The Software Development Process

Agile Framework

AGILE METHODOLOGY

- Agile methodology differs significantly from other methodologies
- Agile means the **"ability to move quickly and adapt to change"**
- Agile methodologies are characterized by adaptive planning, early delivery and continuous improvement, with the ability to respond to change quickly and easily
- In traditional software development methodologies like Waterfall model, the customer can see the end product only at the completion of the project

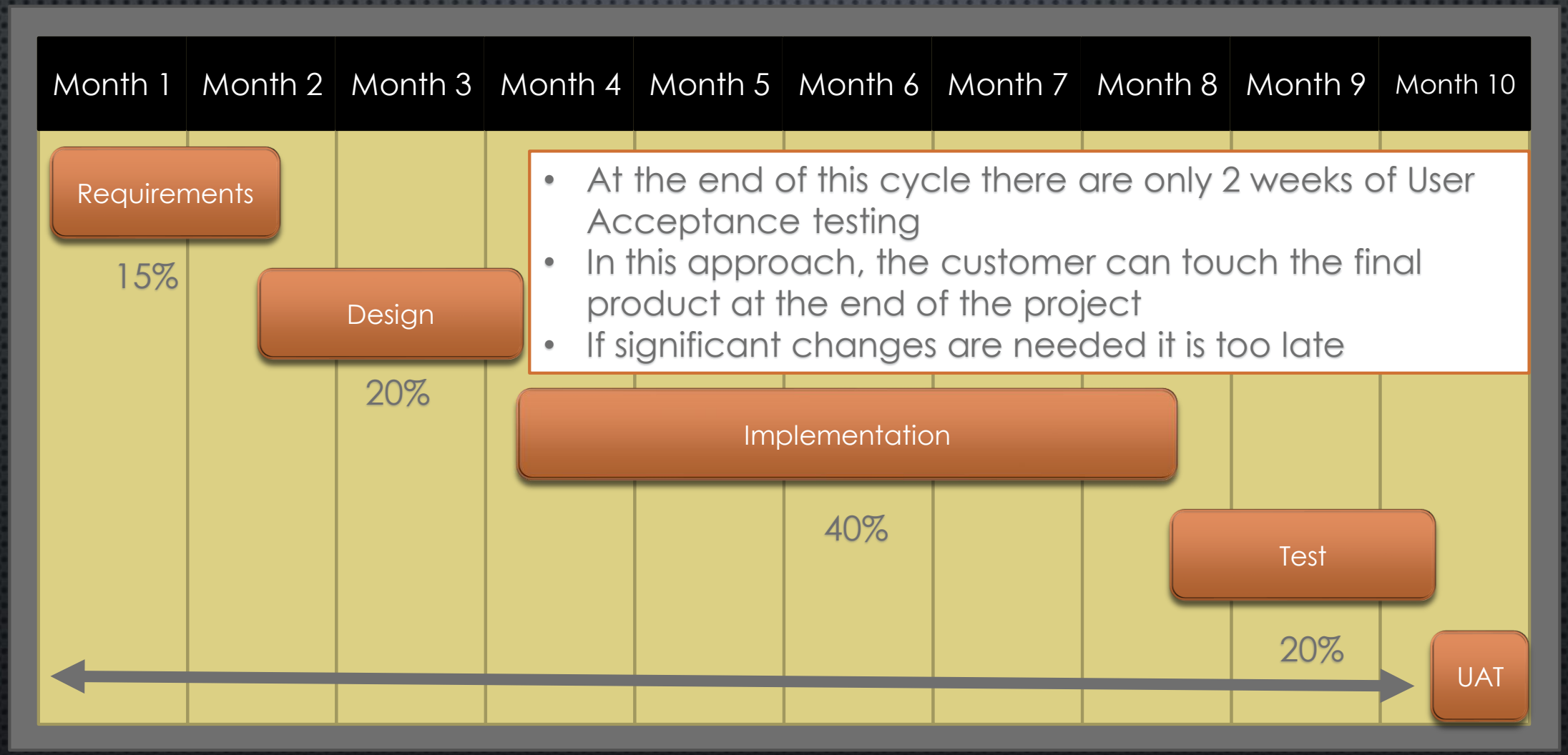
AGILE METHODOLOGY

- In traditional software development methodologies phases e.g. Requirements gathering, design, development, testing and User Acceptance Testing, can take a long time before deploying the project
- On the other hand, Agile projects are characterized by phases known as Sprints or iterations which are shorter in duration
- Sprints/iterations can vary from 2 weeks to 2 months, during which features are developed and delivered according to the sprint plans
- Agile projects can have one or more iterations and deliver the complete product at the end of the final iteration

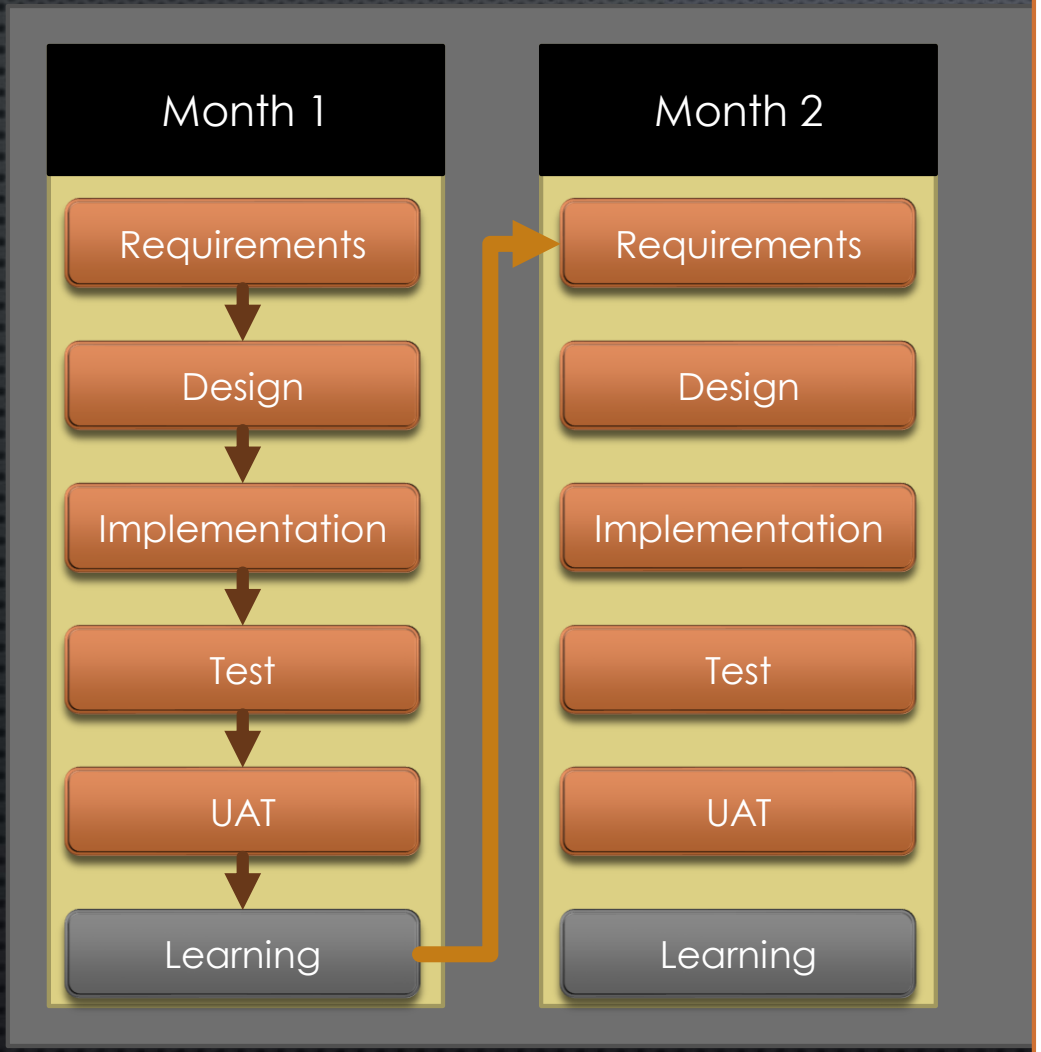
AGILE METHODOLOGY

- In the next lesson I will show you how an Agile software project is implemented
- Now I want to make you understand the difference between an Agile and a traditional software development project/process
- Let us say that we need to deliver a mobile App and we need 10 months

In traditional Waterfall model



In Agile model



- In the Agile methodology, each project is broken up into several 'Iterations'.
- All Iterations should be of the same time duration (between 2 to 8 weeks).
- At the end of each iteration, a working product should be delivered.
- In this approach the project is broken up into X (e.g. 10) releases
- Rather than spending a long time on requirements gathering, the basic core features that are required in the product will be decided soon and which of them can be developed in the first iteration
- Any remaining features that cannot be delivered in the first iteration will be taken up in the next iteration or subsequent iterations, based on priority
- At the end of the first iterations, the team will deliver a working software with the features finalized for that iteration
- There will be X iterations and at the end of each iteration the customer will approve a working software that is incrementally enhanced and updated with the features scheduled for that iteration

AGILE METHODOLOGY

- With Agile approach the customer to interact and test a functioning software at the end of each iteration and provide feedback on it
- Changes are more easily caught corrections are implemented soon
- The software is developed and released incrementally in the iterations
- Agile methodology gives more importance to collaboration within the team, with the customer
- All the stakeholders are involved in the real time process to identify the need for changes and delivering of a working software
- The design, development, testing and rework of each feature is completed before the feature is declared completed and released
- There are no separate phases, one single phase only

AGILE FRAMEWORKS - SCRUM

- Scrum is an agile process used for software development
- Scrum is a project management framework that is applicable to any project with aggressive deadlines, complex requirements and a degree of uniqueness
- In Scrum, projects move forward via a series of iterations called sprints
- Each sprint is typically two to four weeks long

AGILE FRAMEWORKS - SCRUM

- Scrum takes a highly iterative approach that focuses on defining key features and objectives prior to each sprint
- It is designed to reduce risk while providing value quickly
- Scrum starts with a requirement documented in form of user story that outlines how features should perform and be tested
- The project is implemented in a series of iterations called sprints
- To help the team work in this flexible way and avoid shifting priorities, Scrum requires that questions be answered from the very start

AGILE FRAMEWORKS - SCRUM

- Scrum is based on the collaboration between testers, developers and Business Analysts
- The communication between the team members and stakeholders is done in form of daily standups and sprint retrospectives
- Scrum Master is a role define for scrum projects, similar to the project manager role in traditional methods
- He is the one who helps keep the project on track by removing blockers
- The Scrum Master can be anyone in the team (e.g. a developer or a tester)
- All in all it requires faster iterations and stronger collaboration

AGILE FRAMEWORKS - SCRUM

- Scrum is best suited when customers and stakeholders want to be actively involved regularly to touch and validate the working product at its intermediate development stages
- Stakeholders meet during showcases where validations and change implementation are simultaneously performed
- Key team members of a Scrum approach include:

Product Owner	The product owner is the project's key stakeholder and represents users, customers and others in the process The product owner is often someone from product management or marketing, a key stakeholder or a key user
Scrum Master	The Scrum Master is responsible that the team is as productive as possible The Scrum Master does this by helping the team use the Scrum process, by removing impediments to progress, by protecting the team from outside, etc.
Developers	The software developers
Automation Engineers	experts who have the knowledge and ability to design, create, develop and manage systems
Testers	The testers of the software
Stakeholders	People, group or organization that have interest or concern in an organization

AGILE FRAMEWORKS – SCRUM – DEFINITIONS

- When working with Scrum you must know the following terms:

Scrum Team

A typical scrum team has between five and nine people, but Scrum projects can easily scale into the hundreds
However, Scrum can easily be used by one-person teams and often is
This team does not include any of the traditional software engineering roles such as programmer, designer, tester or architect
Everyone on the project works together to complete the set of work they have collectively committed to complete within a sprint
Scrum teams develop a deep form of camaraderie and a feeling that "we're all in this together."

Product Backlog	The **product backlog** is a list of features containing information e.g. priority new or change, risks and effort The term **backlog** is used in two different way The **product backlog** is a list of desired features for the product The **sprint backlog** is a list of tasks to be completed in a sprint
Sprint planning meeting	A **sprint planning meeting** is held at the beginning of each sprint during the meeting the product owner presents the top items on the product backlog ordered by priority The Scrum team analyzes and identifies what of the product backlog can be completed for that sprint The selected items will then be moved from the product backlog to a sprint backlog
Daily Scrum	A brief meeting called the daily scrum is conducted before starting the daily activities for each sprint The scope of this meeting is to set the context for each day's work, verify if the project is on track and motivate the team to keep the deadlines All team members are required to attend to the daily scrum

Sprint review meeting	A sprint review meeting is held at the end of each sprint, during which the team demonstrates what has been delivered and get the approval to close the sprint A demo version of the final product is displayed and the new functionalities or fixes of previous bugs or changes are demonstrated to the stakeholders
Sprint retrospective	At the end of each sprint, the team conducts a sprint retrospective This is done in form of a meeting during which the team (with the Scrum Master and the Product Owner) outline the lessons learned points collected while accumulating experience during the sprint execution (what has been done well and what has not been done well) This analysis helps understand how well Scrum is working for them and what changes they need to adopt in order to improve their future work

AGILE FRAMEWORKS – SCRUM – ORGANIZATION STRUCTURE

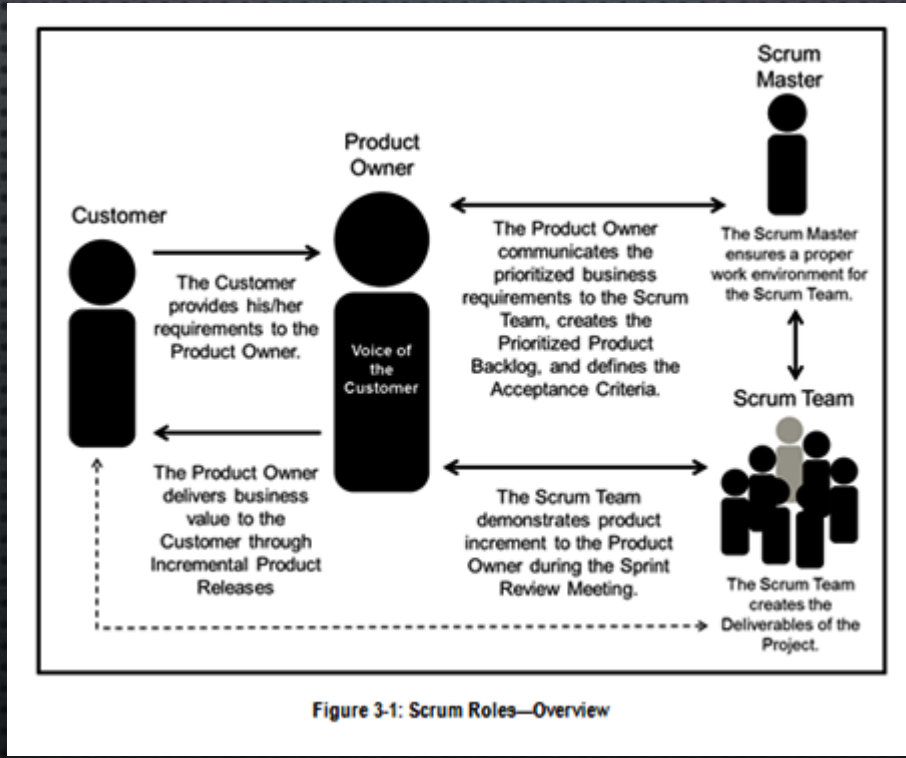


Figure 3-1: Scrum Roles—Overview

AGILE FRAMEWORKS – SCRUM – BEST PRACTICES

- We have already mentioned strong communication, collaboration and adaptability are important best practices for Scrum
- Requirement and acceptance criteria definition based on the communication (in the form of a user story) with the customer
- This process will assure the delivery of documented requirements without misunderstandings or misinterpretations
- Using the acceptance criteria to develop code and approve deliveries
- Testing the code in sandbox-like environments as well as production-like environments prior to deploying it to production

AGILE FRAMEWORKS – SCRUM – SCRUM PRINCIPLES

- Scrum principles are the core guidelines for a k and should mandatorily be used in all Scrum
- The six Scrum principles are

Empirical Process Control
Self-organization
Collaboration
Value-based Prioritization
Time-boxing
Iterative Development

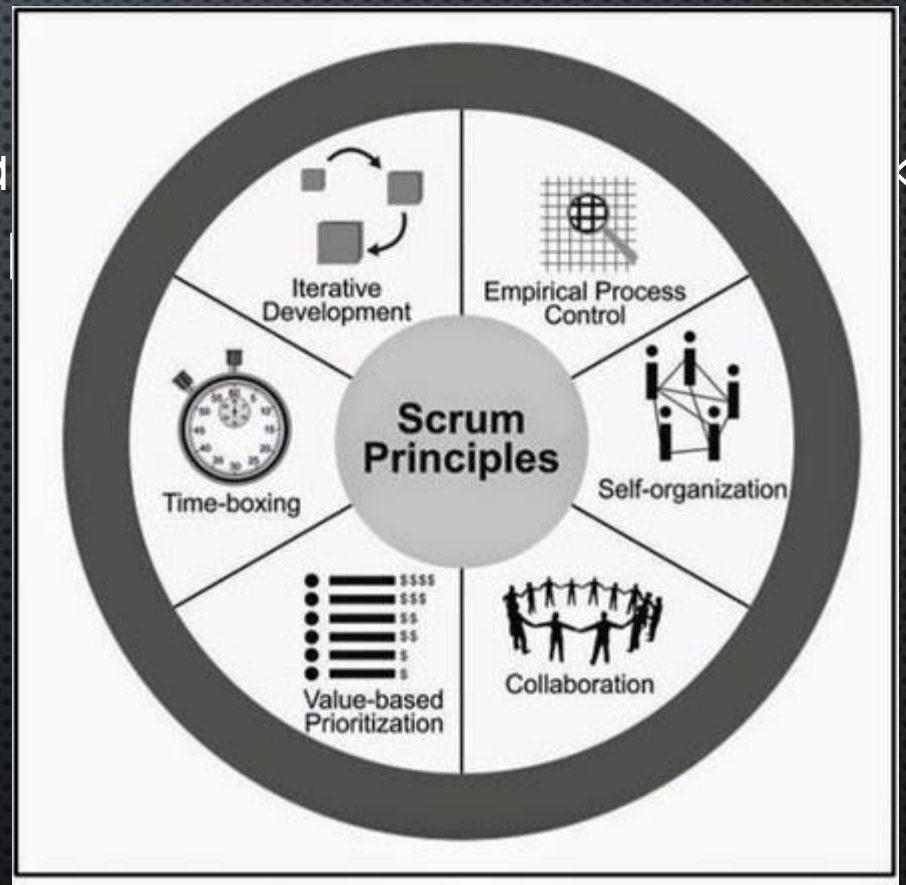


Figure 1-3: Scrum Principles

THE SCRUM SPRINT CYCLE

- Scrum has a two to four week Sprint cycle in which the team delivers their work, driving towards potentially shippable
- A key strength of Scrum lies in its use of cross-functional, self-organized, and empowered teams who divide their work into short, concentrated work cycles called Sprints
- Each Sprint begins with a Sprint Planning Meeting
- The scope of the Sprint Planning Meeting is to define the high priority User Stories to develop and release at that Sprint
- The duration of a Sprint is between one to six weeks
- The scope is to create potentially shippable Deliverables or product increments

Project Business Case

Project Vision Statement

Product Backlog

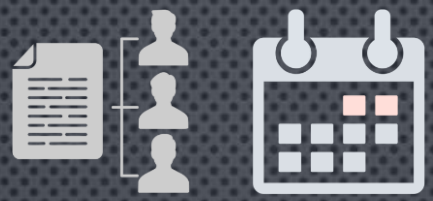
The Scrum cycle begins with a Stakeholder Meeting, during which the **Project Vision** is created

A Scrum project involves a collaborative effort to create a new product, service, or other result as defined in the **Project Vision Statement**

The **product backlog**, which has been prioritized by the product owner and contains everything desired in the product that's known at the time

The **Product Owner** then develops a **Prioritized Product Backlog** which contains a prioritized list of business and project requirements written in the form of User Stories

The team selects some amount of work from the product backlog and commits to completing that work during the sprint
Part of figuring out how much they can commit to is creating the **sprint backlog**



Release Planning Schedule

A very high-level plan for multiple Sprints is created during the **Release planning**
It is a guideline that reflects expectations about which features will be implemented and when they are completed

Project Business Case

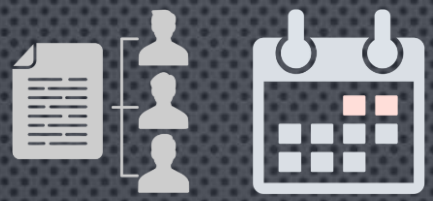
Project Vision Statement

Product Backlog

Sprint Backlog

The **Sprint Backlog** is the set of Product Backlog items selected for the Sprint
It includes a plan for delivering the product Increment and realizing the Sprint Goal

During the Sprint, short, highly focused **Daily Standup Meetings** are conducted where team members discuss daily progress



Release Planning Schedule

Daily Standup

Implementation

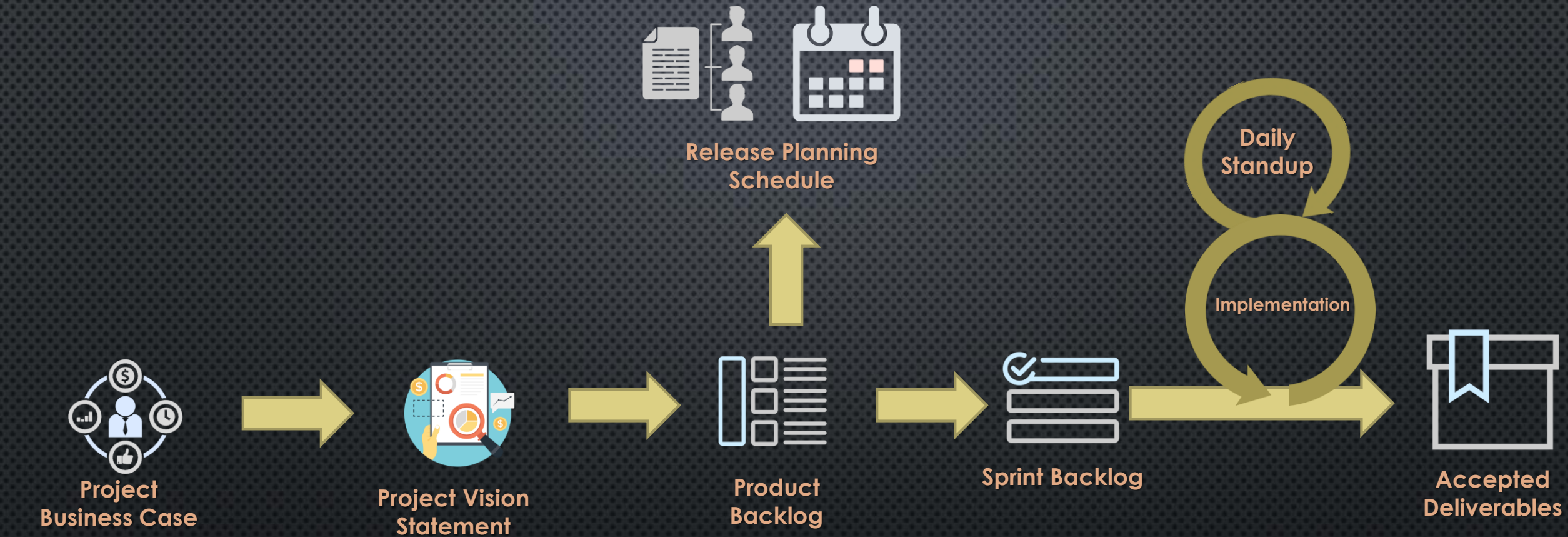
Project Business Case

Project Vision Statement

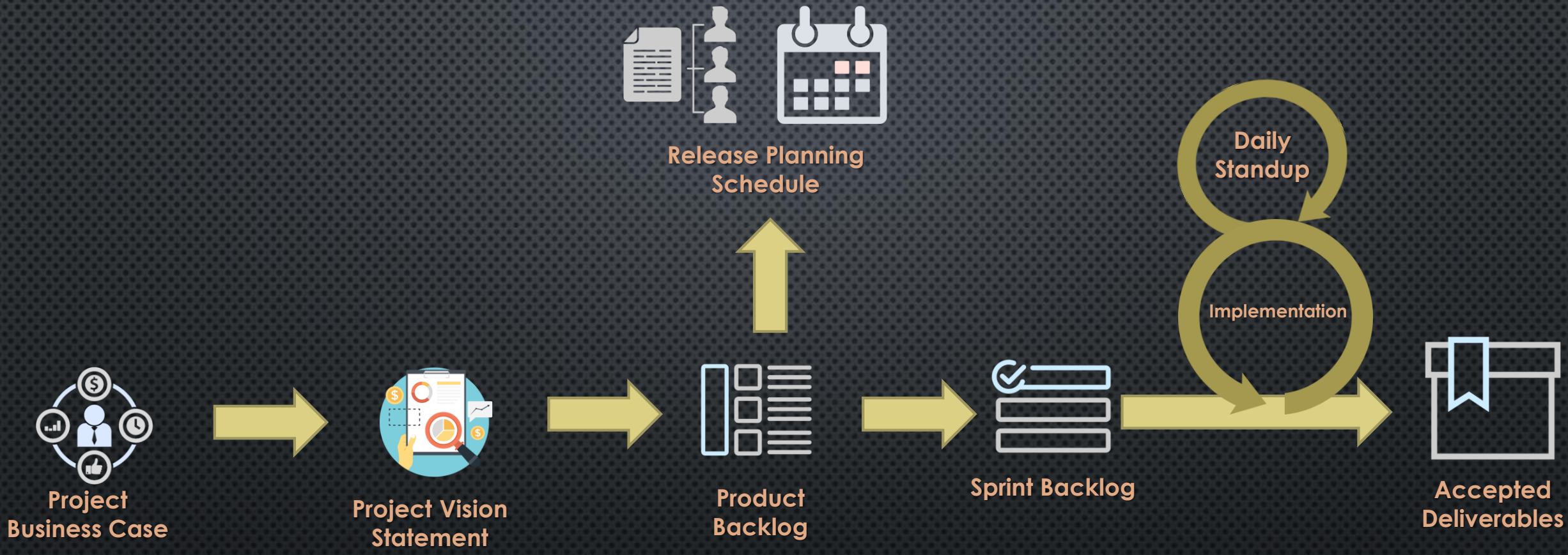
Product Backlog

Sprint Backlog

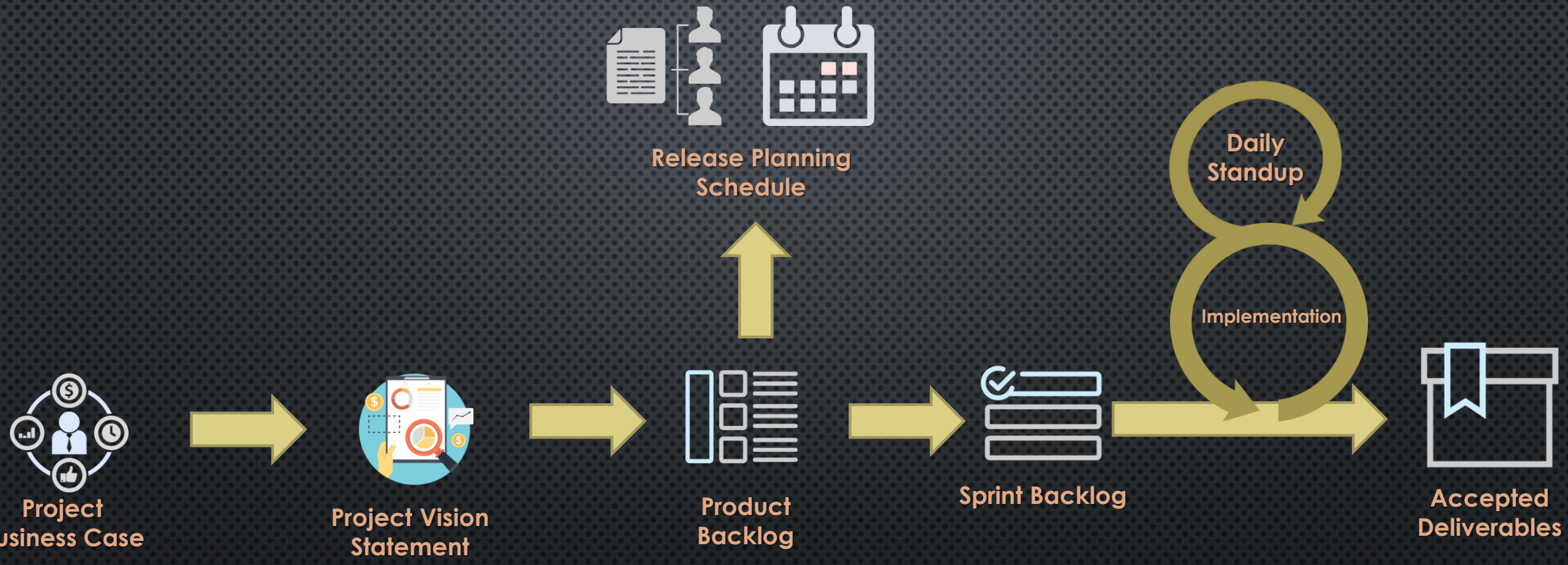
Accepted Deliverables



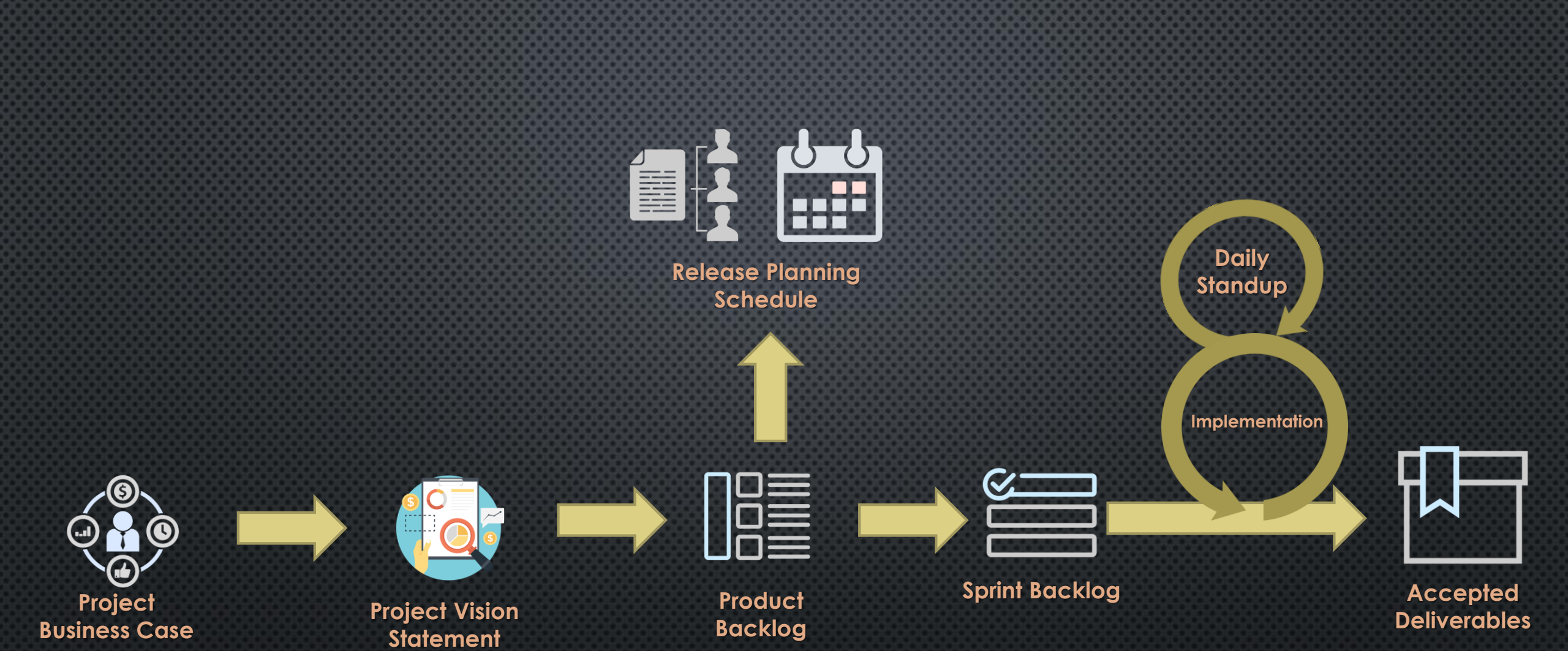
Toward the end of the print, a **Sprint Review Meeting** is held during which the Product Owner and relevant stakeholders are provided a demonstration of the Deliverables



The team has produced a potentially **shippable product increment**
Potentially shippable means that the feature is working and matches the standard of quality specified in the acceptance criteria for that feature



The Product Owner accepts the Deliverables only if they meet the predefined Acceptance Criteria



The Sprint cycle ends with a **Retrospect Sprint Meeting** where the team discusses ways to improve processes and performance as they move forward into the subsequent Sprint

SCRUM RELEASE PLANNING

- Release planning is longer-term planning to understand when each feature will be implemented and at which cost
- Release planning derives from a balance of customer expectations and overall quality against the constraints of scope, schedule, and budget
- In an agile organization you can have 3 different cadence of release planning


Release every feature
Release 1
Feature A
Release 2
Feature B
Release 3
Feature C
Release every Sprint
Release 1
Feature A1
Feature A2
Feature A3
Release 2
Feature B1
Feature B2
Feature B3
Release 3
Feature C1
Feature C2
Feature C3
Release after multiple Sprints
Release 1
Feature A1
Feature A2
Feature A3
Feature B1
Feature B2
Feature B3
Feature C1
Feature C2
Feature C3

SPRINT PLANNING

- Sprint Planning is the result of the task to plan the work to be performed in a Sprint
- This plan is created by the collaborative work of the entire Scrum Team
- Sprint Planning is limited to a maximum of eight hours for a Sprint
- This is valid for one-month-long Sprints
- For shorter Sprints, the event is usually shorter
- The Scrum Master ensures that the planning event takes place and that all members involved understand its purpose
- The Scrum Master teaches the Scrum Team to keep it within the limited duration

Sprint Planning answers the following:

- What can be delivered in the Increment resulting from the upcoming Sprint?
- How will the work needed to deliver the Increment be achieved?

- The work for a sprint is selected from the Product Backlog and pulled into the Sprint Backlog
- The work in the Sprint Backlog is not a commitment, it is a forecast
- A Sprint is assign to a time frame, not to the work planned for it
- In a Sprint some open bugs from previous releases can be fixed and fixes can be included in the Sprint release

THE SCRUM DAILY STAND-UPS

- The Daily Stand Up Meeting serves the daily exchange between the team members and is an elementary component of the SCRUM process
- This meeting is also suitable for other project management models
- This should not be confused with reporting to superiors or the SCRUM master/product owner and therefore does not fulfill any control of the team
- During the daily stand-ups meeting only three questions are answered alternately by the team

1. What did I achieve yesterday?
2. What am I doing today?
3. Were there any problems/risks/ impediments?

THE SCRUM DAILY STAND-UPS

- Excessive discussions should be interrupted by the team and/or the SCRUM Master (depending whom you decided to be the coordinator)
- The reason why the Meeting Daily is called "Stand-Up" is that by standing together long discussions are automatically avoided
- The concentration of the team is given and the duration of the meeting can be kept

THE SCRUM DAILY STAND-UPS: STRUCTURE

Participants	• SCRUM Master • Product Owner • Team
Inputs	• Achieved objectives of the last working day • Planned user stories for the day
Outputs	• If necessary, problems/malfunctions to be eliminated • Dependencies between tasks that have become known through the stand-up
Frequency and duration	• Daily at a fixed time at the beginning of the day • approx. 0.25 hours
Owner	• The Team

TIPS FOR A SUCCESSFUL STAND UP

Ad hoc topics or unplanned tasks

It is utopian to believe that due to SCRUM there will be no ad hoc topics or disturbances in the daily project routine
A good idea here is that the product owner presents such topics in the daily stand up and that they are discussed together with the team and provided with complexity points
The team then decides whether the topic can still be implemented within the sprint or which other topic must be removed from the sprin
In general, of course, only really important and urgent topics should appear here and this should not be used as a "back door"

Status Meeting	The stand-up meeting is not a status meeting It is not about reporting the status to the boss The goal is to exchange information and synchronize with each other Make sure that the SCRUM Master is not addressed: the team members must look at each other The SCRUM Master or Product Owner should stay out In fact it is helpful for him/her to stand apart from the SCRUM Master
Planning Meeting	If new, important requirements arise, the discussion or definition should not be part of the stand-up Keep it short and focus on the above questions The requirements for the new requirement should be discussed at another meeting or immediately afterwards
Discussions	Discussions are especially important when it comes to detailed questions or finding solutions, but should be postponed to the time after the stand-up Depending on the requirements, smaller groups may be formed to continue the discussion

Listen and let speak	Active listening is important at the meeting Even if at first glance you may not be interested in the topic mentioned, there are definitely later starting points In addition, everyone should be allowed to speak and not be interrupted, unless they deviate from the above questions
Buzzer	The introduction of a "buzzer" is a very helpful idea If the discussion in the stand-up degenerates, everyone of the team can press the buzzer and focus the stand-up on the actual questions
What have I achieved vs. what have I done?	Much more important than what was done the day before is the question of what was achieved On the one hand this reduces the flow of words and on the other hand it strengthens the focus on the topics that bring the team and the project forward

THE SCRUM DAILY STAND-UPS: BENEFITS

- The workflow is kept running
- Keeps the meeting short (by standing)
- Problems are discovered earlier
- Increased sense of responsibility, improved communication and collaboration within the team
- Encourages self-organization of the team and personal planning of each team member
- The team can tackle problems better if necessary and make small course corrections
- Personal contact is facilitated (when the respective people are on site)

THE SPRINT REVIEW MEETING

- At the end of each sprint, the team presents the results of their work to the product owner and all interested stakeholders and collects feedback (opinions, suggestions for improvement, praise and criticism)
- The Scrum Team and stakeholders are invited to the Sprint Review Event
- The results of the work are presented in form of demonstration and the goal is to get the approval by the product owner
- The approval is a prerequisite to move to the next sprint
- In case of open points, these can be listed (as bugs, issues, etc.) to be managed as part of the backlog of the next sprint

THE SPRINT REVIEW MEETING

- The Sprint Review is a place to discuss the marketplace changes, examine the completed Sprint as an event, update the release schedule, discuss the Product Backlog and the possible focus for the next Sprint
- This is where the dialog between the Scrum Team and the stakeholders takes place and feedback on product Increment is obtained
- The number of changes to the Product Backlog can be an important sign of how healthy your Scrum is

THE SPRINT REVIEW MEETING – HOW TO CONDUCT

- The Product Owner opens the meeting and tells the attendees what the Development Team completed during the current Sprint (what has been "Done" and “not Done”)
- The Development Team demonstrates the "Done" functionality (Demo), answers the questions and discusses problems they met on their way
- The Product Owner discusses the current state of the Product Backlog, marketplace changes, and forecasts the likely release dates
- The attendees collaborate on the Product Backlog Items, which can be completed during the next Sprint. Thus, the stakeholders get a shared understanding of the future work
- A review of the budget, timeline, potential capabilities follows

The Sprint Review is a great tool for getting feedback, while the number of changes to the Product Backlog after the Sprint Review can be an important indicator of how healthy your Scrum is

THE SPRINT REVIEW MEETING

- The sprint review meeting is intentionally kept very informal, typically with rules forbidding the use of PowerPoint slides and allowing no more than two hours of preparation time for the meeting
- A sprint review meeting should not become a distraction or significant detour for the team; rather, it should be a natural result of the sprint

ROLES AND RESPONSIBILITIES FOR THE EVENT

Product Owner

- Introduction to sprint goals
- Introduction of planned top requirements that team has committed to deliver
- Sprint status overview
- Information of the increase of quality (mention defects solved and improvements)

Scrum Master

- Organization of the meeting, and invitation to participants
- Moderator of the meeting
- Evidence of feedback

Team

- Informing about the sprint status
- Live demonstration of the functionalities

A typical Agenda for a Sprint Review Event can be:

START	DUR.	ACTIVITY	DESCRIPTION	WHO
09:10	10 min	**Goal of the Sprint**	• Introduction into Sprint Goals • Information about an impact of the sprint on release plans • Review of the product roadmap	Product Owner
09:20	5 min	**Review of the top requirements**	• Review of up to three top requirements related to sprint goal	Product Owner
09:25	15 min	**Sprint Status**	• Share information about plan vs reality • Review of the sprint statistics • Review of the important technology changes or improvements, enablers • Statistics of bug fixes • Current program increment (release) statistics	Scrum Master
9:40	15 min	**Demonstration**	• Live demonstration of up to three completed requirements	Team
9:55	10 min	**Feedback**	• Collecting the feedback from stakeholders	Scrum Master
10:05	5 min	**Closing**	• Information about the next sprint review • Publishing sprint review presentation	Scrum Master

THE RETROSPECTIVE MEETING

- The **sprint retrospective** is a meeting facilitated by the Scrum Master at which the team discusses the just-concluded sprint and determines what could be change that might make the next sprint more productive
- The sprint review looks at what the team is building, whereas the retrospective looks at how they are building it
- The sprint retrospective is an important mechanism that allows a team to continuously evolve and improve throughout the life of a project

THE RETROSPECTIVE MEETING

- It is important that everyone, including the team, product owner, and Scrum Master, get a chance to air their opinions in an open, honest, yet constructive atmosphere
- It often also helps management to get feedback from the team about the work and progress of project
- Retrospectives aren't used to record complaints, rather the team tries to find effective solutions to problems and develops action plans

THE RETROSPECTIVE MEETING

- The Sprint Retrospective is held after the sprint review at the end of each sprint
- During the retrospective, the team self-identifies elements of the process that did or did not work during the sprint, along with potential solutions
- It aims to continuously improve the processes
- Sprint Retrospective meetings can be facilitated by asking each person in the team to answer the following questions:

- What went well during the sprint?
- What would we like to change?
- How can we implement that change?

Alternatively, instead of asking what went well and what didn't go well, the following questions may be asked:

- What should we start doing?
- What should we stop doing?
- What should we continue to do?

THE RETROSPECTIVE MEETING – OTHER TOPICS

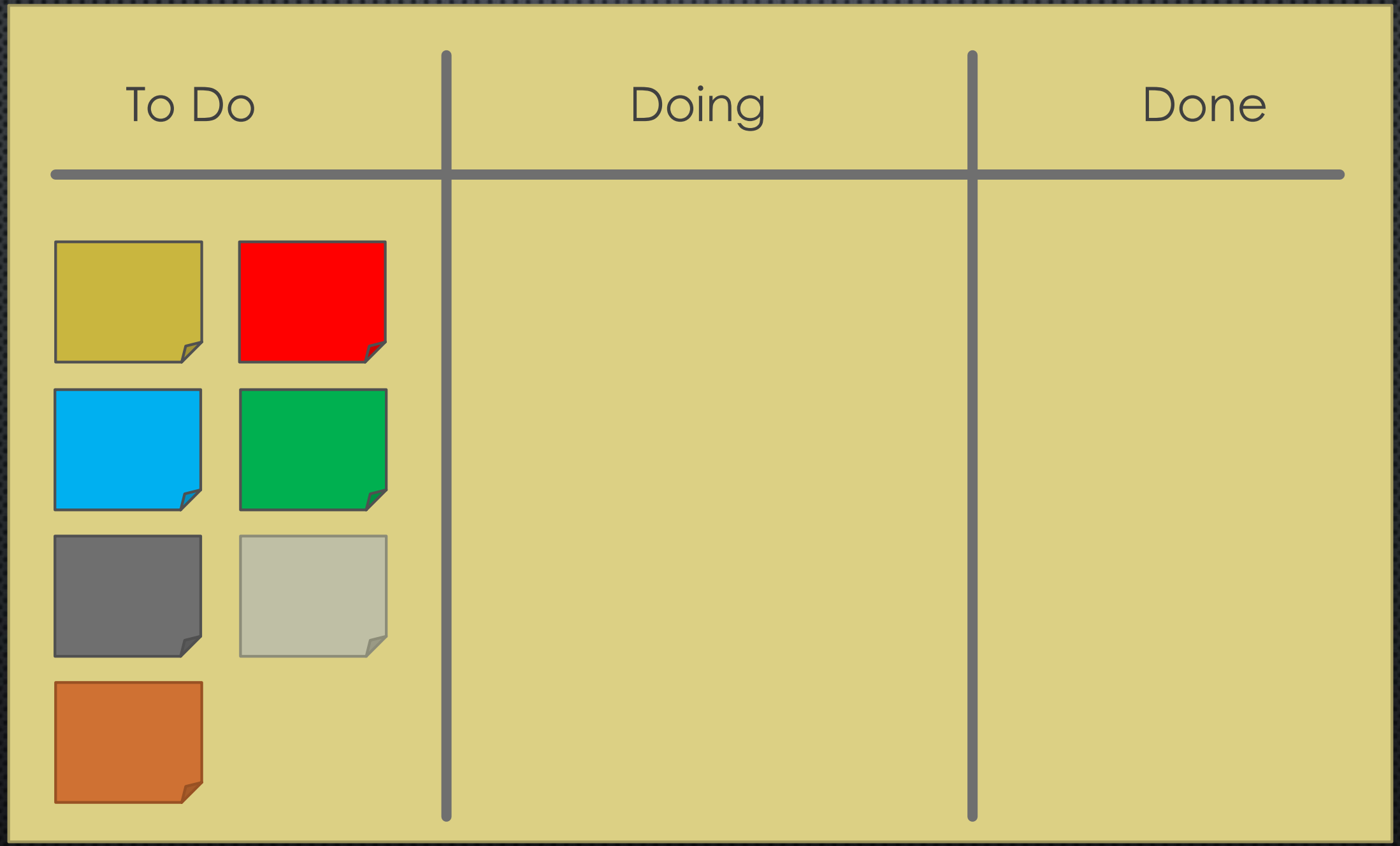
Results	• Compare the amount of work planned with what the development team actually completed • Review the sprint burndown chart and what it tells the development team about how they're working
Resources	• Discuss team composition and alignment
Relationships	• Talk about communication, collaboration, and working in pairs
Processes	• Go over getting support, development, and code review processes
Tools	• How are the different tools working for the scrum team? • Think about the artifacts, electronic tools, communication tools, and technical tools
Productivity	• How can the team improve productivity and get the most work done within the next sprint?

THE RETROSPECTIVE MEETING

- Answers to questions during the meeting have to be very detailed and specific in order to take effective actions afterwards
- The retrospective meeting should be held immediately after the Sprint review meeting
- It is recommended that the entire team participate
- In fact any issues or concerns that the teams face during that Sprint must be described and addressed to improve the processes in upcoming Sprints
- Important, the Scrum Master has the task to prioritize actions and track (document) lessons learned based on the meeting results

THE KANBAN AGILE METHOD

- Kanban is a very simple Agile methodology
- Kanban is basically a way of organizing the work according to the information on a prioritized to-do list
- Requirements in Kanban are tracked by their current stage in the process (to-do, in development, in test, done)
- Unlike scrum, Kanban is not time-based, rather, it is based solely on priority
- When a developer is ready for the next task, he/she pulls it from the to-do list


To Do
Doing
Done

THE KANBAN AGILE METHOD

- Since there are fewer planning meetings, this approach means the team needs to be extremely close
- In this type of environment, if developers work much faster than the testers, bottlenecks will crop up
- In these situations, anyone on the team should jump in and help in different areas
- Of course meeting this need requires a great deal of flexibility and adaptability

THE KANBAN AGILE METHOD

- The difference between Kanban and Waterfall is that the requirements can change
- In fact the testing team plans the testing activity of each requirement once the developer have selected it from the top of the backlog
- This has the advantage to reduce the overhead in planning and can reduce the timing
- With Kanban, releases still get planned, but teams usually don’t promise anyone features by certain dates unless the item in question is near the top of the backlog

THE KANBAN AGILE METHOD

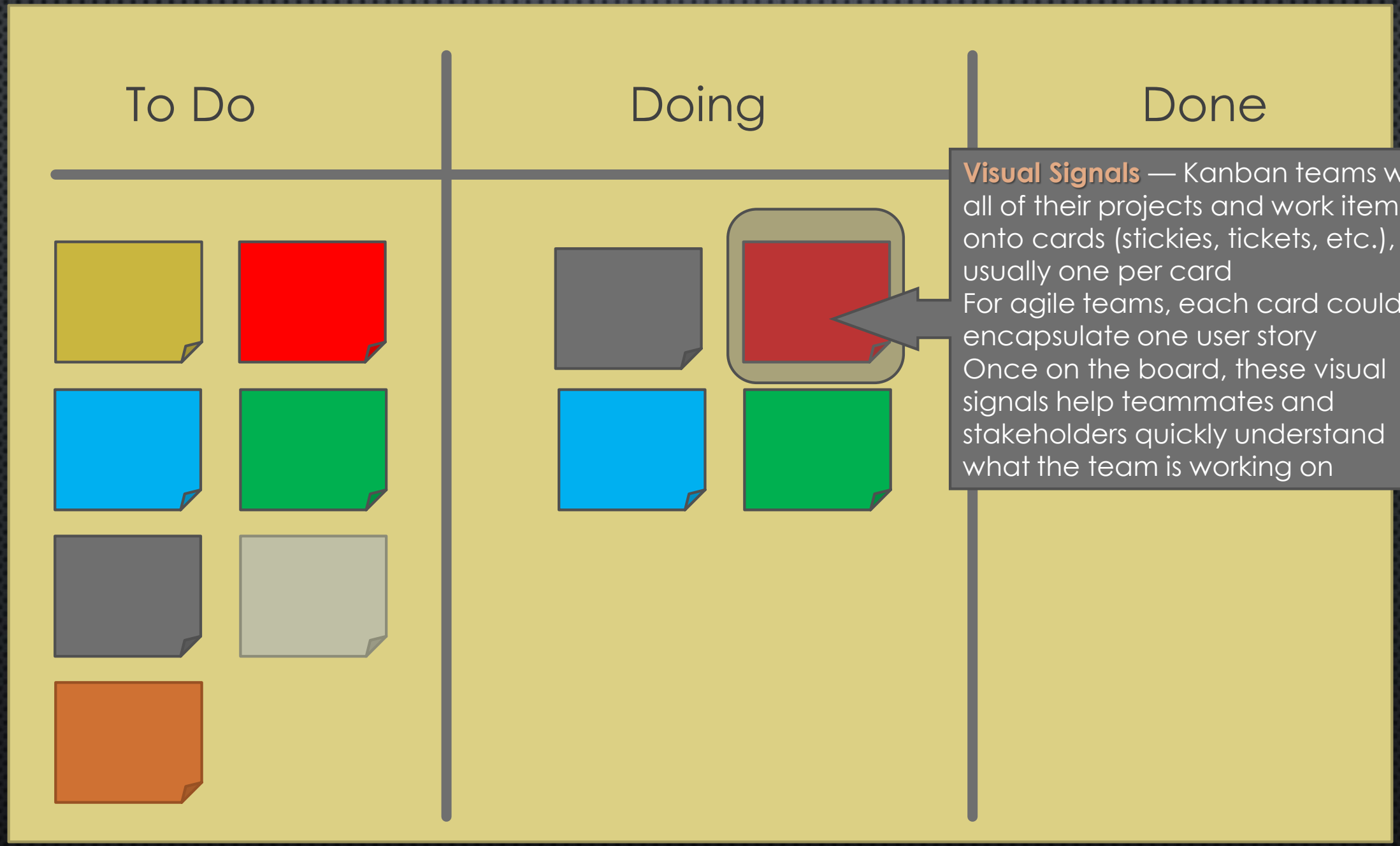
- This methodology offers the quickest way to bring code to production, but has some negative technical effects on the code
- Developing without always knowing what's next can have negative impact if we want to create reusable code
- Kanban is best suited for small teams or when it is not mandatory to define a detailed release plan
- It is a preferable methodology for maintenance work since bugs are not always straightforward and often require research to resolve, which makes time management challenging

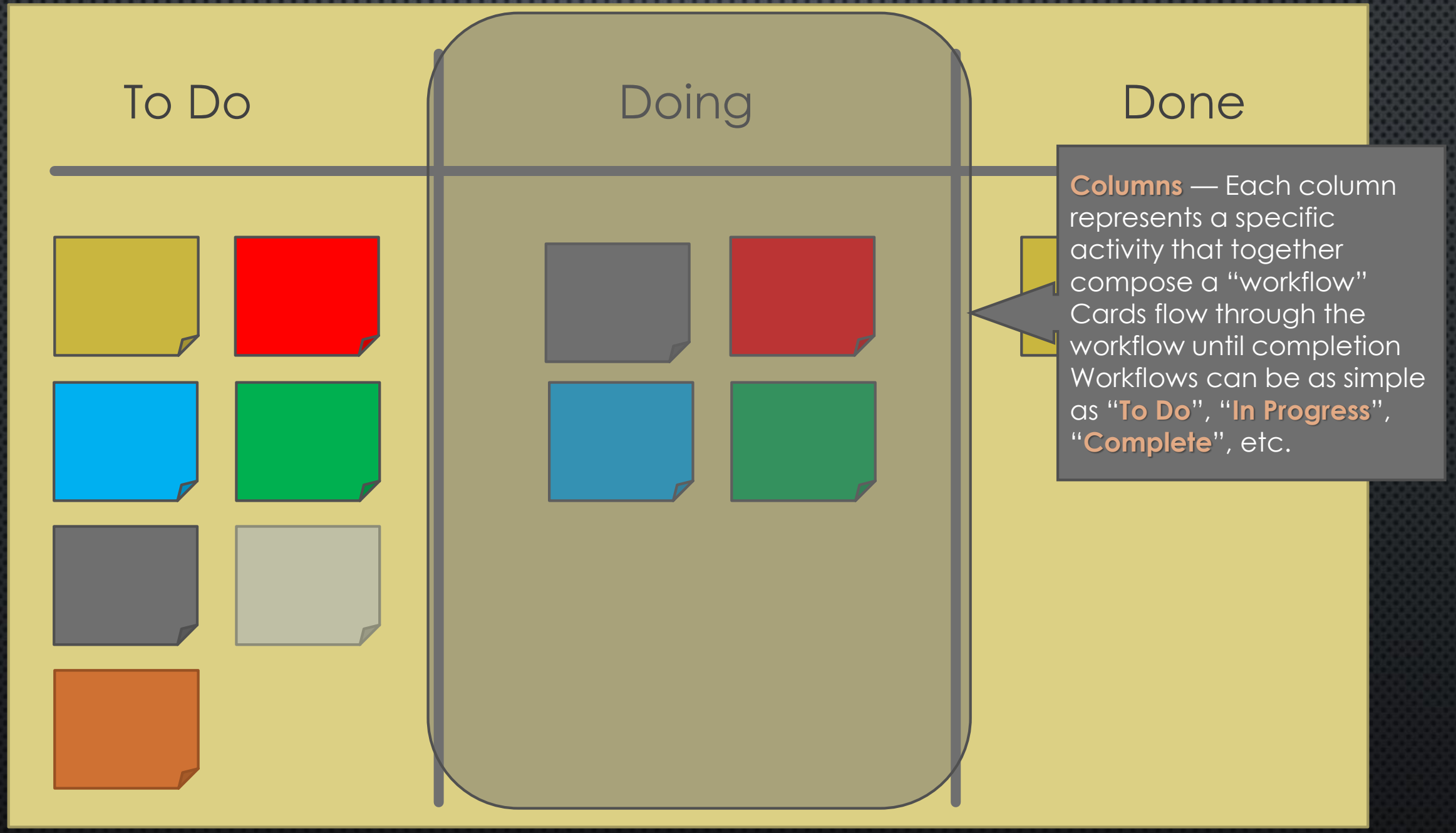
THE KANBAN AGILE METHOD

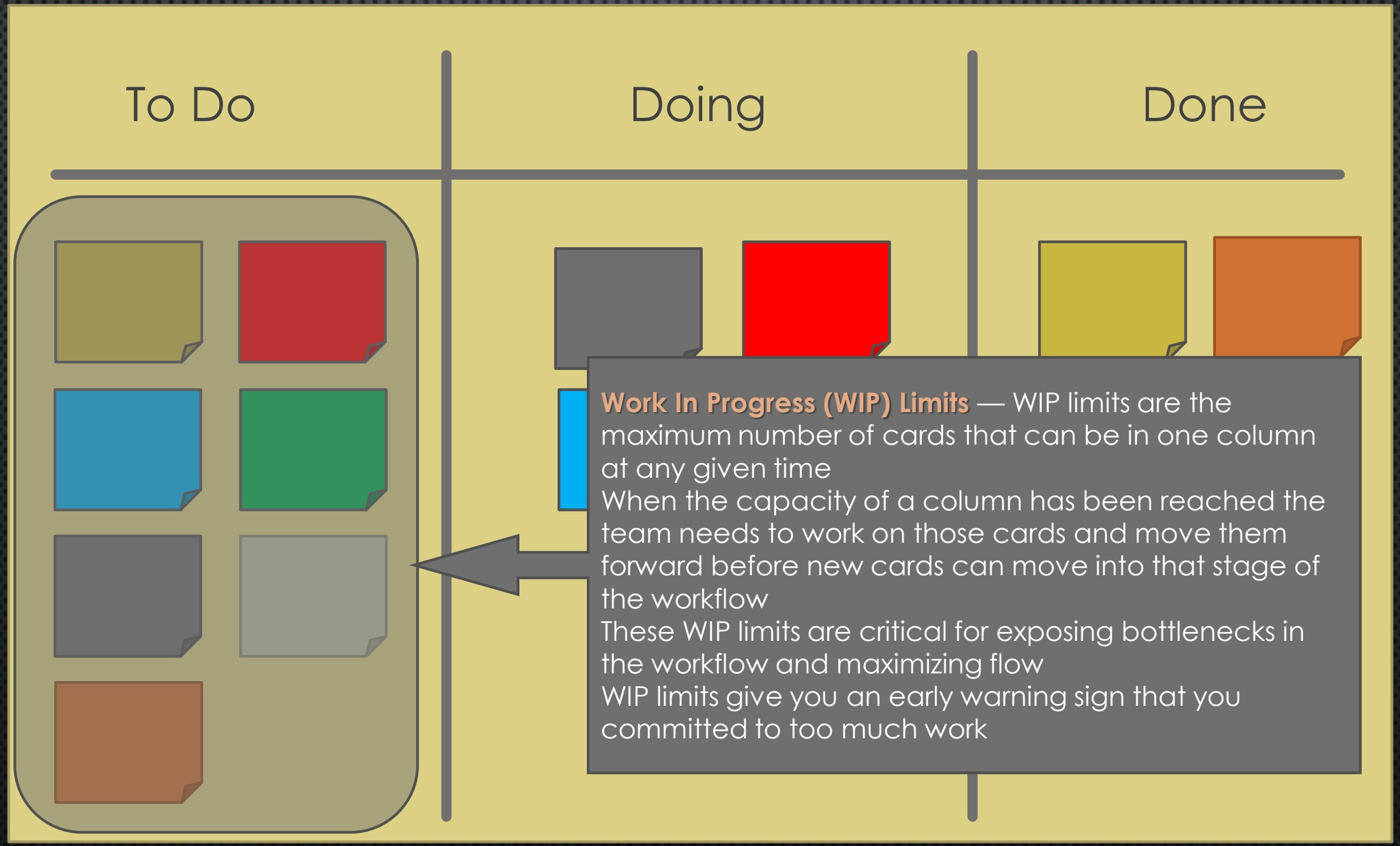
- Teams that cannot minimize the amount of planning for issues are likely better off following a Scrum or Waterfall methodology
- For a successful implementation of this methodology it is recommended to establish an open communication between the business owners, developers and testers
- The team should have the flexibility to take on other roles outside of their core responsibilities in order to help clear bottlenecks
- Making everyone an owner of the product so that they care fully about the result is a key for this method

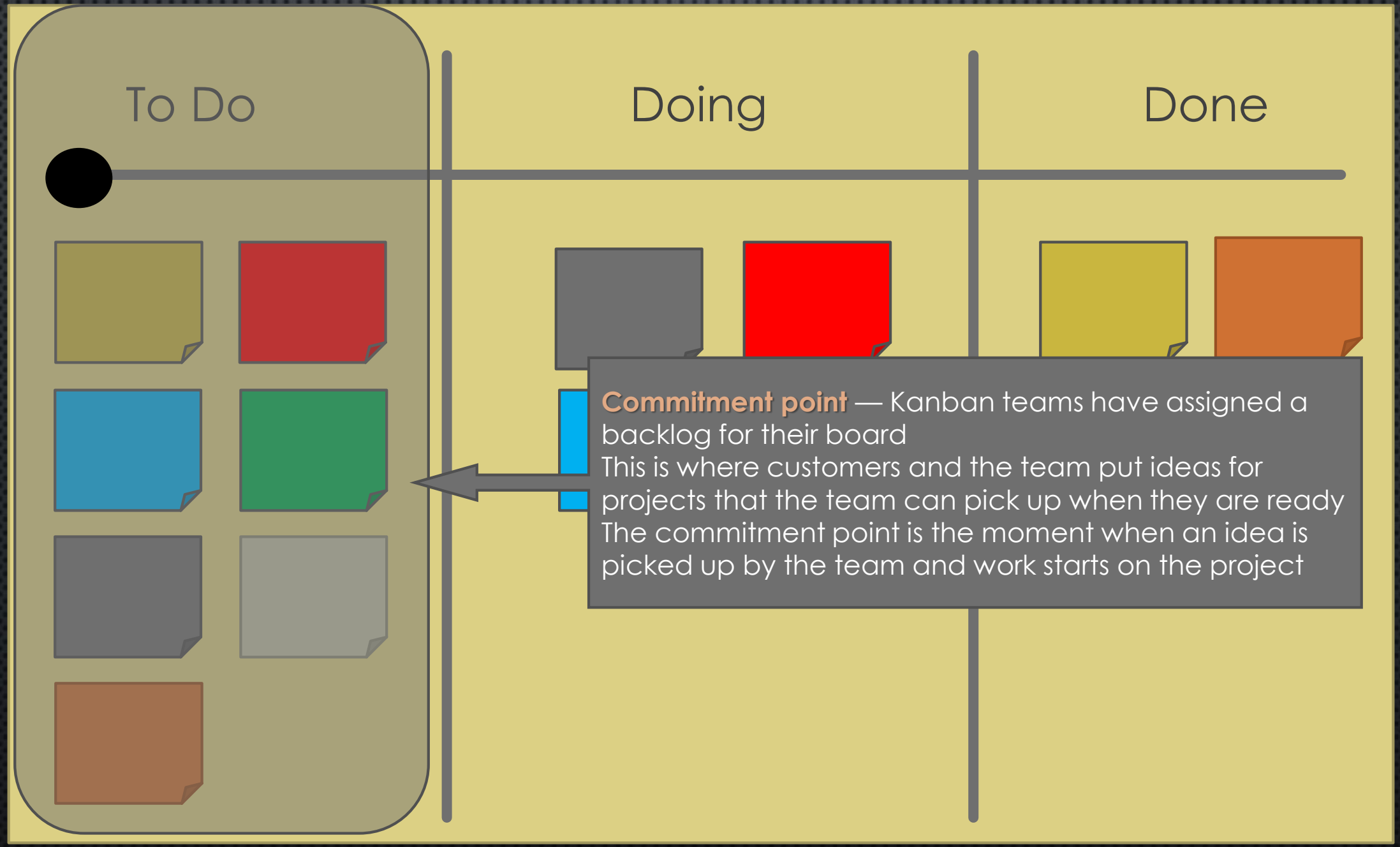
KANBAN BOARDS

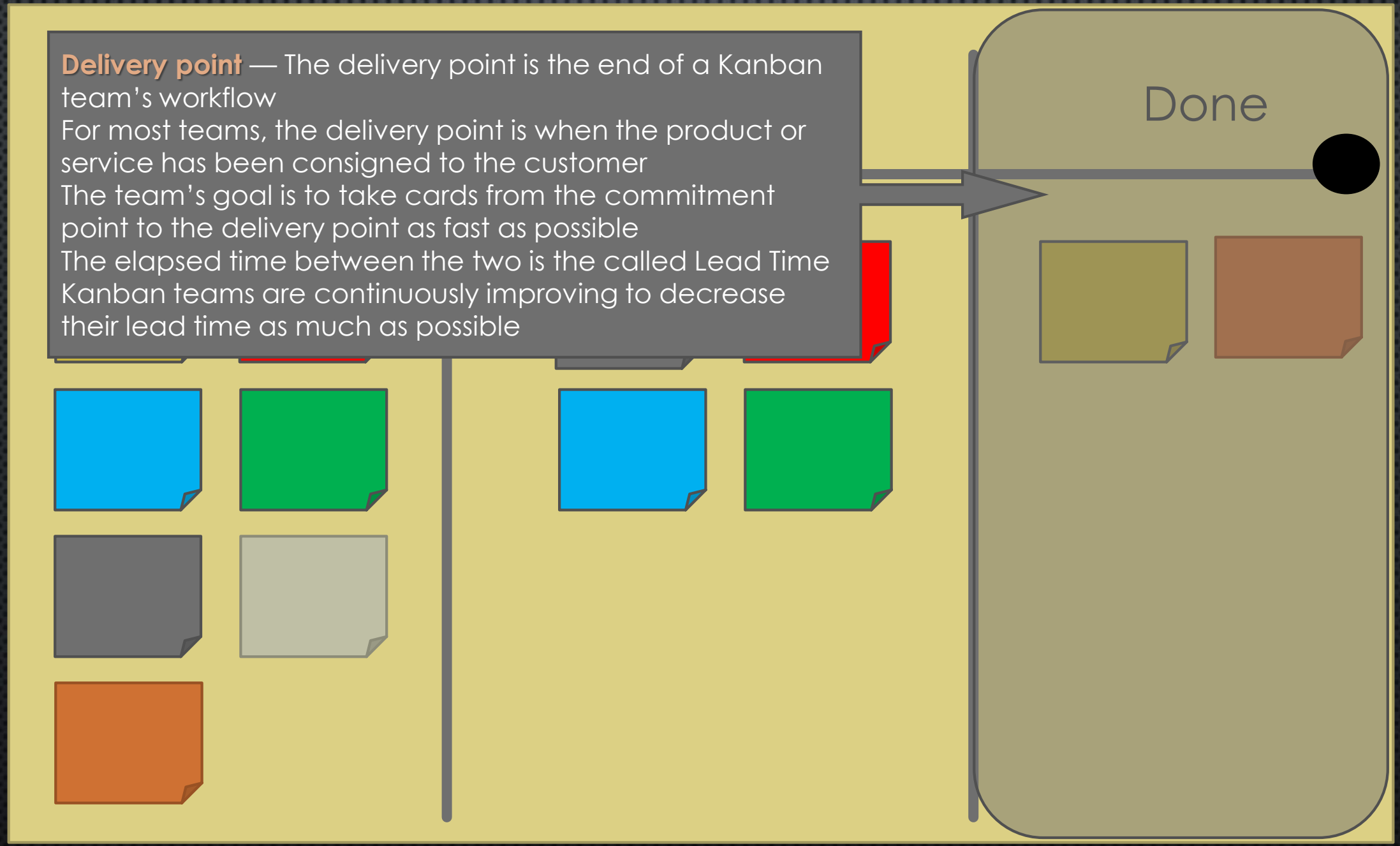
- A kanban board is an agile project management tool designed to help visualize work, limit work-in-progress, and maximize efficiency
- Kanban boards use cards, columns, and continuous improvement to help the teams commit to the right amount of work
- Kanban boards can be broken down into five components:
  - Visual signals
  - Columns
  - work-in-progress limits
  - A commitment point
  - A delivery point


To Do
Doing
Done
Visual Signals — Kanban teams write all of their projects and work items onto cards (stickies, tickets, etc.), usually one per card
For agile teams, each card could encapsulate one user story
Once on the board, these visual signals help teammates and stakeholders quickly understand what the team is working on


To Do
Doing
Done
Columns — Each column represents a specific activity that together compose a "workflow"
Cards flow through the workflow until completion
Workflows can be as simple as "To Do", "In Progress", "Complete", etc.


To Do
Doing
Done
Work In Progress (WIP) Limits — WIP limits are the maximum number of cards that can be in one column at any given time
When the capacity of a column has been reached the team needs to work on those cards and move them forward before new cards can move into that stage of the workflow
These WIP limits are critical for exposing bottlenecks in the workflow and maximizing flow
WIP limits give you an early warning sign that you committed to too much work


To Do
Doing
Done
Commitment point — Kanban teams have assigned a backlog for their board
This is where customers and the team put ideas for projects that the team can pick up when they are ready
The commitment point is the moment when an idea is picked up by the team and work starts on the project

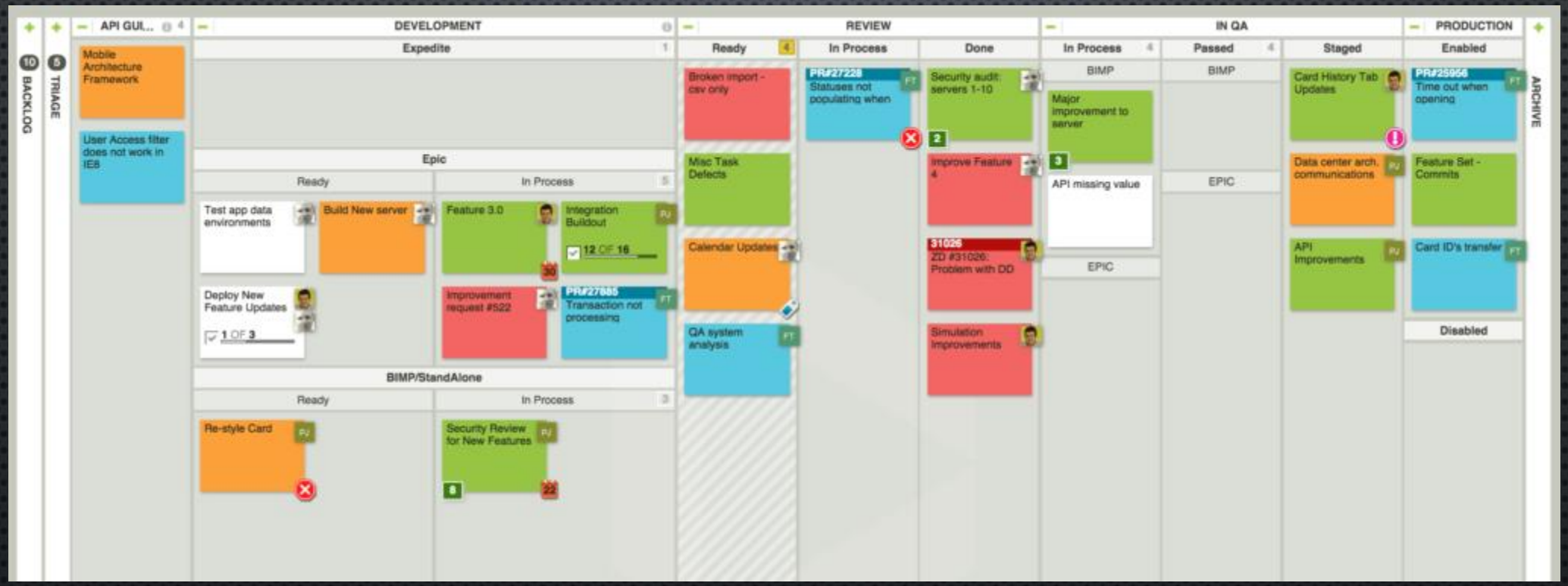

Delivery point — The delivery point is the end of a Kanban team's workflow
For most teams, the delivery point is when the product or service has been consigned to the customer
The team's goal is to take cards from the commitment point to the delivery point as fast as possible
The elapsed time between the two is the called Lead Time
Kanban teams are continuously improving to decrease their lead time as much as possible
Done

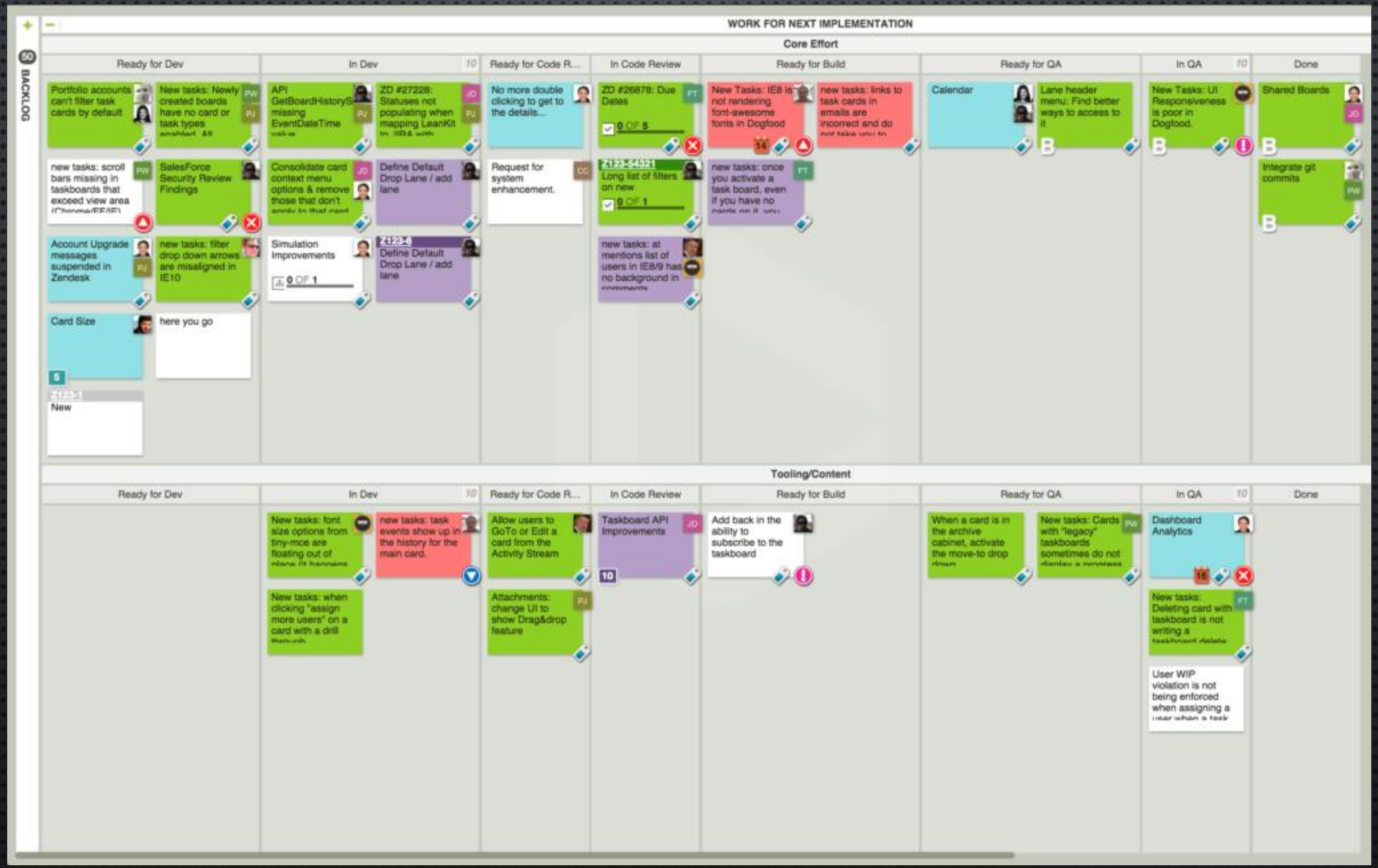
KANBAN VS. SCRUM BOARD

- The difference between Kanban and scrum is actually quite subtle
- By most interpretations, scrum teams use a Kanban board, just with scrum processes, artifacts, and roles along with it
- There are, however, some key differences

- Scrum sprints have start and stop dates whereas kanban is an ongoing process
- Team roles are clearly defined in scrum (product owner, dev team, and scrum master), while Kanban has no formal roles. Both teams are self-organized
- A Kanban board is used throughout the lifecycle of a project whereas a scrum board is cleared and recycled after each sprint
- A scrum board has a set number of tasks and strict deadline to complete them
- Kanban boards are more flexible with regards to tasks and timing. Tasks can be reprioritized, reassigned, or updated as needed

Show from https://leankit.com/learn/kanban/kanban-board-examples-for-development-and-operations/


BACKLOG
TRIAGE
API GUI...
Mobile Architecture Framework
User Access filter does not work in IE8
DEVELOPMENT
Expedite
Epic
Ready
In Process
Test app data environments
Build New server
Feature 3.0
Integration Buildout
12 OF 16
Deploy New Feature Updates
1 OF 3
Improvement request #522
PR#27885
Transaction not processing
BIMP/StandAlone
Ready
In Process
Re-style Card
Security Review for New Features
REVIEW
Ready
In Process
Done
Broken import - csv only
Misc Task Defects
Calendar Updates
QA system analysis
PR#27228
Statuses not populating when
Security audit: servers 1-10
Improve Feature 4
31026
ZD #31026: Problem with DD
Simulation Improvements
IN QA
In Process
Passed
Staged
BIMP
Major improvement to server
API missing value
EPIC
BIMP
EPIC
Card History Tab Updates
Data center arch. communications
API Improvements
PRODUCTION
Enabled
Disabled
PR#25956
Time out when opening
Feature Set - Commits
Card ID's transfer
ARCHIVE


BACKLOG
50
WORK FOR NEXT IMPLEMENTATION
Core Effort
Ready for Dev
In Dev
10
Ready for Code R...
In Code Review
Ready for Build
Ready for QA
In QA
10
Done
Portfolio accounts can't filter task cards by default
New tasks: Newly created boards have no card or task types
API GetBoardHistorySi... missing EventDateTime
ZD #27228: Statuses not populating when mapping LeanKit
No more double clicking to get to the details...
ZD #26878: Due Dates
0 OF 5
New Tasks: IE8 is not rendering font-awesome fonts in Dogfood
14
new tasks: links to task cards in emails are incorrect and do
Calendar
Lane header menu: Find better ways to access to it
B
New Tasks: UI Responsiveness is poor in Dogfood.
B
Shared Boards
B
new tasks: scroll bars missing in taskboards that exceed view area
SalesForce Security Review Findings
Consolidate card context menu options & remove those that don't
Define Default Drop Lane / add lane
Request for system enhancement.
4123-54321
Long list of filters on new
0 OF 1
new tasks: once you activate a task board, even if you have no
Integrate git commits
B
Account Upgrade messages suspended in Zendesk
new tasks: filter drop down arrows are misaligned in IE10
Simulation Improvements
0 OF 1
4123-4
Define Default Drop Lane / add lane
new tasks: at mentions list of users in IE8/9 has no background in
Card Size
5
here you go
4123-1
New
Tooling/Content
Ready for Dev
In Dev
10
Ready for Code R...
In Code Review
Ready for Build
Ready for QA
In QA
10
Done
New tasks: font size options from tiny-mce are floating out of
new tasks: task events show up in the history for the main card.
Allow users to GoTo or Edit a card from the Activity Stream
Taskboard API Improvements
10
Add back in the ability to subscribe to the taskboard
When a card is in the archive cabinet, activate the move-to drop
New tasks: Cards with "legacy" taskboards sometimes do not
Dashboard Analytics
16
New tasks: when clicking "assign more users" on a card with a drill
Attachments: change UI to show Drag&drop feature
New tasks: Deleting card with taskboard is not writing a
User WIP violation is not being enforced when assigning a


3 BACKLOG
5 CURRENT SPRINT BACKLOG
DEVELOPMENT IN PROCESS
Feature Group A
Feature
Ready
In Process 3
Done
Story 1
3
Task 1.3
Task 1.2
Task 1.2
Feature Group B
Feature
Ready
In Process 3
Done
TICKET #2312
Defect 1
2
Task 2.3
Task 2.4
Task 2.5
Task 2.6
Task 2.2
FT
Task 2.7
Task 2.1
Feature Group C
Feature
Ready
In Process 3
Done
Story 3
2
Task 3.3
Task 3.4
FT
Task 3.2
Task 3.1
2 OF 2
6 TESTING IN PROCESS
0 DEPLOY QUEUE
0 PRODUCTION CHECKOUT
0 DONE
ARCHIVE

Design

- As a user, I need to be able to request admin rights.
- As a admin, I need to be able to invite a user to join.
- As a user, I need to be able to change my password.
- As a admin, I need to be able to set permissions for a u...
- As a user, I need to be able to filter catalog items

Development

- As a user, I need to be able to
- As a admin, I need to be able to set
- As a user, I need to be able to
- As a user, I need to be able to sort
- As a user, I need to be able to sort

QA Testing

- As a user, I need to be able to
- As a admin, I need to be able to invite
- As a user, I need to be able to
- As a admin, I need to be able to list all of the users i...
- As a user, I need to be able to
- As a admin, I need to be able to list all
- As a user, I need to be able to login to the application...
- As a user, I need to be able to sort items
- As a admin, I need to be able to delete users.
- As a admin, I need to be able to delete users.
- As a admin, I need to be able to delete users.
- As a user, I need to be able to change my password.
- As a user, I need to be able to filter catalog items

UAT

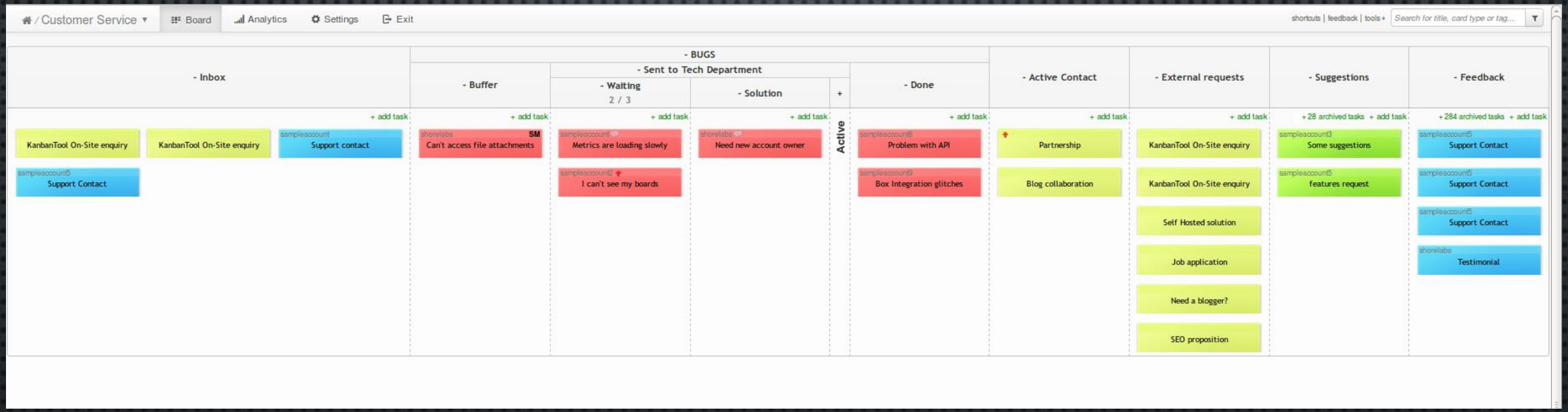
- As a admin, I need to be able to set permissions for a u...
- As a user, I need to be able to filter catalog items
- As a user, I need to be able to sort items
- As a user, I need to be able to login to the application...

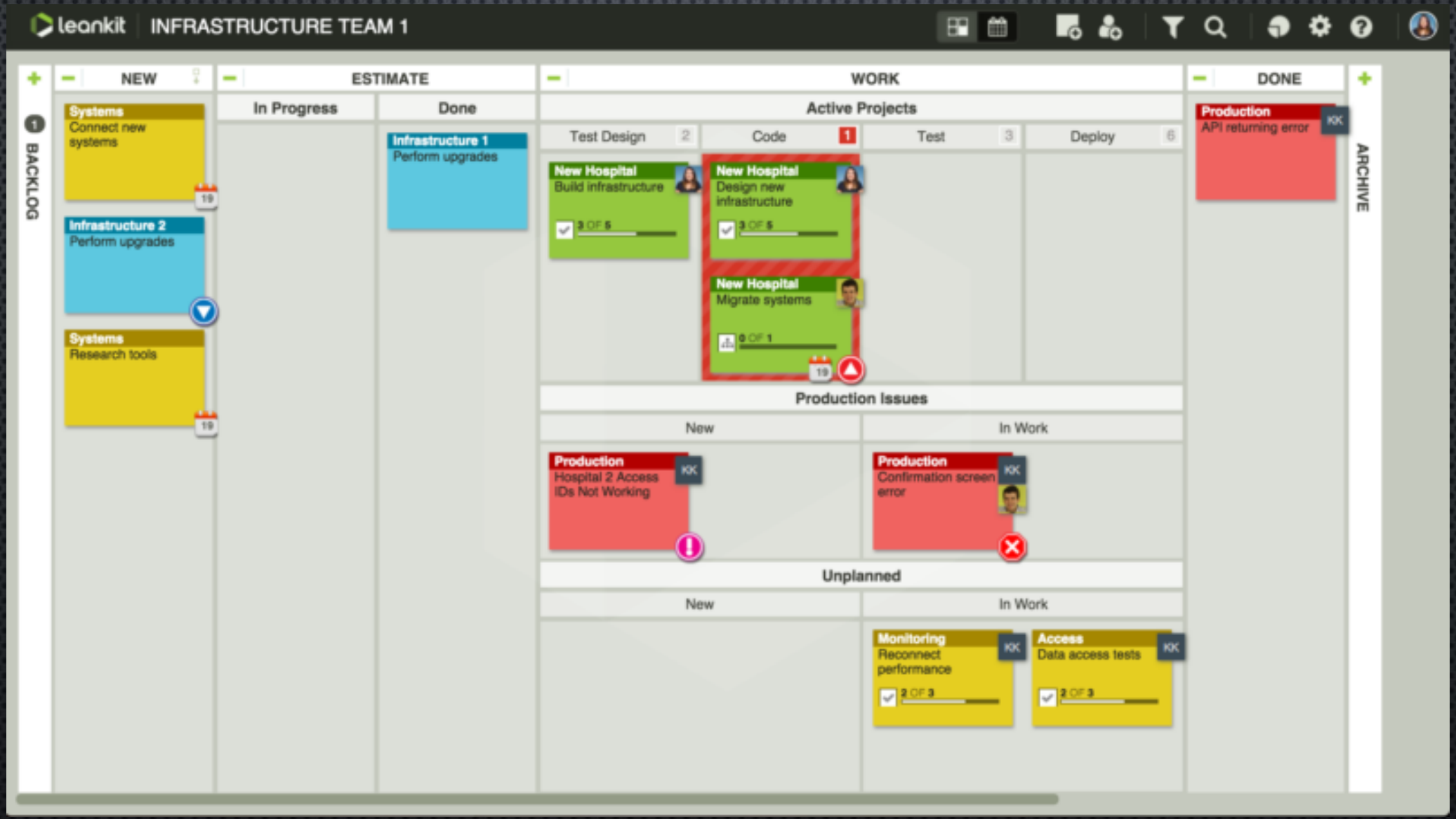
Deployment

- As a admin, I need to be able to delete users.
- As a admin, I need to be able to delete users.
- As a user, I need to be able to login to the application...

Done

- As a user, I need to be able to request admin rights.
- As a admin, I need to be able to invite a user to join.
- As a user, I need to be able to change my password.


/ Customer Service
Board
Analytics
Settings
Exit
shortcuts | feedback | tools +
Search for title, card type or tag...
- Inbox
- BUGS
- Buffer
- Sent to Tech Department
- Waiting
2 / 3
- Solution
+
- Done
- Active Contact
- External requests
- Suggestions
- Feedback
+ add task
+ 28 archived tasks
+ 284 archived tasks
KanbanTool On-Site enquiry
KanbanTool On-Site enquiry
Support contact
Support Contact
Can't access file attachments
Metrics are loading slowly
I can't see my boards
Need new account owner
Active
Problem with API
Box Integration glitches
Partnership
Blog collaboration
KanbanTool On-Site enquiry
KanbanTool On-Site enquiry
Self Hosted solution
Job application
Need a blogger?
SEO proposition
Some suggestions
features request
Support Contact
Support Contact
Support Contact
Testimonial


leankit
INFRASTRUCTURE TEAM 1
BACKLOG
1
NEW
Systems
Connect new systems
19
Infrastructure 2
Perform upgrades
Systems
Research tools
19
ESTIMATE
In Progress
Done
Infrastructure 1
Perform upgrades
WORK
Active Projects
Test Design 2
Code 1
Test 3
Deploy 6
New Hospital
Build infrastructure
3 OF 5
New Hospital
Design new infrastructure
3 OF 5
New Hospital
Migrate systems
0 OF 1
19
Production Issues
New
In Work
Production
Hospital 2 Access IDs Not Working
KK
Production
Confirmation screen error
KK
Unplanned
New
In Work
Monitoring
Reconnect performance
KK
2 OF 3
Access
Data access tests
KK
2 OF 3
DONE
Production
API returning error
KK
ARCHIVE

TEAM SIZE IN A SCRUM PROJECT

There are no prescriptive limits or guidelines to agile team size

There are general guidelines

The preferred size of a team has always been recommended to be:

- **7 +/- 2 or**
- **between 3 and 9**
- **But direct experience is always valuable**

The size can be 10+ if the primary reasons is cross-team dependencies or multi-site development

On the other hand we should not go below 4 to effectively pair, collaborate, or review each other's work

The key point is that you need to discover the right size on your own, starting with 6 or 7 members for your first project