Mentimeter

Software Management Plan

1

Mentimeter

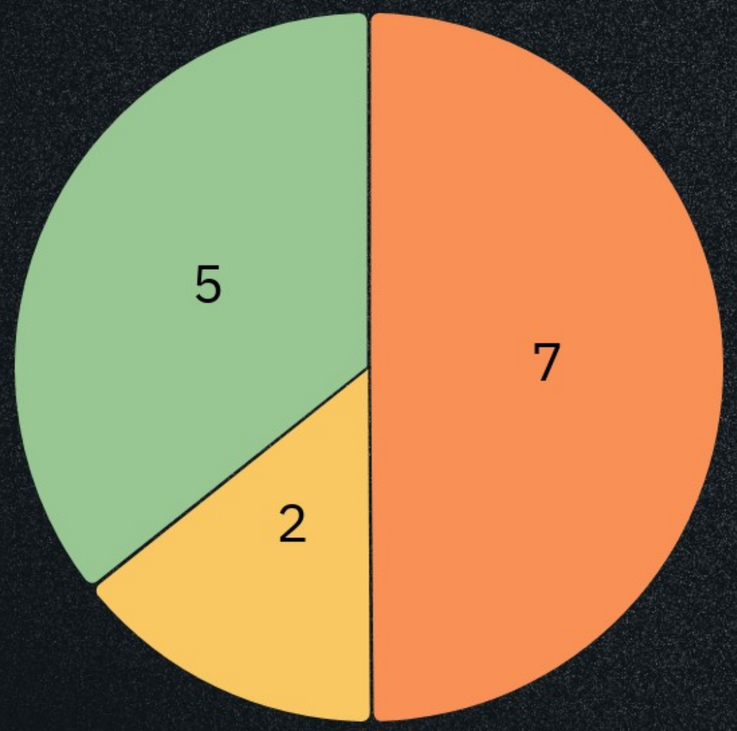
Instructions

Mentimeter

Let's start with two questions...

Mentimeter

Have you, or your team, ever developed a software for your research?

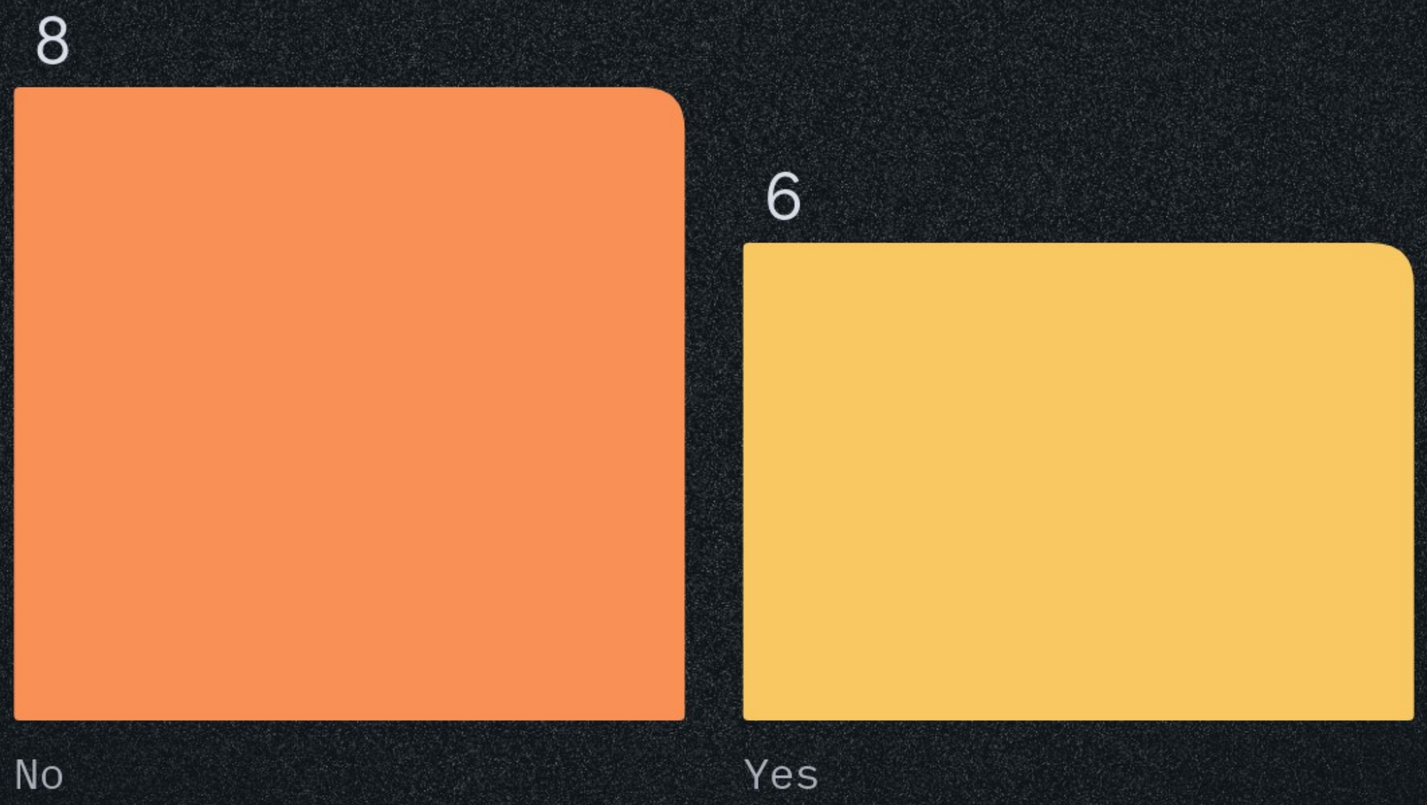


7	I (we) have
2	I am (we are) in the middle of it now
5	I am (we are) planning to do it as part of a future project
0	None of the above

12

Mentimeter

Have you ever written a project management plan?



14

Mentimeter

What is a software Management plan [SMP]?

Framework that outlines ***how software development, releases, maintenance, and support will be conducted throughout its lifecycle.***

Mentimeter

What is a software Management plan [SMP]?

Outlines how the key aspects like ***goals, resources, milestones, risks, and quality*** are effectively controlled.

Mentimeter

What is a software Management plan [SMP]?

The plan is usually created early in the project, ideally before work is undertaken, and is essential for both small and large projects to ***ensure clear communication and expectations among team members***.

Mentimeter

The *traditional* Software Lifecycle

Mentimeter

What do you think are the main steps of a software development lifecycle?

- Planning
- definition of requirements
- Testing
- Defining objectives and requirements
- framework of data flow
- plan of data pipeline
- bug fixing
- testing

9

Mentimeter

What do you think are the main steps of a software development lifecycle?

1. Plan and structure project2. Develop3. testing4. Debug

Updates

Capturing Requirements, Developing, Testing, Deploying,Maintaining

design abstraction

Yes

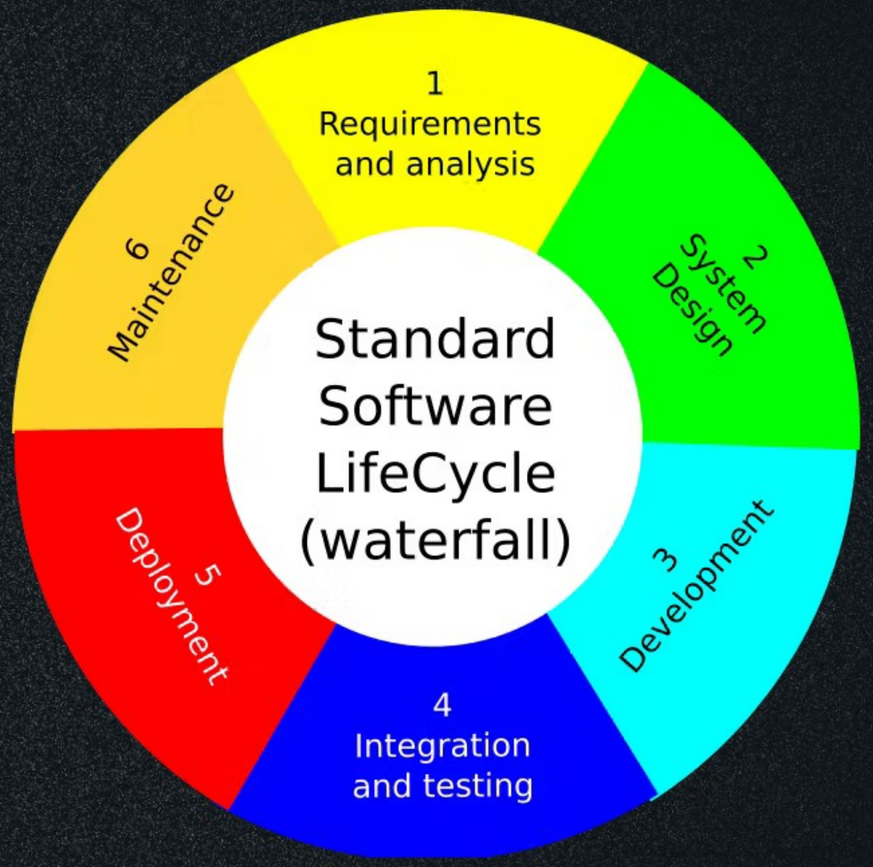
9

Mentimeter

1 - Gathering requirement from stakeholders and create software specification document. Understanding what the system is supposed to do.

2 - Created the blueprint of the system. Architecture, components and their interactions

3 - The coding takes place here, based on the software specification document

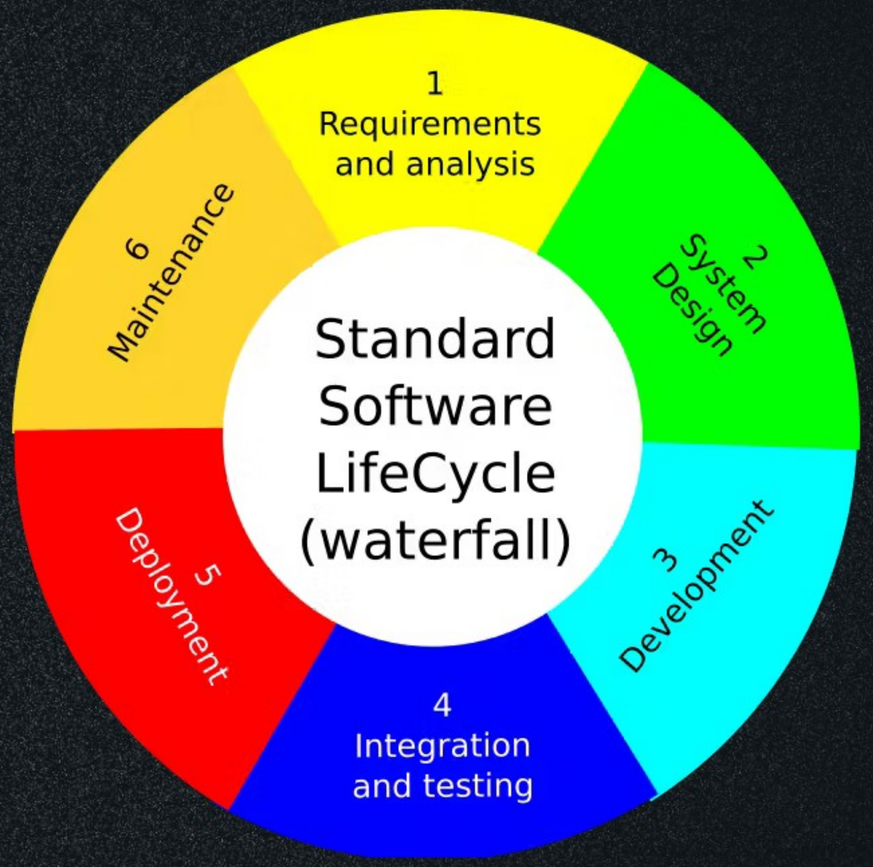


Mentimeter

4 - the individual block are integrated to a complete system and various testing are conducted.

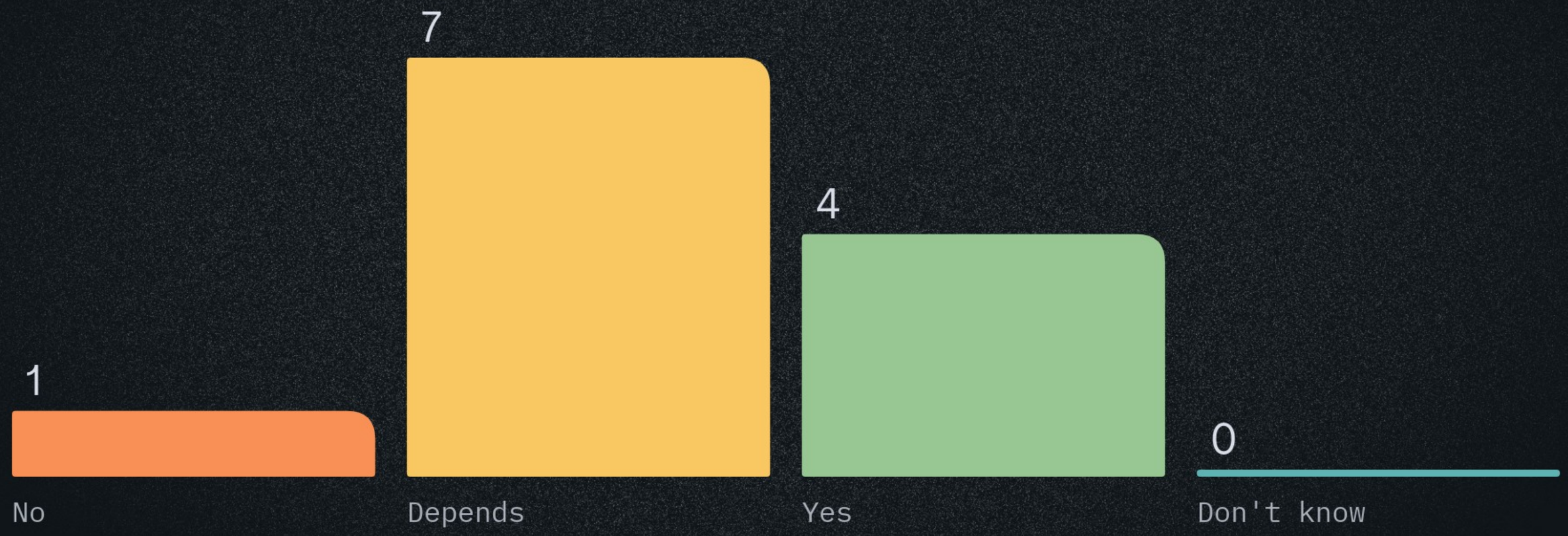
5 - Once ready, the code is deployed to the production environment and delivered to the client.

6 - Monitoring of the software, bug fixes, update.



Mentimeter

Do you think it applies well to research?



12

Mentimeter

This software lifecycle model is quite rigid as it does not handle changes very well.

It is called the 'waterfall' model and is often used in the private sector.

1

Mentimeter

What do you think are the main differences between software development in research and the private sector?

- Often I will be the client and developer
- profit
- Open Source, Data Confidentality
- The need to be adapted for different projects
- The requirements/interests/constraints of the funder body (research) versus client (private sector)
- The best way to solve the problem is not known yet
- stages change as the research brings out new outcomes (along with challenges)
- more of a focus on ethics and perhaps more iterative in the research sector

11

Mentimeter

What do you think are the main differences between software development in research and the private sector?

usability

change in experiment/analysis design over time

check you can access the data that you need

11

Mentimeter

Developing software in academia is very different from the private sector

Industry: Driven by market needs and profitability. Funding is well structured and aligned with the timeline. Built for longevity. Large teams. Often closed or limiting license.

Research: Driven by research question, innovation, advancing knowledge. More dynamic. Relies on grants, harder to maintain after end of project/funding. Often smaller teams with overlapping roles. Open-source is favoured.

Mentimeter

Software Management plan:

Answering the right questions

-What?

-Who?

-How?

-What can go wrong?

-Am I doing it right?

-Who will see it?

-How long will it live?

Mentimeter

Software Management plan:

Answering the right questions

What? --> Software definition [FAIR]

Who? --> Human and Technical resources [FAIR]

How? --> Development approach, tools and timeline [FAIR]

What can go wrong? --> Risk Analysis [FAIR]

Am I doing it right? --> Quality Insurance [FAIR]

Who will see it? --> Dissemination [FAIR]

How long will it live? --> Long-term Maintenance [Reproducibility]

Mentimeter

What?

Software definition

Mentimeter

Software Overview: Goal of a research software?

--> Help answer a research question

Mentimeter

Software Overview: Goal of a research software?

Other constraints are coming with this:

Development must be made in a way that allows other researchers (***and yourself***) to carry the research forward.

--> Must meet academic standards for **collaboration, documentation and reproducibility**

Mentimeter

You have a research question, and an idea for a software. What is the next step?

- Find out if it has been done before
- feasibility research of functions or algorithms you want to build
- Try it
- identify the data pipeline
- lit review
- Data gathering and cleaning
- Set out/Plan the required inputs, outputs and the process needed
- find scenarios for testing

9

Mentimeter

You have a research question, and an idea for a software. What is the next step?

If its open source you could build on it

See if you can iterate using this software (if the license allows)

9

Mentimeter

Next step:

--> Look at what is already available.

The case where people are re-implementing something that already exists is far too common and wastes a lot of previous research time and funding.

In short: **DO NOT RE-INVENT THE WHEEL**

Mentimeter

During your browsing, you find another software that does similar things, but not with all the functionalities you need. What are your options?

- depends on the licensing of the other software
- fork the project
- Check the licence, see if it is possible to adapt it for my analysis
- Us that software if you can
- Try to align my research ideas to fit with software capabilities
- develop a branch from it if it is open source
- See if you can use it as a foundation to iterate from...depending n the licence

7

Mentimeter

Options #1 --> Use it!

- If the code is open source, you might just be able to clone it and continue the development (as long as you respect the terms of the licence).
- Alternatively, contribute directly to that software by adding new features.
- If the code is NOT open source (or unlicensed), contact the author and ask if you can have access to it!

--> You will forge new collaboration on the way!

Mentimeter

Option #2 --> Ignore it and rewrite your own version

- In that case, ask yourself these questions:
  - What would make people use my solution instead of the one that is already available?
  - What benefits will my solution bring to the research community?

This might be valid option!

Mentimeter

Do Not start coding straight away!

Ask yourself what you are going to code, who is going to do it and how.

Mentimeter

Simple questions to ask yourself:

- What data will be used as input?
- What will the output of the software look like?
- Any standard that you should stick to?

If you stick to what is generally used in your research domain you will make your software more **Interoperable and Reusable**

Mentimeter

Who?

Human and technical ressources

Mentimeter

- Identify team members and define their roles in the development of the tool.

- Extend the distribution of roles to other aspects of the project, such as data management & publication

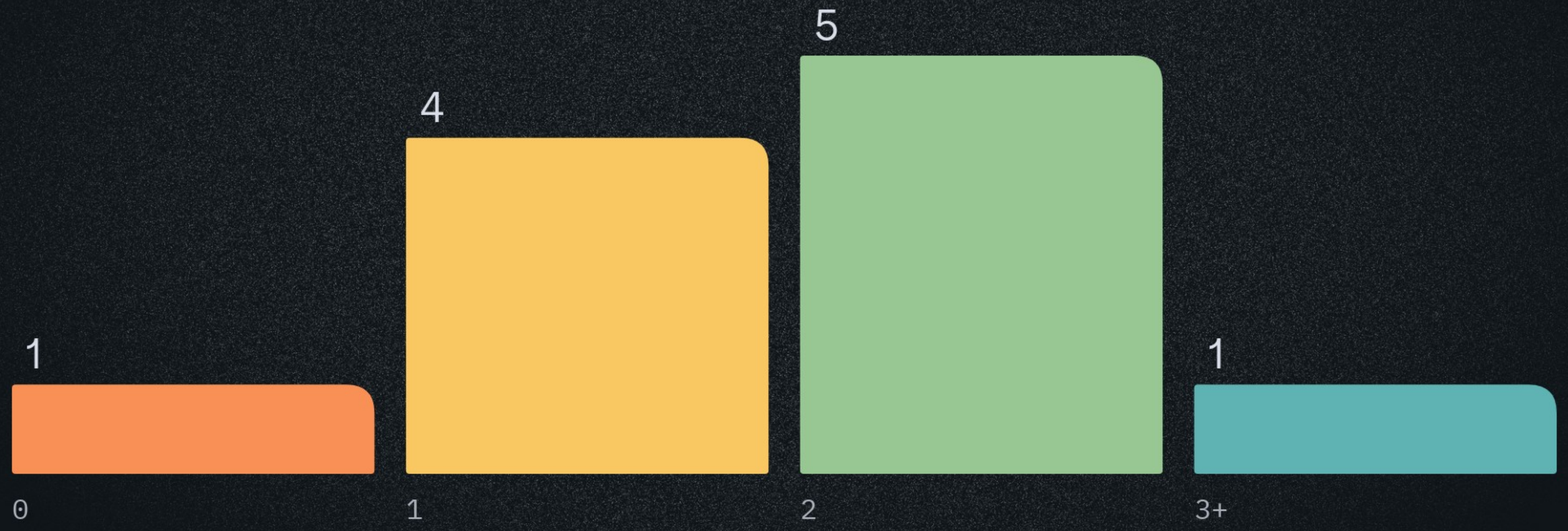
Doing so will help you:

-Identify what skills are available.

- Identify what skills are missing (training needs)

Mentimeter

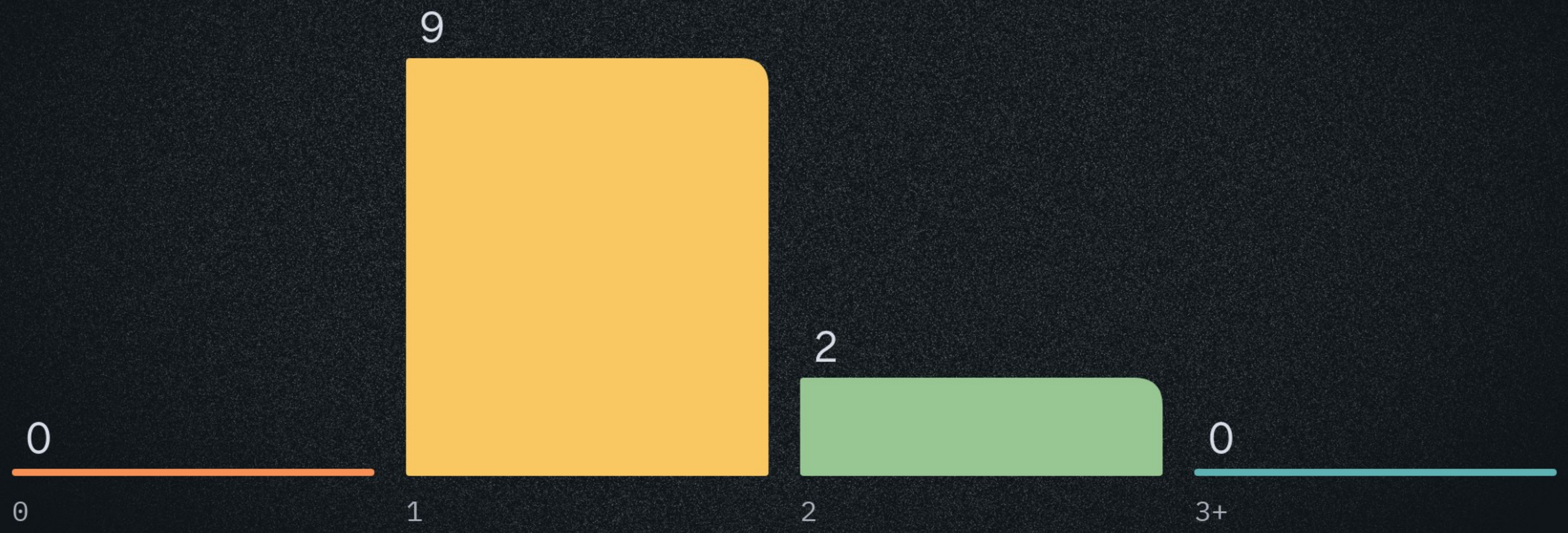
How many people (in your current project or in a previous one) are aware of how the software works?



11

Mentimeter

And how many people are able to fix it in case of problem?



11

The bus factor:

Measurement of risk that arises if a member of the team were ***"hit by a bus"*** and the information and capabilities they hold about the project has not been shared with the team.

Possible Mitigation:

- Different people in the team know the code and where it is!
- Peer review during the development
- Very clear and up-to-date documentation on what the software does ***<u>and</u>*** how it does it.

This will make your code **findable** and ***reusable*** by your team!

Mentimeter

Other ressources?

The university offers a wide range of options that you can collaborate with:

- Research Software Engineering
- Data analytics service
- High Performance Computing support
- Data steward service

etc...

It is important that you are aware of them and include the ones you need from the start of the project *(some need budget!)*

Mentimeter

How?

Development approach, timeline and tools

Mentimeter

Development Approach

Waterfall / Agile

Linear and sequential project management. / Iterative and flexible project management

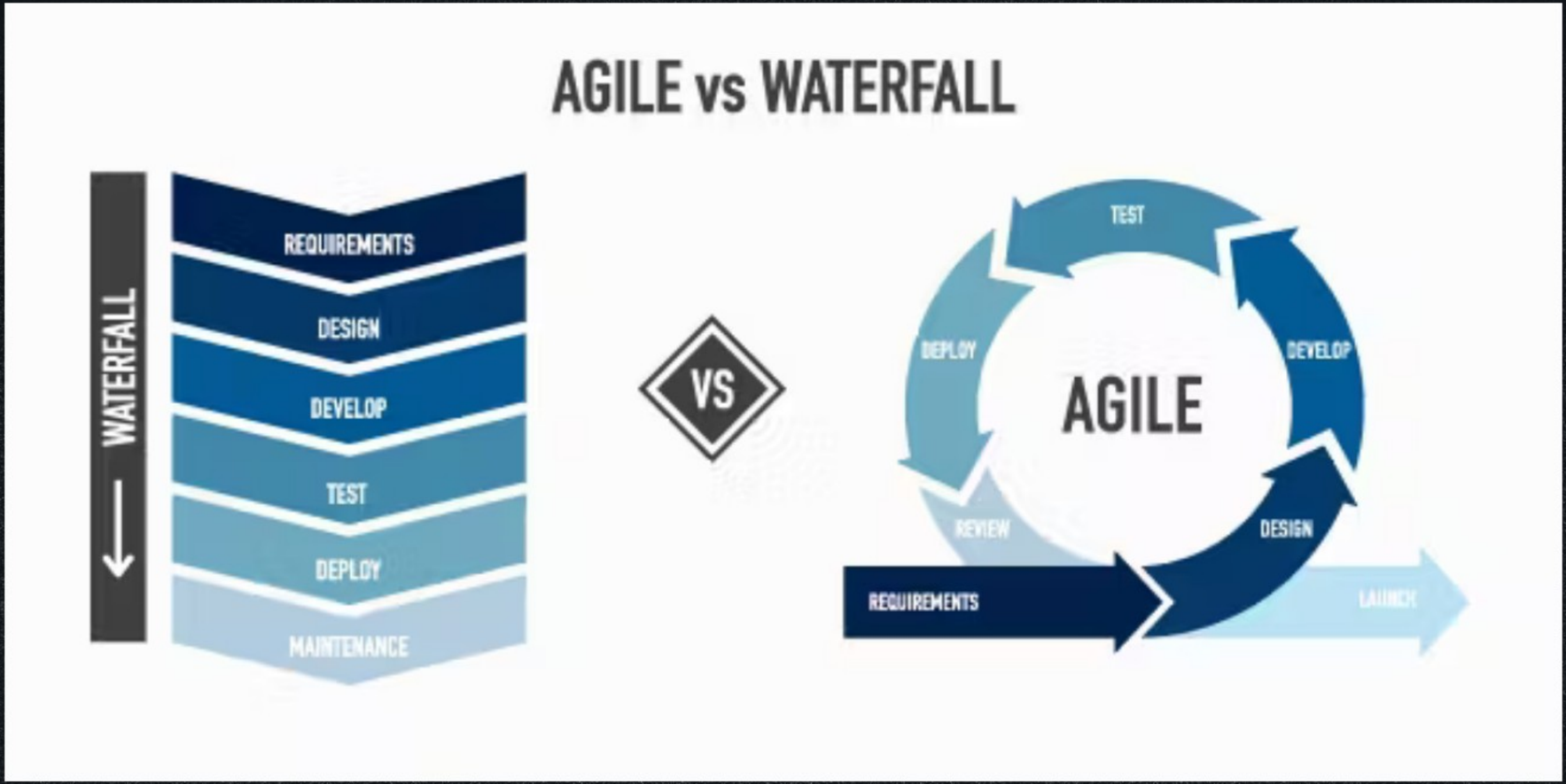
Each phase depends on / technique focused on delivering small,

the completion of the previous one. / incremental improvements.encourages

Progress flows in one direction, similar / collaboration between cross- functional teams

to a waterfall. / functional teams, frequent feedback,

/ and continuous adaptation to changes.



Source: https://generaldigital.com/

Mentimeter

- Whatever method of project management you choose, ensure that it is fostering collaboration within the group and a sustainable software -

1

Mentimeter

Research is by definition exploratory, regular review are important (both code AND documentation)

--> They might actually uncover publishable intermediate results!

Mentimeter

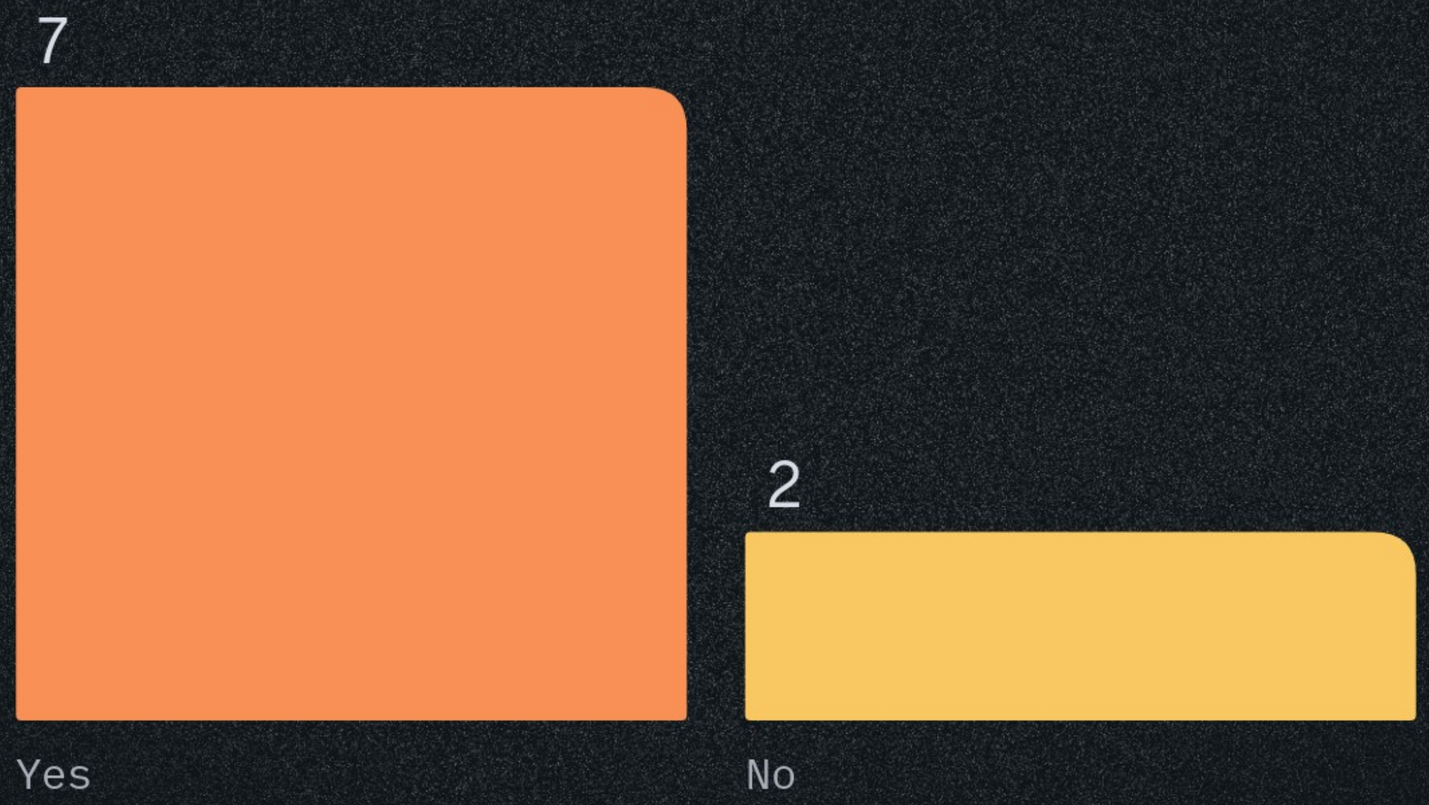
Timeline?

It is often possible to split the development of a software package into multiple intermediary states. --> This states are a good review point

Don't forget: This is research! You research timeline is the priority and comes with its own milestones (conference, paper submission, funding deadlines, PhD thesis submission). If you align research milestones with the development of the software it will make it easier to track both.

Mentimeter

Have you ever had this type of file naming: ***file1, file1_version2, file1_final....***



9

Mentimeter

Tools you use for software development are also very important.

--> Version control allows you to track ALL the changes made to your software.

This will make you software more FAIR

Multiple options are available for version control: git, mercurial, SVN (old).

Mentimeter

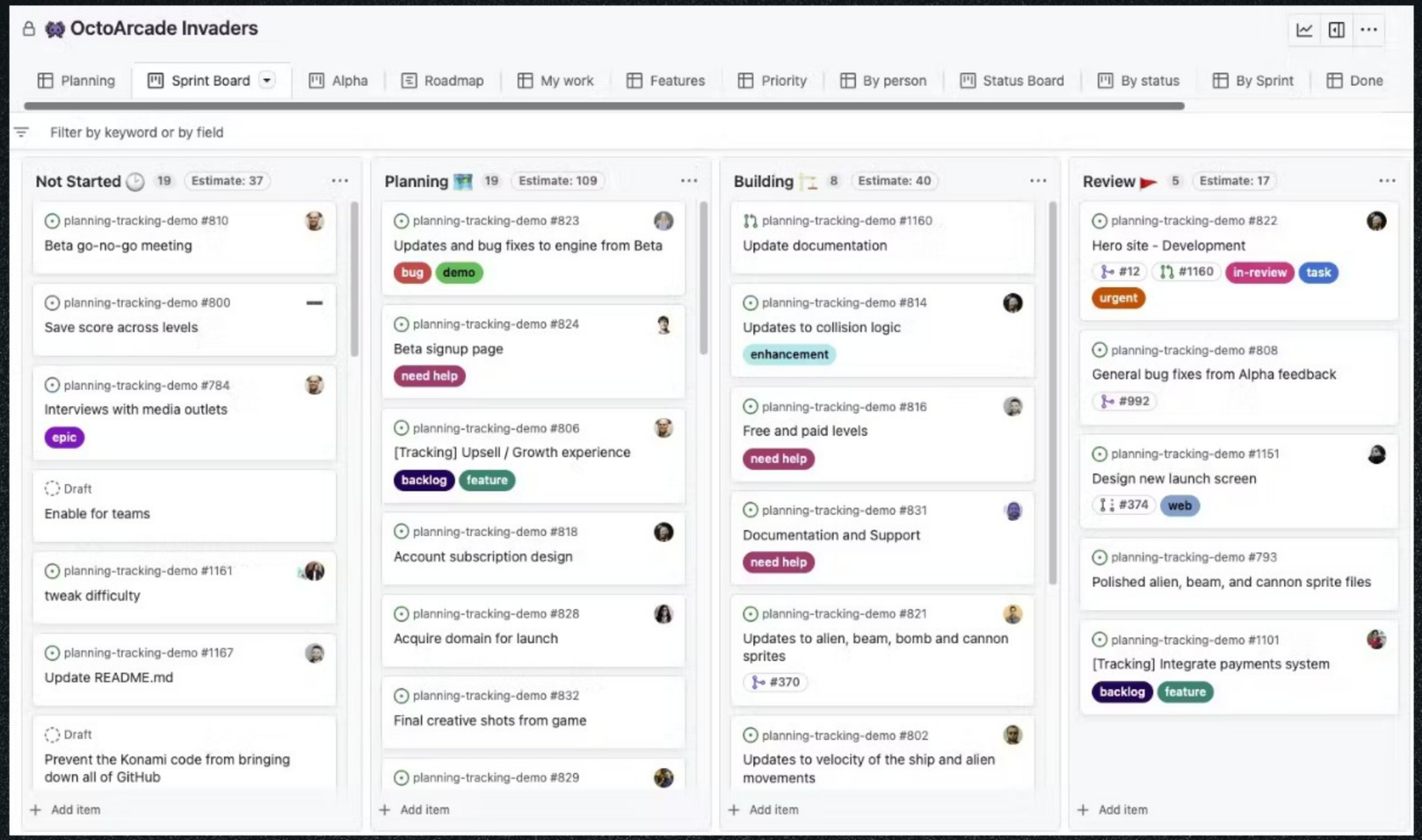
Tools you use for software development are also very important.

--> Version control allows you to track ALL the changes made to your software.

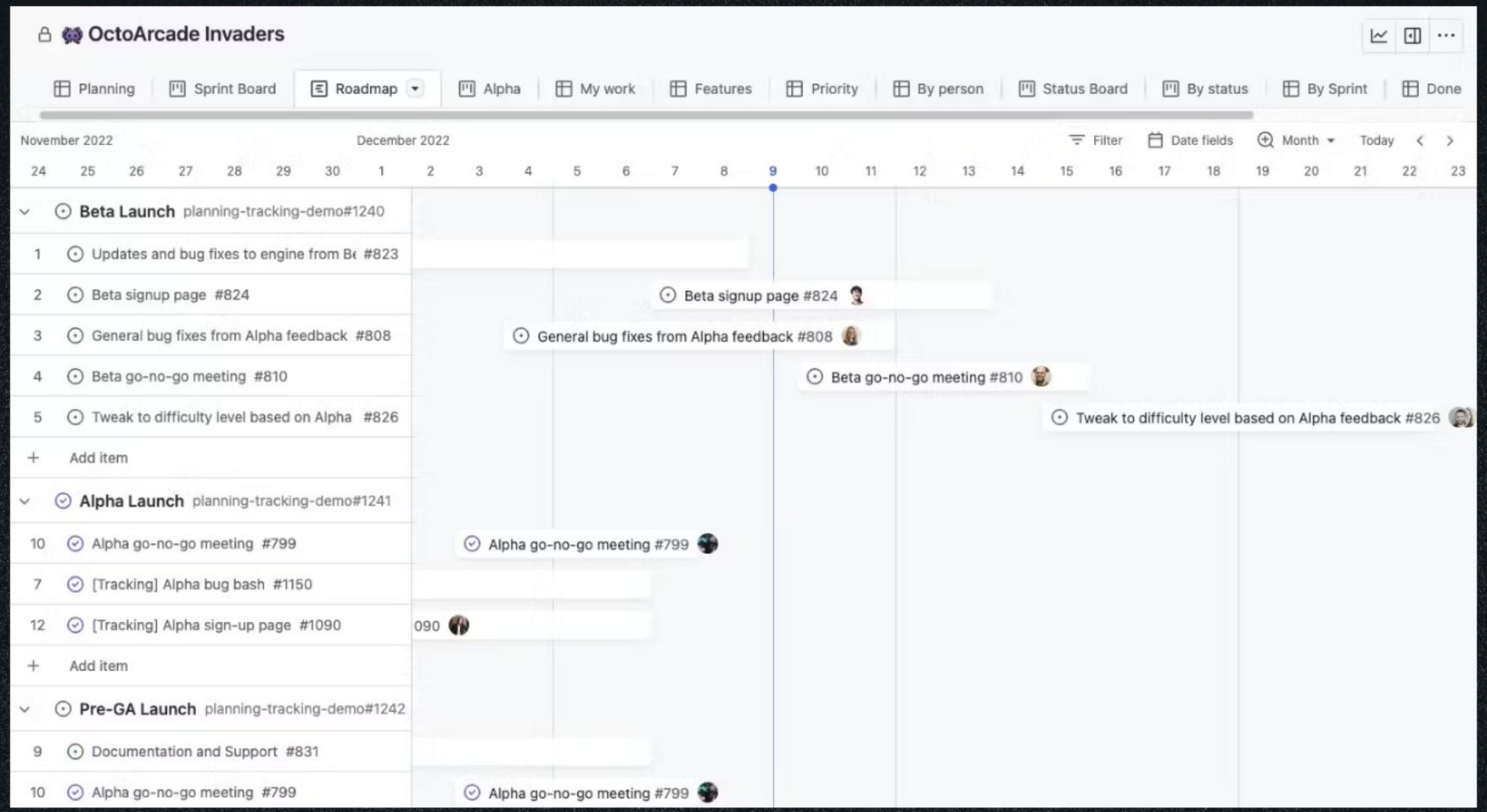
This will make you software more FAIR

Multiple options are available for version control: **git,** mercurial, SVN (old).

Mentimeter



Github gives a lot of project management tools



Github gives a lot of project management tools

Mentimeter

What can go wrong?

Risk analysis

Spoiler Alert:

Nothing will work at the first try!

--> Roadblocks will be discovered along the way.

It is good practice to try to **identify them in advance.**

Mentimeter

Based on your experience, what risk can you foresee regarding a software project?

- Struggling to complete in the timescale
- changing requirements
- Scope creep
- quality issues
- Accidentially making some raw data public
- Software isn;t our domain expertise
- customer expectations
- version dependencies issues/clashes later on

8

Mentimeter

Based on your experience, what risk can you foresee regarding a software project?

allow for feedback when used

8

Mentimeter

Examples:

- Data issues (sometimes data will never arrive!)
- Tool/library obsolescence
- Turnover in the team

Mentimeter

Mitigation:

- Looking at other available open dataset
- Try to stick to standard and well supported libraries.
- Code design
- Documentation!!!

Mentimeter

Is my code robust?

Ultimately, your research will be as robust your research software...

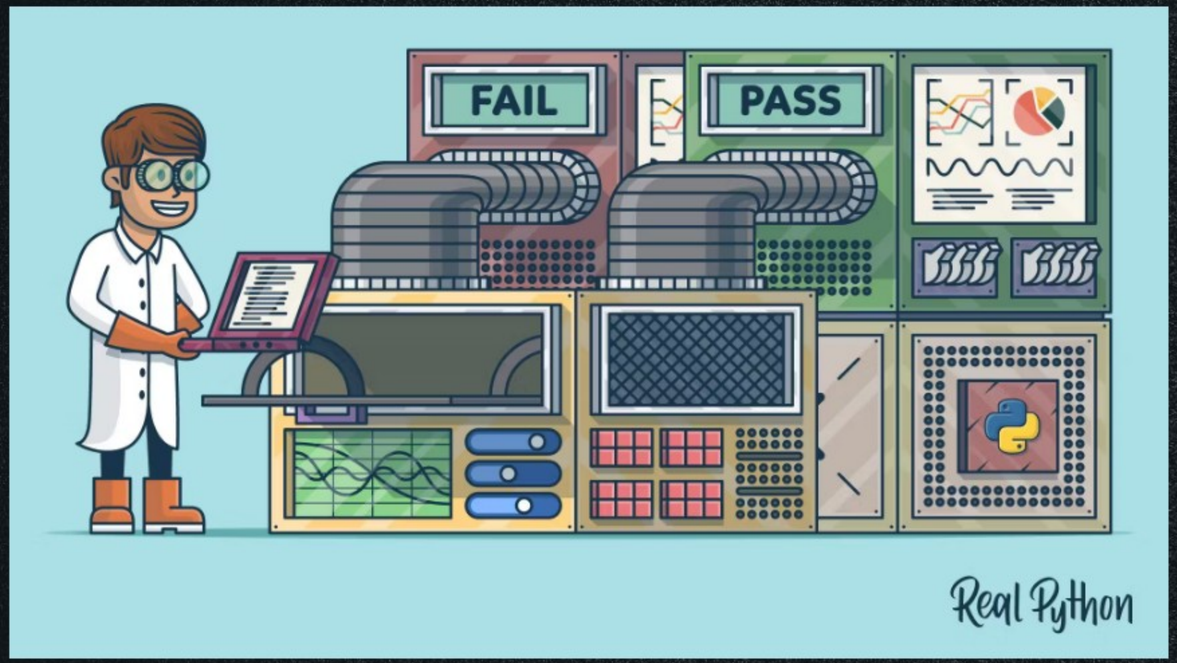
Mentimeter

How can you be sure that your code is doing what you think it is doing?

- step checks
- Validate intermediate outputs against known data
- give it "bad" data
- positive and negative test
- unit tests, validation against known data
- Unit testing, Integration Testing, Functional Testing
- Test with inputs where you know the expected output
- ask someone else to look at it

7

Mentimeter



Quality Insurance:

-Testing: Allows you to test that each bit of your code is working as expected

-User testing: If someone else can reproduce your results with your software it will give you a good idea of how reusable your software is.

-Replication: If you can replicate other results with your software that will help you gain confidence on what you are doing.

-Peer review: Checking code quality, disseminating knowledge

-Data Version Control: Increase reproducibility, helps track all version of the data you have

Dissemination:

One of the corner stone of the research process!

- **Documentation is paramount**
  - It will help you first
  - Increase greatly the potential of reuse
  - Should be tailored to your audience

Mentimeter

Dissemination:

One of the corner stone of the research process!

- Documentation is paramount
- Licences (See Kate's Talk)
- Software repository (See Ric's Talk)
- Publication
- Training

Mentimeter

How long will it live?

Long term maintenance

Mentimeter

Two possible situations:

A - There is still funding:

--> Identify who is in charge of the maintenance and updates and make sure that they get credit for it! **You need a clear code and documentation to allow this to happen**. [You might be working with totally different people than at the beginning of the project]

Mentimeter

Two possible situations:

B - There is not funding anymore:

The most likely situation (founders are not taking this into account). **How to make sure that your code will work 5-10 yrs after the project?**

--> **One option is to distribute your software with its own computational environment. This ships your software with the required dependencies and put all this in its own bubble.**

--> This greatly increase the **long-term reproducibility of your work!**

Mentimeter

Software Management plan:

Answering the right questions

What? --> Software definition [FAIR]

Who? --> Human and Technical resources [FAIR]

How? --> Development approach, tools and timeline [FAIR]

What can go wrong? --> Risk Analysis [FAIR]

Am I doing it right? --> Quality Insurance [FAIR]

Who will see it? --> Dissemination [FAIR]

How long will it live? --> Long-term Maintenance [Reproducibility]

Mentimeter

Minimal version:

You might not need all of this, depending on the type of software you will develop.

1 - Define objectives

2- Information on what software outputs (including documentation and other related material) are expected to be produced.

3 - Who is responsible for the development work?

4 - Which version control system is to be used?

5 - What license will be used for each produced piece of code/data?