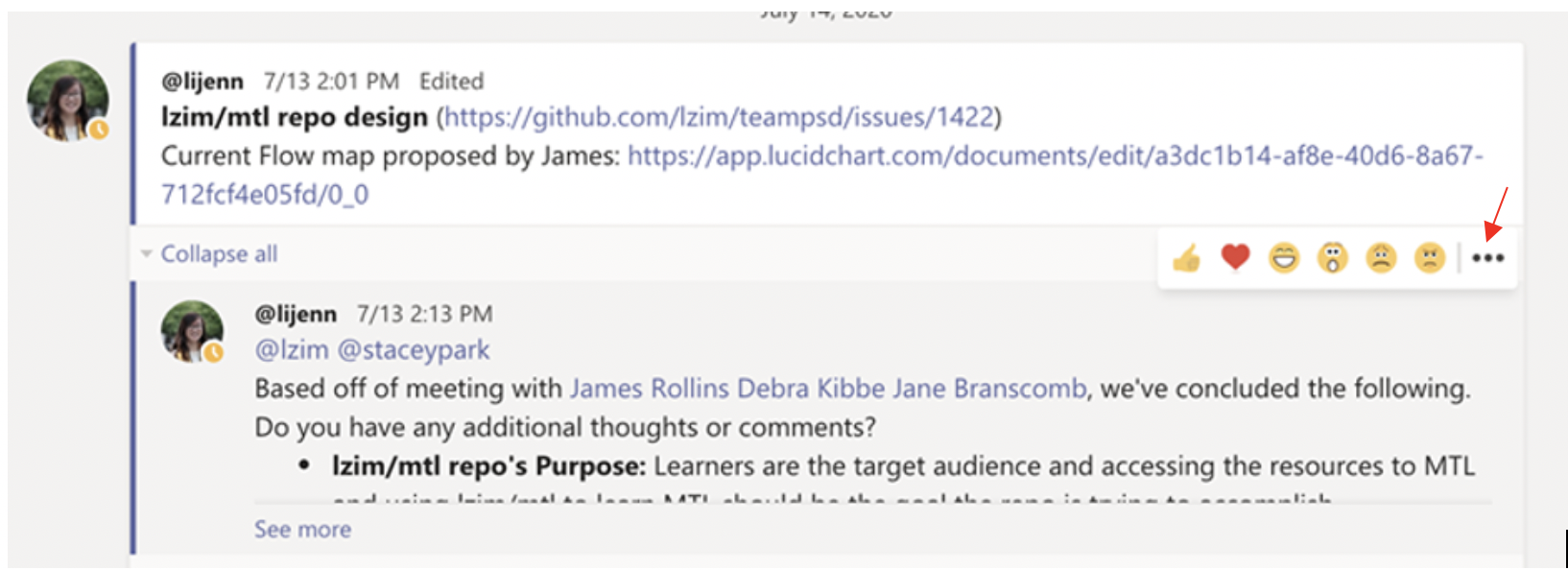
### “React” to a Comment

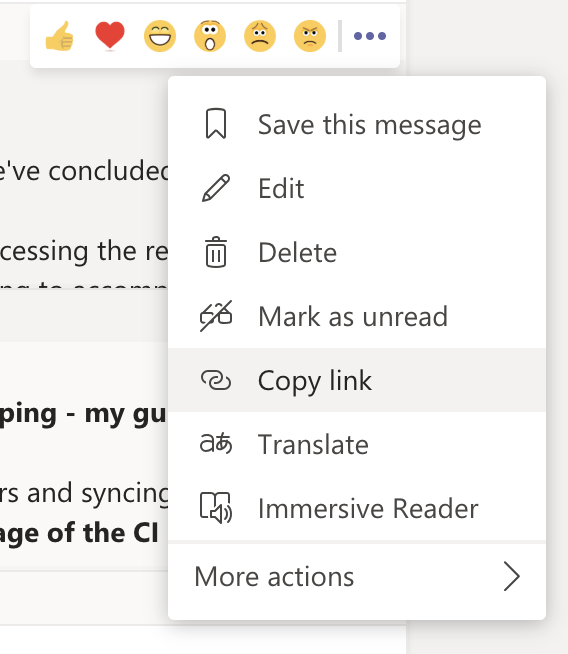
To “React” to a comment or reply to acknowledge it:

1. Drag your cursor over to an existing comment and **select 1 of the 6 emojis** to “react” to it. 

### Link a Comment/Discussion/File

To link to a specific thread or comment or file:

1. Drag your cursor over to an existing comment and **click on the 3 dots** next to the 6 “react” emojis. 
2. Click on **hyperlink icon copy link button** to copy the link to the location of that thread/comment/file.



# GitHub

TeamPSD’s main platform for project tracking is GitHub + the extension of [ZenHub](https://chrome.google.com/webstore/detail/zenhub-for-github/ogcgkffhplmphkaahpmffcafajaocjbd?hl=en-US) for project management.

We :heart: GitHub for sharing the same commitment and values in having a free and open science workflow.

## Standard Operating Procedure (SOP)

Before beginning to contribute to TeamPSD’s workflow, ensure that you read through our [Standard Operating Procedures (SOP)](https://github.com/lzim/teampsd/wiki/TeamPSD-Policy-&-Procedure-SOP) to understand our rules and policies and orient to our team using GitHub.

## GitHub Repositories

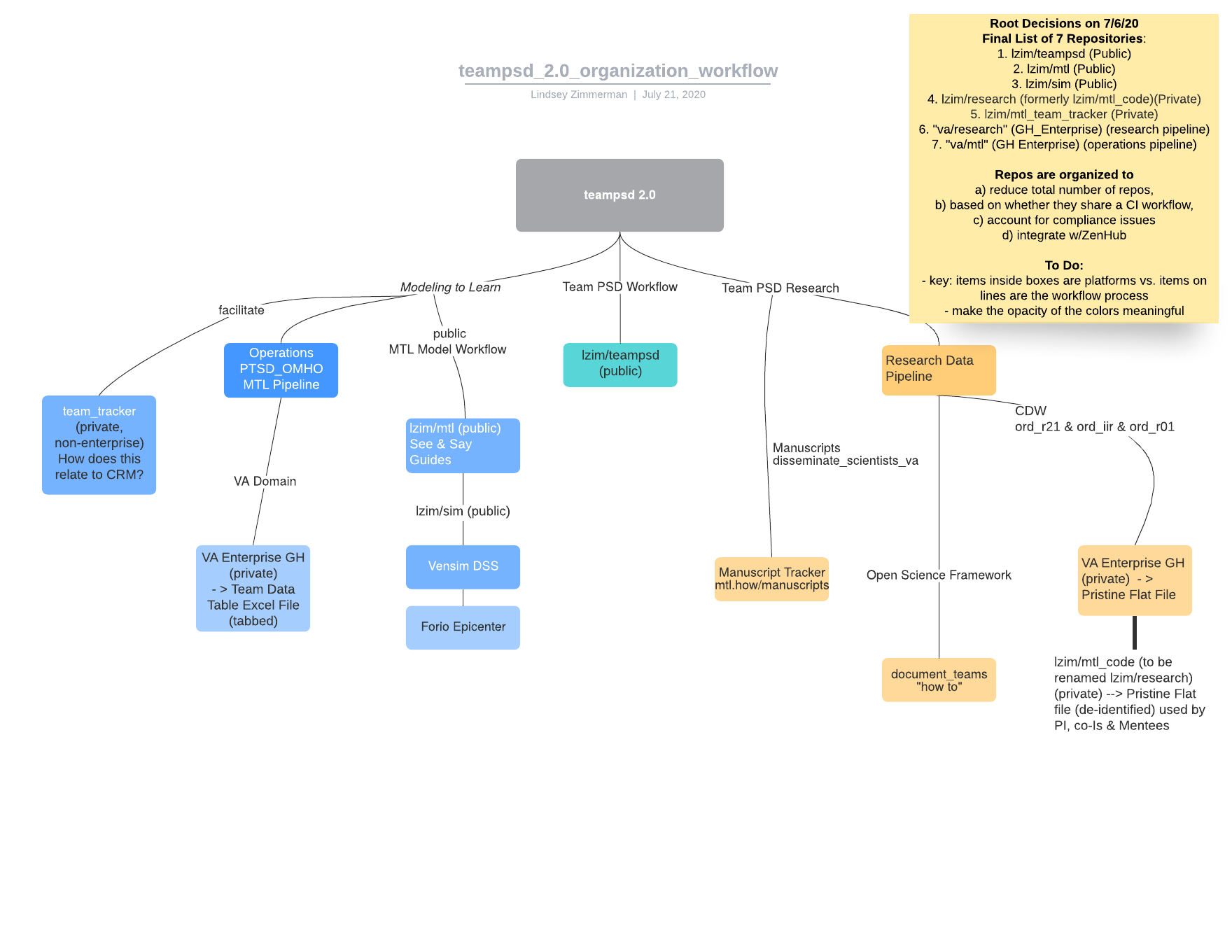
### Non-VA Repositories

1. [lzim/teampsd](https://mtl.how/teampsd) (Public) This repository contains TeamPSD resources and track the work that the team uses internally (i.e. Bookdown, document\_team, depend\_products, etc.)
2. [lzim/mtl](https://mtl.how/) (Public) This repository supports *Modeling to Learn* and contain the content in describe\_learners, detail facilitators, and SIM/Model Code.
3. lzim/sim (Public) This repository supports the *Modeling to Learn* Simulation User Interface plaform, Forio Epicenter, & and model files from Vensim DSS.
4. [lzim/research](https://github.com/lzim/mtl_code) (Private) This repository holds the analysis for the (R01, IIR, R21) and disseminate science.
5. [lzim/mtl\_team\_tracker](https://mtl.how/mtl_team_tracker) (Private) This repository is used by VA Clinicians and *Modeling to Learn* Co-facilitators to track team progression.

### VA Enterprise Repositories

1. “va/mtl” (operations pipeline) This repository stores the operations code used to support *Modeling to Learn* teams.
2. “va/research” research pipeline) This repository holds the backend code that needs to stay internal for our grants.

### Workflow



teampsd\_2 0\_organization\_workflow

## Basics of GitHub

To go through a quick overview on repositories, branches, commits, and opening/merging pull requests, click [here](https://guides.github.com/activities/hello-world/).

For a glossary of GitHub terms, click [here](https://docs.github.com/en/github/getting-started-with-github/github-glossary).

## Style

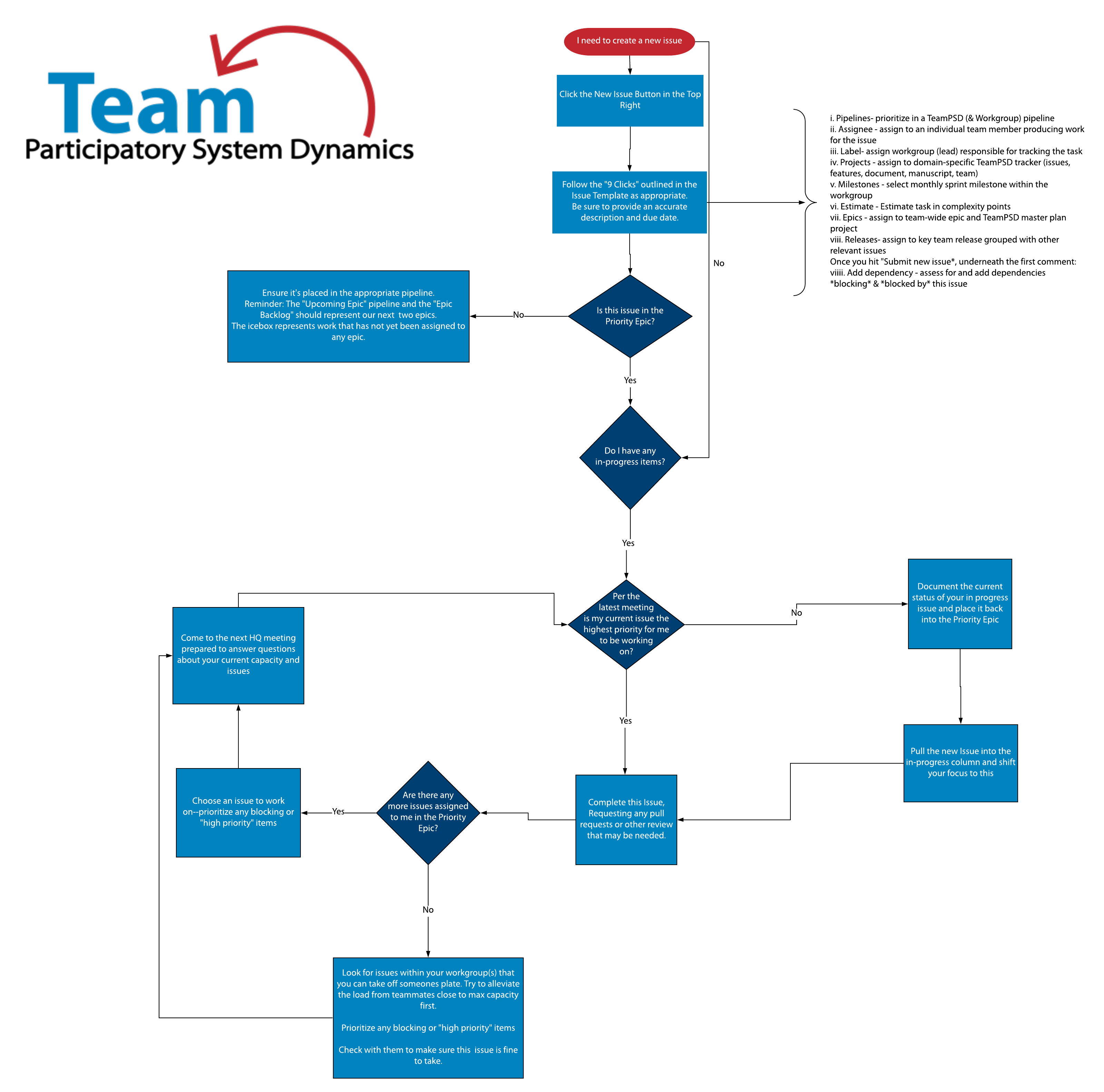
We use R Markdown format for syntax and style which you can refer to here (ADD HYPERLINK). Use a human and computer readable format to name files, meaning all lowercase, underscores, and dates (as necesary) with a name that will be meaningful to whoever reads it.

We will update with links to style guides.

## ZenHub for Project Management

Our team uses the GitHub Google Chrome extension, [ZenHub](https://chrome.google.com/webstore/detail/zenhub-for-github/ogcgkffhplmphkaahpmffcafajaocjbd?hl=en-US), for project management.

#### ZenHub Flow - Issue Card

To create an issue card, begin [here](https://github.com/lzim/teampsd/issues/new/choose) and follow the flow to create and maintain your issue card. 

## Files

### Create and Edit Files

1. To create files, choose the branch or folder you want and click “Create New File.”
2. To view the file’s code, simply click “Raw” in the top right corner of the file, next to “Blame” and “History.”
3. To view comments that you/others have added, you can click on “History” and then check for an “ellipses” symbol. This symbol indicates that a comment has been added to the commit. To see the comment, click on the ellipses. ![](data:text/plain; charset=utf-8;base64,NDA0OiBOb3QgRm91bmQ=)

### Upload Files

To upload files, choose the branch or folder you want and upload the file. The file will be named as it was in your computer, so make sure to give the file a name according that is simple and reflective of its contents to anyone that reads the file name.

### Download Files

Files cannot be downloaded individually - you have to download the entire branch. You can find the green “Download” button on the main page of each branch.

## Branches

### Create and Edit Branches

1. Open up the specific file you want to edit. Create a new working branch and name it using all lowercase, no spaces (use underscores), and your initials - i.e. “facilitation\_guide\_sp”.  
   ![](data:text/plain; charset=utf-8;base64,NDA0OiBOb3QgRm91bmQ=)
2. Make sure you’re in the branch you just created instead of the original branch. Click the pencil icon to edit. Edit the file and add comments on changes you made. Click on “Commit”.  
   ![](data:text/plain; charset=utf-8;base64,NDA0OiBOb3QgRm91bmQ=)

## Pull Requests

### **Create and Edit Pull Requests**

1. Go to “Pull Requests” and click on “New Pull Request”. The “Base” branch should be the original branch (most likely “master”). The “Compare” branch will be the branch you created. Click on “Create a New Pull Request”. Add “Reviewers” and click on “TeamPSD” for the “Project section. Click on”Create Pull Request".  
   ![](data:text/plain; charset=utf-8;base64,NDA0OiBOb3QgRm91bmQ=)
2. To add line specific comments, click on “Files Changed”. Hover your mouse in front of the specific line in question and a blue plus sign will appear. Click on that and add your comment.  
   ![](data:text/plain; charset=utf-8;base64,NDA0OiBOb3QgRm91bmQ=)