

Design Docs

17-313 Spring 2025

Foundations of Software Engineering

<https://cmu-313.github.io>

Michael Hilton, Austin Henley, and Nadia Nahar

Administrivia

- P2B due Tuesday, Feb 11th, 11:59pm

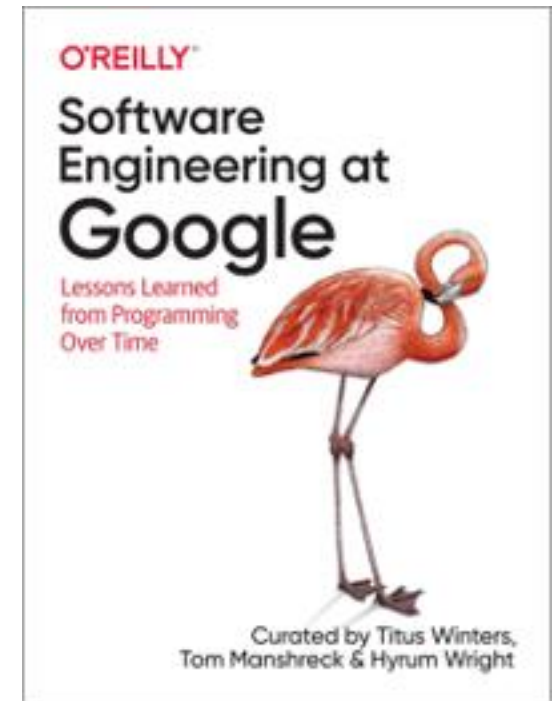
Smoking Section

- Last full row



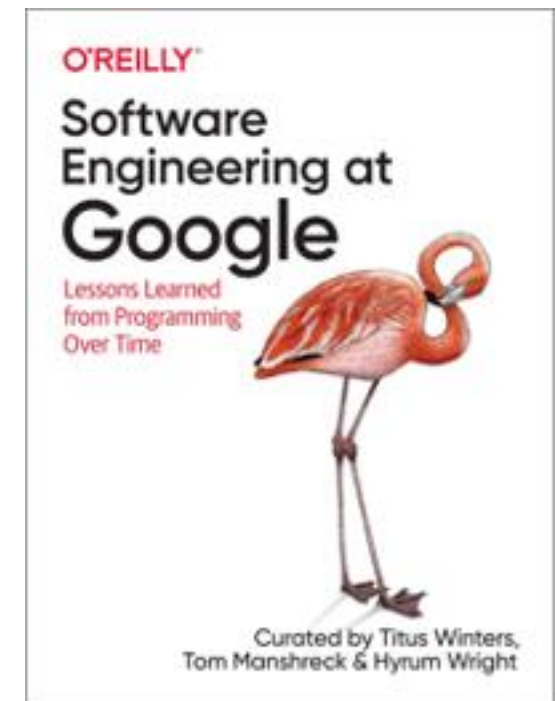
Types of documentation

- Reference documentation (incl. code comments)
- Design documents
- Tutorials
- Conceptual documentation
- Landing pages



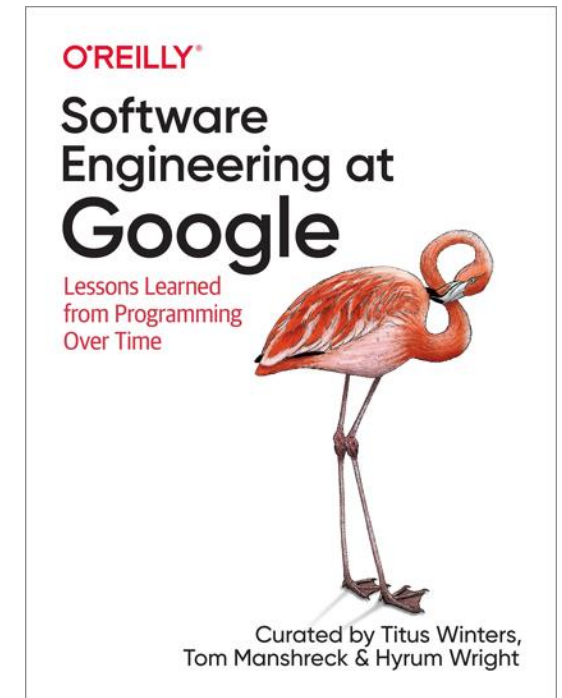
Design documents

- **Code review before there is code!**
- Collaborative (Google Docs)
- Ensure various concerns are covered, such as: security implications, internationalization, storage requirements, and privacy concerns.
- A good design doc should cover
 - Goals and use cases for the design
 - Implementation ideas (not too specific!)
 - Propose key design decisions with an emphasis on their individual tradeoffs



Design Documents

- The *best* design docs suggest design goals, and cover alternative designs, documenting the strengths and weaknesses of each.
- The *worst* design docs accidentally embed ambiguities, which cause implementors to develop contradictory solutions that the customer doesn't want.



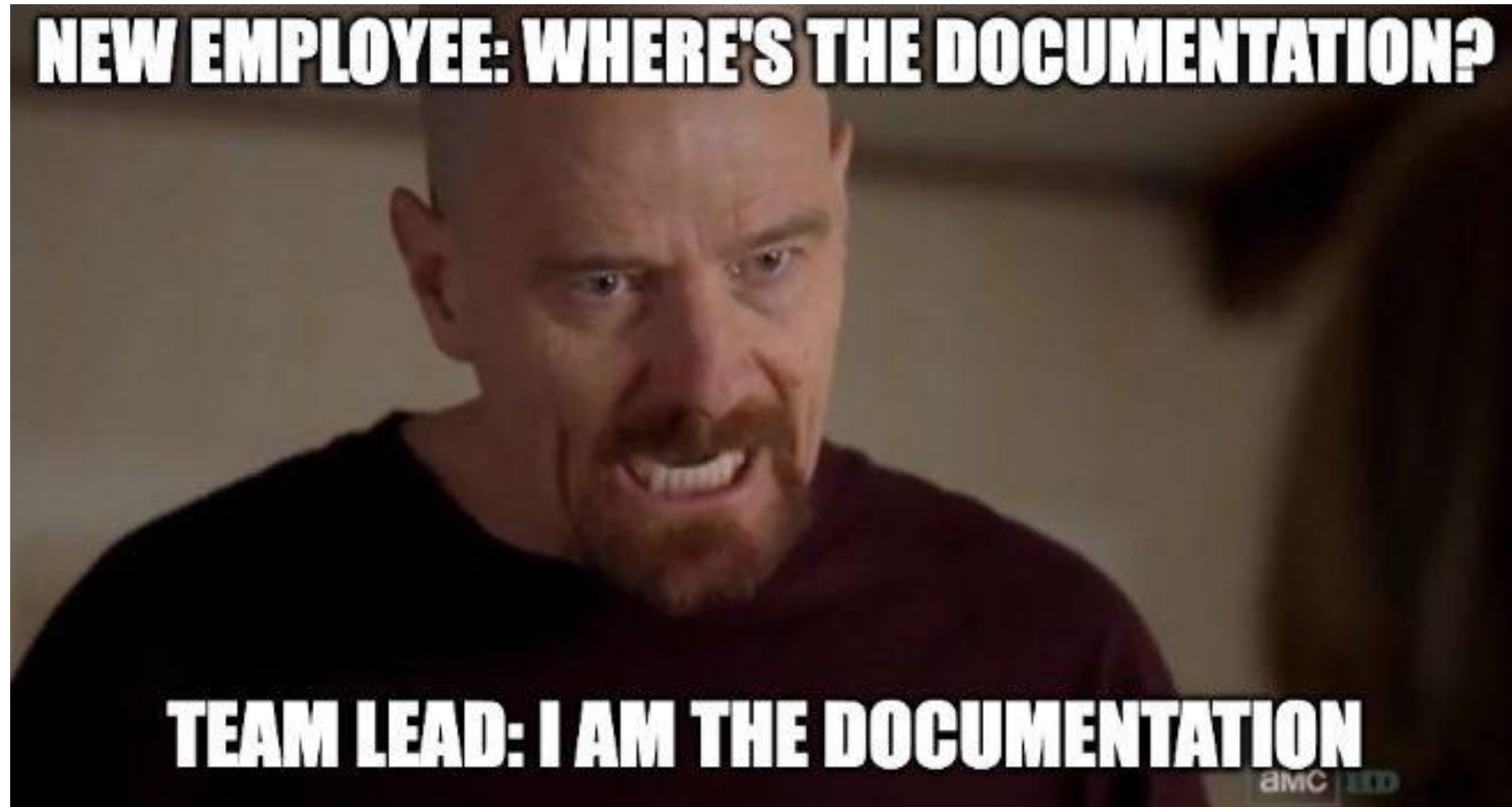
Companies using an RFC-like engineering planning process*

<ul style="list-style-type: none">• Airbnb• Affirm• Algolia• Amazon• AutoScout24• Asana• Atlassian• Blue Apron• Bitrise• Booking.com• Brex• BrowserStack• Canonical• Carousell• Catawiki• Cazoo• Cisco• CockroachDB• Coinbase• Comcast Cable• Container Solutions• Contentful• Couchbase• Criteo• Curve• Daimler• Delivery Hero	<ul style="list-style-type: none">• Doctolib• DoorDash• Dune Analytics• eBay• Ecosia• Elastic• Expedia• Glovo• Gojek• Grab• Faire• Flexport• GitHub• GitLab• GoodNotes• Google• Grafana Labs• GrubHub• HashiCorp• Hopin• Hudl• Indeed• Intercom• LinkedIn• Kiwi.com• Klarna• MasterCard	<ul style="list-style-type: none">• Mews• MongoDB• Monzo• Mollie• Miro• N26• Netlify• Nobl9• Notion• Nubank• Oscar Health• Octopus Deploy• OLX• Onfido• Pave• Peloton• Picnic• PlanGrid• Preply• Razorpay• Reddit• Red Hat• SAP• Salesforce• Shopify• Siemens• Spotify• Square	<ul style="list-style-type: none">• Stripe• Synopsys• Skyscanner• SoundCloud• Sourcegraph• Spotify• Stedi• Stream• SumUp• Thumbtack• TomTom• Trainline• TrueBill• Trustpilot• Twitter• Uber• VanMoof• Virta Health• VMWare• Wayfair• Wave• Wise• WarnerMedia & HBO• Zalando• Zapier• Zendesk• Zillow
---	---	---	--

*not a complete list

pragmaticengineer.com

Why is this important?



Common parts/templates

1. Metadata: *version, date, authors*
2. Executive Summary: *problem being solved, project mission*
3. Stakeholders (and non-stakeholders)
4. Scenarios / User Stories
5. User Experience

1. High-level Requirements: *Functional*
 - Global Requirements: *Quality, Security, Privacy, Ethics*
2. Features and Operations
3. Design Considerations and Tradeoffs
4. Non-Goals
5. Roadmap / Timeline
6. Open Issues

Examples: SourceGraph RFCs

Requests for Comment

<https://about.sourcegraph.com/handbook/communication/rfc>



Meet Cody 

We're building the only AI coding assistant that knows your **entire codebase**

Cody answers technical questions and writes code directly in your IDE, using your code graph for context and accuracy.

Get Started with Cody BETA



By registering, you agree to our [Terms of Service](#) and [Privacy Policy](#).

When to use an RFC:



- You want to frame a problem and propose a solution.
- You want thoughtful feedback from team members on our globally-distributed remote team.
- You want to surface an idea, tension, or feedback.
- You want to define a project or design brief to drive project collaboration.
- You need to surface and communicate around a highly cross-functional decision with our [formal decision-making process](#).

Don't use an RFC when



- You want to discuss personal or sensitive topics one-on-one with another team member.
- You want to make a decision to change something where you are the decider. In the vast majority of cases, creating an RFC to explain yourself will be overkill. RFCs should only be used if a decision explicitly requires one of the bullets in the previous page.

RFC Labels



- **WIP:** The author is still drafting the RFC and it's not ready for review.
- **Review:** The Review label is used when the RFC is ready for comments and feedback.
- **Approved:** When the RFC is for the purpose of making a decision, the Approved label indicates that the decision has been made.
- **Implemented:** When the RFC is for the purpose of making a decision, the Implemented label indicates that the RFC's proposal has been implemented.
- **Closed:** When the RFC is for the purpose of collaboration or discussion but not necessarily to make a decision or propose a specific outcome that will eventually become Implemented, the Closed label indicates that the RFC is no longer an active collaborative artifact.
- **Abandoned:** When the RFC is for the purpose of making a decision, and there are no plans to move forward with the RFC's proposal, the Abandoned label indicates that the RFC has been purposefully set aside.

Observe Sourcegraph Design Docs

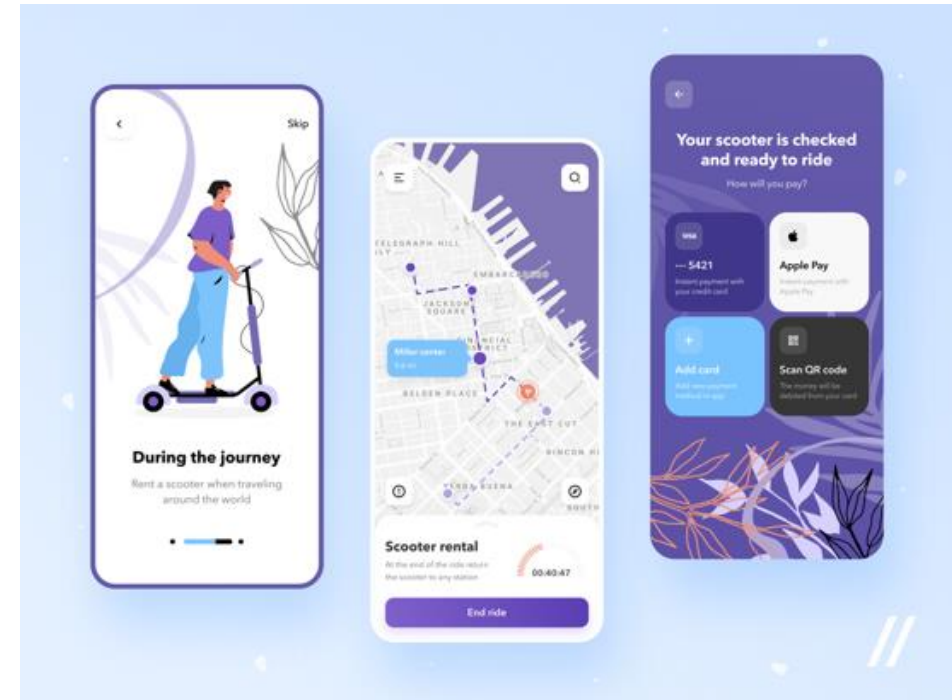
- Docs are publicly available

<https://drive.google.com/drive/folders/1zP3FxdDlcSQGC1qvM9IHZRaHH4I9Jwwa>

- Let's take a look at one!

Exercise

- 4 Proposed Features:
 - Add Payment Method
 - More Secure Authentication
 - Add Android Support
 - Internationalization (i18n)



Time to write our own design docs!

- Divide up into 4 sections –NOTE: you should be signed in w/Andrew to google
- Your mission:
 - Brainstorm a feature to add to a scooter app and write a design spec, together, in real time!
 - Review the design doc, collaborate around text
 - Review another team's design doc, ask questions/leave comments

