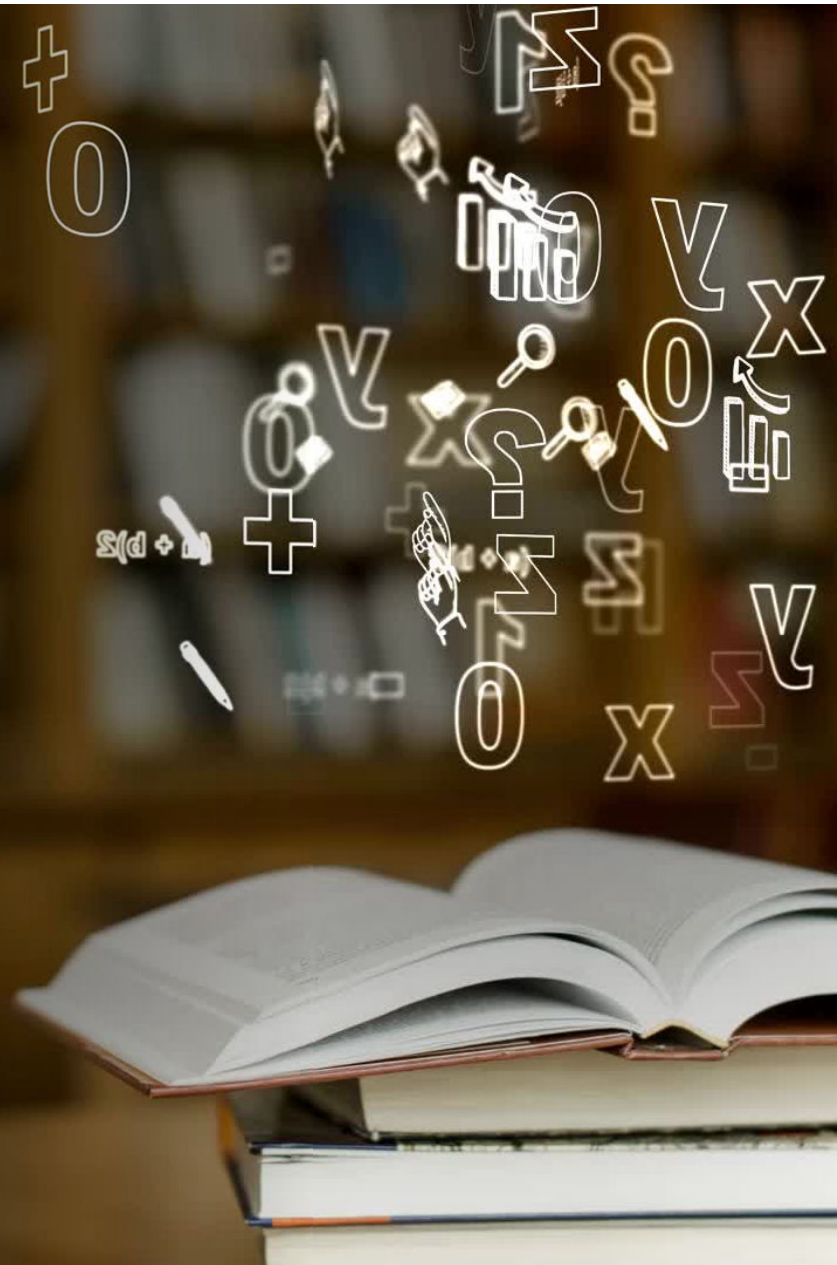




# Introduction to Agile project management for libraries

---



---

## Your Tech Camp trainer

- Nerida Cain
  - Certified PMP and Agile Practitioner
- Project Manager who's worked in
  - State Library Victoria
  - Banking and media sectors
- Specialisation in software implementations
- Most recently lead the implementation of SLV's Library Services Platform
- Alma, Primo VE, included over 100 integrations and new customer authentication process



# How can Agile be used in Libraries?



WHAT'S AGILE'S DEAL



HOW IT CAN BE USED IN  
LIBRARY PROJECTS



ADOPTING TOOLS TO  
RUN AGILE PROJECTS



DEEP DIVE INTO KEY  
AGILE TECHNIQUES



ADOPTING KEY AGILE  
TECHNIQUES TO USE IN  
YOUR OWN PROJECTS

# What is a project?

Project Management Body of Knowledge (PMBOK):

*A temporary endeavor with a beginning and an end and it must be used to create a unique product, service or result.*



# What does Agile mean?

Macquarie Dictionary defines *agile* as:

- *adjective*: quick and light in movement;
- of or relating to a management strategy in which development stages are flexible, allowing for quick reaction to immediate needs while maintaining long-term goals.







---

# Agile v. Waterfall

## **Agile**

Iterative approach to delivering a project, continuous releases incorporating stakeholder feedback.

---

## **Waterfall**

A linear approach and sequential approach; distinct project phases must be completed before moving to the next one i.e.

Atlassian notes that Agile teams may follow a similar sequence yet do so in smaller increments with regular feedback loops.

# Why use Agile project management?

---

Higher product quality; lots of testing; regular reviews and opportunities to identify improvements

Improves communication between project team and organisation

Reduces risk; working in sprints allows teams to develop a working product or fail fast and take another approach

Better visibility into project performance – frequent meetings and reviews provide increased transparency to everyone on the team

Team members have control throughout the project with more opportunities to test and adapt

# Key Agile principles

---

Customers or key stakeholders, should receive project deliverables or iterations at regular intervals throughout the project's life cycle, rather than just one delivery at the end.

Welcome changing requirements, even late in development.

Delivery working software frequently from a couple of weeks or months, with a preference for shorter timescales. This allows for a fast turnaround of workable products.

Regular communication! Business people and technical staff work together daily throughout the project.



# Key Agile principles

---



Build teams with motivated individuals and give them autonomy. Build teams on capabilities rather than job positions or titles.



Promote face-to-face conversation; co-locate teams and stakeholders whenever possible.



Working product is the primary measure of progress; the aim is to provide complete, working deliverables. Prioritise this over additional requirements, such as project documentation.



Agile processes promote sustainable development; the project team should be able to maintain a constant pace indefinitely.

# Agile is dead


---

- Pure Agile isn't suited to libraries; it's for software development, website development teams.


But

- Many aspects of Agile will transform your project management practices

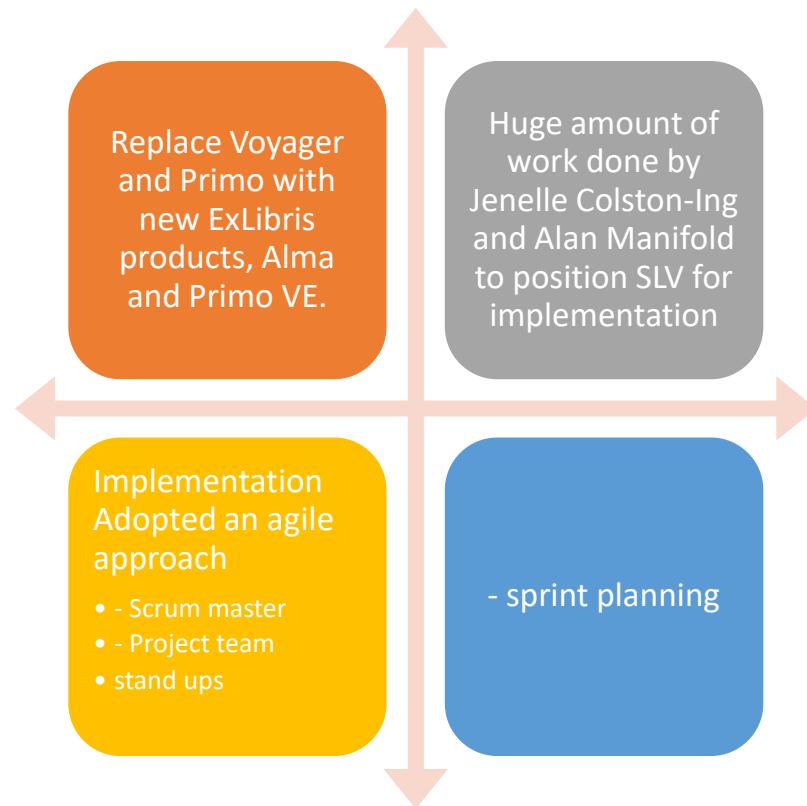




# What can Agile bring to a library setting?

- Create high performing, autonomous teams that have a clear understanding of what they need to achieve and by when
  - Foster excellent communication which will identify issues before they become major blockers
  - Regular showcases provide opportunities to identify design and configuration issues
  - Retrospectives offer opportunities to correct issues within the project team or project framework to foster continuous improvement
- 

# Implementing a new LMS at SLV





# Meet Wrike

- Who / What / Where?
- How is it going to be used today?

Choose your project

- BYO
- Library Services Platform implementation

Let's talk  
about  
projects

---

What projects have you  
worked on?

---

What worked?

---

What didn't work?

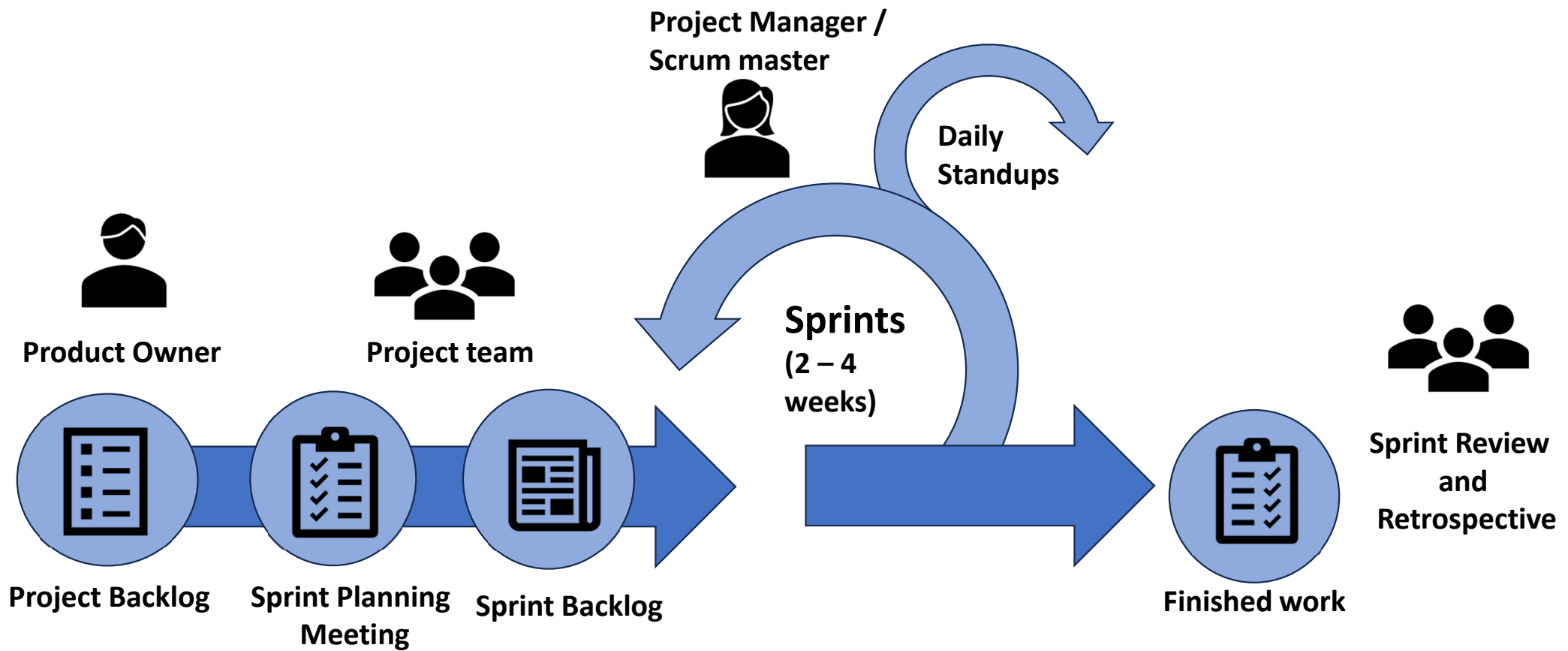


# Nuts and bolts of Agile

The tools of Agile project management




# Agile cycle with a Scrum focus



# The tools

- Why do you need a project management tool?
- What do you need in a project management tool?
- How to use the tools?





## Case study: Implementation of an LMS

- How to implement a Library Management System using Agile?
- Case study included in materials
- 7 month implementation
- The Library will be working closely with its software vendor
- Key tasks included in case study notes; not a comprehensive list



ACTIVITY  
Get to know  
Wrike

- Establish your project team
- Select project to work on
- Review case study OR brief your team on your project
- Set up Wrike board

10 minutes

# Agile project team

AGILE TEAM ROLES	RESPONSIBILITIES
Project Manager (or Scrum master)	Manage working environment in which the project is evolving; co-ordinate project at a high level but leave detailed planning to Developers and SMEs. Adopt a facilitative style rather than a “command and control” style
Product Owners <i>(Product = software)</i>	Understand the customer and business requirements, then create and manage backlog based on requirements Manages product roadmap and prioritises backlog of enhancements or project work
Business Analyst	Supports the functional and technical teams; helps the organisation to define requirements.
Developers SMEs	Significant input into creation and prioritisation of requirements. Work on tasks prioritised by the product owner.

- Three key tasks
  - Balancing teams
  - Improving communications
  - Mitigating failures or obstacles
- 
- All teams need a product owner and developers and key stakeholders. In smaller teams, the product owner can act as the scrum master. In larger teams, the workload would be too large.
  - Key to success – all team members accept personal responsibility and commit to doing the work. Each team member needs to maintain good communication; spot and report obstacles and feedback on potential issues.
  - Must do; at the beginning of the project, hold a workshop with your project team to discuss roles and responsibilities, and expectations from team members.

# Identify the work that needs to be done

- Define the backlog
- Product backlog





## ACTIVITY

# Define your project backlog

In your groups and using Wrike, define your project's backlog

10 minutes

# Agile ceremonies

- Agile ceremonies are meetings; they provide transparency and regular communication with the team.
- Not every agile scrum team needs to practice all scrum meetings.
- Scrum ceremonies are attended by project manager (scrum master), product owners and development team.
- Agile ceremonies
  - Sprint Planning
  - Daily stand-up
  - Sprint review
  - Sprint retrospective



# Agile tool deep dive



## Time boxing (Sprints)

Definition: a fixed period of time, at the end of which an objective has been met.

Optimal length for time box or sprint is 2 – 4 weeks. Long enough to achieve something and short enough that the team doesn't lose focus.

Timebox structure: Kick off -> Do the work -> Close out



# Planning your Sprint

- Before the sprint starts, schedule a sprint planning session with the team to identify and agree the work to be completed
- With input from the product managers, who determine the priorities, the team will draw work from the project backlog
- How much work can be completed? This will be uncertain at the beginning of the project, but as it progresses, you will develop an understanding of this.



# Planning your sprint

## Key Agile tenets

- The project team needs to determine how much work can be completed, not the project manager. The project manager can challenge and coach the team, but shouldn't set the agenda.



# Daily standups

- Key tool to ensure momentum in Sprints
- Limit them to 15mins; ideally in the morning
- Agenda for Daily Stand ups
  - What I have been doing since the last stand-up that helps achieve the Sprint's objectives
  - What I will be doing between now and the next stand-up to help achieve the Sprint's objectives
  - What problems, risks or issues (blockers) I have that will prevent me or the team achieving the Sprint's objectives

# Daily Stand Ups

- **Project Manager or Scrum Master's role**

- Encourage everyone to stand up! It helps to keep updates to a minimum.
- Share the project's board in the meeting so everyone can see it and can speak to the tasks that are assigned to them.
- Keep the team to time
- Ensure they stick to the agenda
- Take on issues and blockers and resolve them, as soon as possible, provide updates.



# Sprint review

- Objective: reveal what was accomplished during the sprint.
- Attendees: Project manager, product owner, stakeholders, project team members
- Duration: 45mins for each week of the sprint
- Agenda:
  - Showcase the team's work. Demo what has been completed. Work must meet agreed quality standards to be considered complete and ready to showcase.
  - Good opportunity to provide feedback

# Sprint retrospective

- Objective: review what was successful during the sprint and identify what can be improved.
- Attendees: Project manager, product owner, project team members
- Duration: 45mins for each week of the sprint
- Purpose: Rapid feedback ensures the project result and project culture improves; continuous improvement is what drives success.
- Tools:
  - Many retrospective tools available online. Miro, Teams



# Delivery Plan



A high level schedule of project increments and the sprints contained within the increment.



Doesn't include task-level detail; tasks are included in your project tool e.g. Wrike.



Delivery Plan Template available for use.

# LMS Delivery Plan



Project manager creates realistic Delivery Plan in conjunction with Product Owners and Developers/SMEs prior to implementation kick off



Delivery plan and progress should be visible to all; project manager keeps it up to date



Details to include for an LMS implementation:

Agile ceremonies the team will use  
How many sprints will you need/have before the Go Live date  
Schedule of sprints and high level objectives to be achieved within each sprint



## ACTIVITY

### Develop a delivery plan

In groups and using Wrike:

- Determine which Agile ceremonies you will use
- Define duration of your sprints
- Identify which sprints in which the key tasks would be completed.



# Break time

---

5 minutes

# Requirements

The key to  
success

- What are requirements?
- A service, function or feature that a user needs.
- Examples for an LMS:
  - a Library patron must be able to request an item.
  - An Acquisitions Officer, must be able to create a new record for a new acquisition.
  - A cataloguer must be able to describe the item






# Functional (FRs) vs. Non-functional (NFRs)

- Functional requirements express function or feature and define what is required
  - e.g. request item or create new record
- Non-functional requirements define how well, or to what level a solution needs to behave.

They describe solution attributes such as security, reliability, performance, response time.



- 
- Identify stakeholders; product owners and SMEs who will use the system
  - Business analyst and project manager meet with key stakeholders product owner and SMEs to gather requirements

#### Tools

- User stories
- Brainstorming sessions
- Process diagramming
- Surveys
- One-on-one meetings



How to gather requirements?

# Top tips for gathering requirements

---



Focus on key stakeholders; who are the key people and teams who will be directly impacted by the project



Leave enough time; it will always take longer than planned



Summarise and confirm understanding of requirements by repeating them back to stakeholders. Never make assumptions.



Requirements are iterative; most requirements gathering processes will miss something; there will always be unknowns; allow for this.



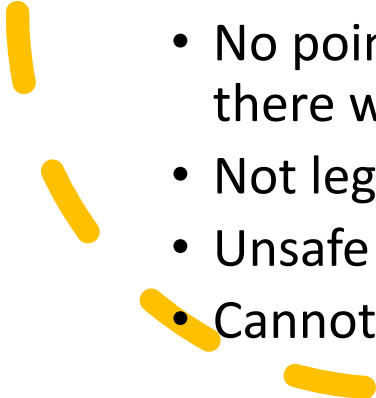
# Prioritising requirements – MoSCoW Rules

**M**ust Have, **S**hould Have, **C**ould Have, **W**on't Have this time

- **Must Have**

These provide the Minimum Useable SubseT of requirements which the project guarantees to deliver.

- No point in delivering on target date without this; if it were not delivered, there would be no point deploying the solution on the intended date
- Not legal without it
- Unsafe without it
- Cannot deliver a viable solution without it





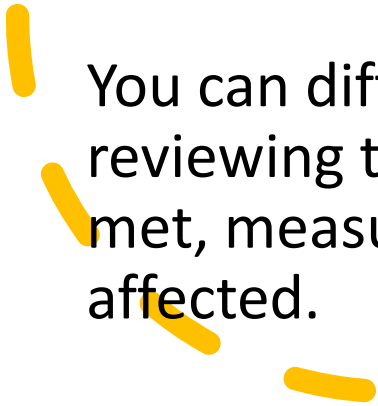
# MoSCoW Rules

- **Should Have**

Important but not vital

- May be painful to leave out, but the solution is still viable
- May need some kind of workaround e.g. some inefficiencies

You can differentiate between Should Have and Could Have by reviewing the degree of pain caused by the requirements not being met, measured, measured in terms of value or number of people affected.



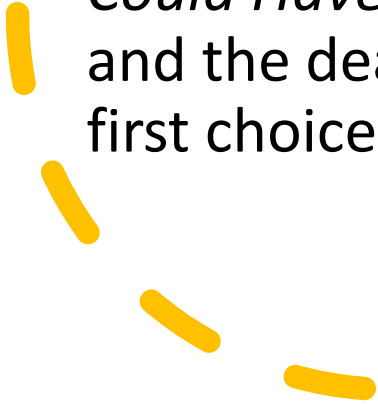


# MoSCoW Rules

## **Could Have**

- Wanted or desirable but less important
- Less impact if left out (compared to Should Have)

*Could Haves* provide main pool of contingency. When a problem occurs and the deadline is at risk, one or more of the *Could Haves* provides the first choice of what is to be dropped from this timeframe.





# MoSCoW Rules

## **Won't Have this time**

These are requirements which the project team has agreed will not be delivered (in this timeframe). They are recorded in the Prioritised Requirements List where they help clarify the scope of the project.

Helps manage expectations.





# MoSCoW Rules

- A few guidelines
  - Must Have requirements should make up 60% of available effort. If they exceed this, the project is at risk.
  - Could Have requirements should make up 20% of effort. Could Have requirements provide the primary contingency that makes delivery of the higher priority requirements more likely.







## Documenting requirements

- Word or Excel documents, template included in Techcamp docs
- User Stories within project management software e.g. Wrike

# User Stories

- A requirement expressed from the perspective of an end-user goal. User stories may also be called Epics.
- The format of a User Story:  
As a <role>  
I need <requirement or feature>  
So that <goal/value>

E.g. As a Library patron  
I need to request Library items  
So that I can borrow them.

# User stories

- Each user story/requirement, becomes a task within the project management tool. It includes acceptance criteria, which is used for testing.
- Acceptance criteria are written questions that expect an answer 'Yes'.
- These Acceptance criteria define, at a high level, the test criteria which will confirm the User Story is working as expected.



ACTIVITY  
Document  
requirements  
for your project

In your groups and using your preferred software:

- Document requirements for a key part of your project

# Testing

- Functional testing

# Testing

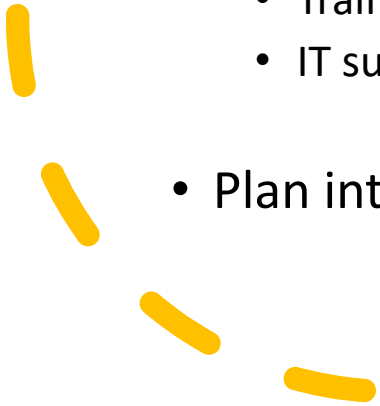
- User testing



# Go Live Planning

## Go live planning tasks

- Compile go live runsheet
- Plan support required for impacted stakeholders:
  - Training support for frontline and back of house staff
  - IT support to troubleshoot software bugs/issues
- Plan internal and external communication to stakeholders



# Go Live Runsheet



Excel spreadsheet that lists every task leading up to and post go live



The runsheet guides go live schedule



Project Manager's responsibility to lead and manage this activity



Start this process at least 1 month ahead of the go live date



Schedule weekly or twice weekly meetings with project team to compile list



Work through each activity on the sheet as a group



Ensure runsheet is accessible by all team members; the project manager will mark items off as complete





# Go Live Planning

## **Go live decision**

- Schedule meeting with project sponsor, product owner, project owner, and key stakeholders to determine if the project deliverable can go live.
- Go live decision should be documented. This can be a memo or a ticket in your project management tool i.e. Wrike.

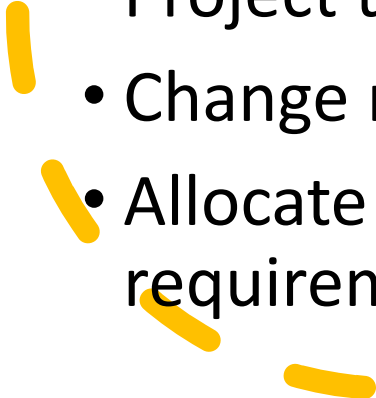


Let's talk about documentation



# Key learnings from SLV

- Agile delivery method ensured success
- Include experienced product leads/SMEs in the project team (even if you have recruit)
- Project team must be self managed and have authority to make decisions
- Project team must be well resourced, dedicated to project
- Change must be driven by peers
- Allocate *plenty* of time to define and document detailed requirements prior to implementation



# Key Agile takeaways