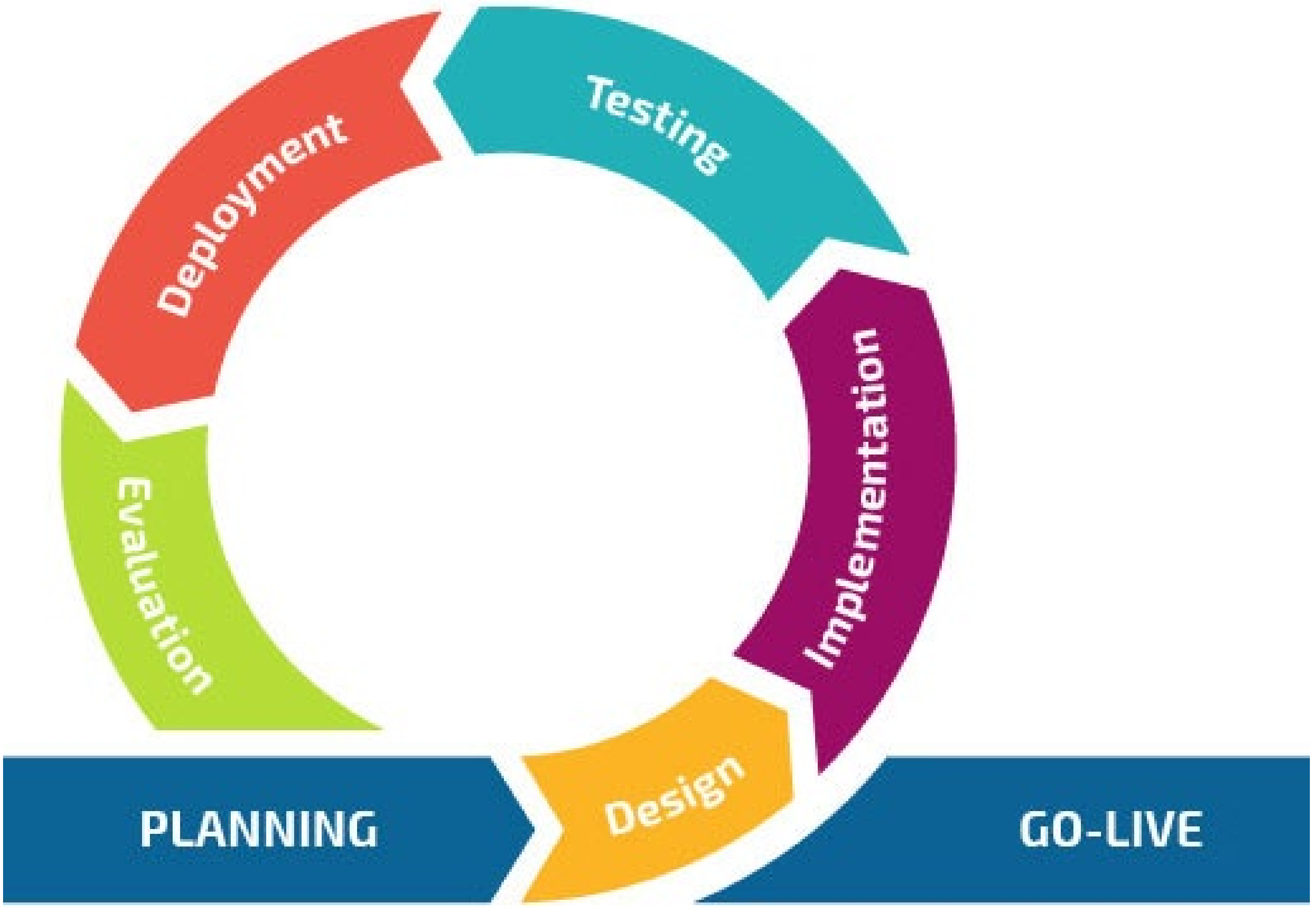
Agile and User Experience

Danielle Favreau

What's inside?



What is Agile?

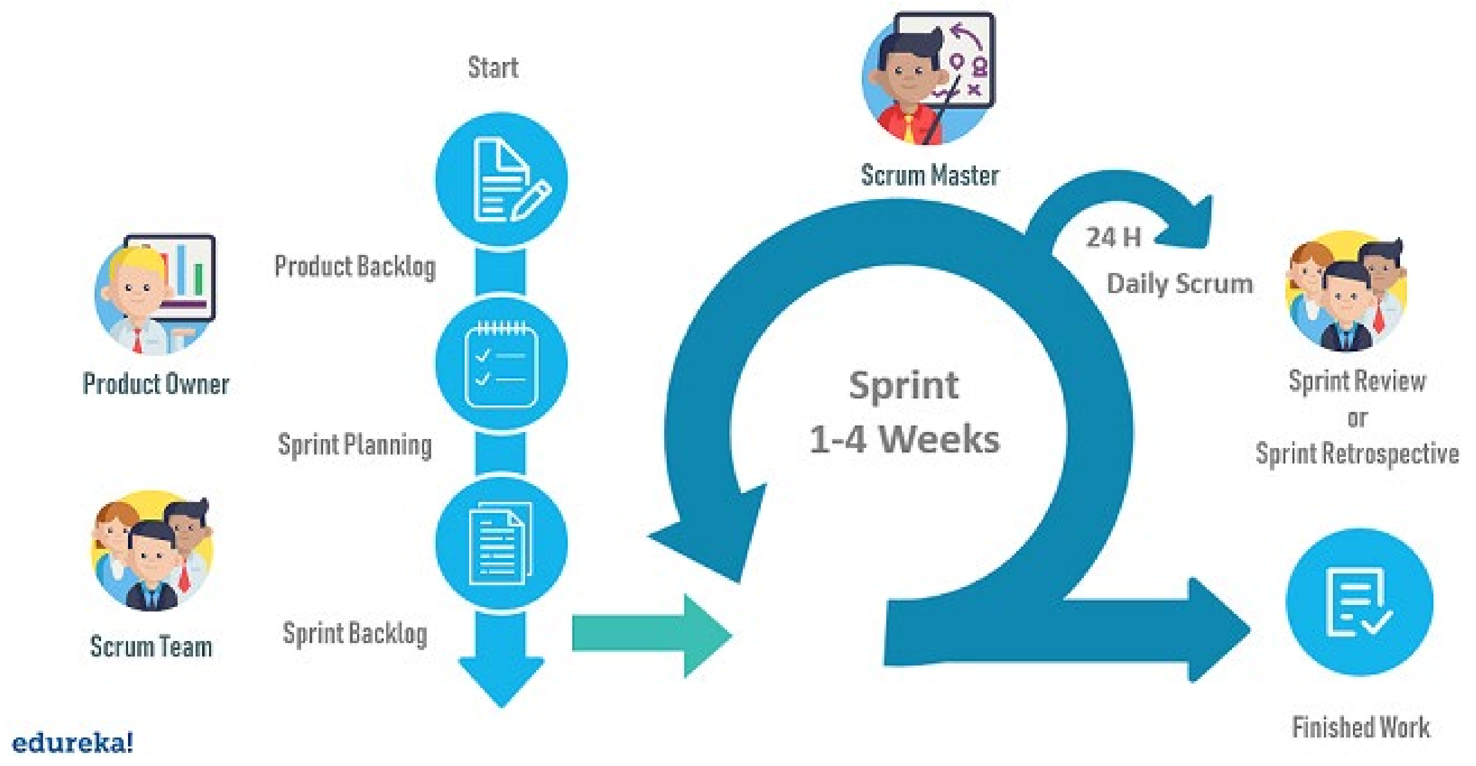
Agile is a methodology for iterative development where requirements and solutions evolve through collaboration between self-organizing cross-functional teams.

What is Agile Methodology?

- It is a method used to deliver solutions **incrementally**.
- Agile development applies to **<u>any</u> development process** that is aligned with the Agile Manifesto.
- While Agile was developed by 14 people in the software industry, it **can be used for almost any process**.
- Agile **values individuals and interactions** over processes and tools.
- Agile **values working solutions** over comprehensive documentation.
- Agile **values customer collaboration** over contract negotiation.
- Agile **values responding to change** over following a plan.

What Agile Methodology isn’t:

- Agile is <u>not</u> Scrum.
- Agile is <u>not</u> Kanban.
- Agile isn’t driven by rules.
- Agile isn’t anti-architecture.
- Agile is not against documentation.
- Agile doesn’t lack planning or structure.
- Agile isn’t right for everyone or everything.
- Agile doesn’t mean putting things on the back burner.



What is Scrum?

Scrum is a subset of Agile.
It is a process framework for agile development.
It is known for its roles, artifacts, and time boxing.

What is Scrum?

- **Scrum** is the **most widely used** framework for agile development.
- It follows a very particular set of practices, rules, and a **strict format**.
- Scrum has **time-boxed ceremonies** that must be adhered to.
- Scrum is often used for **complex software and product development**.
- It significantly **increases productivity** and reduces time to benefits.
- Scrum **expects changes** and readily accommodates them.
- Scrum **increases the quality** of deliverables.
- Scrum can provide a better estimate, therefore **better scheduling**.

What is the Scrum Process?

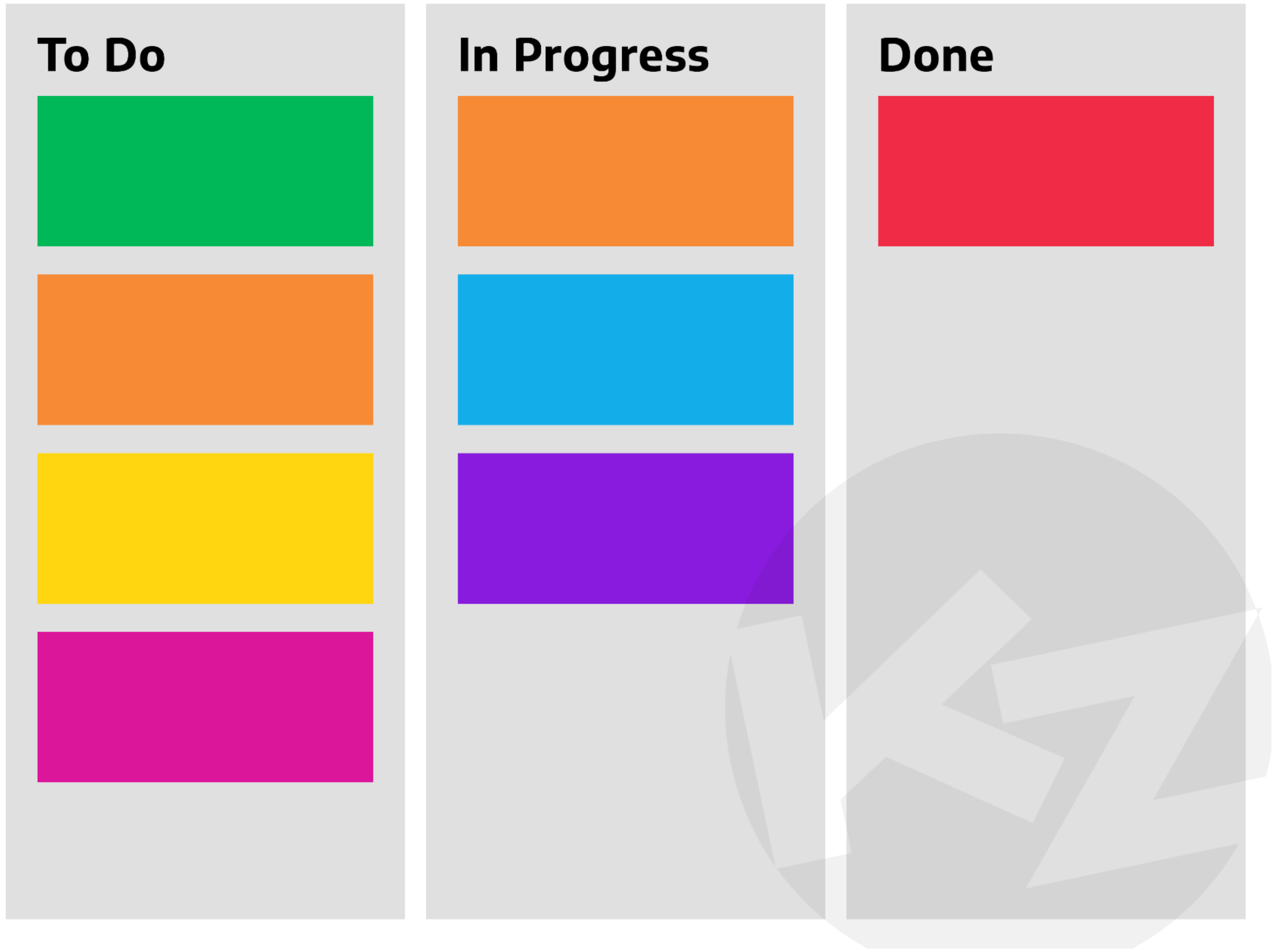
- The process starts with a **list of tasks** to be done (the **Backlog**).
- Those **tasks get organized** in order of priority (the **Refinement**).
- Those **tasks get fleshed out** to contain details of what to do and how to determine if they are done (the **Stories**).
- **Tasks are chosen** (the **Planning**) to do during **a 2-week time block** (the **Sprint**).
- The **team meets daily for 15 minutes** to talk about what they did yesterday, what they're doing today, and if they have anything preventing them from getting things done (the **Stand-up**).
- Once a Sprint, the team **shows the work they completed** to anyone with interest in the product created and they get feedback (the **Review**).
- The team also gets together at the end of the Sprint to **talk about what they did well and what they can do better** (the **Retrospective**).

Who is on the Scrum Team?

- Scrum has three roles: **Scrum Master/Agile Coach**, **Product Owner/Manager** , and the **Team**.
- The **Scrum Master (SM)** is responsible for making sure the process runs smoothly, they remove obstacles for the team, and they organize and facilitate the meetings (ceremonies).
- The **Product Owner (PO)/Product Manager (PM)** is the “single source of truth” for the team.   The PO/PM is responsible for maintaining the Backlog, buffering the team from requests, and determining what takes priority each Sprint.
- The **Team** is a self-organizing and cross-functional group of people who do the hands-on work chosen for the Sprint.

What is self-organizing and cross-functional?

- **Self-organizing** means that the Team members decide how to break the work into tasks and how to allocate those tasks to members of the team throughout the Sprint.
- **Self-organizing is not** the Team choosing which tasks to do during the Sprint, that direction comes from the PO.
- **Cross-functional** means that the Team is comprised of people with all the skills necessary to complete the work to be done in the Sprint.
- **Cross-functional does not** mean that everyone on the Team can do everyone else's work.



What is Kanban?

Kanban is an agile workflow management method designed to help visualize the work to be done by the team.

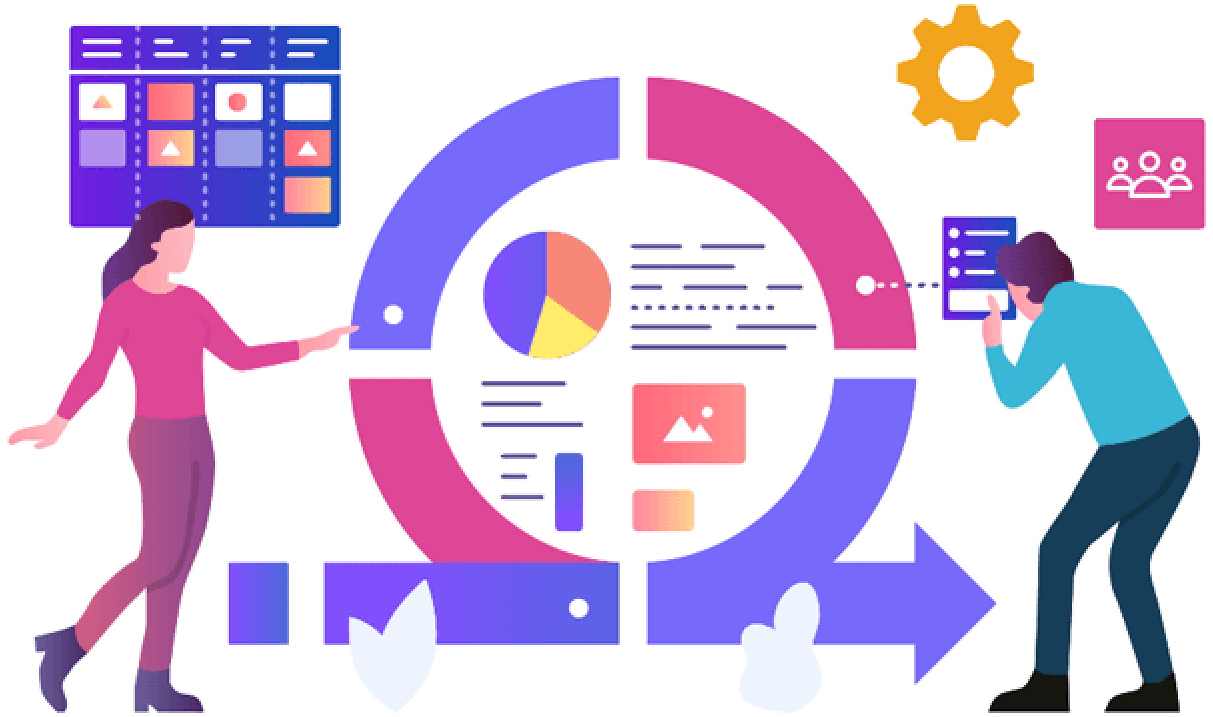
What is Kanban?

- Kanban respects the current process, roles, and responsibilities.
- Kanban is flexible and can be used with any existing workflow, system, or process. It will naturally highlight issues that need to be noticed and addressed. It focuses on getting things done.
- Kanban encourages continuous incremental and evolutionary changes.
- It discourages sweeping large changes as they encounter resistance due to fear and uncertainty.
- It encourages acts of leadership at all levels, everyone strives for optimal and continuous improvement.

What are the six practices of Kanban?

1. **Visualize the Workflow:** We do this by using the Kanban view on Jira, where we see each of our stories and tasks on little cards.
2. **Limit Work in Progress:** We try to do this by limiting the number of stories that team members are working on at any given time.
3. **Manage Flow:** Managing the flow is about managing the work but not the people, so we encourage the work to move through the columns.
4. **Make Process Policies Explicit:** You can’t manage or improve something you don’t understand, we try to foster understanding in all of our work.
5. **Feedback Loops:** The entire team needs to both give and receive feedback; ceremonies like the standup and review allow us to do this.
6. **Improve Collaboratively:** to achieve continuous improvement and sustainable change, we need a shared vision and a collective understanding.

It is not Scrum vs. Kanban!



It is Scrum and Kanban!

What is Scrumban?

Scrumban uses the prescriptive nature of Scrum to be agile, but uses the process improvement of Kanban to allow the team to continually improve its process.

Why use Scrumban?

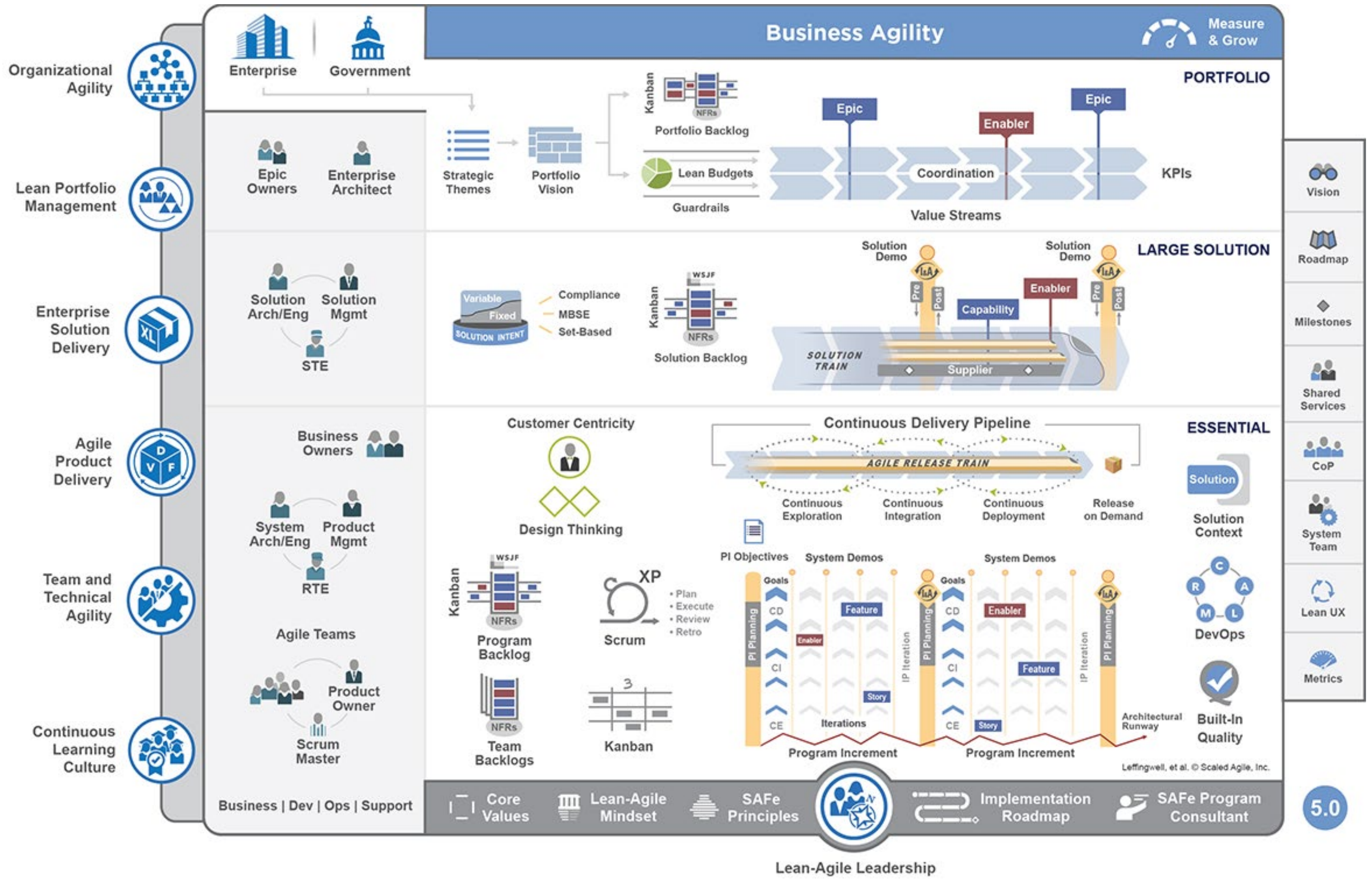
- Scrumban is ideal for maintenance projects.
- It’s best used for event-driven work.
- It works very well for support work or work tied to other team’s work.
- It thrives in projects with frequent or unexpected user stories.
- It is ideal for work that precedes sprint development (think research, testing, design, packaging, deployment).
- It can be used effectively by teams that may have workflow issues, resource issues, or process issues.
- It’s also ideal for teams where agile is being introduced.

How do should you use Scrumban?

- Use a board layout on Jira with columns that allow you to visually track work through a series of stages.

  To Do | Researching / Refining | Dev in Progress | Ready for Review / Test | Ready to Demo | Done

- Minimize work that each person pulls into the sprint using SAFe Agile Pointing where each person gets 8 points per two-week Sprint.
- Only allow items into the Sprint that the PO deems important for that time period (sometimes other items make it in out of necessity).
- Allow for Just-in-Time decisions to allow the team to do the work needed to do at the time it is needed (this doesn’t mean we leave things to the last minute).
- Have roles like Scrum Master and Product Owner.
- Have ceremonies: Standup, Refinement, Planning, Review, and Retro.



What is SAFe Agile?

SAFe Agile Framework ® is a way to scale agile across an entire enterprise.

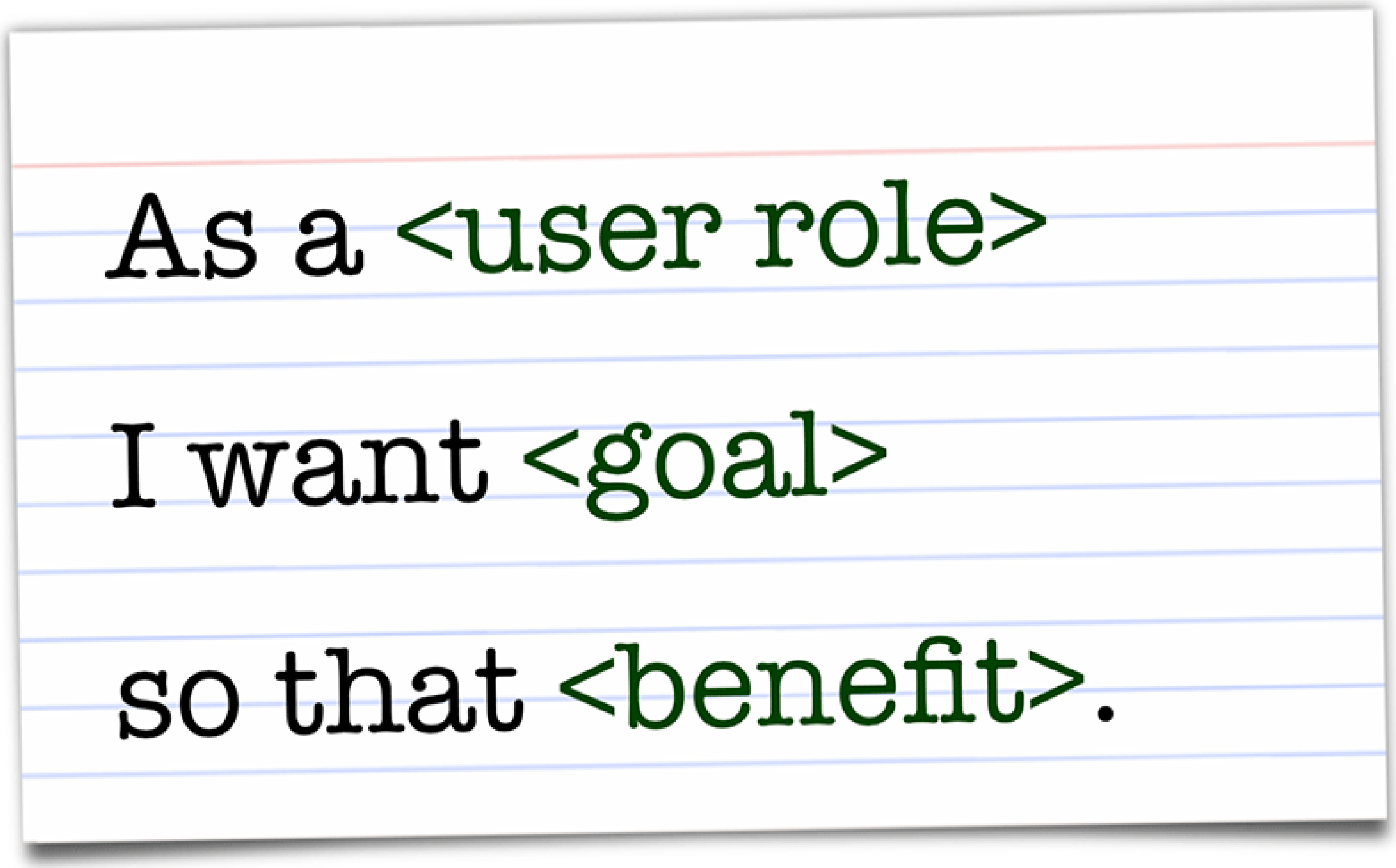
And while that graphic makes it look very complicated, it doesn't have to be.

What does SAFe Agile mean for User Experience?

- You can see SAFe Agile in the hierarchy of Jira stories.
  Stories should be associated with a Feature.
  Each Feature should be associated with a Capability.
  Each Capability should be associated with an Epic.

  Epic → Capability → Feature → Stories
- The Features are the source of the Stories and every story must be associated with a feature.
- Capacity must be consistent across all Scrum teams across the enterprise; to achieve this, SAFe story pointing is used, where each person on the team gets 1 point per working day of the sprint (minus 2 ceremony days).

What is a Story?



As a <user role>
I want <goal>
so that <benefit>.

Stories are vertical slices of work that include all of the details necessary for the team to achieve the work by the end of the Sprint.

A vertical slice means that the story includes all aspects to make the end-product release ready.

How should you write Stories?

- To create an effective JIRA story there are several questions you should ask. The most important being:

  If someone had to take over my task - and couldn't talk with me to get a hand off - would they have enough information to do the job correctly?

- You may have heard of the story format:
  As a user I want a goal so that benefit.
  That’s the start of a user story, but isn’t where user stories end.

- Stories should provide enough information for someone with no technical knowledge to understand what’s being asked and how it should be validated.

What does an effective Story contain?

- An effective user story answers many questions:
  - What is the Definition of Ready for the story? (What’s needed to get started?)
  - Who is the audience of the deliverable?
  - What do I need in order to complete the story? (Applications? Access?)
  - Who else needs to work on this story with me?
  - What are the expected deliverables?
  - Is this story dependent on any others or does it create dependencies?
  - Can the work defined in the story be completed within the Sprint?
  - When is the deliverable expected? (It is rarely at the end of the Sprint.)
  - What is the Definition of Done for the story?
  - Where will the deliverable(s) live?

What is the Definition of Ready and Done?

- The **Definition of Ready** (DoR) defines the criteria that the story must meet before being considered for pointing or including in a Sprint.
- The **Definition of Done** (DoD) is the criteria that the story must meet to be considered done and ready for release (release may simply mean posting on Confluence or emailing to someone or pushing to a server for the world to see).
- The Team(s) who have a stake in the outcome of the story are who will determine the DoD for the story.
  - For an analytics story, the DoD should be determined by the analytics team, user experience team, and the web team.
  - For a UX story, the DoD should be determined by both by the UX team and the team who requested the UX artifact.

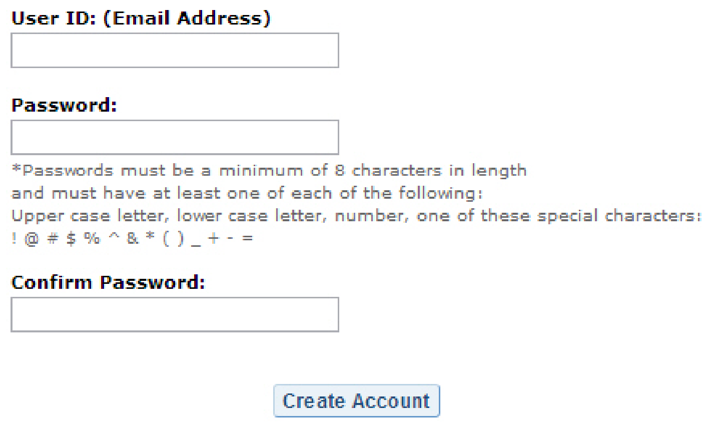
What does a good Story look like?

As a member, I want to know if I have completed the account sign up form correctly, so that when I click the submit button I know that it will be accepted.

Label	Field Type / Length	Restrictions	Validation
User ID (Email Address)	Text / 140 varchar	Regex for email	Your user ID must be a valid email address.
Password	Text / min 8 varchar	* Passwords must be a minimum of 8 characters in length and must have at least one of each of the following: Upper case letter, lower case letter, number, one of these special characters: ! @ # $ % ^ & * ( ) _ + - =	
Confirm Password	Must Equal Password	Must equal password field.	Both passwords must match.
Create Account	n/a	Do not disable, if a user clicks it without completing the above fields, show the errors.	

Expected Deliverable:
Working registration page at domain.com/registration.html

UAC:
Form meets criteria in table above.
Data saves to registration database table.
Natural language validations appear if fields not completed correctly.
Approval by PO.


User ID: (Email Address)

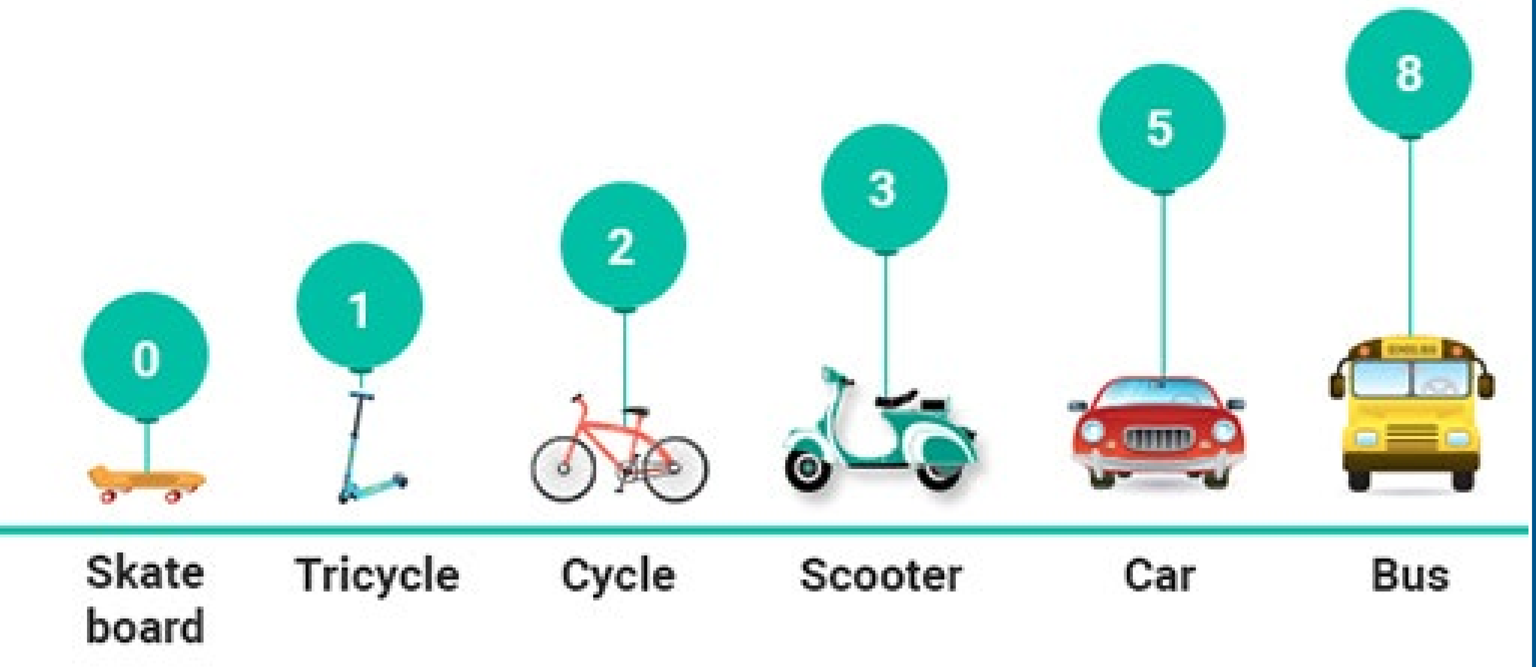
Password:

*Passwords must be a minimum of 8 characters in length and must have at least one of each of the following: Upper case letter, lower case letter, number, one of these special characters: ! @ # $ % ^ & * ( ) _ + - =

Confirm Password:

Why should you write stories?

- Story writing allows everyone to fully understand the tasks available and ensures that people who can complete the task know what's involved.
- Stories provide clarity and a chance to identify potential issues, dependencies, or blockers.
- An effective story allows each person on the team to accurately determine their capacity by ensuring that they can be completed within a single Sprint.



What is Story Pointing?

A Scrumban team can only take on so much work during a Sprint, story pointing allows teams to choose just enough work to meet their capacity, and will help them determine their velocity.

Why point stories?

Think of pointing stories like making a budget. Pointing allows you to know how much you have to spend each Sprint and tells you how much you actually spent. There are two methods of pointing stories, and this will explain the difference between the two, and why I recommend the one I do.

Method 1

You know you have to budget for the month, but you don't know how much you'll make. It could be $0 or $250,000. Each month your landlord gives you a different amount you have to pay for your rent so you never know how much you'll have to pay. Your insurance company changes your premium every month so it could be $0 or $300. Gas and groceries fluctuate dramatically so you never know what to budget. And it changes if you do the shopping or if your partner does. Each month you end up owing money to someone because you're never sure how much you'll make or how much you'll have to pay.

Method 2

You have to budget for the month, you made $4,000. $800 will go straight to rent. You have $3,200 left to spend for the month. You know that gas will cost you $160. Insurance will cost $120. Groceries will be $800. Childcare is $1,200. Utilities will cost you $500. And you know you have $420 left for miscellaneous costs.

Based on these two scenarios, which would allow you to budget most effectively?

Story Pointing Methods

Method One shows traditional pointing, where we would use the Fibonacci sequence: 0, 1, 2, 3, 5, 8, 13, 21, 34, 55, … (add the number to the number before).
Points = effort + complexity.

Method Two shows normalized pointing, where we point from 0, 0.5, 1, 2, 3, 4, 5, 6, 7, or 8; where each point is equal to one working day.
Points = time.

Arbitrary pointing makes it difficult to accurately point stories.
With normalized pointing, pointing is more accurate and makes it easier to track across separate teams within a large organization or across new teams, like those at a Start-Up.

*Note that even the creator of story points, Ron Jeffries, is sorry he ever created them. Stories were originally estimated in "Ideal Days" (the number of days it would take if no one bothered you). Ideal Days was then multiplied by a load factor, usually 3, meaning it would take 3 Real Days to do 1 Ideal Day's work. Estimates were done in days, not t-shirts, not dogs but days. This allowed the team to easily know that they couldn't do 50 "days" worth of work in three weeks. Eventually the word "days" became "points." See the Appendix for the entire article.

How do the two pointing systems work?

Now let's see how that applies to a team of 6 people.

Method 1 (points assigned / points in sprint / completed)

	Sprint 1	Sprint 2	Sprint 3
John	62 / 13	84 / 61	24 / 21
Abbey	72 / 84	60 / 48	42 / 42
Marisa	102 / 134	34 / 48	63 / 42
Lincoln	24 / 32	48 / 52	75 / 72
Carl	48 / 96	28 / 34	81 / 96
Simone	36 / 48	21 / 36	71 / 68
Capacity	344 / 456 / 407	275 / 314 / 279	356 / 371 / 341
Velocity	407	343	342.3

Capacity: Expected / Assigned / Actual

Method 2 (fewer than 8 on the left means days off)

46	Sprint 1	Sprint 2	Sprint 3
John	8 / 7	8 / 11	6 / 6
Abbey	8 / 8	8 / 8	8 / 7
Marisa	8 / 8	6 / 6	8 / 8
Lincoln	8 / 10	8 / 8	8 / 7
Carl	8 / 8	8 / 9	8 / 8
Simone	8 / 8	8 / 10	8 / 8
Capacity	48 / 50 / 49	46 / 56 / 52	46 / 46 / 44
Velocity	49	50.5	48.3

What does story pointing tell us?

What can we tell about the teams from the previous slide?

Method 1

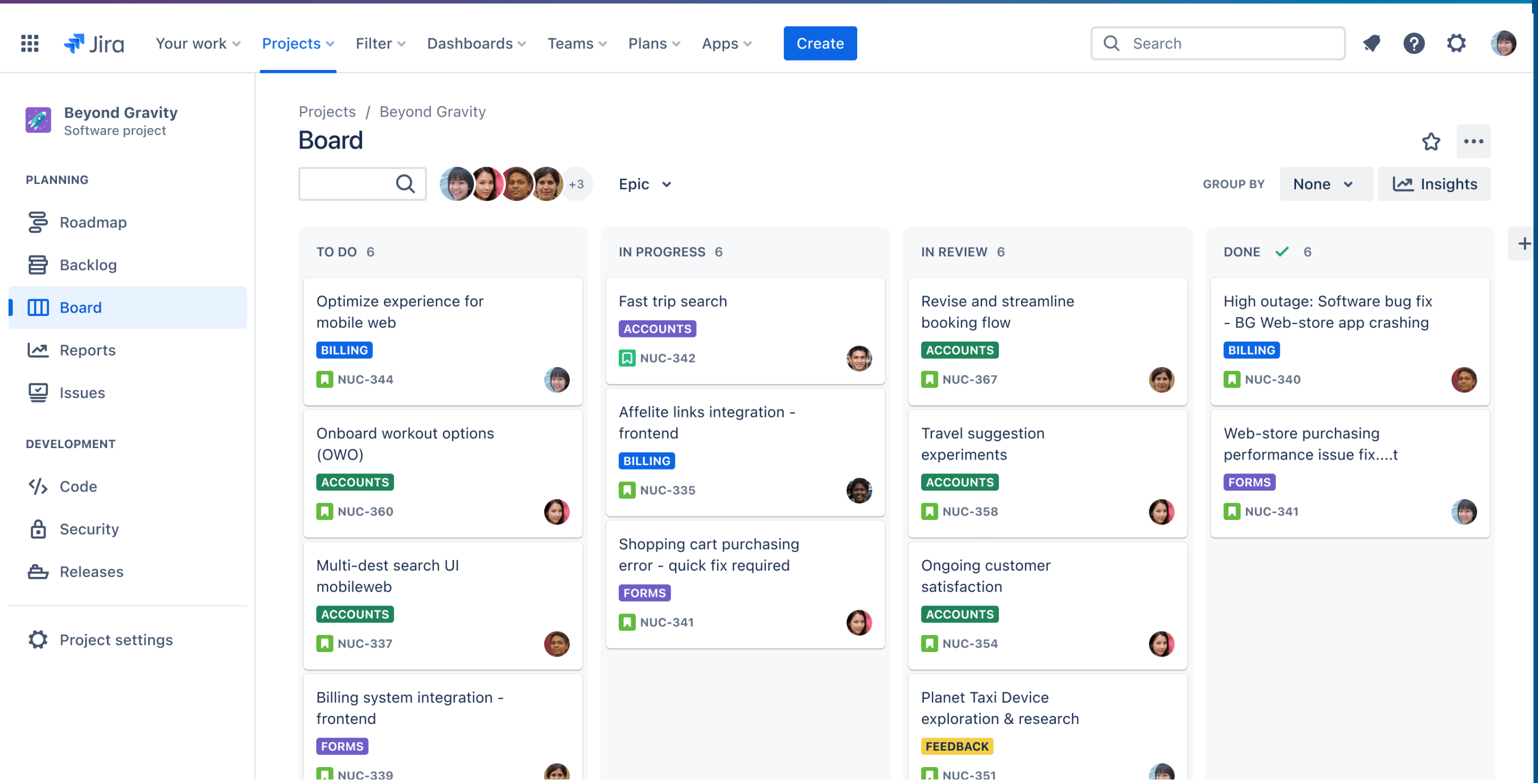
Sprint Capacity: 275 – 456
Days off: Unknown
Individual Capacity: 31.6 – 75.3

Method 2

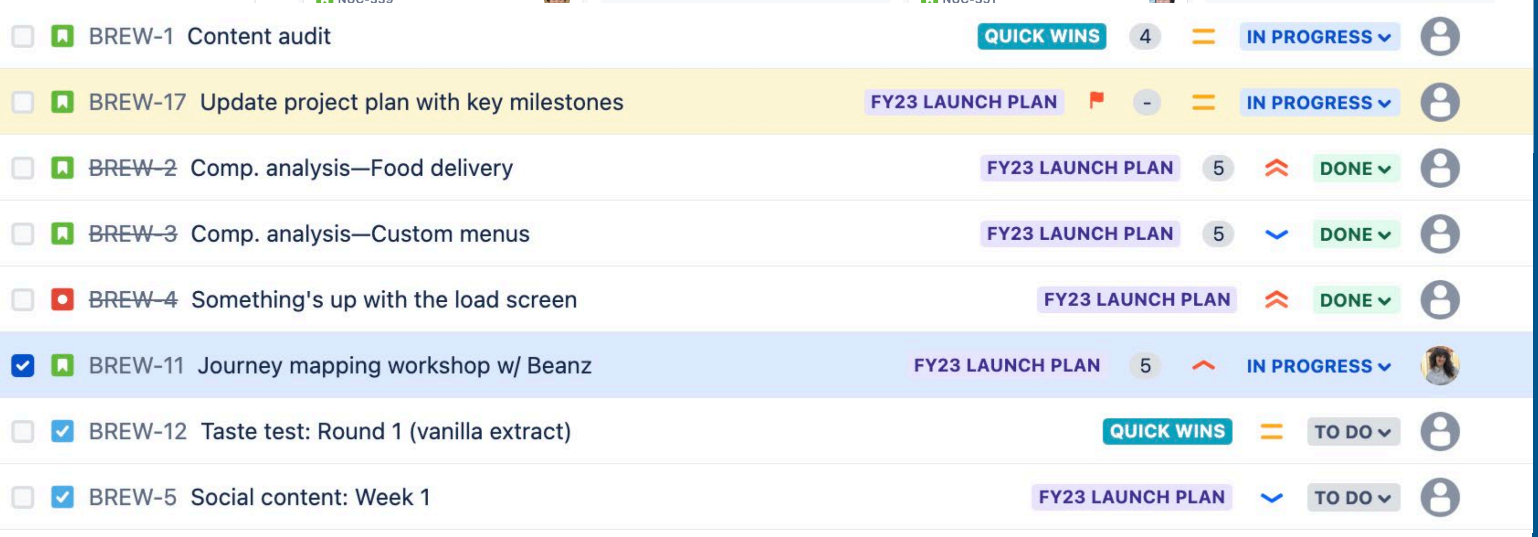
Sprint Capacity: 48 (-1 for each vacation day)
Days off: Marisa 2, John 2
Individual Capacity: 7.6 - 8.6

When the capacity fluctuates so heavily, it's difficult to tell when you need more manpower, especially when you can't tell if time off is being taken into account. When individual capacity has a huge range (31 – 75) you can't tell when someone is overwhelmed (like Carl seems to be) or underperforming (like John seems to be) or if it's just that the points are off.

When you use normalized pointing, you can quickly calculate the team's sprint capacity based on their working days, determine if you need additional workers, see if people are under or over performing, and when used across the enterprise it's easy for upper management to determine the overall capacity and expected velocity of the entire organization.



TO DO 6	IN PROGRESS 6	IN REVIEW 6	DONE ✓ 6
Optimize experience for mobile web — BILLING — NUC-344	Fast trip search — ACCOUNTS — NUC-342	Revise and streamline booking flow — ACCOUNTS — NUC-367	High outage: Software bug fix - BG Web-store app crashing — BILLING — NUC-340
Onboard workout options (OWO) — ACCOUNTS — NUC-360	Affelite links integration - frontend — BILLING — NUC-335	Travel suggestion experiments — ACCOUNTS — NUC-358	Web-store purchasing performance issue fix....t — FORMS — NUC-341
Multi-dest search UI mobileweb — ACCOUNTS — NUC-337	Shopping cart purchasing error - quick fix required — FORMS — NUC-341	Ongoing customer satisfaction — ACCOUNTS — NUC-354	
Billing system integration - frontend — FORMS — NUC-339		Planet Taxi Device exploration & research — FEEDBACK — NUC-351	



BREW-1	Content audit	QUICK WINS	4		IN PROGRESS
BREW-17	Update project plan with key milestones	FY23 LAUNCH PLAN	-		IN PROGRESS
~~BREW-2~~	Comp. analysis—Food delivery	FY23 LAUNCH PLAN	5		DONE
~~BREW-3~~	Comp. analysis—Custom menus	FY23 LAUNCH PLAN	5		DONE
~~BREW-4~~	Something's up with the load screen	FY23 LAUNCH PLAN			DONE
BREW-11	Journey mapping workshop w/ Beanz	FY23 LAUNCH PLAN	5		IN PROGRESS
BREW-12	Taste test: Round 1 (vanilla extract)	QUICK WINS			TO DO
BREW-5	Social content: Week 1	FY23 LAUNCH PLAN			TO DO

What do I recommend?

I recommend Scrumban and using a normalized pointing system.

What does a Sprint schedule look like?

You should expect to attend the following ceremonies:

	Monday	Tuesday	Wednesday	Thursday	Friday
Week 1			Sprint Planning **Start of Sprint**	Refinement Stand Up	Stand Up
Week 2	Stand Up	Stand Up	Stand Up	Refinement Stand Up	Stand Up
Week 3	Stand Up	Demo Retrospective			

Everyone is expected to attend Stand Up, Sprint Planning, the Demos, and the Retrospective.

If you have multiple teams you may break up their planning, refinements, and demos.

What are your responsibilities?

Each team member is responsible for:

- Attending the daily standups.
- Updating a standup page on Confluence every day.
- Logging time in 15-minute increments to the stories you work on.
- Create stories based on the things they hear in meetings or while working on projects.
- Flesh out said stories with as much detail as possible.
- Presenting those stories to the PO to ensure that they are properly fleshed out.
- Update stories on a regular basis as you learn new details or have deliverables.
- Upload your deliverables in a working state and final state to your code repository and/or Confluence and Link those deliverables to the JIRA stories (do not upload files to JIRA). Branches should be named the Jira Key.
- Completing the work you were assigned during the sprint. If you can't, you must talk with the PO.

Agile and Documentation will Help Lead Your Team to Success!

While there are a lot of documentation items related to our flavor of agile, it’s because documentation is each team’s job. UX, Design, Research, Analytics, and Dev provide the data and the information that fuels the decisions made by the rest of your company.

When a PO is approached with questions, your documentation provides them with the data and information they needs to show a compelling story about what is being done and why they should continue to support the mission. The more visuals, the more documentation, the more hours logged in JIRA, the more details you include, the better – and more well supported – the story is that the PO can share with them.

Welcome to Agile!

References

- Agile Myths: http://www.agilenutshell.com/agile_myths
- Agile Manifesto: http://agilemanifesto.org/iso/en/principles.html
- What is Scrum: https://www.cprime.com/resources/what-is-agile-what-is-scrum/
- Kanban: https://kanbanize.com/kanban-resources/getting-started/what-is-kanban
- Scrumban: https://www.agilealliance.org/what-is-scrumban/
- Scrumban: https://www.planview.com/resources/articles/lkdc-what-is-scrumban/
- SAFe Agile: https://www.scaledagileframework.com/story/
- Normalized Pointing: https://www.scaledagileframework.com/iteration-planning/#:~:text=In%20SAFe%2C%20however%2C%20story%20point,basis%20for%20economic%20decision%2Dmaking.
- Why Story Points Don't Work: https://medium.com/@xevix/why-story-points-dont-work-5f10c5d5a0f0
- Story Points by the creator of Story Points: https://ronjeffries.com/articles/019-01ff/story-points/Index.html