PLATINUM

Octopus Deploy

GOLD

auth0
by Okta

elabor8®

GitHub

Insight

Lynkz

MAKERX

Mantel group

PATIENT ZERO

technologyone
Transforming business, making life simple

Telstra Purple

VALD

workday®

SILVER

Progress®

Seq
Machine data, for humans.

Loan Market Group.

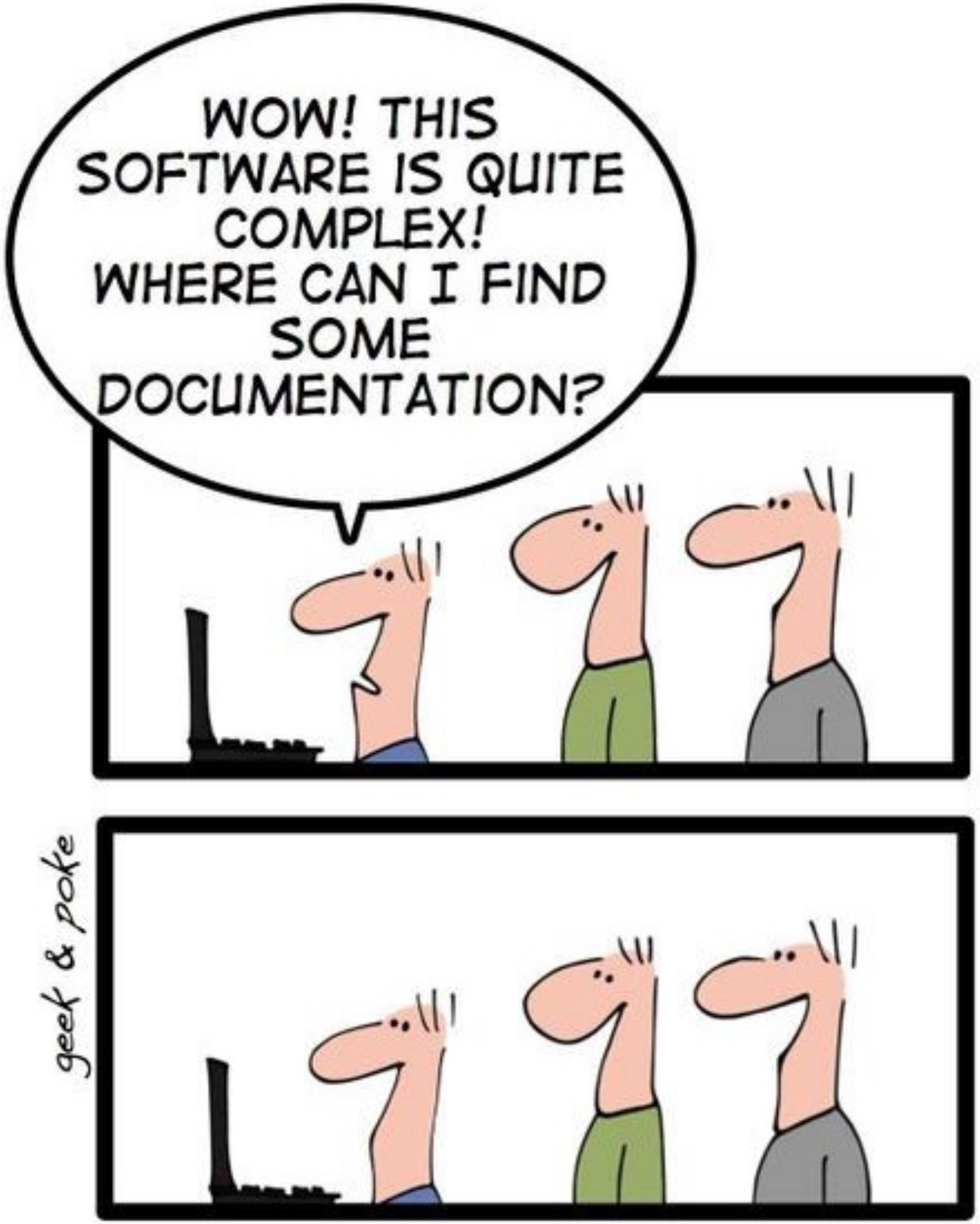
SixPivot

SOULSOLUTIONS

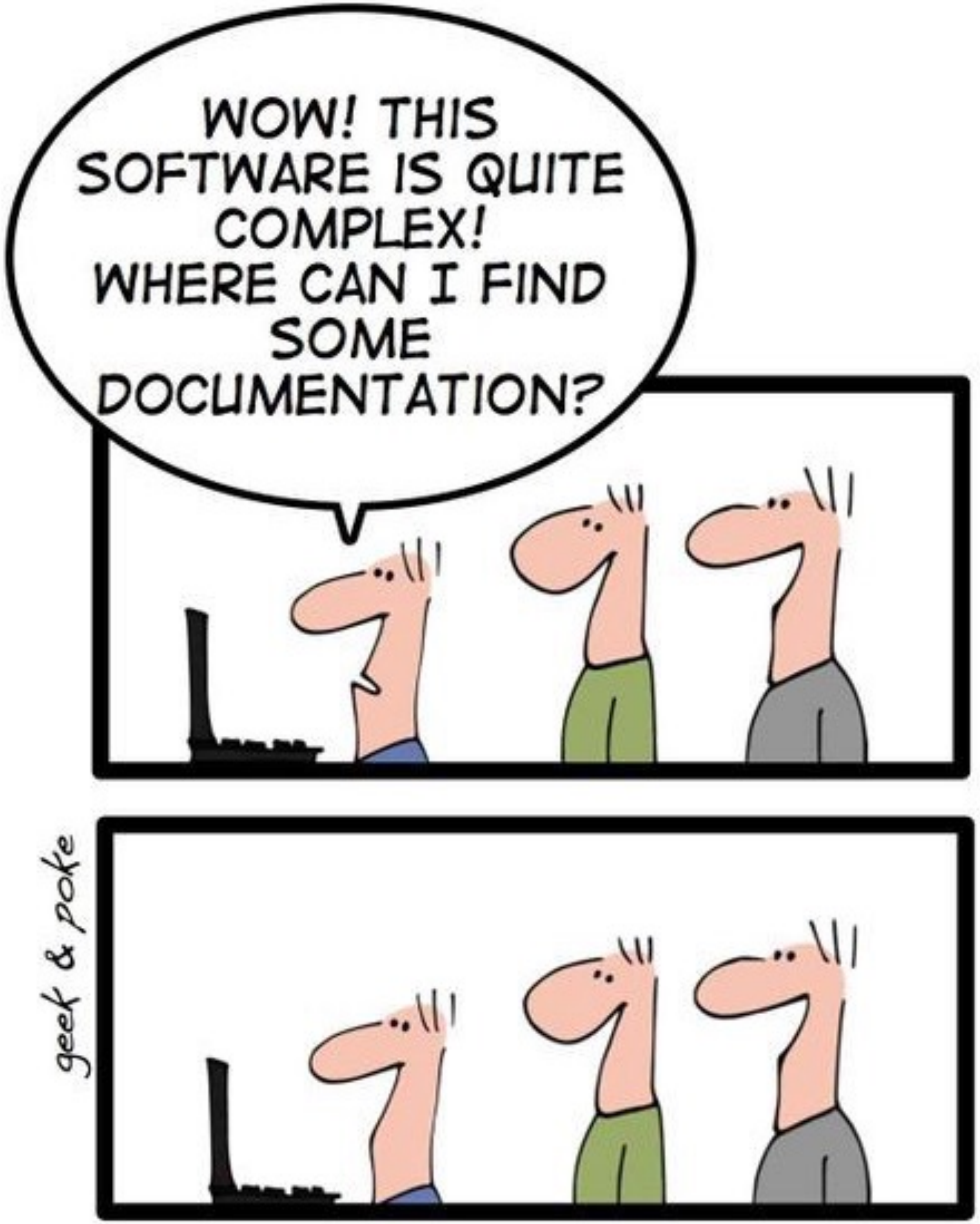
STANDARD

Calyptia

Documentation for Developers

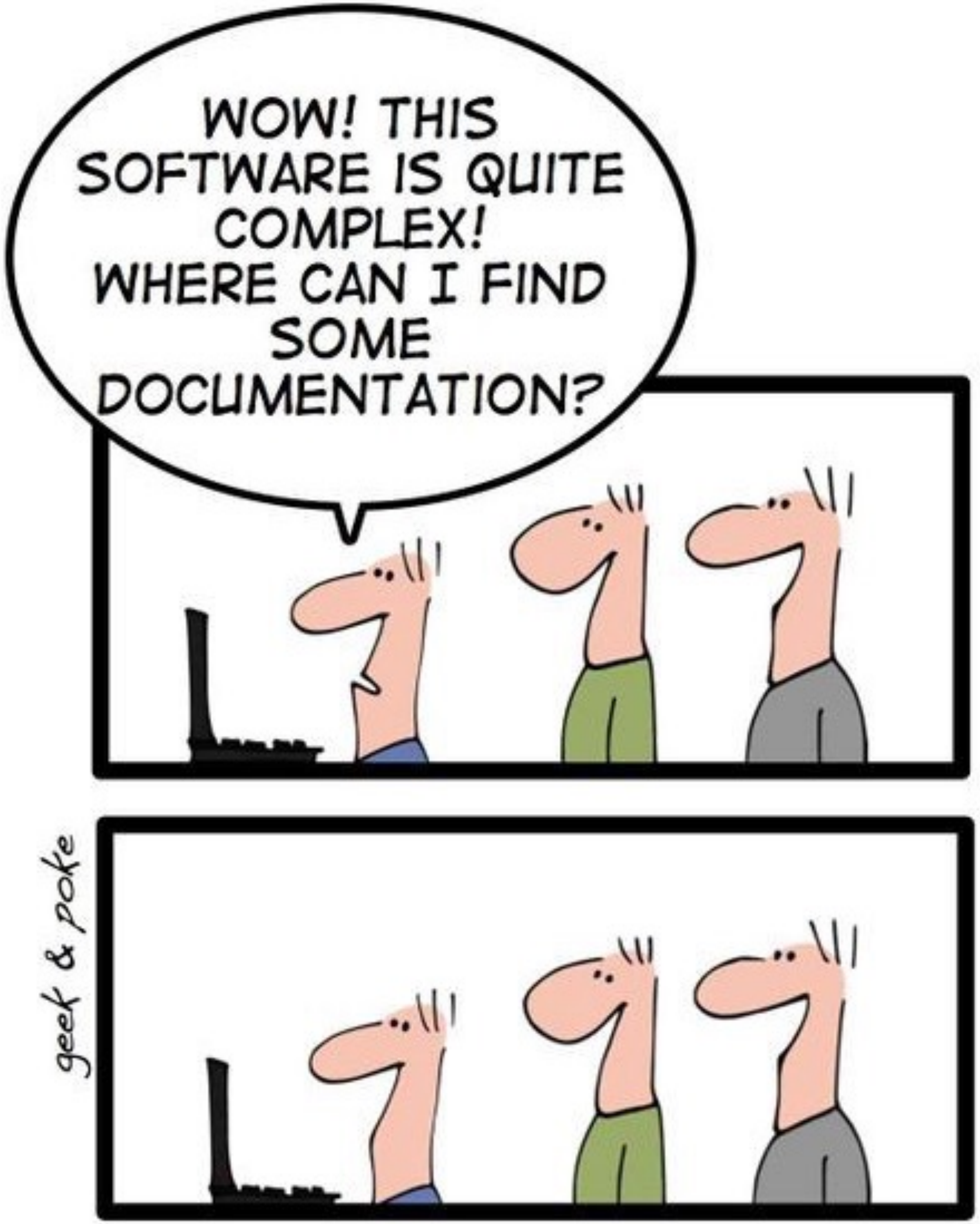

WOW! THIS SOFTWARE IS QUITE COMPLEX! WHERE CAN I FIND SOME DOCUMENTATION?
geek & poke

~~Documentation for Developers~~


WOW! THIS SOFTWARE IS QUITE COMPLEX! WHERE CAN I FIND SOME DOCUMENTATION?
geek & poke

Developers
Do
Documentation

BRISBANE


WOW! THIS SOFTWARE IS QUITE COMPLEX! WHERE CAN I FIND SOME DOCUMENTATION?
geek & poke

https://geekandpoke.typepad.com/geekandpoke/2009/12/we-all-had-our-dreams.html

About Me

Currently

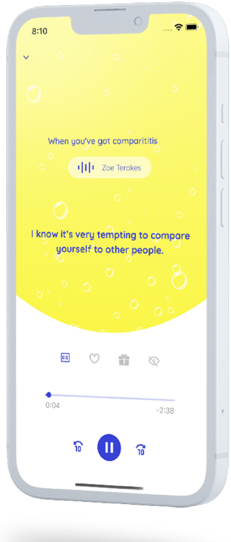
- Principal Consultant & Practice Director
- CTO

Previously

- Various Enterprise, Startup and Freelance

SixPivot

HEY lemonade



When you've got comparititis

Zoe Terokes

I know it's very tempting to compare yourself to other people.

Why do we document things?

- Communicate complex systems easier
- Reduce cognitive load
- Developer On-boarding
- Software and Solution Architecture
- Architectural Guidelines and Principles
- Decisions and rationale
- Disaster recovery

Why *DON’T* we document things?

- Documentation is boring
- Often requires special tooling, or licenses
- Diagramming tools are often hard to version control
- Perfect is the enemy of done

Traditional approach

- Wikis, Confluence
- Word Documents
- Specialised tools

- Visio
- Draw.io
- Paint
- Illustration Tools

My ethos about documentation?

A possible solution?

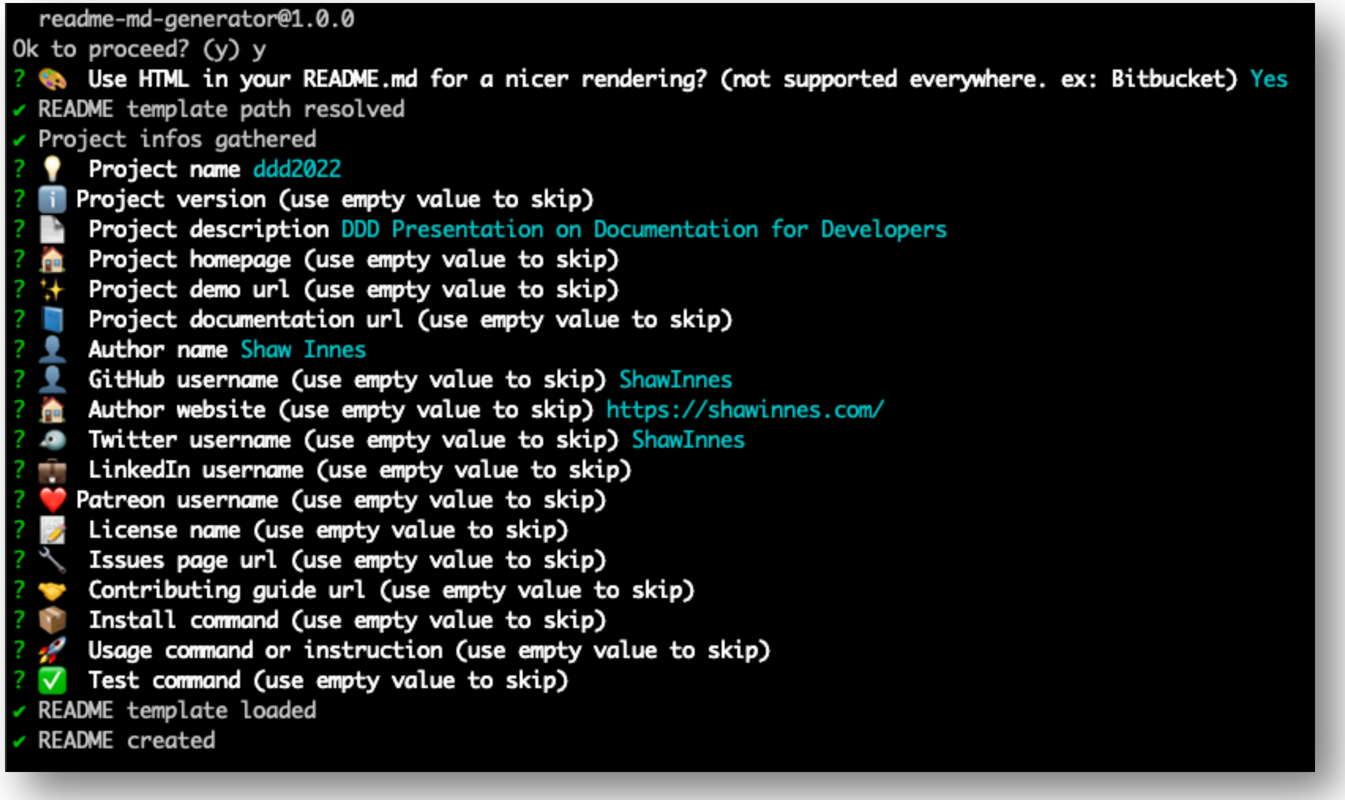
- Be pragmatic
- Adopt “Documentation as Code”
- Favour the separation of structure from presentation
- Minimal requirement for extra tooling

Pros and cons

- Version controllable / diffable
- Simple to use/learn
- Using a DSL vs just diagrams means there is context
- Cross platform
- Minimal tooling
- Can generate images / pdf / ppt
- Minimal licensing

- Downside is the learning curve
- Code sprawl
- Harder to search & aggregate
- Less accessible to non-tech folk

Markdown Readmes



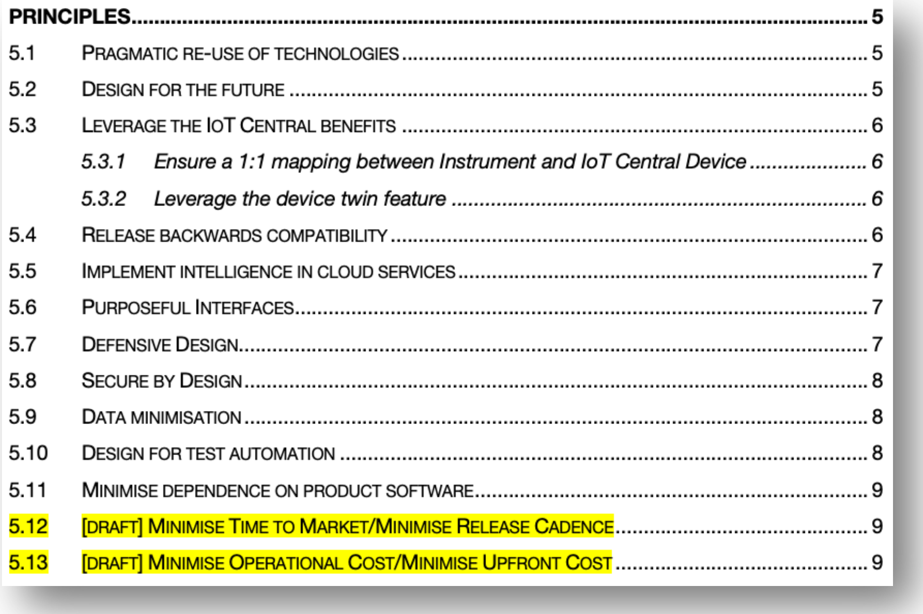
```
  readme-md-generator@1.0.0
Ok to proceed? (y) y
? 🎨  Use HTML in your README.md for a nicer rendering? (not supported everywhere. ex: Bitbucket) Yes
✔ README template path resolved
✔ Project infos gathered
? 💡  Project name ddd2022
? ℹ️ Project version (use empty value to skip)
? 📄  Project description DDD Presentation on Documentation for Developers
? 🏠  Project homepage (use empty value to skip)
? ✨  Project demo url (use empty value to skip)
? 📘  Project documentation url (use empty value to skip)
? 👤  Author name Shaw Innes
? 👤  GitHub username (use empty value to skip) ShawInnes
? 🏠  Author website (use empty value to skip) https://shawinnes.com/
? 🐦  Twitter username (use empty value to skip) ShawInnes
? 💼  LinkedIn username (use empty value to skip)
? ❤️ Patreon username (use empty value to skip)
? 📝  License name (use empty value to skip)
? 🔧  Issues page url (use empty value to skip)
? 🤝  Contributing guide url (use empty value to skip)
? 📦  Install command (use empty value to skip)
? 🚀  Usage command or instruction (use empty value to skip)
? ✅  Test command (use empty value to skip)
✔ README template loaded
✔ README created
```

- A quick intro for onboarding
- Use a template for consistency
- Google for examples
- Customise for your organization
- readme-md-generator

https://github.com/kefranabg/readme-md-generator

Architectural Guidance

- Architecture Principles
  - Assumptions
  - Make it easier for developers to make decisions
  - Could this meeting have been a Principle?



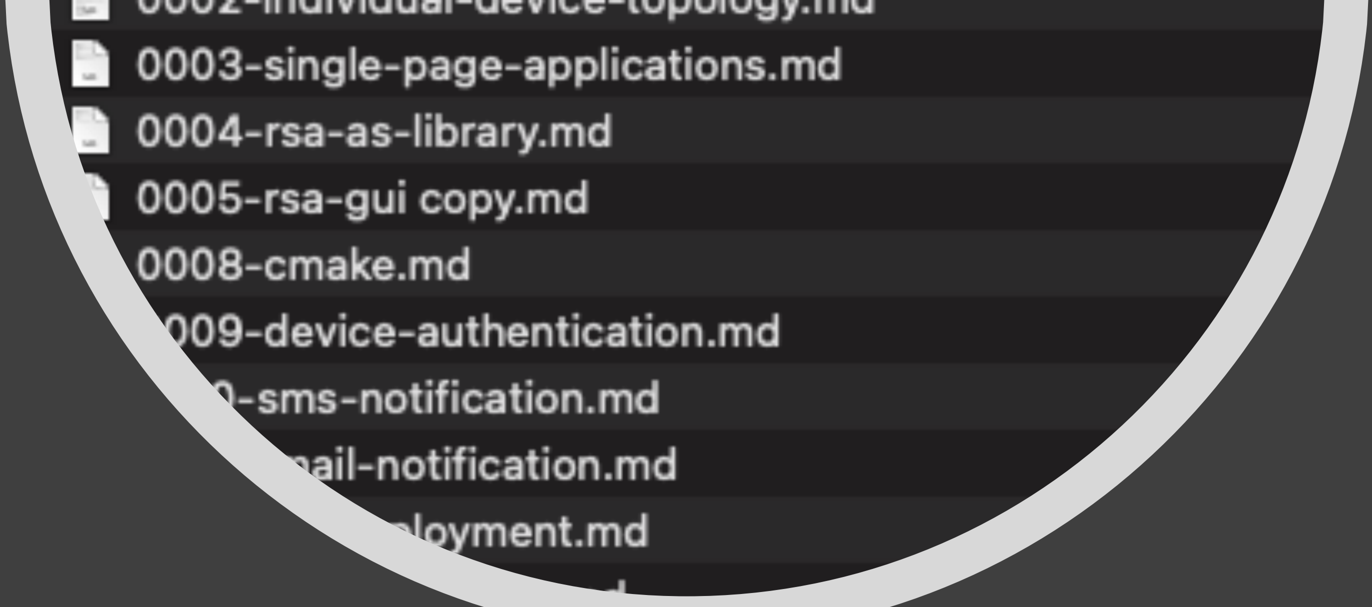
Principles 5

Well-defined principles

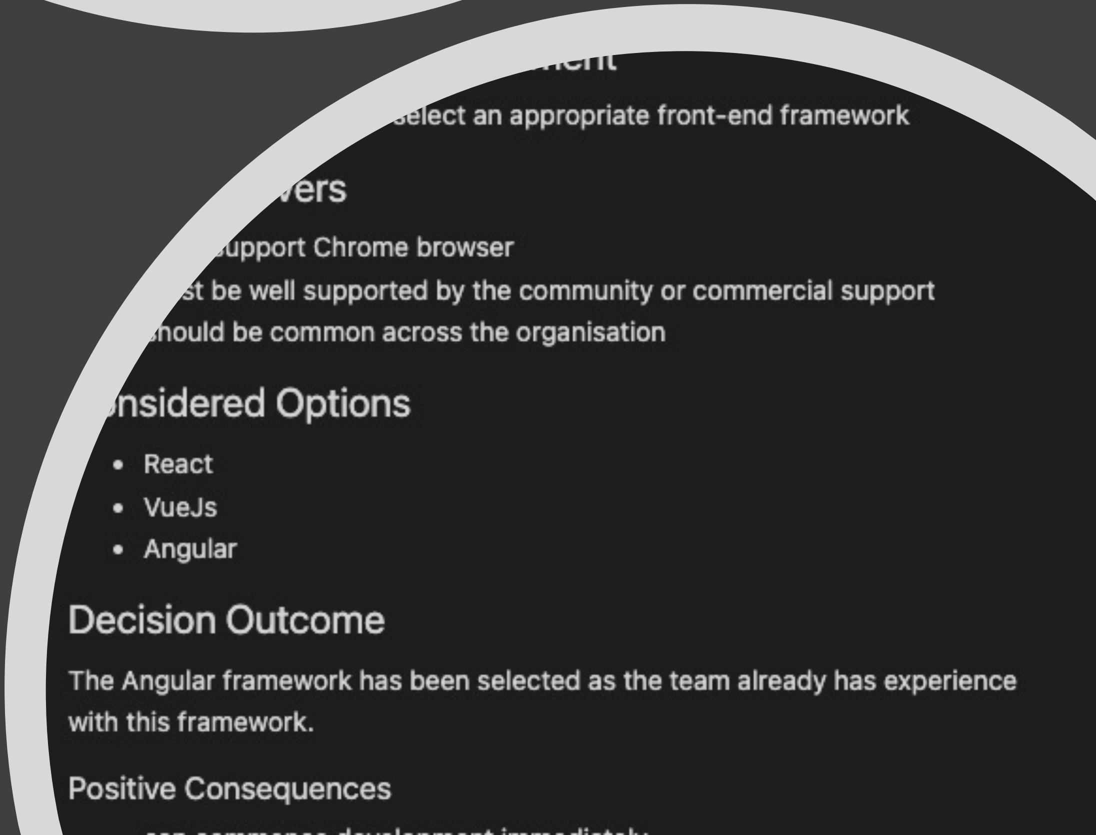
- Name: Succinct, easy to remember, and carries the essence of the rule (few words)
- Statement: Unambiguous statement defining the rule (1 to 3 sentences)
- Rationale: Why is this rule important, particularly in terms of business benefits.
- Implications: Identifies explicitly how the rule affects the business, teams, etc.

Tracking Decisions

- [Markdown Decision Records (MADR)]
    - Context/Problem
    - Considered Options
    - Decision
    - Consequences



0002-individual-device-topology.md
0003-single-page-applications.md
0004-rsa-as-library.md
0005-rsa-gui copy.md
0008-cmake.md
009-device-authentication.md
0-sms-notification.md
ail-notification.md
loyment.md



elect an appropriate front-end framework

vers

upport Chrome browser
st be well supported by the community or commercial support
hould be common across the organisation

nsidered Options

- React
- VueJs
- Angular

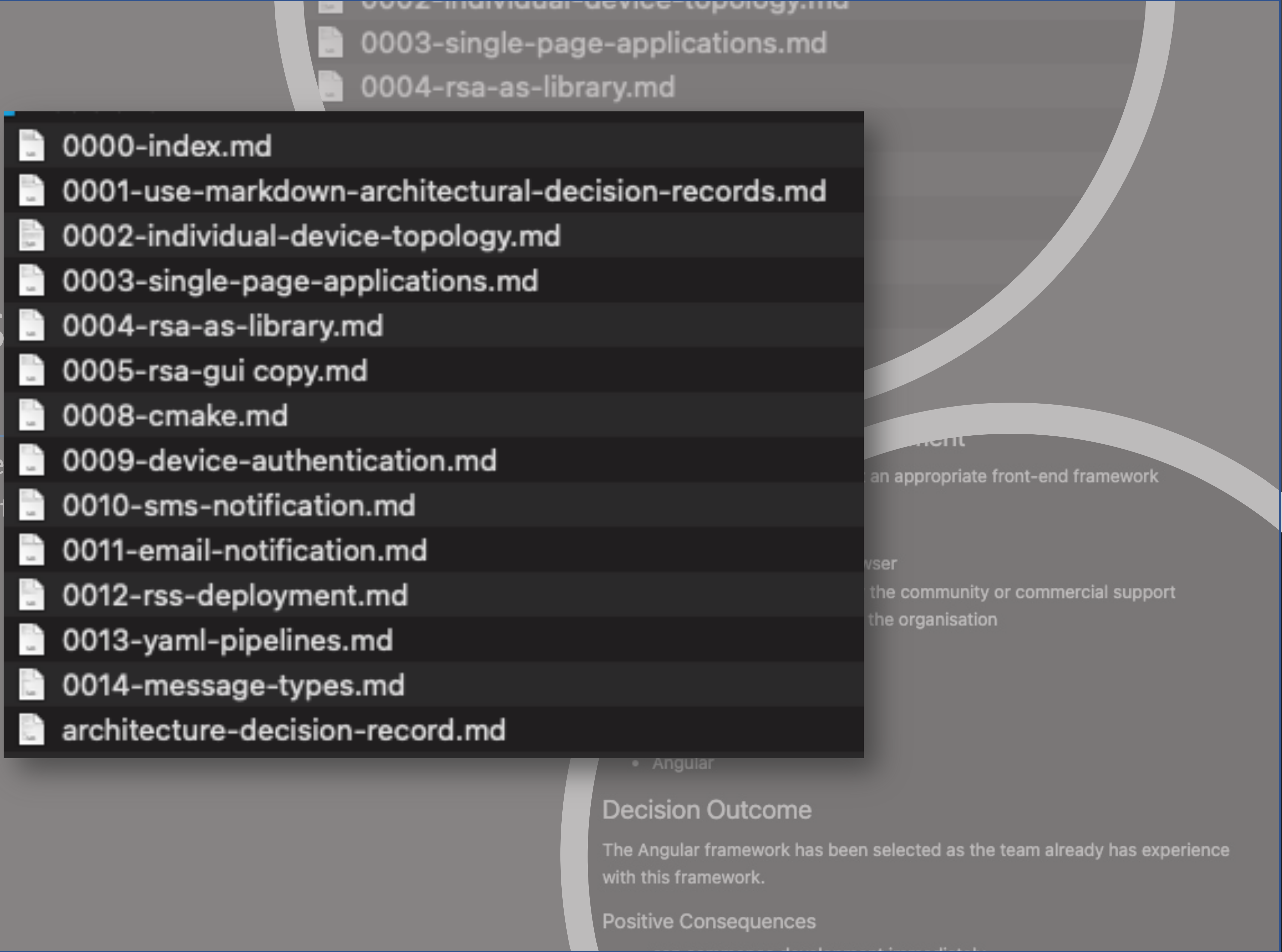
Decision Outcome

The Angular framework has been selected as the team already has experience with this framework.

Positive Consequences

Tracking Decisions

- Markdown Decision
  - Context/Proble
  - Considered Opt
  - Decision
  - Consequences



```
0000-index.md
0001-use-markdown-architectural-decision-records.md
0002-individual-device-topology.md
0003-single-page-applications.md
0004-rsa-as-library.md
0005-rsa-gui copy.md
0008-cmake.md
0009-device-authentication.md
0010-sms-notification.md
0011-email-notification.md
0012-rss-deployment.md
0013-yaml-pipelines.md
0014-message-types.md
architecture-decision-record.md
```

0003-single-page-applications.md

0004-rsa-as-library.md

an appropriate front-end framework

wser

the community or commercial support

the organisation

- Angular

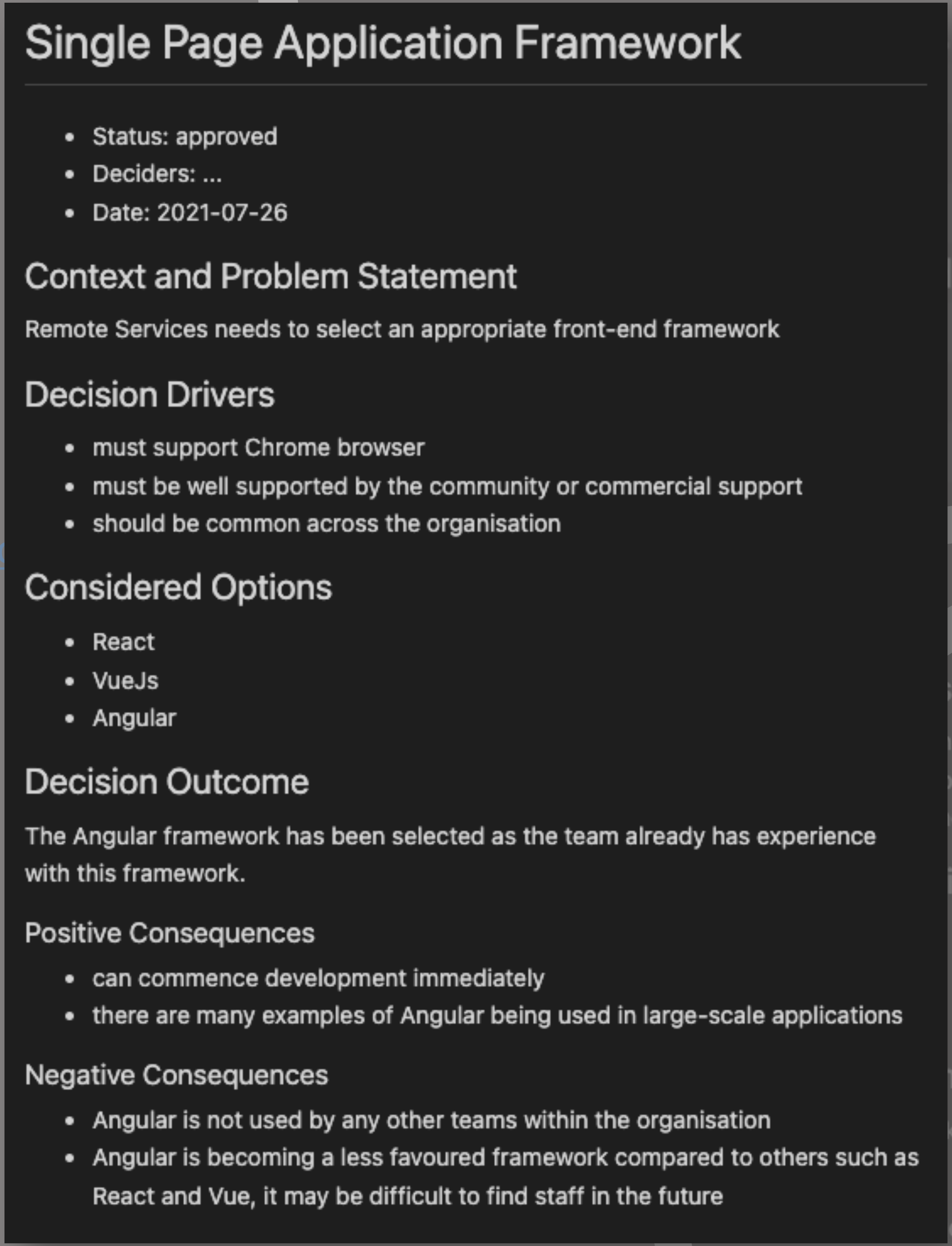
Decision Outcome

The Angular framework has been selected as the team already has experience with this framework.

Positive Consequences

Tracking Decisions

- Markdown Decision Recor
  - Context/Problem
  - Considered Options
  - Decision
  - Consequences



Single Page Application Framework

- Status: approved
- Deciders: ...
- Date: 2021-07-26

Context and Problem Statement

Remote Services needs to select an appropriate front-end framework

Decision Drivers

- must support Chrome browser
- must be well supported by the community or commercial support
- should be common across the organisation

Considered Options

- React
- VueJs
- Angular

Decision Outcome

The Angular framework has been selected as the team already has experience with this framework.

Positive Consequences

- can commence development immediately
- there are many examples of Angular being used in large-scale applications

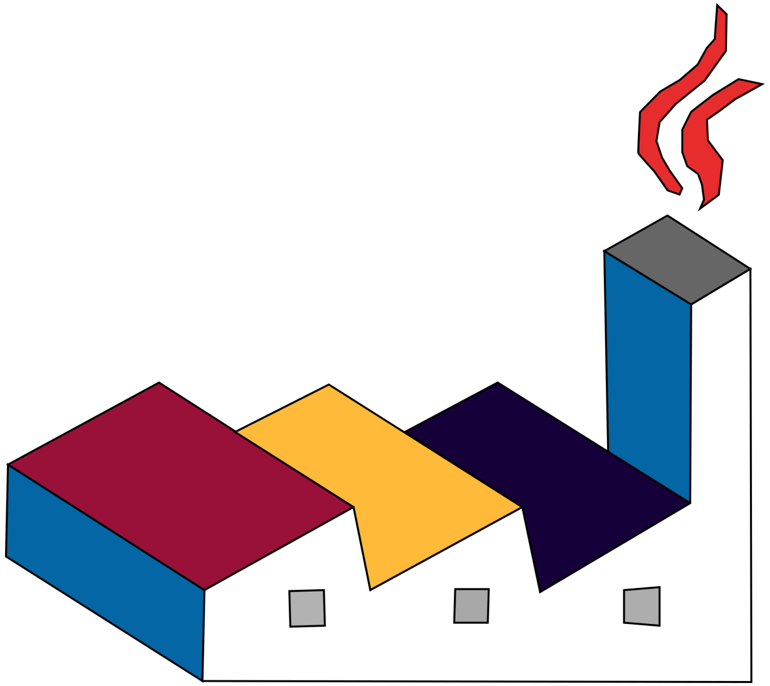
Negative Consequences

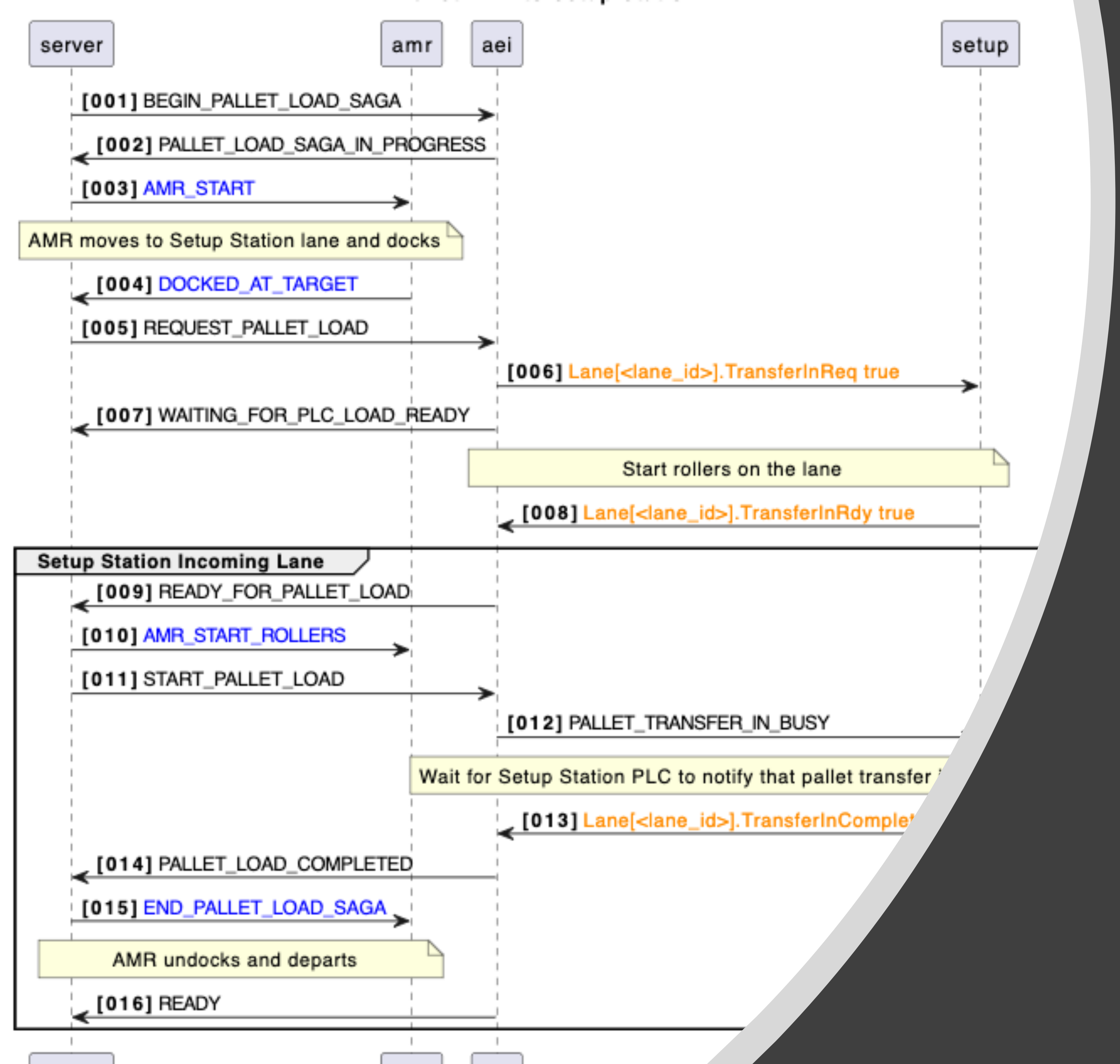
- Angular is not used by any other teams within the organisation
- Angular is becoming a less favoured framework compared to others such as React and Vue, it may be difficult to find staff in the future

Diagramming - PlantUML

- Text-based diagramming
- Support for multiple diagram types
  - Sequence diagram
  - Class diagram
  - Activity diagram
  - Entity Relationship diagram
- Extensible with Libraries
  - Azure, Kubernetes, AWS, C4…
- Tooling
  - Visual Studio Code + Extensions
  - Supported by JetBrains and others also
  - Supported in various Markdown wikis

https://plantuml.com/
https://crashedmind.github.io/PlantUMLHitchhikersGuide/
https://github.com/plantuml/plantuml-stdlib





[001] BEGIN_PALLET_LOAD_SAGA

[002] PALLET_LOAD_SAGA_IN_PROGRESS

[003] AMR_START

AMR moves to Setup Station lane and docks

[004] DOCKED_AT_TARGET

[005] REQUEST_PALLET_LOAD

[006] Lane[<lane_id>].TransferInReq true

[007] WAITING_FOR_PLC_LOAD_READY

Start rollers on the lane

[008] Lane[<lane_id>].TransferInRdy true

Setup Station Incoming Lane

[009] READY_FOR_PALLET_LOAD

[010] AMR_START_ROLLERS

[011] START_PALLET_LOAD

[012] PALLET_TRANSFER_IN_BUSY

Wait for Setup Station PLC to notify that pallet transfer i

[013] Lane[<lane_id>].TransferInComplet

[014] PALLET_LOAD_COMPLETED

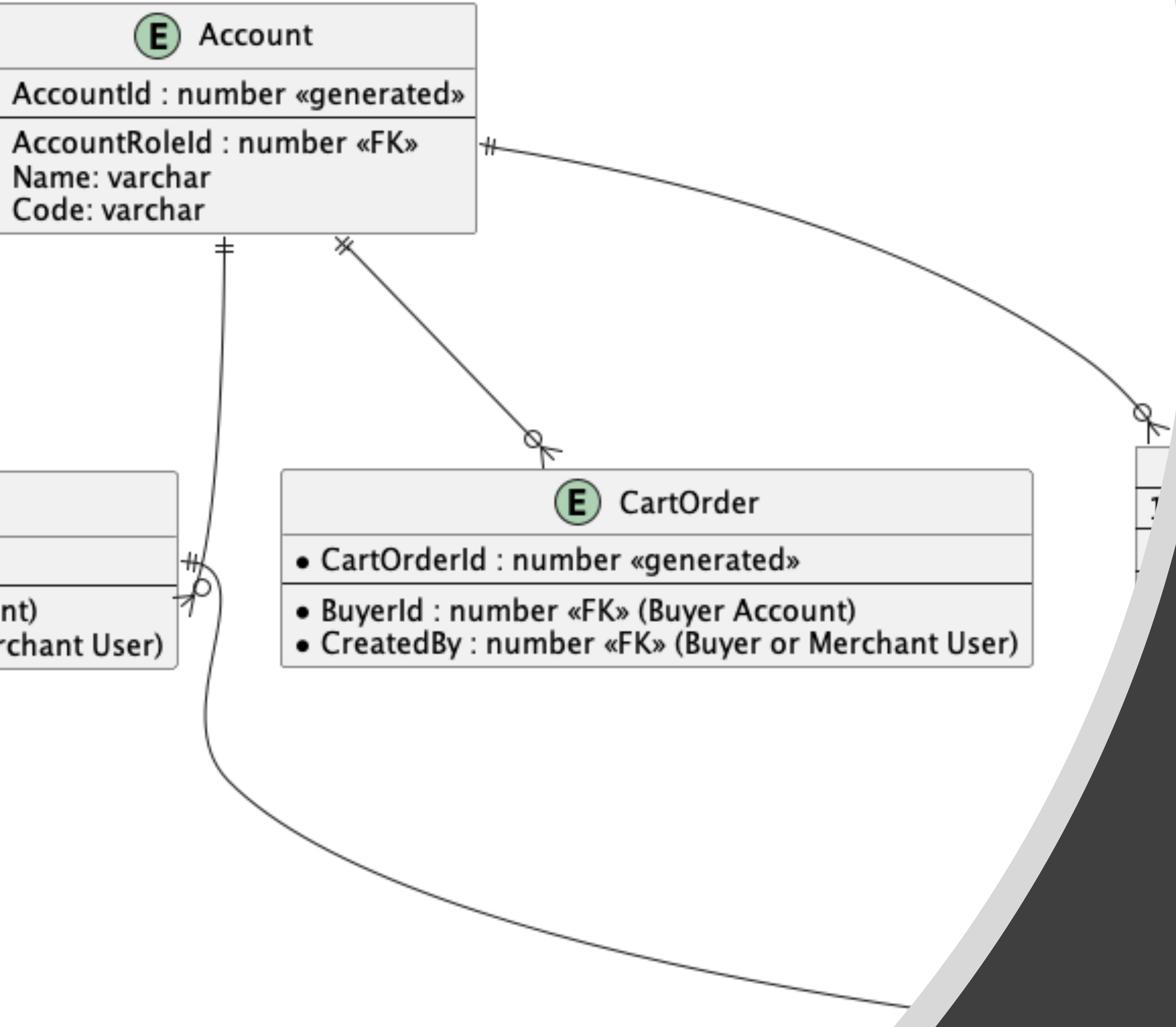
[015] END_PALLET_LOAD_SAGA

AMR undocks and departs

[016] READY

PlantUML Demo

Sequence Diagram



E Account

AccountId : number «generated»

AccountRoleId : number «FK»
Name: varchar
Code: varchar

E CartOrder

- CartOrderId : number «generated»

- BuyerId : number «FK» (Buyer Account)
- CreatedBy : number «FK» (Buyer or Merchant User)

nt)
rchant User)

PlantUML Demo

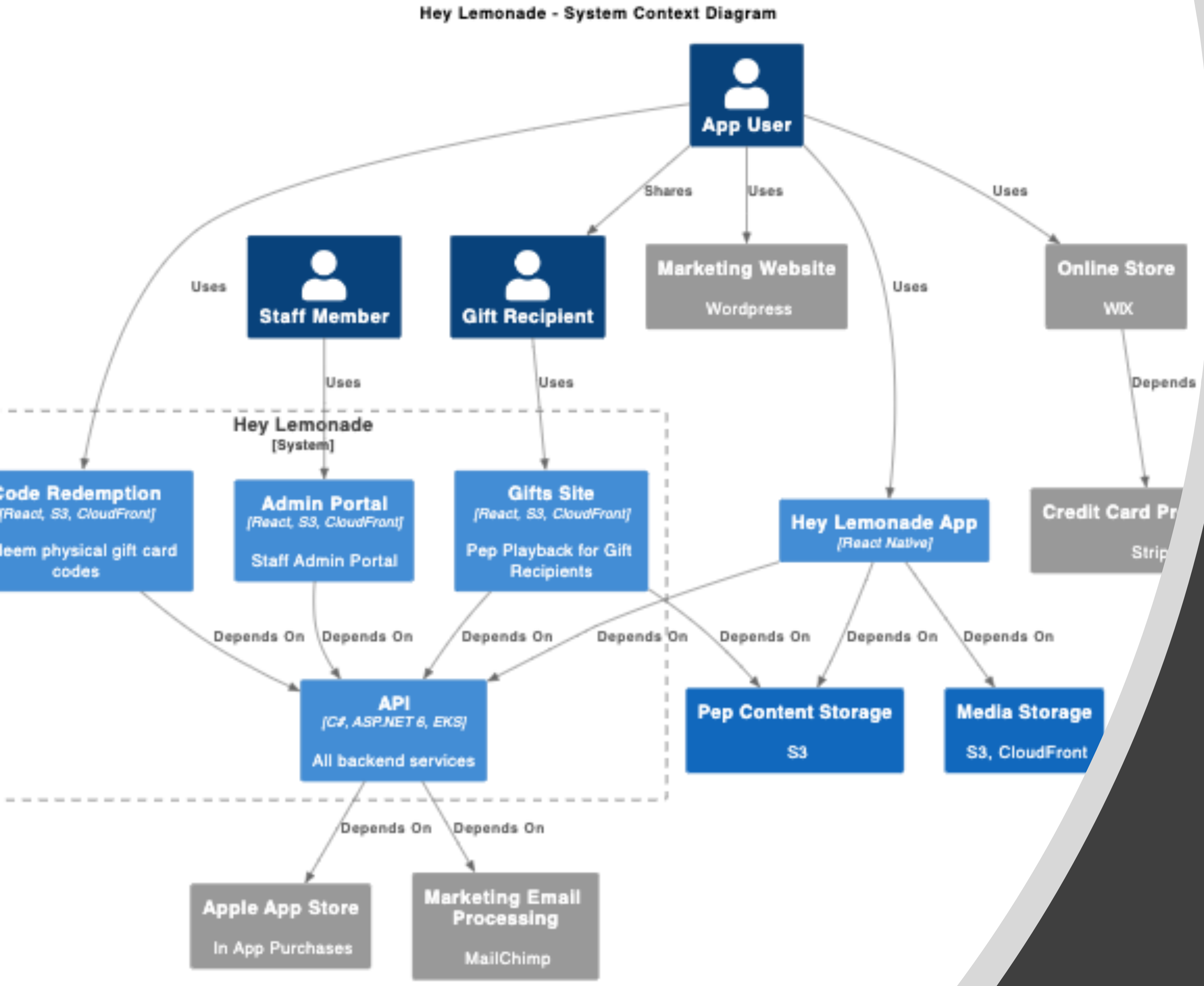
Entity Relationship Diagram

C4 Model

- Created by Simon Brown (see YOW 2022)
- Helps avoid overly-complex diagrams

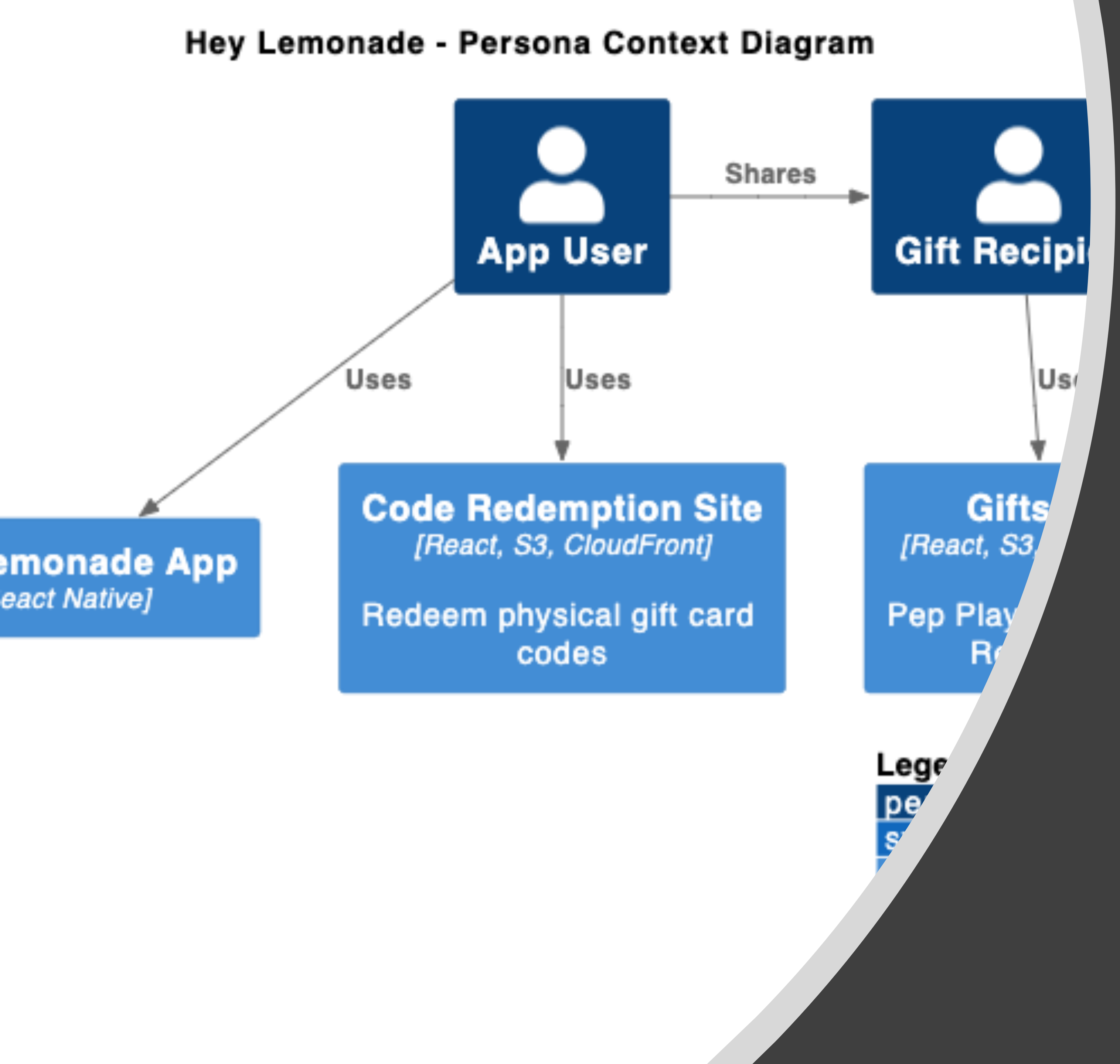
- Context (level 1): system scope, users and other systems
- Container (level 2): represents an application or a data store
  - web application, mobile app, serverless function, embedded device
- Component (level 3): internals of a container
- Code (level 4): class-level decomposition

https://c4model.com/
https://adrianvlupu.github.io/C4-Builder/
https://github.com/plantuml-stdlib/C4-PlantUML
https://c4model.com/review/



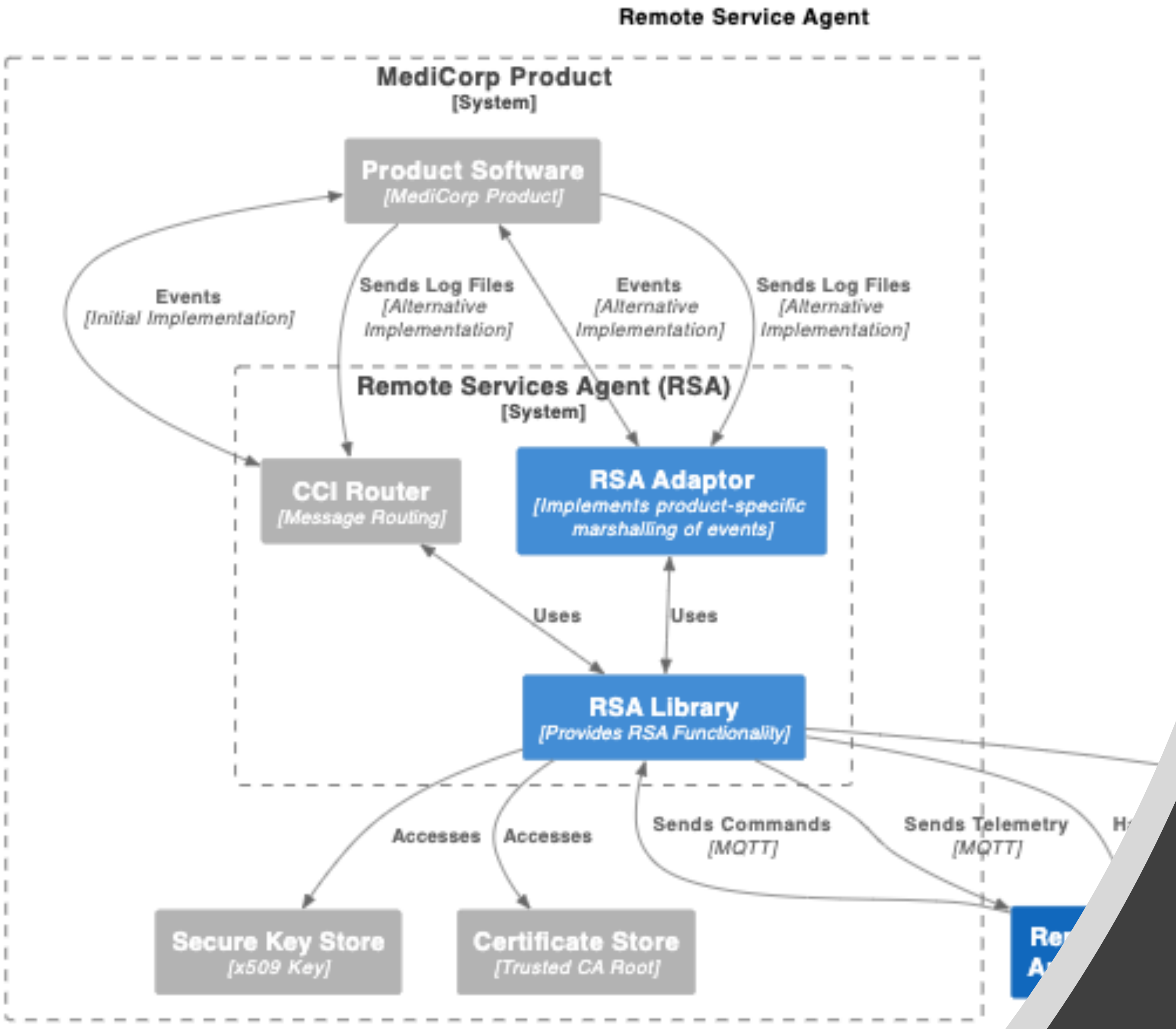
C4 Model Demo

Bad Context Diagram



C4 Model Demo

Good Context Diagram



C4 Model & Markdown Demo

Context & Container Diagrams

Bringing it all together

- Roll your own scripts
- Markdown front-matter for metadata
- Output to PDF in CI/CD pipelines
- Tools like C4 Builder

https://adrianvlupu.github.io/C4-Builder/
https://pandoc.org/

Structurizr

- Tooling to extend on PlantUML & C4 & ADRs
- More control over rendering
- Linking between different elements
- Model and element reuse
- Team sharing

(Also by Simon Brown)

https://structurizr.org/
https://structurizr.com/
https://structurizr.com/help/decision-log

How do you encourage developers to document their code inline? So that code doco can be rendered in the build pipeline.

It’s not something I actively encourage, but it depends on the type of software you’re writing. As with most of the stuff I’ve presented here, be pragmatic.

Through the pull request process is a good way to ensure that code is documented, if required. I would suggest documenting code inline if the intent of the code can’t be clearly understood by a mid-level developer.

However, if you’re creating a library or API that is to be consumed by 3rd parties, then inline documentation might be more useful. In this case I’d look at using convention tests or some other process in your CI build to break the build if a developer adds new code without the expected documentation.

Can you recommend a good starting template for an Architectural Guidance doc?

Software Architecture for Developers – Simon Brown

http://leanpub.com/software-architecture-for-developers

Creating Software with Modern Diagramming –Ashley Peacock

https://pragprog.com/titles/apdiag/creating-software-with-modern-diagramming-techniques/

As much as I would hate to recommend TOGAF…

https://pubs.opengroup.org/architecture/togaf8-doc/arch/chap29.html

Any alternatives to PlantUML?
Have you looked at mermaid?

Yep, I’ve used mermaid. It’s a good option because it seems to be natively supported by more wikis and github etc.

The reason I didn’t talk about mermaid much is because it’s a bit limited in functionality compared to PlantUML. Having said that, if it’s supported by your platforms then mermaid is a good option. It’s also relatively straight forward to move from mermaid to PlantUML later if you need the extended functionality.

From a documentation point of view, a couple of people suggested Asciidoc during the Q&A session. I haven’t looked into this heavily but I think it’s worth a look as well, especially for creating large-scale user documentation.

Another project I recently came across is kroki.io – it seems to be an amalgamation of a bunch of other text based documentation projects.

Any tools for exporting tables from Excel to markdown?

There’s actually a Visual Studio Code extension for that. If you just want to copy/paste tables this is a good option:

https://marketplace.visualstudio.com/items?itemName=csholmq.excel-to-markdown-table

I’m not sure about what you’d do if you wanted to extract a table as part of a CI process though.

What would you recommend for BPMN?

Good question. I haven't done a lot with BPMN.

When I did have to view some BPMN documents recently I used Visual Studio Code plugin to visualise the diagrams, but that's about the extent of my experience unfortunately.

I know PlantUML doesn't currently support it because there's an open issue discussing this.

Can you do cloud infrastructure diagram?

Also, can you add images (logos)?

Absolutely. There are a ton of PlantUML libraries for this:

https://github.com/plantuml-stdlib/Azure-PlantUML
https://github.com/awslabs/aws-icons-for-plantuml
https://github.com/dcasati/kubernetes-PlantUML

https://crashedmind.github.io/PlantUMLHitchhikersGuide/aws/aws.html