Software Teams and Communication

17-313: Foundations of Software Engineering

https://cmu-313.github.io

Josh Sunshine and **Michael Hilton**

Spring 2026

Smoking Section

- Last full row

Administriva

- **Project 2A - Team Contract & Project Planning**
  - Due Friday, January 30, 2026 @ 11:59PM
    - **Project 2A - Teamwork Contract – Team Gradescope Submission**
    - **Project 2A - Project Planning – Individual Gradescope Submission**
    - **Project 2A - Extra Credit – Group Submission (Please add your teammates if you have already submitted**
- **#copilot_archiver_support**

Participate in a study on debugging Java and patch suggestions!

We're conducting an eye tracking study to explore how developers debug with patch suggestions.

🔍 **Key Details:**

- **Duration:** At most 2 hours
- **Location:** TCS Hall
- **Compensation:** $50 for non-professionals
  $100 for professionals

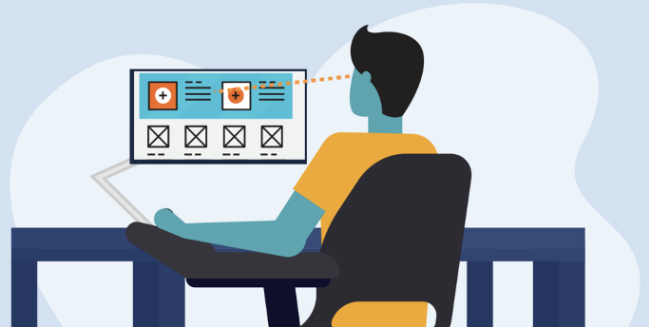
👥 **Participant Eligibility:**

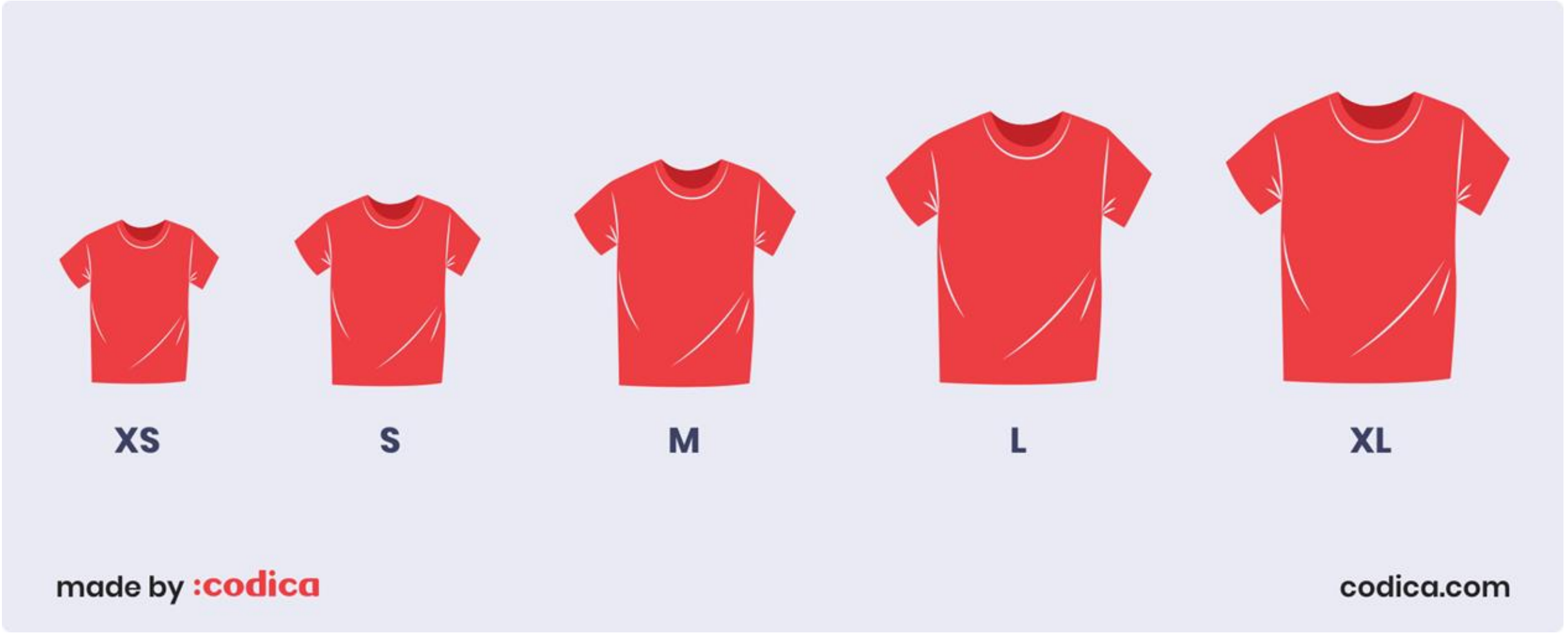
- Age 18+
- 1+ year(s) of Java experience
- Touch typists with normal or corrected-to-normal vision

💡 **Interested?**

Fill out our prescreening form at the link: https://tinyurl.com/4c7cd5ka or scan our QR code! Questions can be emailed to abegel@cmu.edu.

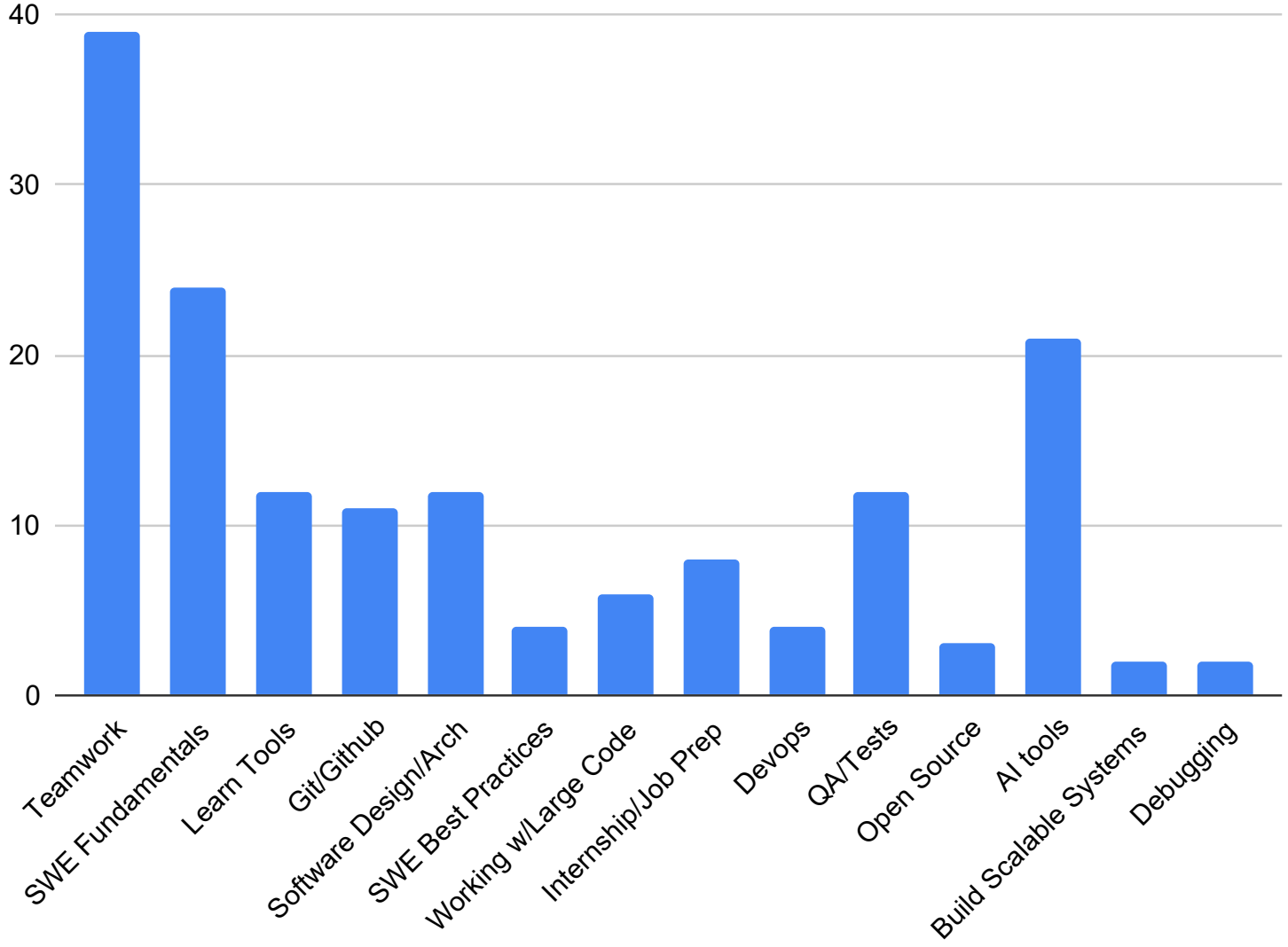
scan me!




XS
S
M
L
XL
made by :codica
codica.com

Learning Goals

- Describe the pros and cons of working as a team
- Recognize the importance of communication in collaboration
- Recognize the need of having multiple communication channels
- Select an appropriate communication tool for a given communication goal
- Ask technical questions effectively
- Write clear and specific Github issues, pull requests, and comments


40
30
20
10
0
Teamwork
SWE Fundamentals
Learn Tools
Git/Github
Software Design/Arch
SWE Best Practices
Working w/Large Code
Internship/Job Prep
Devops
QA/Tests
Open Source
AI tools
Build Scalable Systems
Debugging

https://www.productplan.com/glossary/bubble-sort/

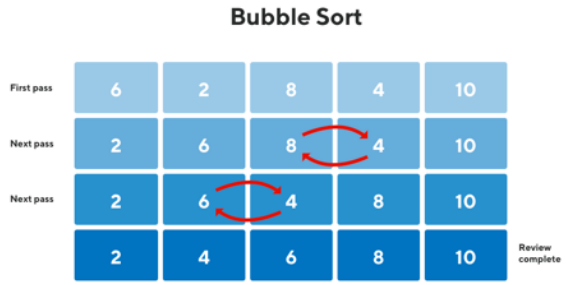
https://www.amazon.com/Monopoly-Classic-Replacement-Board-Hasbro/dp/B00DOZ2IBM

https://www.forbes.com/sites/bernardmarr/2020/07/17/5-ways-self-driving-cars-could-make-our-world-and-our-lives-better/

Https://dribbble.com/shots/12512417-Scooter-Rental-App/

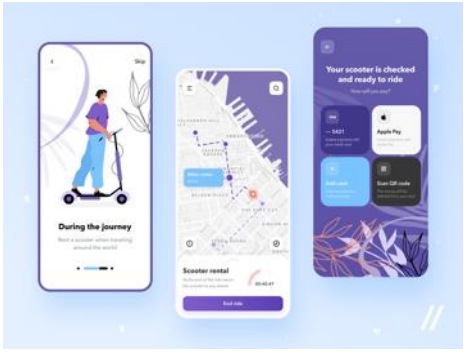
https://teedy.io/en/#!/

We all work in a team



Bubble Sort

Monopoly Game



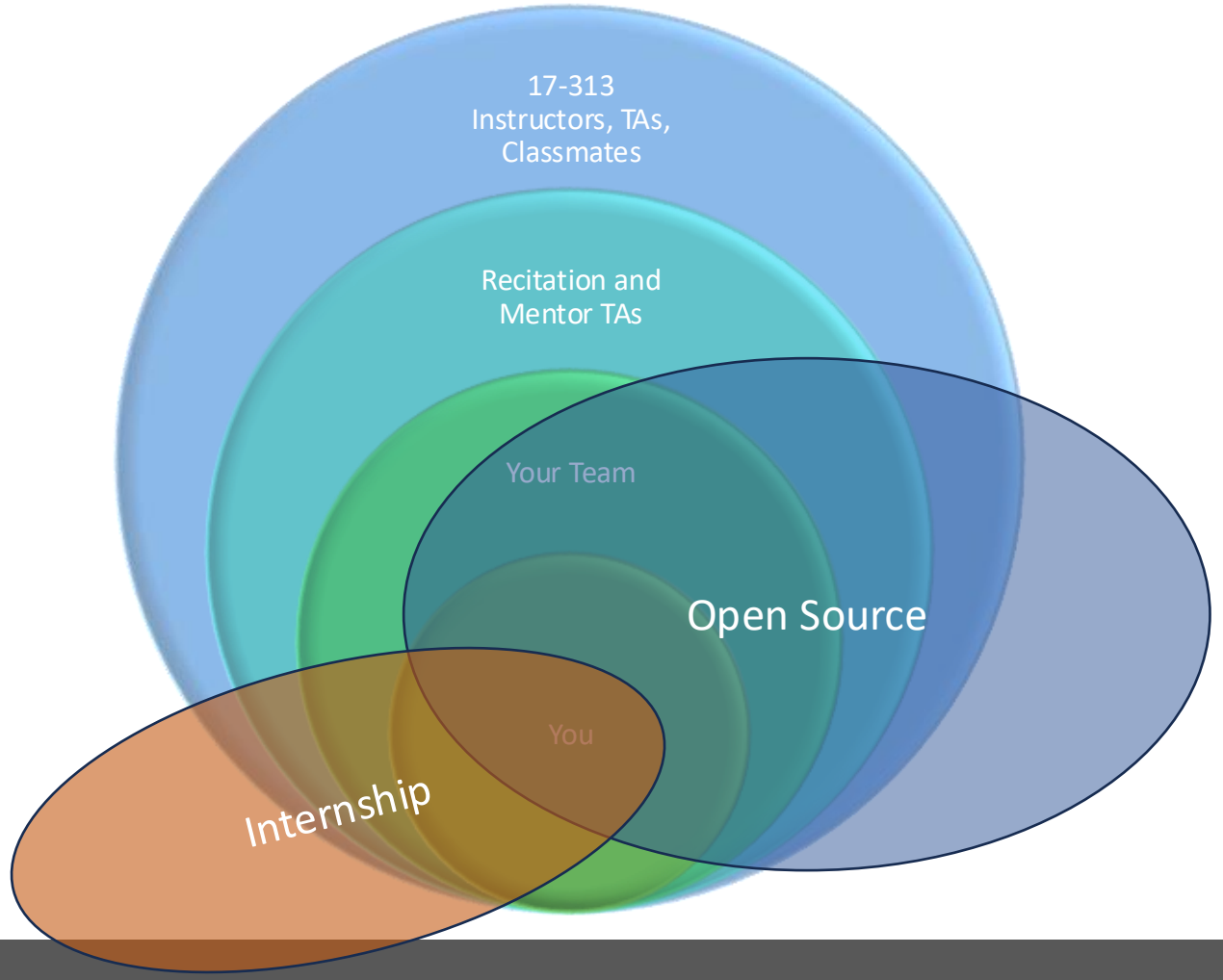
Scooter App

NodeBB

Autonomous Vehicle

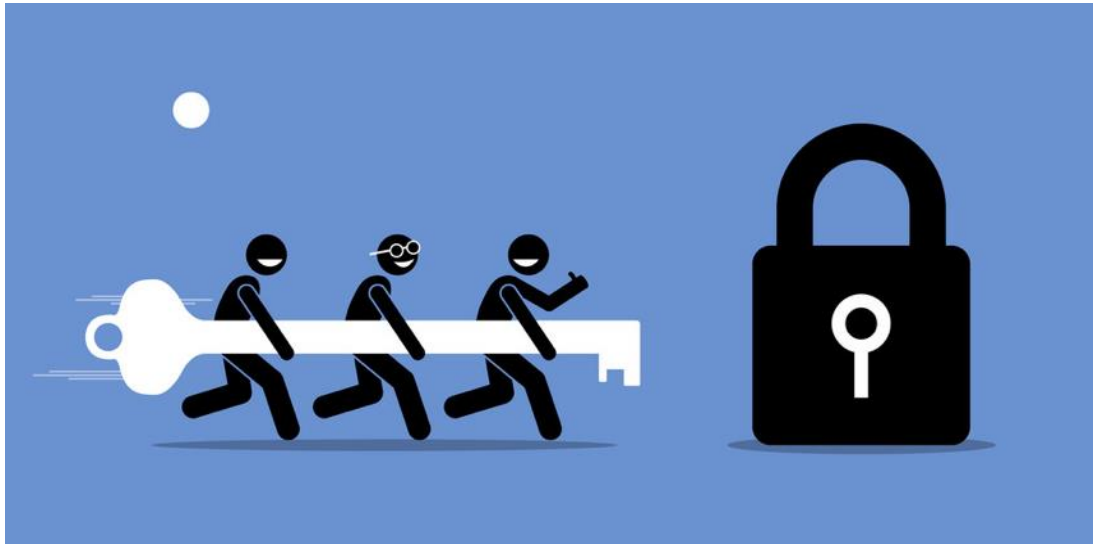
We all work in a team

We all work in a team



vectorstock.com/23988884

Working solo vs. as a team (human/AI?)



Participation Activity

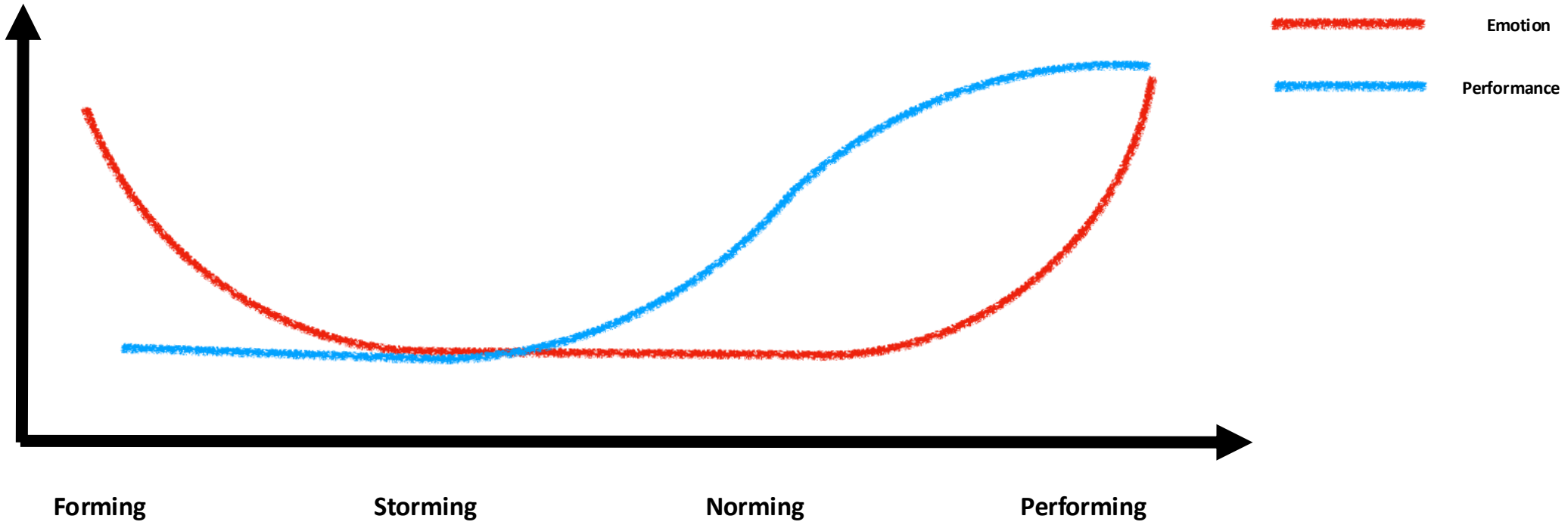
- Name in top left hand corner
- Three cases:
  - Solo
  - Team with humans
  - Team with AI

- For each case, come up with 1 benefit and 1 drawback

Working as a team

- Design & implement software
  - Establish a collaboration process
  - Meet with the team
  - Choose a leader
  - Divide work and integrate
  - Share knowledge
  - Resolve conflicts

Stages of Team Formation



Tuckman, B. W. (1965). Developmental sequence in small groups. Psychological Bulletin, 63, 384-399.

Norming

- When working with someone who is remote, how do you like to work together?
- How do you manage your time when you get busy with a lot of tasks?
- How do you feel about chatting by text message, audio call, video call?
  - Exchange phone numbers with your project partner(s) in case your Internet goes out and you still want to work on the project together.
- Negotiate when you can work on the project together outside of class.
- Have you had a positive prior teaming experience?
  - How often did your team meet?
  - Did your team have a leader? If yes, what did that leader do?
  - What was your role on the team?
  - How well did you get along with your teammates related to work, or related to non-work?

Projects

Project 1: Hello, NodeBB!

Project 2: Collaborative Development

2A Team Process & Planning

2B First Sprint

2C Second Sprint

how to improve it. The first one will be released on **Friday, September 8th** and due the following **Friday, September 15th** (both at 11:59pm).

Main Deliverables

Teamwork Contract (35 pts)

When working with a team, it is important to discuss each team member's background, and establish common expectations of the team. Miscommunication or the general lack of communication are often the most common causes of team conflict.

> **Team Conflict Example**
>
> A common conflict in working style is when there are team members who always want to get a headstart on their work, while there are team members who are fine with doing work a few days before the deadline. It causes panic in the former team members, while the latter team members feel frustrated as to why they are being rushed.

As such, your first process task of the semester will be creating a teamwork contract with your teammates. It is a **1 - 2 page** document containing information that all teammates agree to follow. You should work on the contract **with all members present**. We recommend that you keep it to around 1 page, 2 page is a **hard limit.**

Additionally, **it is more important that you only include statements that the team will adhere to** than it is to fulfill the length requirement (quality over quantity!) You do not need to write full sentences (bullet points are okay), but your decisions must be clearly conveyed in the document.

You are free to include anything that your team deems necessary, but you should minimally address the following sections:

1. **Expectations**

On This Page

Deliverables

Team Setup

Slack Channel

GitHub Repository

Teamwork Self-Assessment

Main Deliverables

Teamwork Contract (35 pts)

Project Planning (35 pts)

Extra Credit (7 pts)

Grading

S3

Carnegie Mellon University

(11) *General Interference with Organizations an Production*

(a) Organizations and Conferences

(1) Insist on doing everything throug "channels." Never permit short-cuts to be take in order to expedite decisions.

(2) Make "speeches." Talk as frequently possible and at great length. Illustrate yo "points" by long anecdotes and accounts of pe sonal experiences. Never hesitate to make a fe appropriate "patriotic" comments.

(3) When possible, refer all matters committees, for "further study and consider tion." Attempt to make the committees as lar as possible — never less than five.

(4) Bring up irrelevant issues as frequent as possible.

(5) Haggle over precise wordings of cor munications, minutes, resolutions.

(6) Refer back to matters decided upon the last meeting and attempt to re-open t question of the advisability of that decision.

(7) Advocate "caution." Be "reasonabl and urge your fellow-conferees to be "reaso able" and avoid haste which might result embarrassments or difficulties later on.

(8) Be worried about the propriety of a decision — raise the question of whether su action as is contemplated lies within the jur diction of the group or whether it might confl with the policy of some higher echelon.

(b) Managers and Supervisors

(1) Demand written orders.

(2) "Misunderstand" orders. Ask endles questions or engage in long correspondenc about such orders. Quibble over them when yo can.

(3) Do everything possible to delay the delivery of orders. Even though parts of an orde may be ready beforehand, don't deliver it unti it is completely ready.

(4) Don't order new working materials until your current stocks have been virtually ex hausted, so that the slightest delay in filling your order will mean a shutdown.

(5) Order high-quality materials which are hard to get. If you don't get them argue abou it. Warn that inferior materials will mean in ferior work.

(6) In making work assignments, always sign out the unimportant jobs first. See that the important jobs are assigned to inefficient workers of poor machines.

(7) Insist on perfect work in relatively un important products; send back for refinishing those which have the least flaw. Approve other defective parts whose flaws are not visible to the naked eye.

(8) Make mistakes in routing so that parts and materials will be sent to the wrong place in the plant.

(9) When training new workers, give in complete or misleading instructions.

(10) To lower morale and with it, produc tion, be pleasant to inefficient workers; give them undeserved promotions. Discriminate against efficient workers; complain unjustly about their work.

(11) Hold conferences when there is more critical work to be done.

(12) Multiply paper work in plausible ways. Start duplicate files.

(13) Multiply the procedures and clearances involved in issuing instructions, pay checks, and so on. See that three people have to approve everything where one would do.

(14) Apply all regulations to the last letter.

(c) Office Workers

(1) Make mistakes in quantities of material when you are copying orders. Confuse similar names. Use wrong addresses.

(2) Prolong correspondence with government bureaus.

(3) Misfile essential documents.

(4) In making carbon copies, make one too few, so that an extra copying job will have to be done.

(5) Tell important callers the boss is busy or talking on another telephone.

(6) Hold up mail until the next collection.

(7) Spread disturbing rumors that sound like inside dope.

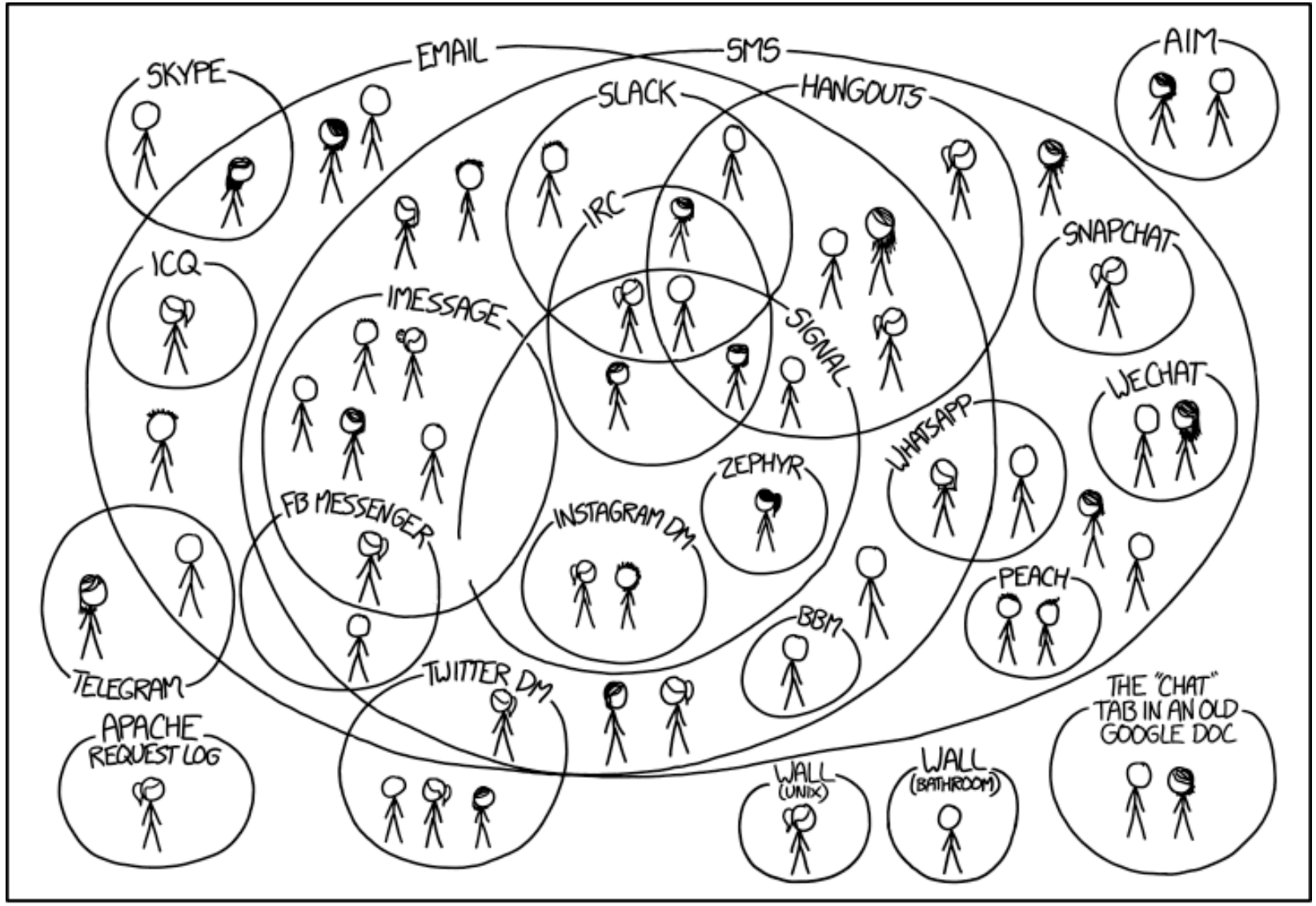
(d) Employees

(1) *Work slowly.* Think out ways to increase the number of movements necessary on your job: use a light hammer instead of a heavy one, try to make a small wrench do when a big one is necessary, use little force where considerable force is needed, and so on.

(2) Contrive as many interruptions to your work as you can: when changing the material on which you are working, as you would on a lathe or punch, take needless time to do it. If you are cutting, shaping or doing other measured work, measure dimensions twice as often as you need to. When you go to the lavatory, spend a longer time there than is necessary. Forget tools so that you will have to go back after them.

S3D Software and Societal Systems Department
Carnegie Mellon University

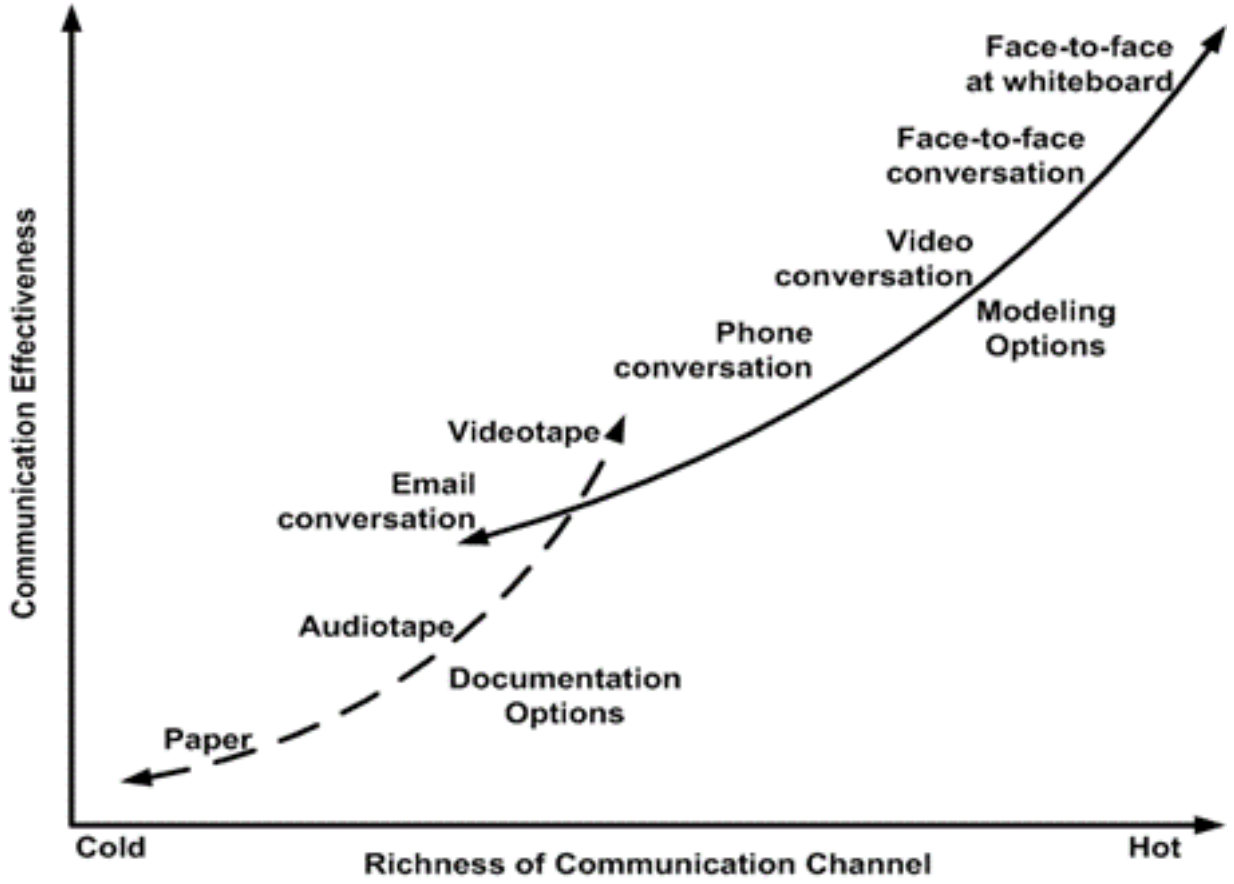
Establish a collaboration process



I HAVE A HARD TIME KEEPING TRACK OF WHICH CONTACTS USE WHICH CHAT SYSTEMS.

Ambler, S. (2002). Agile modeling: effective practices for extreme programming and the unified process. John Wiley & Sons.

Select the right comm. tools



Copyright 2002-2005 Scott W. Ambler
Original Diagram Copyright 2002 Alistair Cockburn

Establish communication patterns

- Asana, Trello, Microsoft Projects, …
- Github Wiki, Google Docs, Notion, …
- Github Issues, Jira, …
- Email, Slack, Facebook groups, …
- Zoom, Microsoft Teams, Skype, Phone call, …
- Face-to-face meetings

17-313 Communication channels

- Slack
- Regular meeting (Lectures, Recitations)
- Office Hours
- Canvas, Gradescope
- Webpage

Check out other projects

Communication

- Forums: Discuss implementations, research, etc. https://discuss.pytorch.org
- GitHub Issues: Bug reports, feature requests, install issues, RFCs, thoughts, etc.
- Slack: The PyTorch Slack hosts a primary audience of moderate to experienced PyTorch users and developers for general chat, online discussions, collaboration, etc. If you are a beginner looking for help, the primary medium is PyTorch Forums. If you need a slack invite, please fill this form: https://goo.gl/forms/PP1AGvNHpSaJP8to1
- Newsletter: No-noise, a one-way email newsletter with important announcements about PyTorch. You can sign-up here: https://eepurl.com/cbG0rv
- Facebook Page: Important announcements about PyTorch. https://www.facebook.com/pytorch
- For brand guidelines, please visit our website at pytorch.org

Communication expectation

- Quality of service guarantee
  - How soon will you get back to your teammates?
  - Weekend? Evening?
- Emergency
  - Tag w/ 911
  - Notify everyone with @channel

Running a (good) meeting

How to run a meeting

- The Three Rules of Running a Meeting
  - Set the Agenda
  - Start on Time. End on Time.
  - End with Action Items (and share them - Github Issues, Meeting Notes, ...)

How to run a meeting

- Set and document clear responsibilities and expectations
- Make everyone contribute
  - Possible Roles: Coordinator, Scribe, Checker
  - Manage Personalities
  - Be Vulnerable

Random Advice

- Note takers have a lot of power to steer the meeting
  - Collaborative notes are even better!
- Different meeting types have different best practices
  - Decision-making meeting
  - Brainstorming meeting
  - One-on-one meeting
  - Working sessions
- PLEASE RECORD ATTENDENCE

WHY DO YOU WANT TO CALL A MEETING?

DON'T KNOW OTHER WAYS TO COLLABORATE

SHARE INFORMATION

BUILD RELATIONSHIPS

CAN'T SELF-MOTIVATE

MAKE A DECISION
SOLVE A PROBLEM
GET FEEDBACK
GENERATE IDEAS
MAKE A PLAN
REFLECT & IMPROVE

PROVE HOW IMPORTANT I AM

Would email or a Confluence page do the trick?

Like a manager 1-1 thing?

Does this need to happen in real time?

Seriously?

GREAT. NO MEETING!

Can you find a time that works for everyone fairly soon?

BOOM. SCHEDULE IT!

Does everyone have enough background info to get it done?

VERY FUNNY. NO MEETING!

OK, FINE. BUT THIS BETTER BE IMPORTANT.

Do you just want an excuse to eat donuts?

FIX THAT FIRST. NO MEETING.

Has this been discussed before?

SO EAT A DONUT. NO MEETING!

You're procrastinating, aren't you?

Should we believe you?...

You really think this time will be different?

BOOM. SCHEDULE IT!

DON'T BE LAME. NO MEETING!

LIAR. NO MEETING!

BOOM. SCHEDULE IT!

GET REAL. NO MEETING!

atlassian.com/effectivemeetings

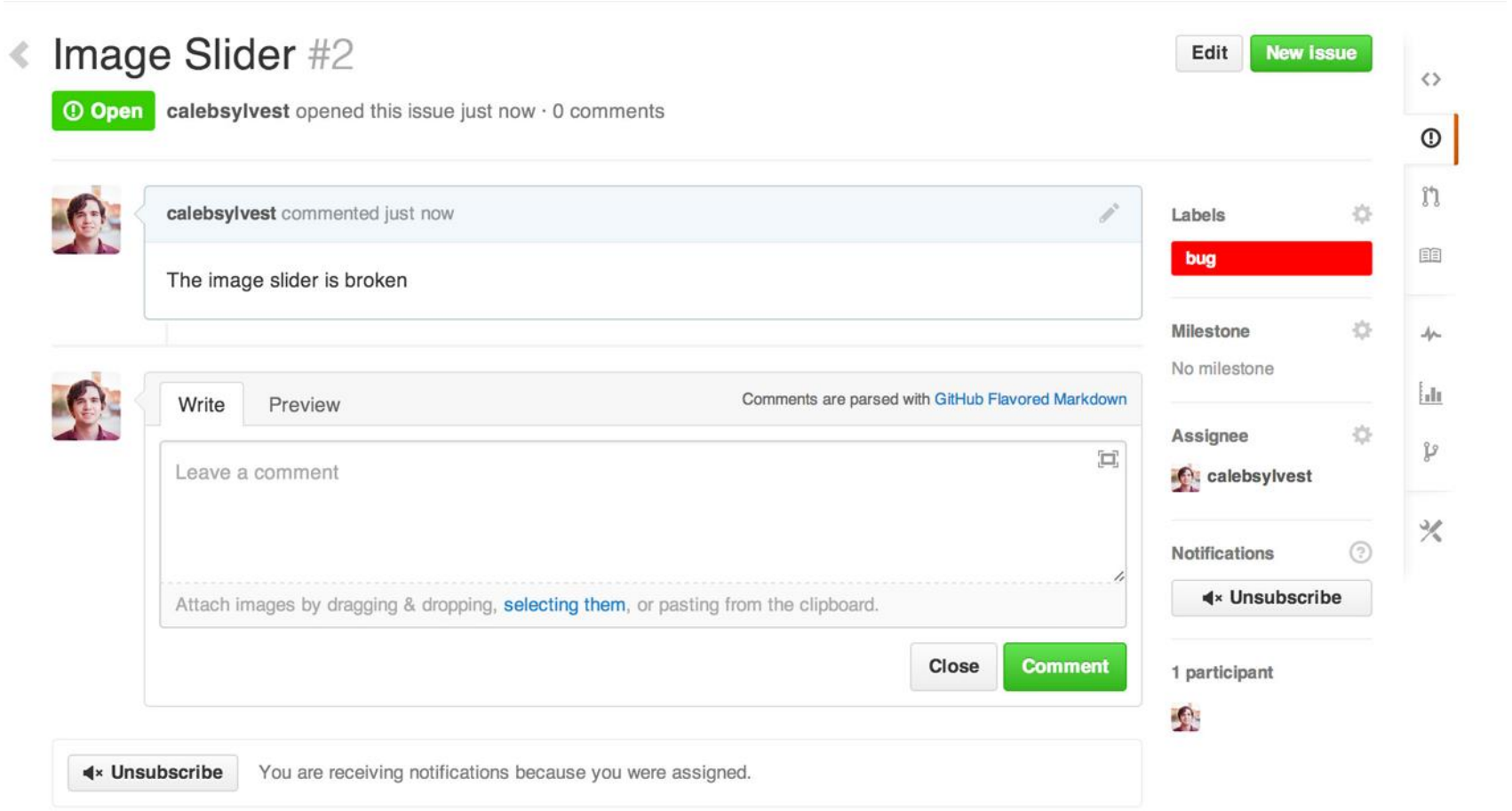
ATLASSIAN

S3D So Sy

Carnegie Mellon University

Async Communication via artifacts

Is this issue useful?



https://upthemes.com/blog/2014/02/writing-useful-github-issues/

https://upthemes.com/blog/2014/02/writing-useful-github-issues/

Writing useful Github issues

Cropping of Image Slider Pics #3

Edit | New Issue

Open **calebsylvest** opened this issue just now · 0 comments

calebsylvest commented just now

http://calebsylvest.com/

The cropping of the images in the slideshow seem to be off. The text is not visible and partially hidden by content below. The Developer Tools show the full-size un-cropped image is being loaded, but obviously not displaying.

Browser: Google Chrome
OS: Mavericks
Hardware: MacBook Pro Retina

Labels

bug

Milestone

No milestone

Assignee

calebsylvest

Notifications

Unsubscribe

https://wiredcraft.com/blog/how-we-write-our-github-issues/

https://docs.github.com/en/issues/tracking-your-work-with-issues/about-issues

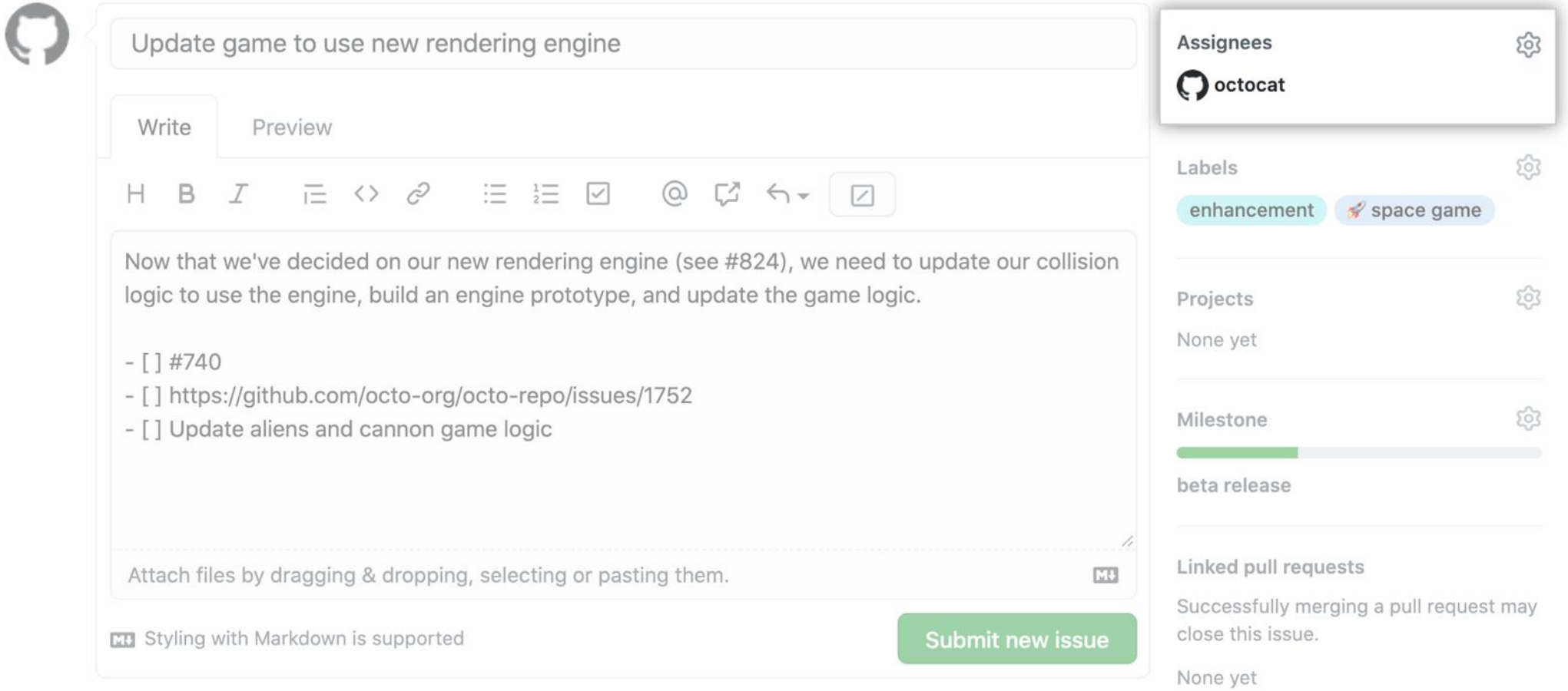
Writing useful Github issues

- Issue should include
  - Context: explain the conditions which led you to write the issue
  - Problem or idea: the context should lead to something
  - Previous attempts to solve
  - Solution or next step (if possible)
- Be specific!
  - Include environment settings, versions, error messages, code examples when necessary

Writing useful Github issues

- Check out guidelines
  - Google: https://developers.google.com/issue-tracker/concepts/issues
  - Rust: https://rustc-dev-guide.rust-lang.org/contributing.html#bug-reports
- Don’t assume the solution
- One issue per issue
- Keep titles short and descriptive
- Format your messages

@Mention or assign appropriate people

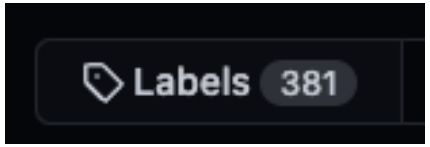


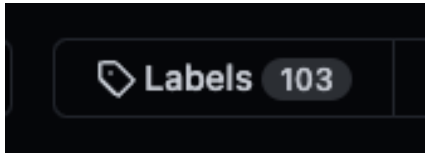
https://nira.com/common-github-issues/

https://rewind.com/blog/best-practices-for-using-github-issues/

Use labels

- Break the project down by areas of responsibility
- Mark non-triaged issues
- Isolate issues that await additional information from the reporter
- Example:
  - Bug / Duplicate / Documentation / Help Wanted / Invalid / Enhancement
  - status: wip, status: ready to implement, status: needs discussion

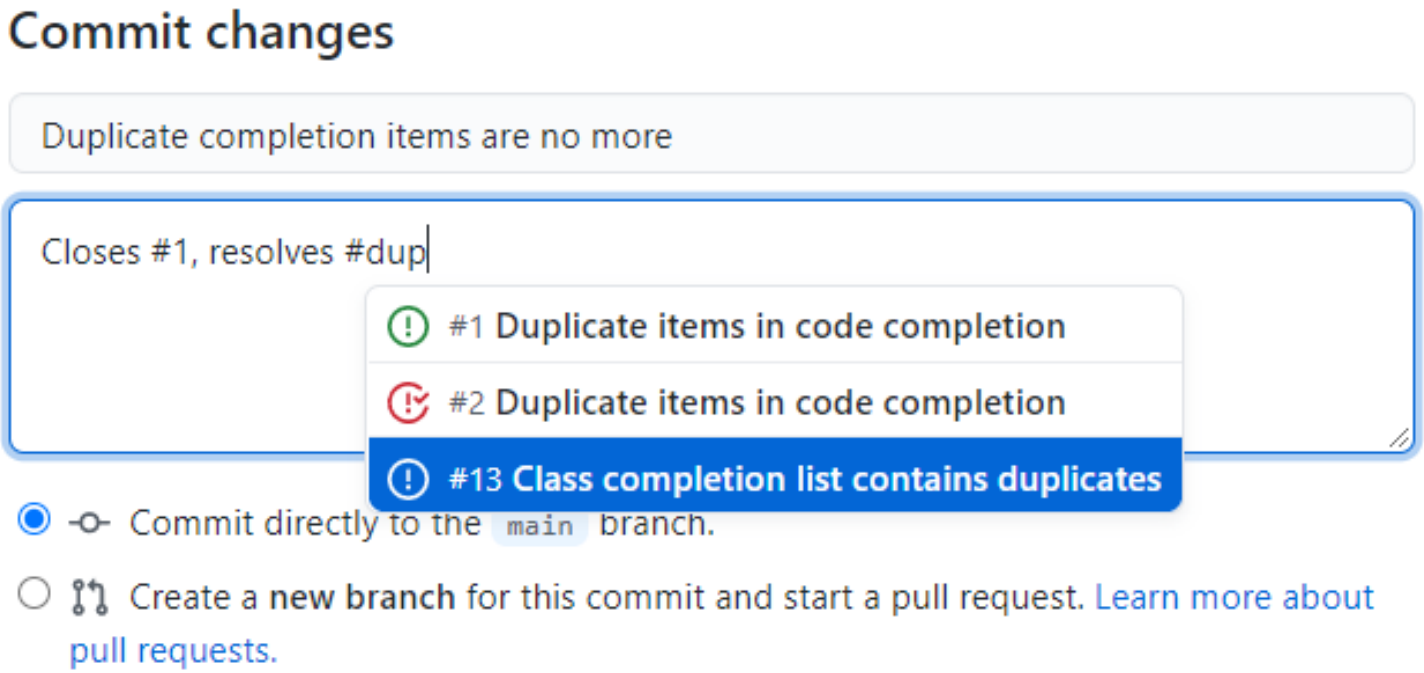




https://rewind.com/blog/best-practices-for-using-github-issues/

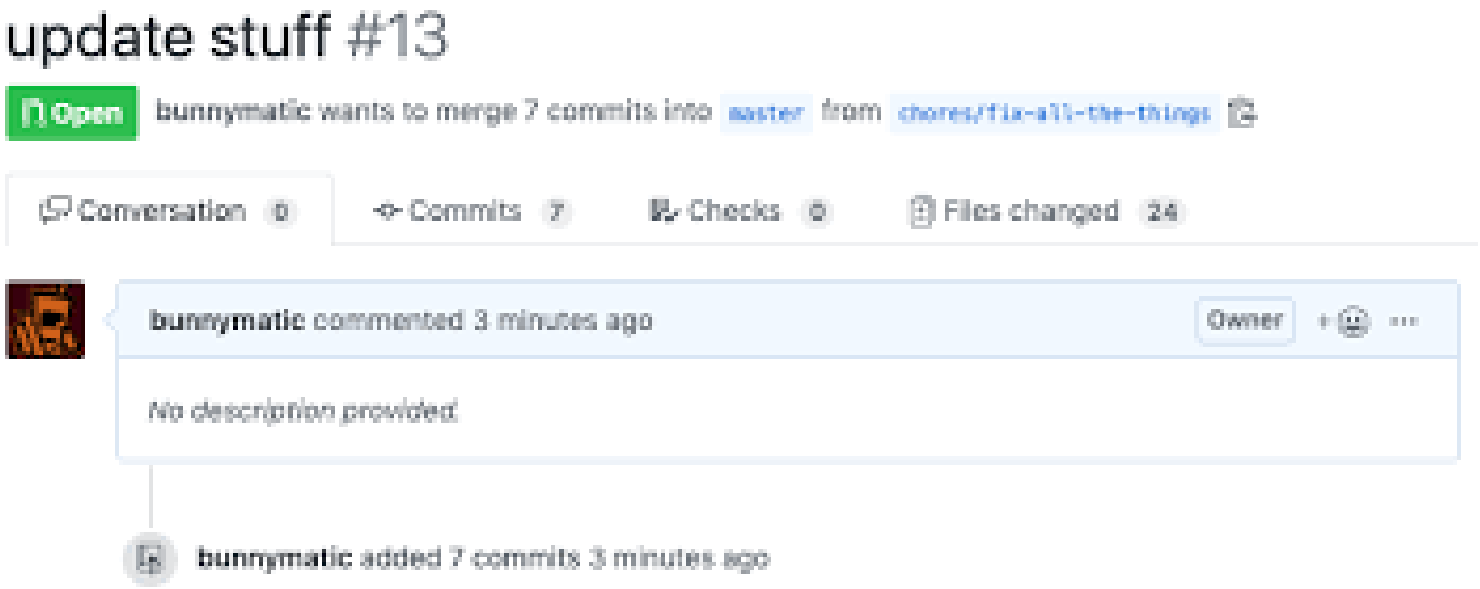
Don't forget to follow-up and close issues

- closes/resolves #issue_number



https://blog.carbonfive.com/why-write-good-pull-requests/

Pull requests



https://www.pullrequest.com/blog/writing-a-great-pull-request-description/

How to write good pull requests

```
## What?

## Why?

## How?

## Testing?

## Screenshots (optional)

## Anything Else?
```

```
## What?
I've added support for authentication to implement Key Result 2 of OKR1. It includes model, table,
controller and test. For more background, see ticket
#JIRA-123.
## Why?
These changes complete the user login and account creation experience. See #JIRA-123 for more
information.
## How?
This includes a migration, model and controller for user authentication. I'm using Devise to do the
heavy lifting. I ran Devise migrations and those are included here.
## Testing?
I've added coverage for testing all new methods. I used Faker for a few random user emails and
names.
## Screenshots (optional)
0
## Anything Else?
Let's consider using a 3rd party authentication provider for this, to offload MFA and other
considerations as they arise and as the privacy landscape evolves. AWS Cognito is a good option, so
is Firebase. I'm happy to start researching this path. Let's also consider breaking this out into
its own service. We can then re-use it or share the accounts with other apps in the future.
```

https://github.blog/2015-01-21-how-to-write-the-perfect-pull-request/

https://betterprogramming.pub/how-to-make-a-perfect-pull-request-3578fb4c112

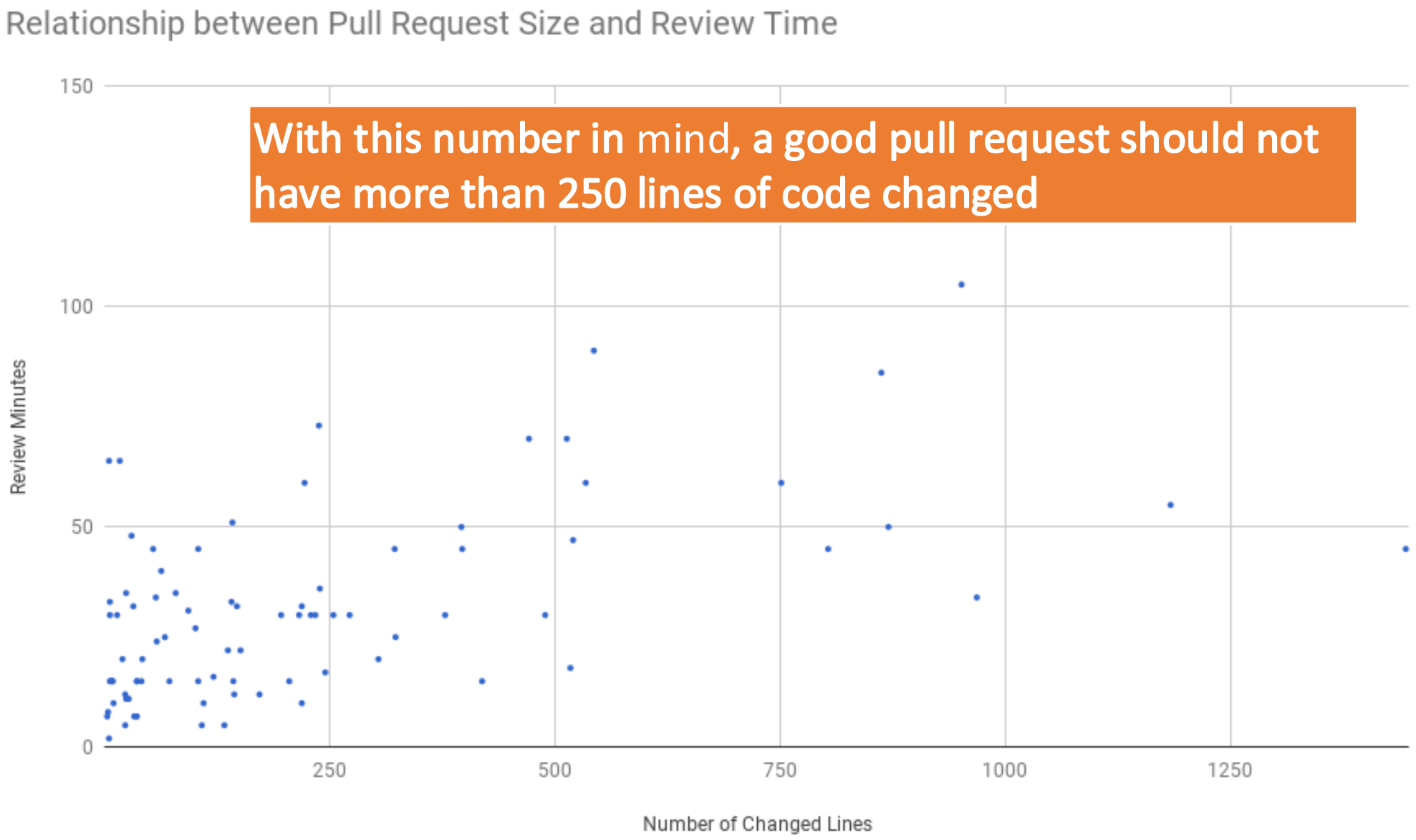
How to write good pull requests

- Remember that anyone (in the company) could be reading your PR
- Be explicit about what/when feedback you want
- @mention individuals that you specifically want to involve in the discussion, and mention why.
  - “/cc @jesseplusplus for clarification on this logic”

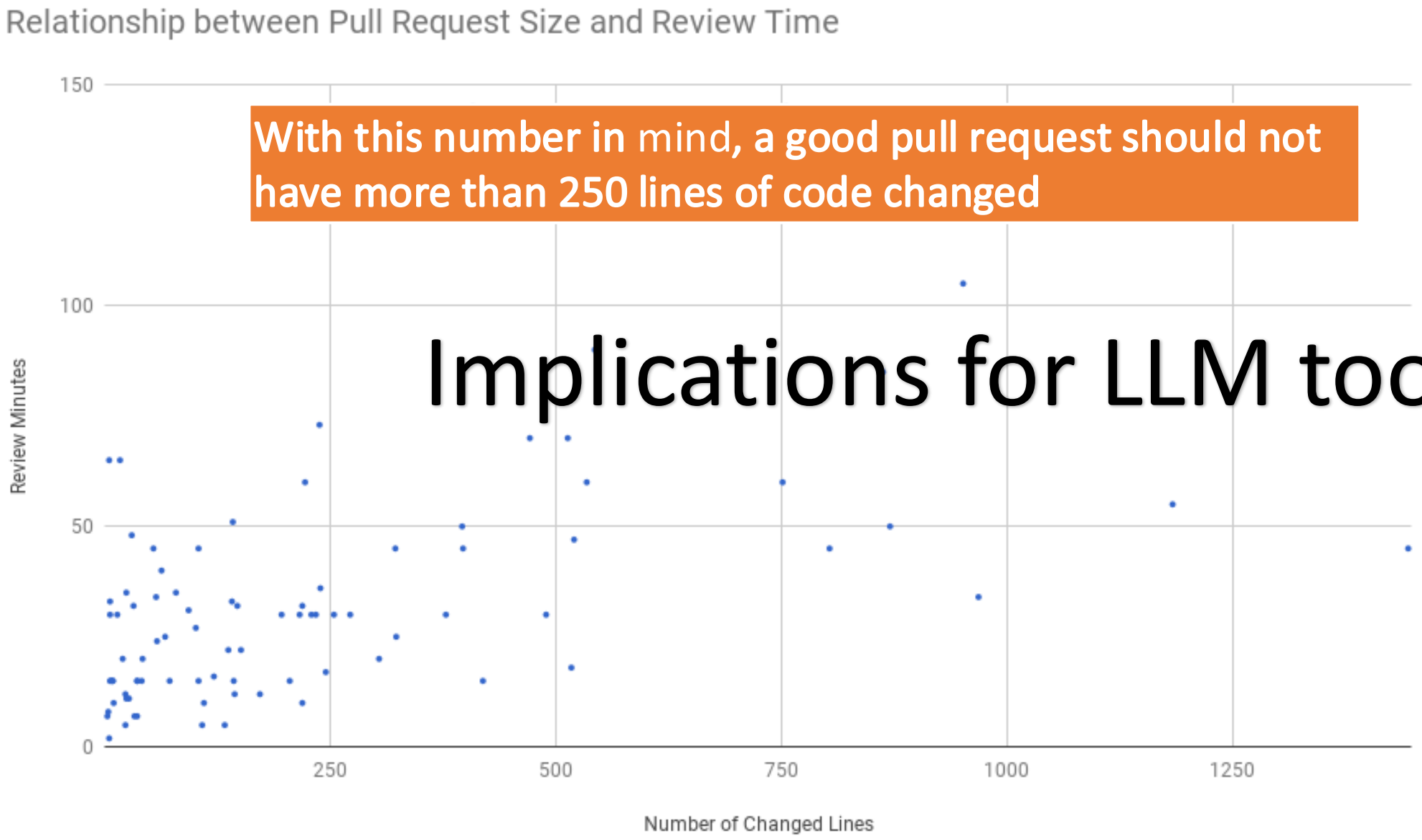
https://twitter.com/iamdevloper/status/397664295875805184?s=20&t=EckdvW89m8RJgndHvWvV8Q

Keep your PRs small

https://hugooodias.medium.com/the-anatomy-of-a-perfect-pull-request-567382bb6067
https://smartbear.com/learn/code-review/best-practices-for-peer-code-review/



https://hugooodias.medium.com/the-anatomy-of-a-perfect-pull-request-567382bb6067
https://smartbear.com/learn/code-review/best-practices-for-peer-code-review/



With this number in mind, a good pull request should not have more than 250 lines of code changed

Implications for LLM tools?

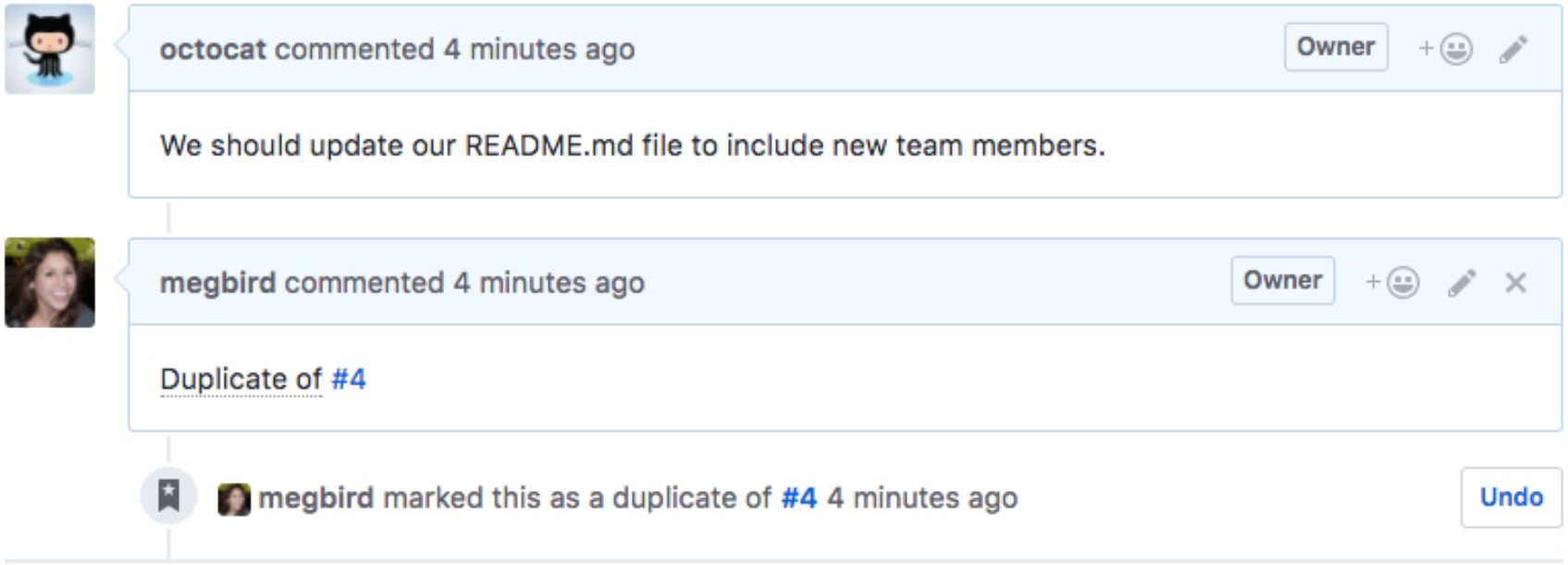
https://github.blog/2015-01-21-how-to-write-the-perfect-pull-request/

Offer useful feedback

- If you disagree strongly, consider giving it a few minutes before responding; think before you react.
- Ask, don't tell. ("What do you think about trying…?" rather than "Don't do…")
- Explain your reasons why code should be changed. (Not in line with the style guide? A personal preference?)
- Be humble. ("I'm not sure, let's try…")
- Avoid hyperbole. ("NEVER do…")
- Be aware of negative bias with online communication.

Avoid Duplicates

- “Duplicate of” issue/pull request number



Be a nice person

Date Sat, 13 Jul 2013 15:40:24 -0700
Subject Re: [GIT pull] x86 updates for 3.11
From Linus Torvalds <>

share 638

```
On Sat, Jul 13, 2013 at 4:21 AM, Thomas Gleixner <tglx@linutronix.de> wrote:
>
>    * Guarantee IDT page alignment

What the F*CK, guys?

This piece-of-shit commit is marked for stable, but you clearly never
even test-compiled it, did you?

Because on x86-64 (the which is the only place where the patch
matters), I don't see how you could have avoided this honking huge
warning otherwise:

  arch/x86/kernel/traps.c:74:1: warning: braces around scalar
initializer [enabled by default]
   gate_desc idt_table[NR_VECTORS] __page_aligned_data = { { { { 0, 0 } } }, };
   ^
```

Using AI to write commit messages

- Stage your changes in the Source Control panel by clicking the + icon next to the files you want to include in the commit.
- Locate the commit message box at the top of the Source Control view.
- Select the sparkle icon (✨) next to the input box. This will prompt the AI to analyze your staged changes and generate a commit message.
- Review the generated message and edit it as needed to ensure accuracy and clarity.
- Click "Commit" to finalize the commit with the AI-generated message.

Knowledge Sharing

https://about.gitlab.com/topics/version-control/software-team-collaboration/

No matter the format, documentation is important

Building on top of others' work in a community-like way can be an accelerator, both in open source and in companies. Documentation often signals if a repository is reliable to reuse code from, or if it's an active project to contribute to. What signs do developers look for?

In both open source projects and enterprises, developers see about

50%

productivity boost with easy-to-source documentation

What the data shows: At work, developers consider documentation trustworthy when it is up-to-date (e.g., looking at time-stamps) and has a high number of upvotes from others. Open source projects use READMEs, contribution guidelines, and GitHub Issues, to elevate the quality of any project, and to share information that makes them more attractive to new contributors. Enterprises can adopt the same best practices to achieve similar success.

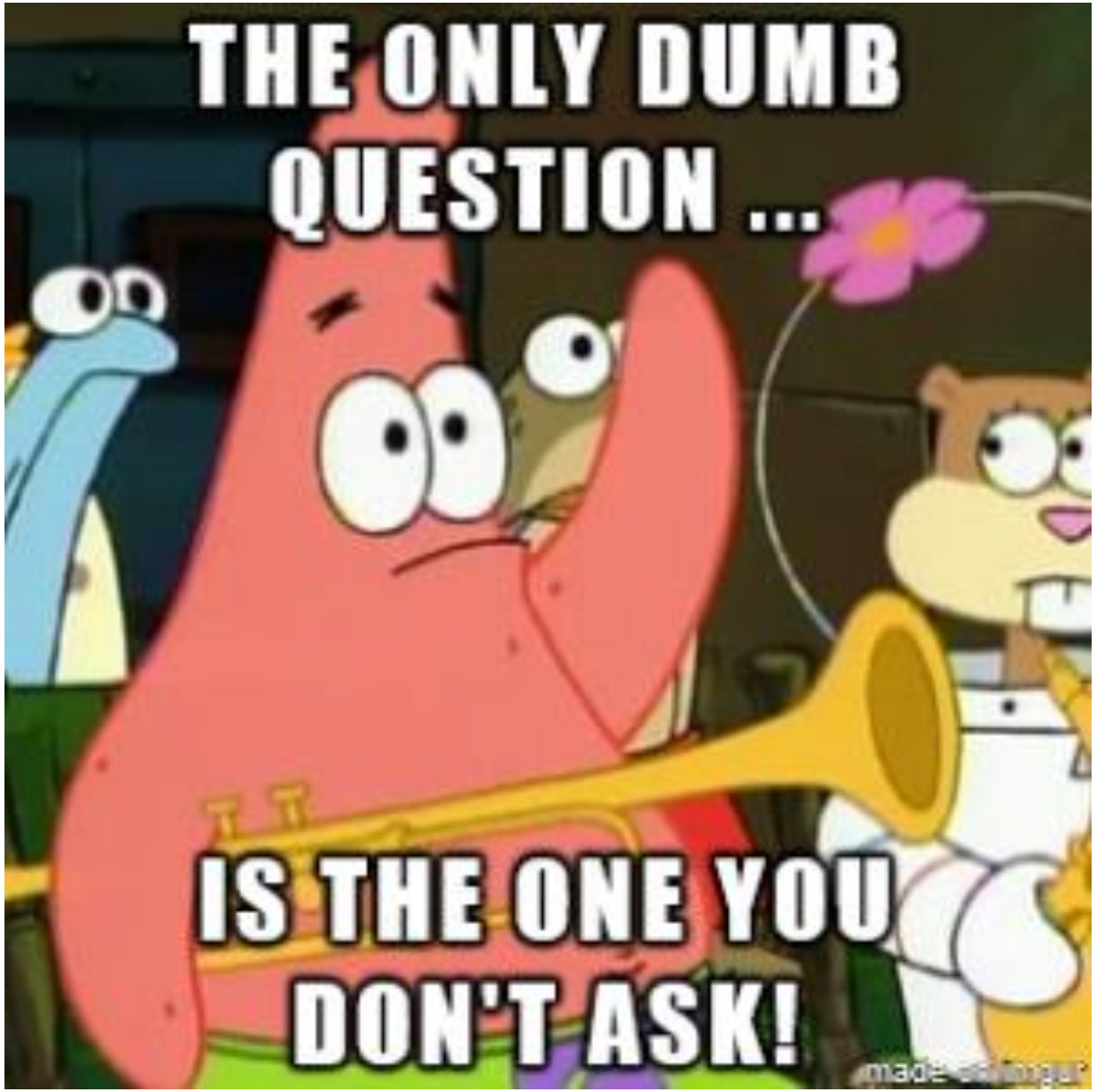
In both environments, developers see about a 50% productivity boost when documentation is up-to-date, detailed, reliable, and comes in different formats (e.g. articles, videos, forums).

Using the data: Review the documentation your team consumes: When was the last time it was updated? Can everyone on your team improve the documentation? Check this frequently to stay on track.

Knowledge Type	Description (Excerpt)
Functionality and Behavior	Describes what the API does (or does not do) in terms of functionality or features. Describes what happens when the API is used (a field value is set, or a method is called).
Concepts	Explains the meaning of terms used to name or describe an API element, or describes design or domain concepts used or implemented by the API.
Directives	Specifies what users are allowed / not allowed to do with the API element. Directives are clear contracts.
Purpose and Rationale	Explains the purpose of providing an element or the rationale of a certain design decision. Typically, this is information that answers a "why" question: Why is this element provided by the API? Why is this designed this way? Why would we want to use this?
Quality Attributes and Internal Aspects	Describes quality attributes of the API, also known as non-functional requirements, for example, the performance implications. Also applies to information about the API's internal implementation that is only indirectly related to its observable behavior.
Control-Flow	Describes how the API (or the framework) manages the flow of control, for example by stating what events cause a certain callback to be triggered, or by listing the order in which API methods will be automatically called by the framework itself.
Structure	Describes the internal organization of a compound element (e.g. important classes, fields, or methods), information about type hierarchies, or how elements are related to each other.
Patterns	Describes how to accomplish specific outcomes with the API, for example, how to implement a certain scenario, how the behavior of an element can be customized, etc.
Code **Examples**	Provides code examples of how to use and combine elements to implement certain functionality or design outcomes.
Environment	Describes aspects related to the environment in which the API is used, but not the API directly, e.g., compatibility issues, differences between versions, or licensing information.
References	Includes any pointer to external documents, either in the form of hyperlinks, tagged "see also" reference, or mentions of other documents (such as standards or manuals).
Non-information	A section of documentation containing any complete sentence or self-contained fragment of text that provides only uninformative boilerplate text.

Know your audience

- Internal document for your team (e.g., meeting note)
- Documentation for project contributors
- Documentation for non-developer collaborators (e.g., UX researchers)
- Documentation for developer users
- Documentation for clients with no software knowldge
- User manual for end users


THE ONLY DUMB QUESTION ...
IS THE ONE YOU DON'T ASK!

https://stackoverflow.com/questions/31279359/new-to-coding-can-anyone-assist-me
https://www.memesmonkey.com/topic/help+me+help+you#&gid=1&pid=3

What is wrong with this question?

New To Coding. Can anyone assist me?

Asked 7 years, 1 month ago Modified 7 years, 1 month ago Viewed 47 times

-4

I am trying to make a word counter and I just cant seem to get it. Can anyone help?

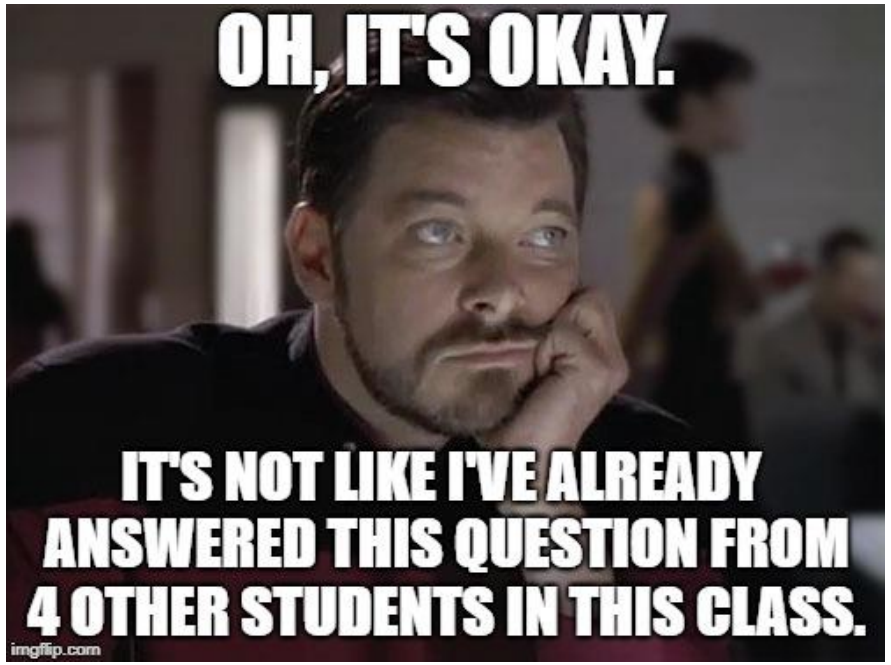
```
import re
print("Welcome To This Software Made By Aaron!")
word = raw_input("Enter Your Words: ")
Check = 0
Right = 0
Length = len(word)
while True:
    if Right == 1:
        if Length < Check:
            Check = Check + 1
            print(Check)
    if Length == Check:
        Right = 1

print("Your Word Count Is " +Check)
```

Make it easy for people to help you

- I am trying to ___, so that I can ___.
  I am running into ___.
  I have looked at ___ and tried ___.
- + I'm using this tech stack: ___.
- + I'm getting this error/result: ___.
- + I think the problem could be ___.

http://kwugirl.blogspot.com/2014/04/how-to-be-better-junior-developer_25.html

Avoid Duplication



RESEARCH-ARTICLE

Mining duplicate questions in stack overflow

Authors: Muhammad Ahasanuzzaman, Muhammad Asaduzzaman, Chanchal K. Roy, Kevin A. Schneider

Authors Info & Claims

Published: 04 November 2015

Studying the needed effort for identifying duplicate issues

Mohamed Sami Rakha, Weiyi Shang & Ahmed E. Hassan

Empirical Software Engineering **21**, 1960–1989 (2016) | Cite this article

748 Accesses | **19** Citations | **1** Altmetric | Metrics

Abstract

Many recent software engineering papers have examined duplicate issue reports. Thus far, duplicate reports have been considered a hindrance to developers and a drain on their resources. As a result, prior research in this area focuses on proposing automated approaches to accurately identify duplicate reports. However, there exists no studies that attempt to

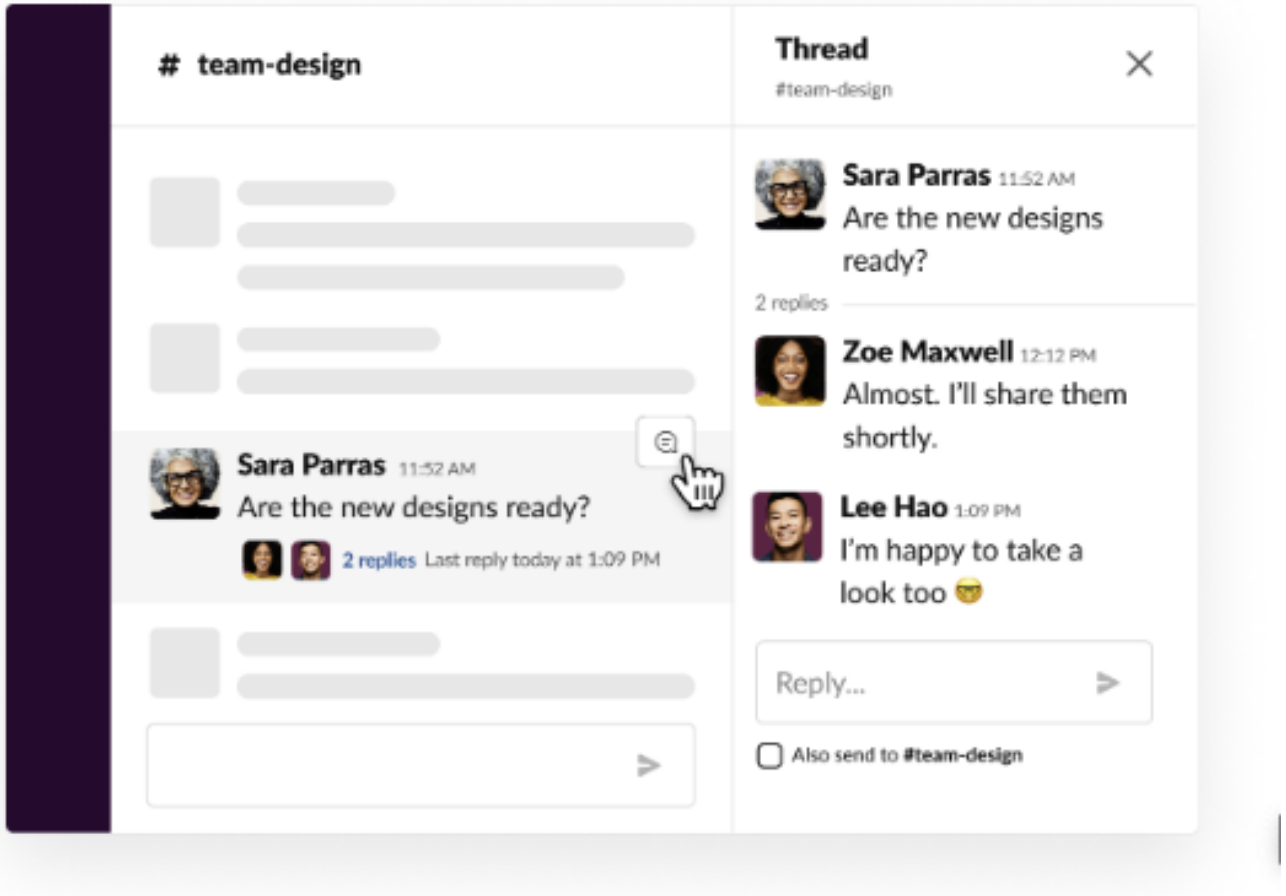
Avoid Duplication - Slack

- Add quotation marks to search a specific phrase
  - “Connection refused errors” will find results containing the entire phrase
- Add from: in front of a display name to search for information shared by someone specific
  - P1 from:@Michael Hilton
- Add is:thread to search within threads
  - WSL is:thread
- Recap problem in caption to enable searching (if using screenshots)

https://slack.com/help/articles/202528808-Search-in-Slack#narrow-your-search-with-modifiers

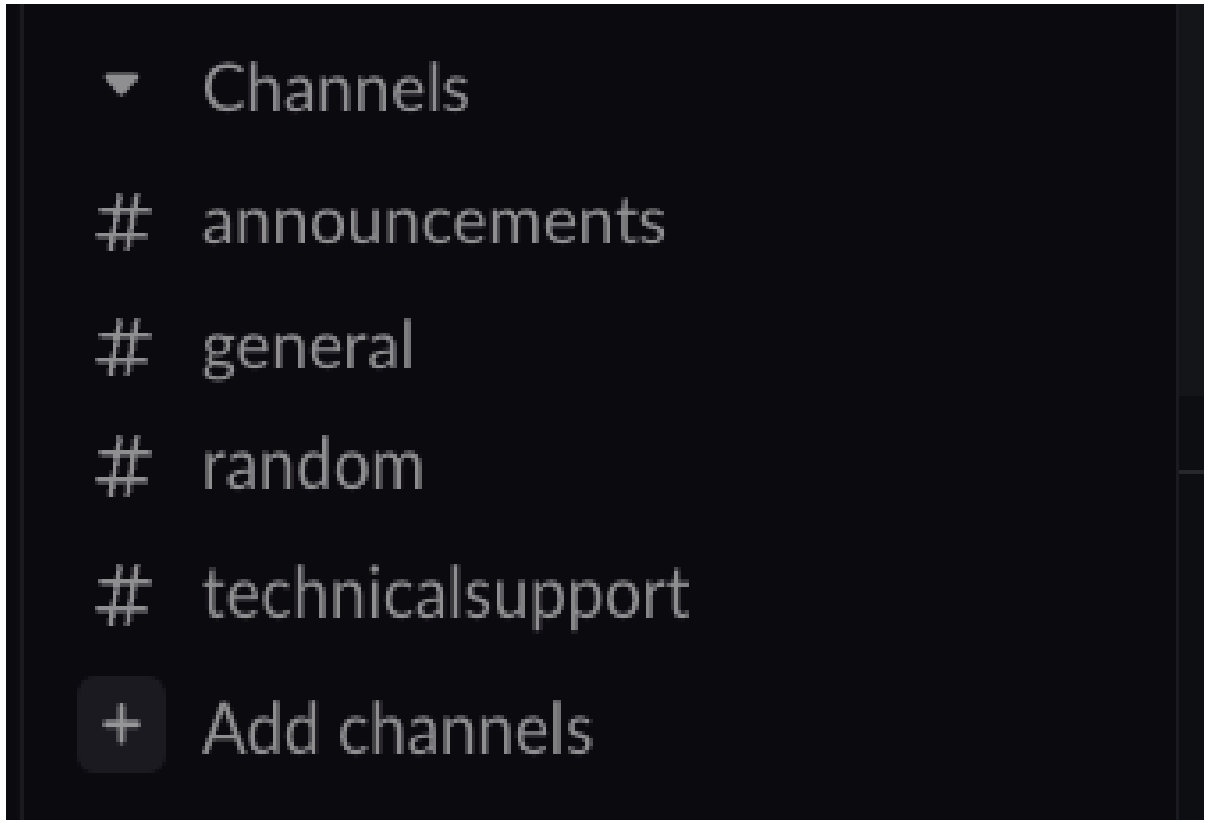
Use threads

- Threads help us create organized discussions around specific messages without adding clutter to a channel.
- You can manage thread notifications.



Use channels properly

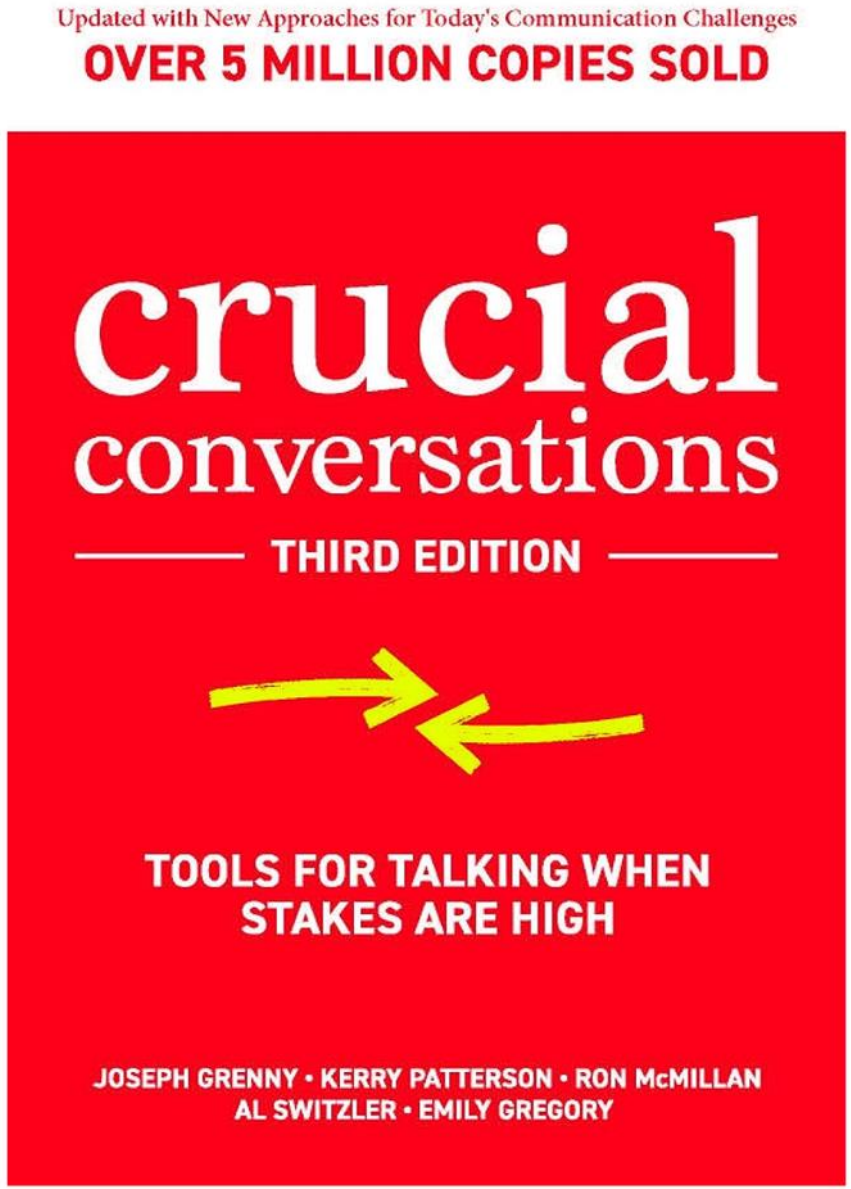
- : Class / homework announcements
- : Administrative / logistics questions
- : Anything! Useful links, memes, …
- : Technical issues (e.g., env setup, errors)



Archive and share the answers

- Avoid duplication!
- You're probably not the only one who's wondering.
- For 313, post your questions in public channels if possible.
  - Feel free to answer too!
- For your team, create a team wiki (e.g., Github project wiki) or shared google document.
- Please do not delete posts

Resolve Conflicts



Communication

Communication

Communication

You can’t solve any problem without communication!

Communication

Communication

Conflict Resolution

- Your goal: Find a solution to the problem and move forward.
  - As a smart person on ”Ted Lasso” once said, “Fight forward, not back.”
- Make sure that everybody works from the same set of facts.
- Establish ground rules for your team’s discussion.
  - Talk about how the situation made you feel. Never presume anything about anyone else.
- Remain calm and rational. If you feel triggered or threatened, extract yourself from the situation, wait an hour to chill out, and then try again.

- If you reach an impasse, talk to your team leader.
- If your team remains in conflict, escalate to your mentor TA.
  - Your mentor TA *will not solve* your problem. They will help *you* to solve your own problems.

Documentation and Visibility

- Document as much about your teamwork process as possible
- If someone is not being a good teammate, please document that
- Make work visible to instructors
- We have made some changes to the course to address specific behaviors
  - Freeloading
  - AI Slop
- Please communicate with us! We want to help you

Team survey

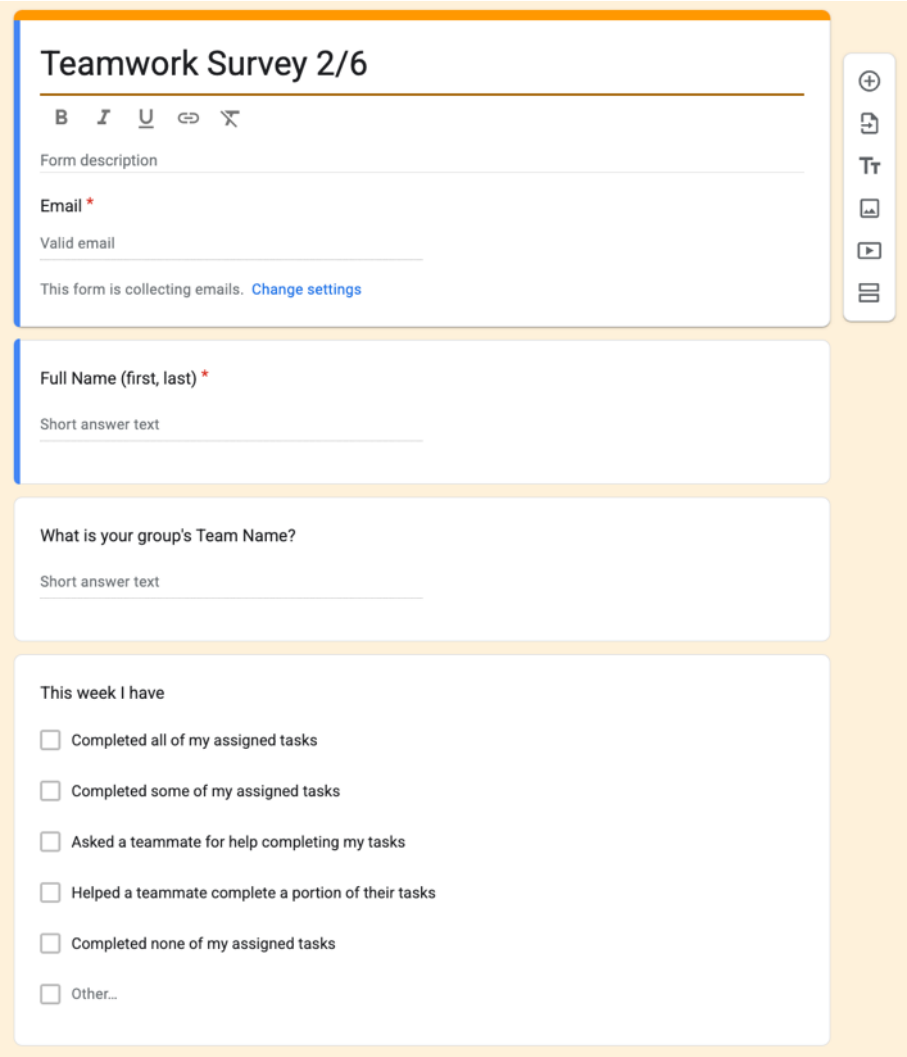
RESEARCH-ARTICLE

Identifying Struggling Teams in Software Engineering Courses Through Weekly Surveys

Authors: Kai Presler-Marshall, Sarah Heckman, Kathryn T. Stolee Authors Info & Claims

SIGCSE 2022: Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1 • February 2022

• Pages 126–132 • https://doi.org/10.1145/3478431.3499367

Generally Due Mondays

- We will assign every week we are working in class (i.e., not spring break)
- Please be honest, we will monitor at first, so this should be early warning, not last resort
- We do not immediately change grades because of results, but may cause us to investigate.
- Please report bad team behaviors