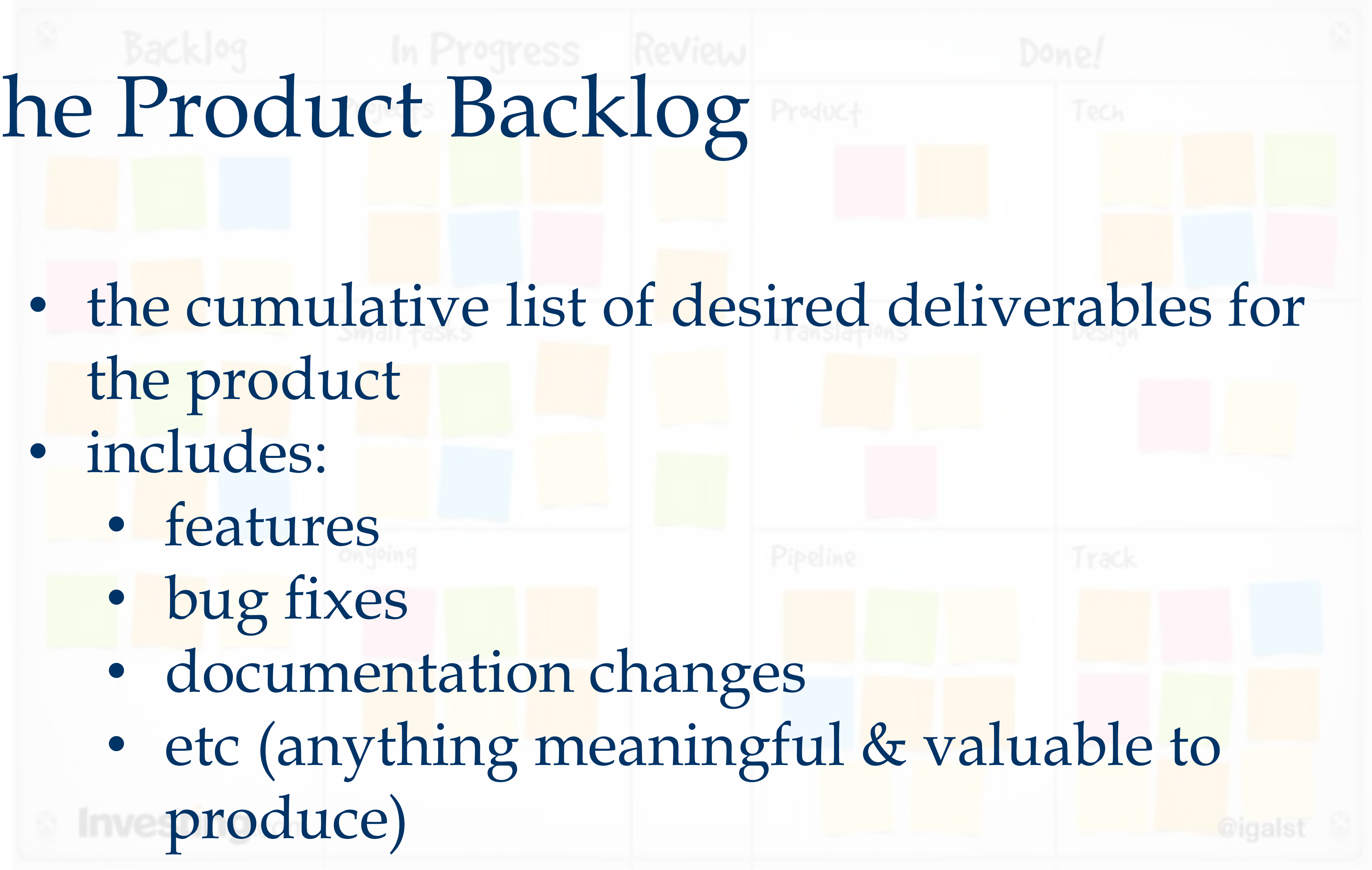
Scrum Methodology: Artifacts

Scrum Artifacts

the tools Scrum practitioners use to make the process visible

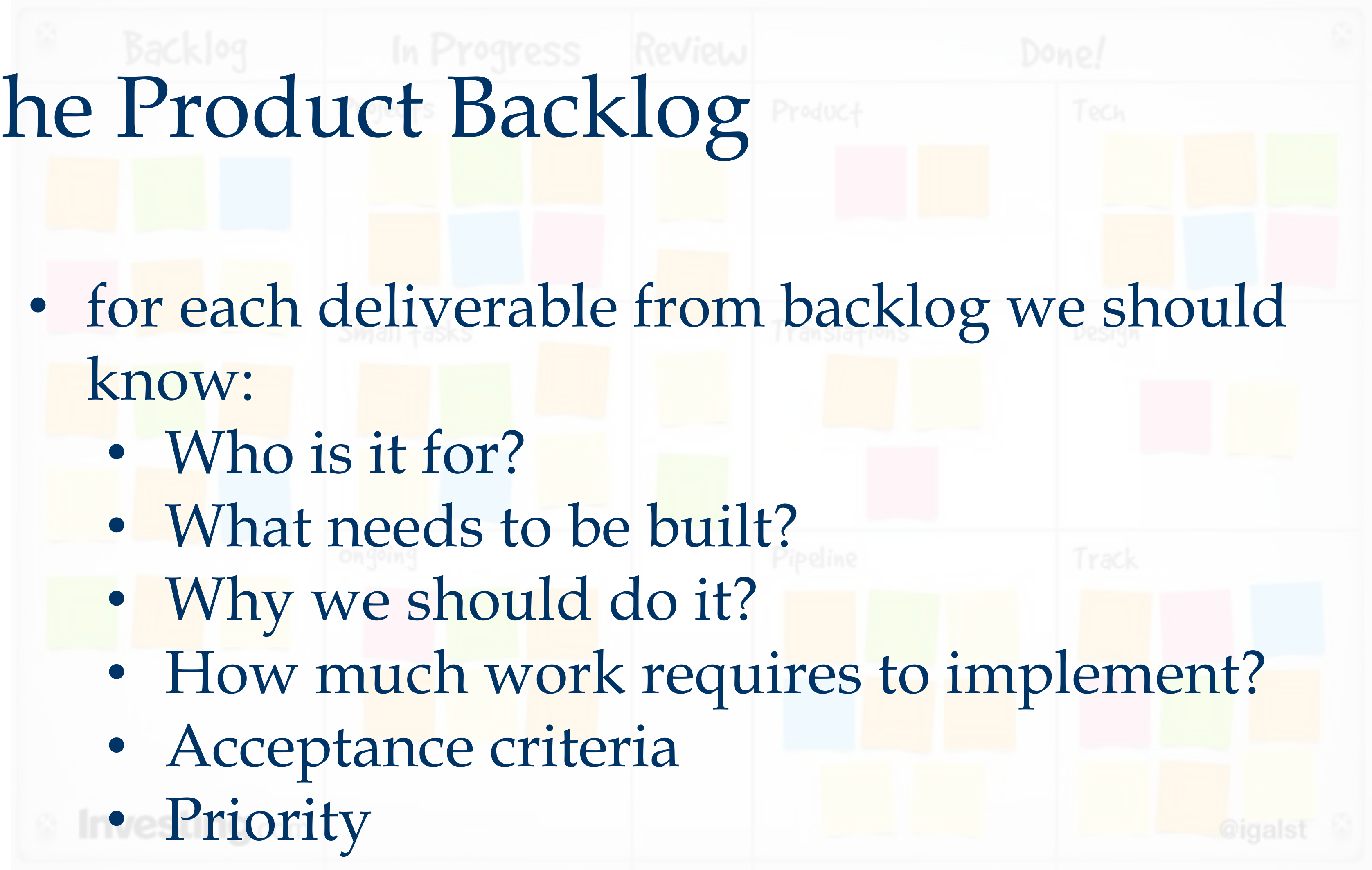
Scrum Artifacts



The Product Backlog

- the cumulative list of desired deliverables for the product
- includes:
    - features
    - bug fixes
    - documentation changes
    - etc (anything meaningful & valuable to produce)

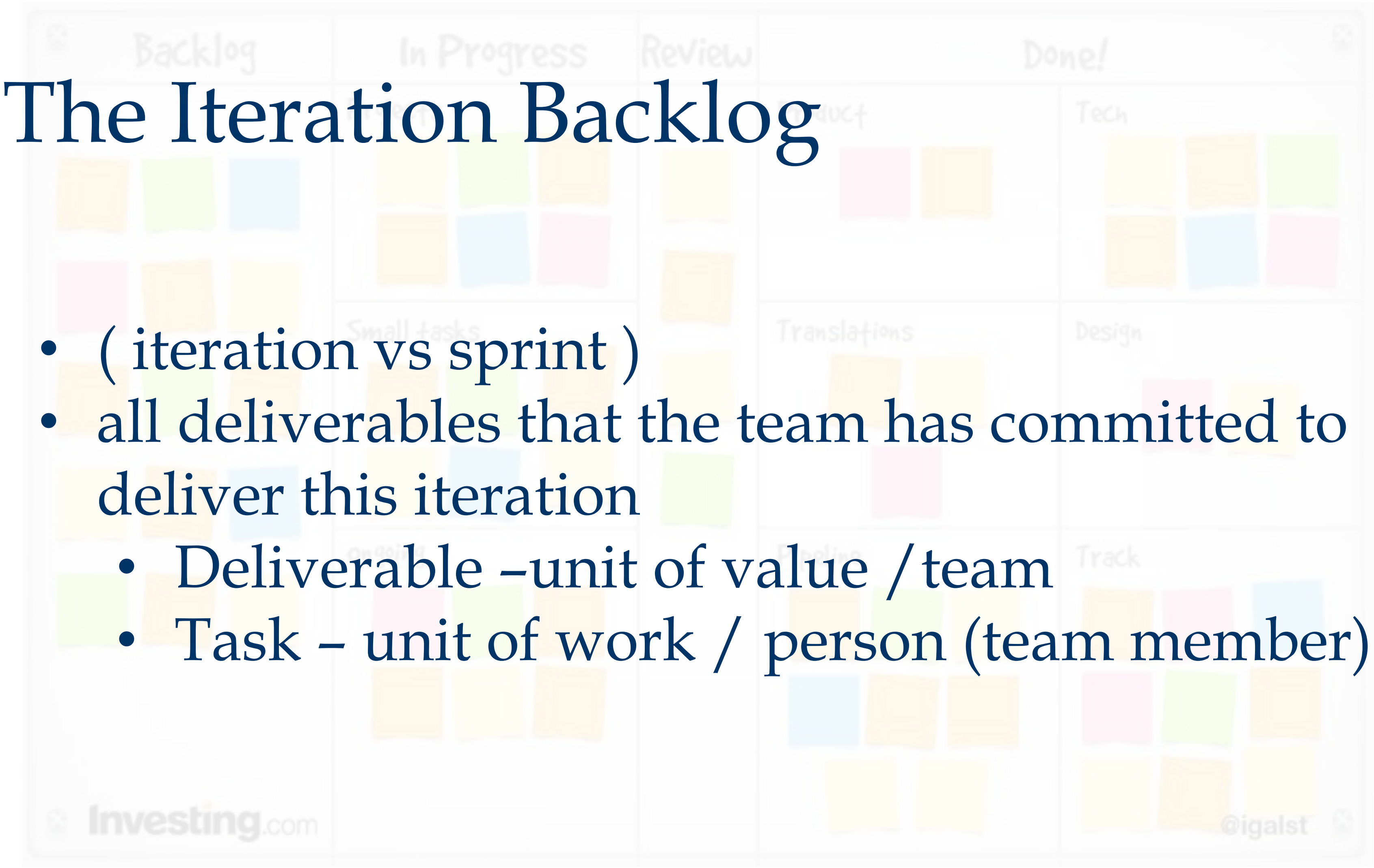
Scrum Artifacts



The Product Backlog

- for each deliverable from backlog we should know:
  - Who is it for?
  - What needs to be built?
  - Why we should do it?
  - How much work requires to implement?
  - Acceptance criteria
  - Priority

Scrum Artifacts



The Iteration Backlog

- (iteration vs sprint)
- all deliverables that the team has committed to deliver this iteration
  - Deliverable –unit of value / team
  - Task – unit of work / person (team member)

Scrum Artifacts

No changes during a Sprint!

1-4 Weeks
Sprint

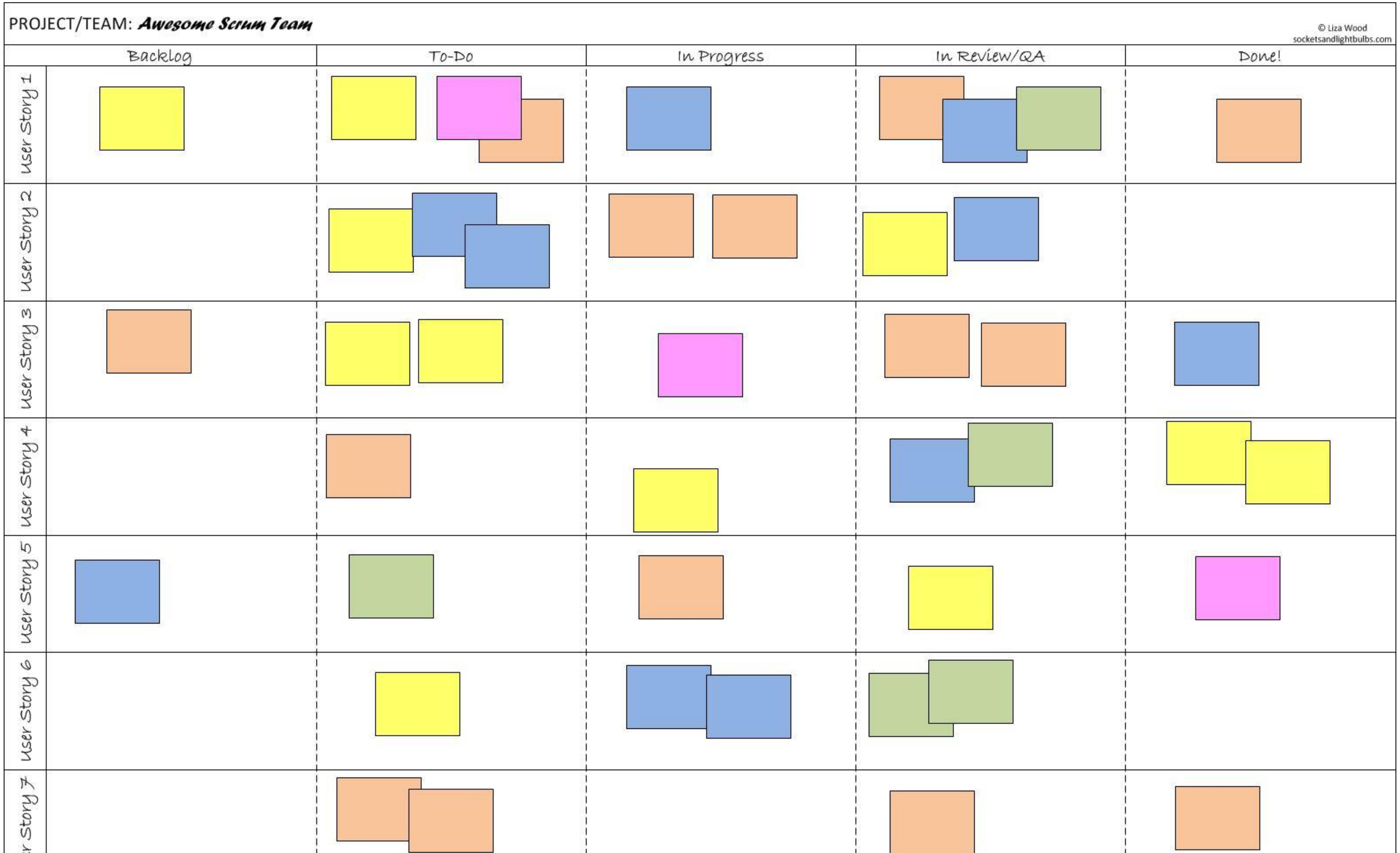
Plan sprint durations around how long you can commit to keeping change out of the sprint!

Scrum Artifacts

Product Increment

- Sum of all the Product Backlog items completed during a sprint and all previous sprints
- At the end of a sprint, the new Increment must be Done

Task Board



PROJECT/TEAM: ***Awesome Scrum Team***

© Liza Wood
socketsandlightbulbs.com

	Backlog	To-Do	In Progress	In Review/QA	Done!
User Story 1	🟨	🟨 🟪 🟧	🟦	🟧 🟦 🟩	🟧
User Story 2		🟨 🟦 🟦	🟧 🟧	🟨 🟦	
User Story 3	🟧	🟨 🟨	🟪	🟧 🟧	🟦
User Story 4		🟧	🟨	🟦 🟩	🟨 🟨
User Story 5	🟦	🟩	🟧	🟨	🟪
User Story 6		🟨	🟦 🟦	🟩 🟩	
r Story 7		🟧 🟧		🟧	🟧

Scrum Artifacts

Definition of Ready

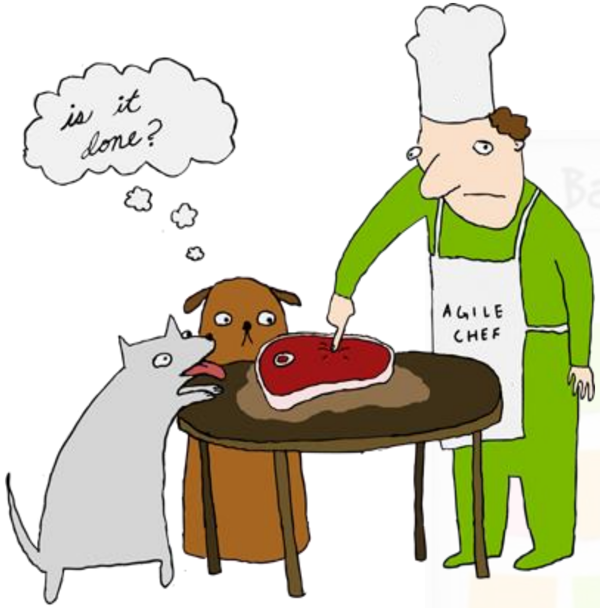
- The Definition of Ready (DoR) sets out the criteria for the stories needed to make the Sprint succeed
- The DoR is drawn up by the Development Team, cooperating with the Product Owner
- The Development Team determines whether Product Backlog items meet the DoR
- The PO respects the DoR. That means that a Product Backlog item is only included in a Sprint if it meets the DoR.

Scrum Artifacts

Definition of Ready

- Example:
  - Story Statement
  - Specification by Example
  - Flow chart, if needed
  - Use Case, if Acceptance Criteria missing
  - Wireframe, if needed, delivered
  - UX [mock-up] if needed, delivered

Scrum Artifacts

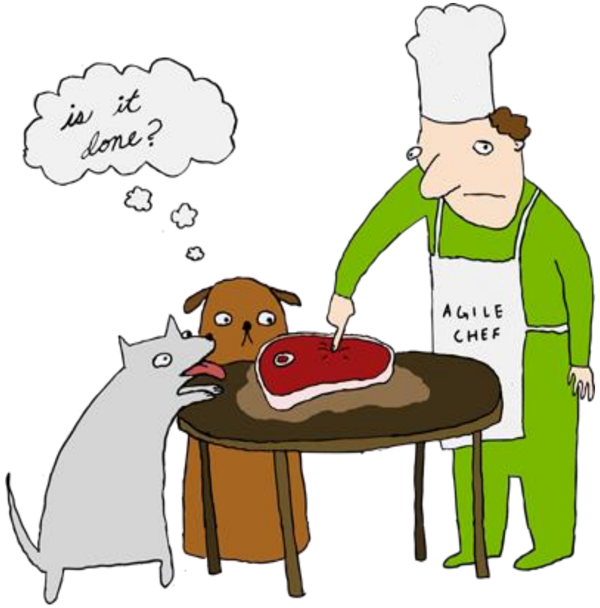


Definition of Done

- “when the code has been written” (programmer)
- “all of the tests have passed” (tester)
- “it’s been loaded onto the production servers” (operations)
- “we can now sell it to customers“(business person)

- Each team creates its own “definition of done”

Scrum Artifacts

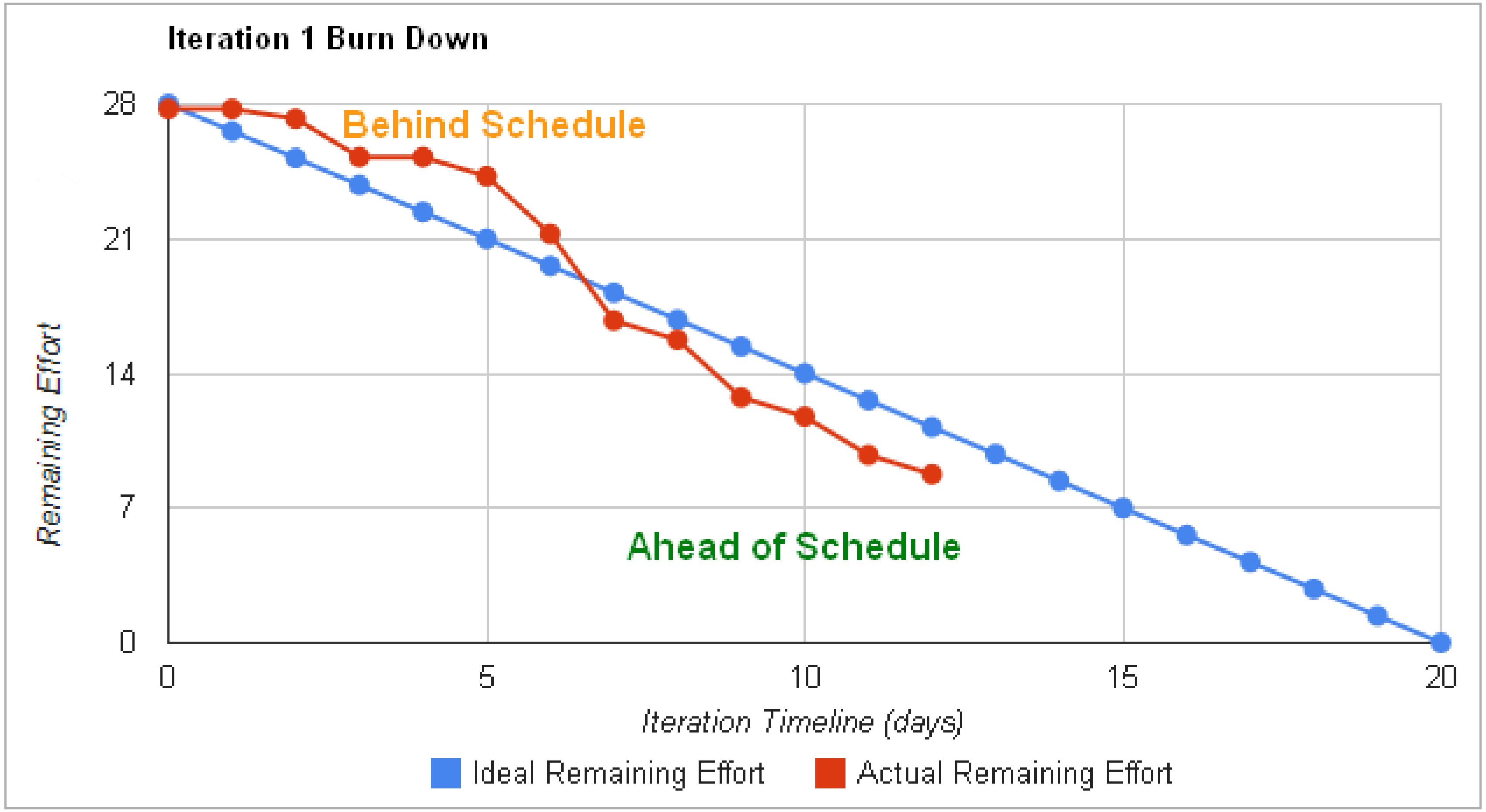


Definition of Done

- Is a crucial tool for making sure that the developed product is **satisfying stakeholders expectations**
- The team is **only finished with a Sprint Backlog item once it meets the DoD**
- The DoD is **drawn up by the Development Team**
- A Definition of Done should exist for:
  - User stories
  - Releases
  - Final project deliverables

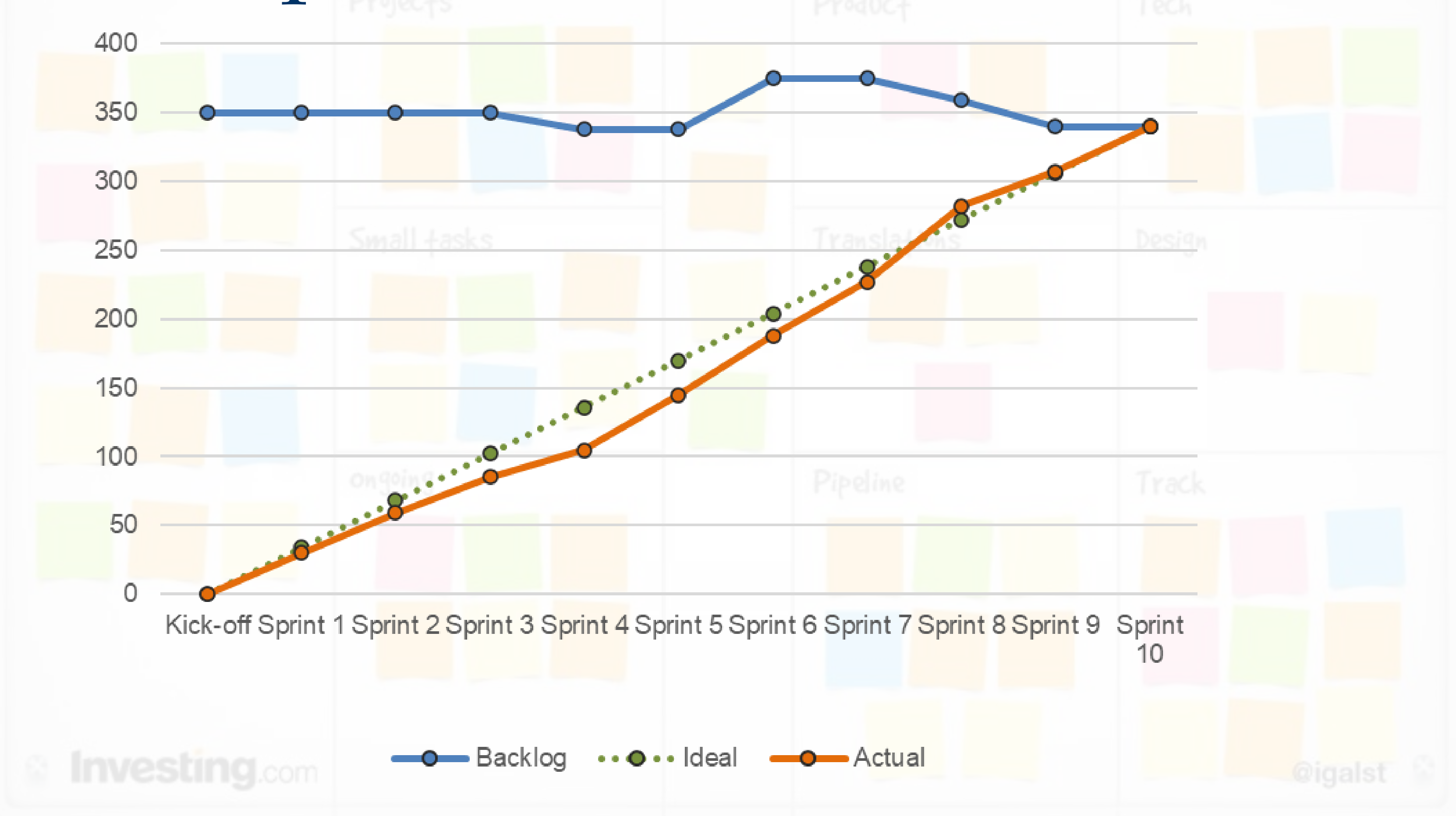
Scrum Artifacts

Burndown Charts



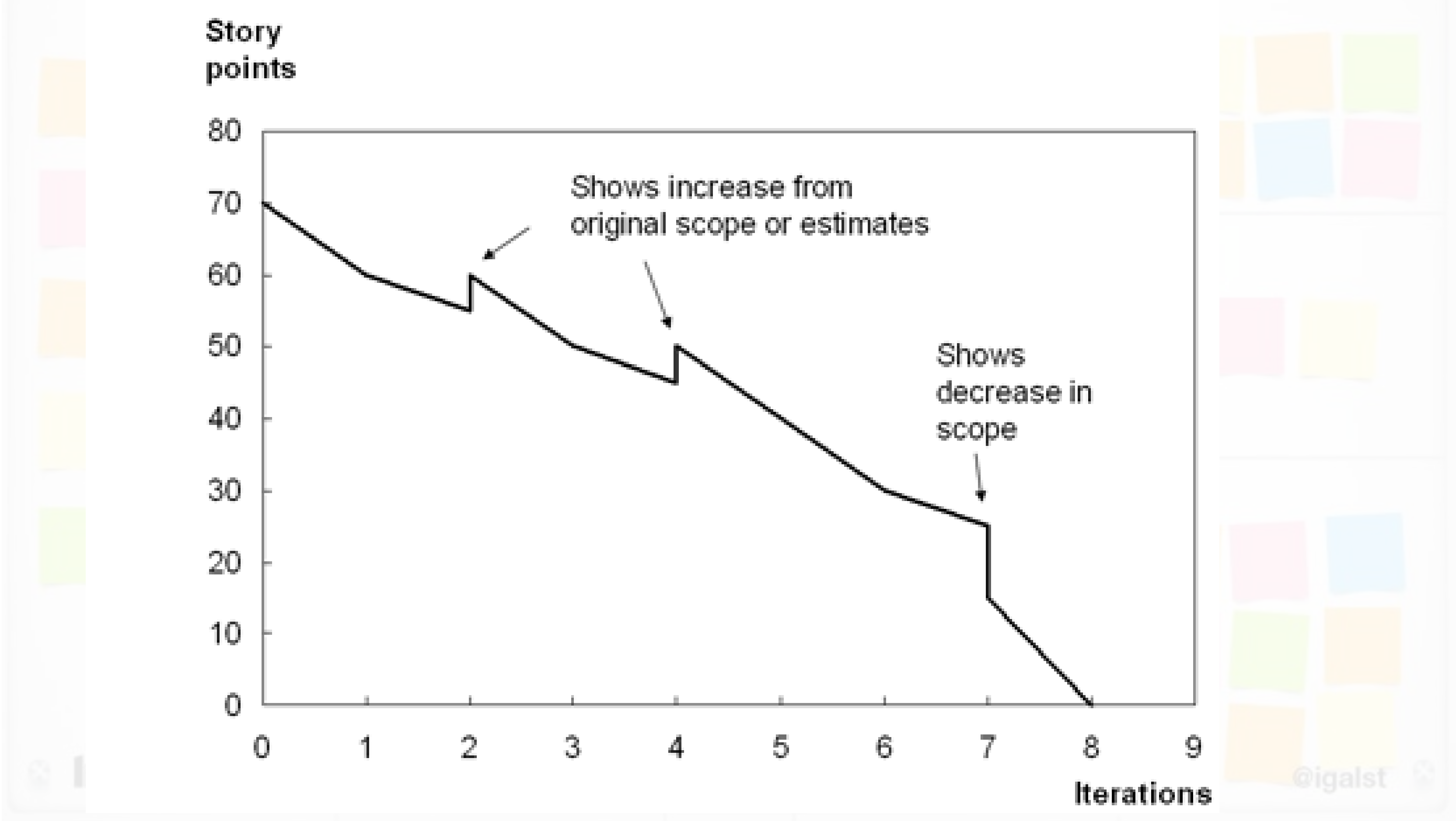
Scrum Artifacts

Burn-up Charts



Scrum Artifacts

Burndown Charts

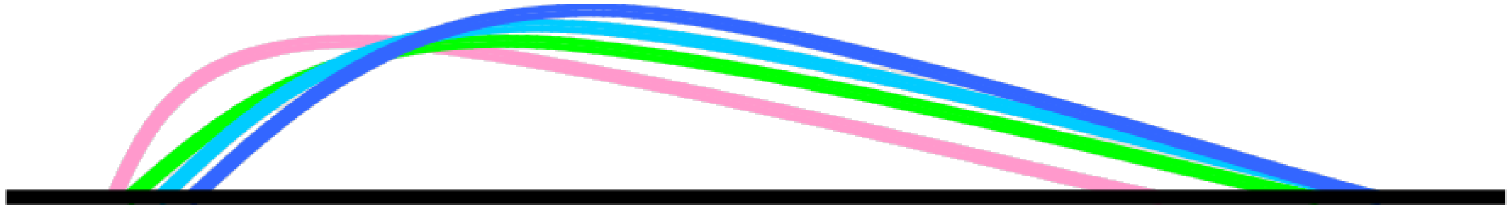


Sequential vs Overlapping Development

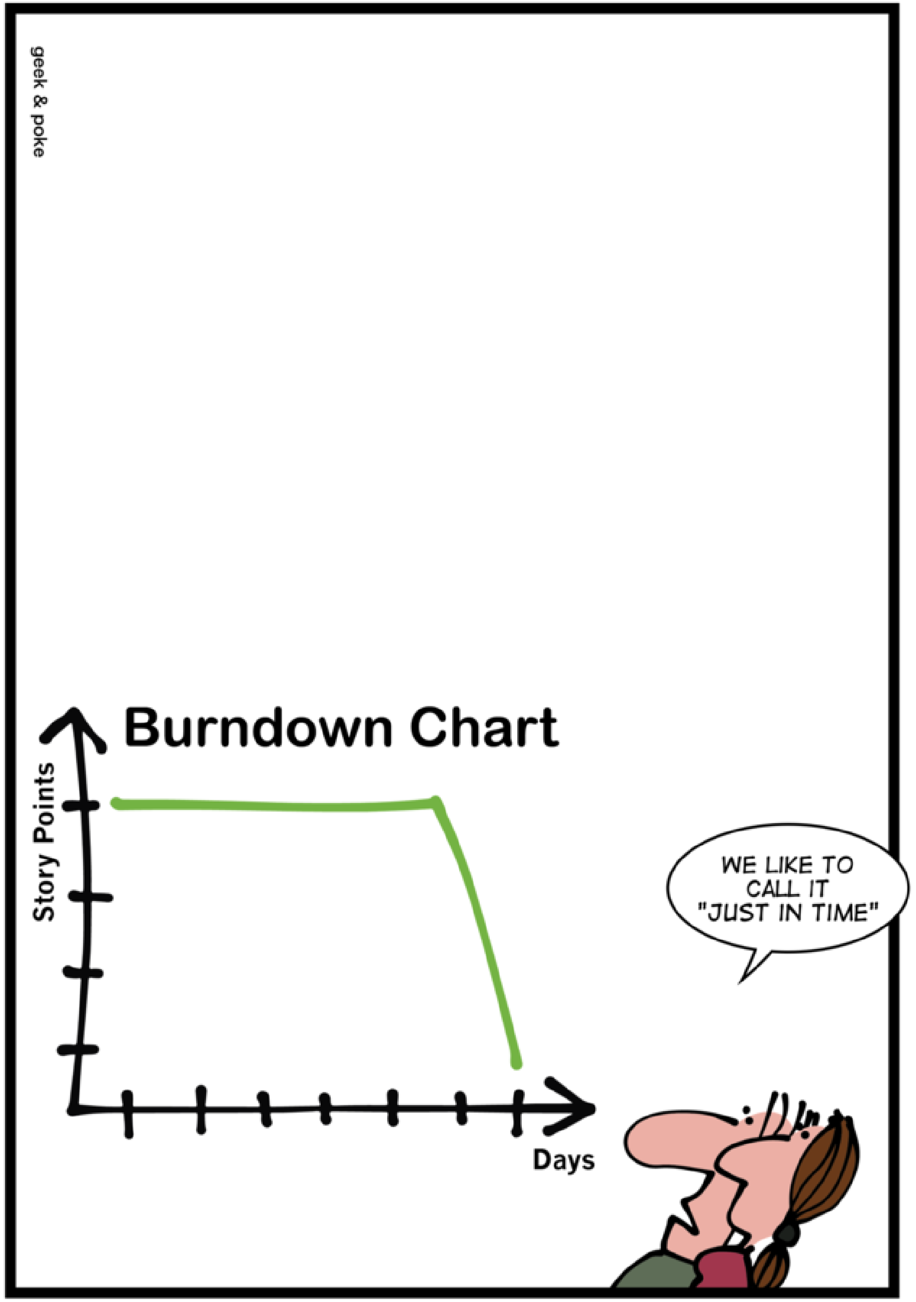
Requirements | Design | Code | Test

Rather than doing all of one thing at a time

...Scrum teams do a little of everything all the time



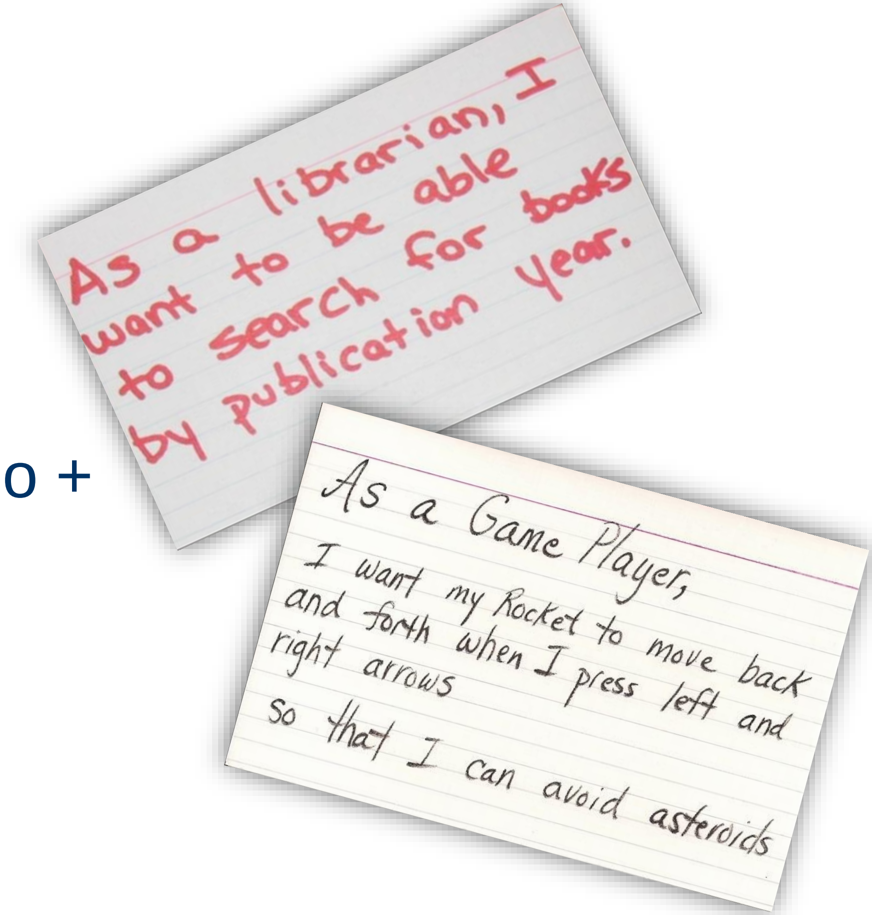
Source: “The New New Product Development Game”
Takeuki, Nonaka, HBR, Jan 1986



WATERFALL IS BACK

User stories

- A user story is one or more sentences in the everyday/business language that captures what a user does or needs to do +
  - a description
  - acceptance criteria



As a librarian, I
want to be able
to search for books
by publication year.

As a Game Player,
I want my Rocket to move back
and forth when I press left and
right arrows
So that I can avoid asteroids

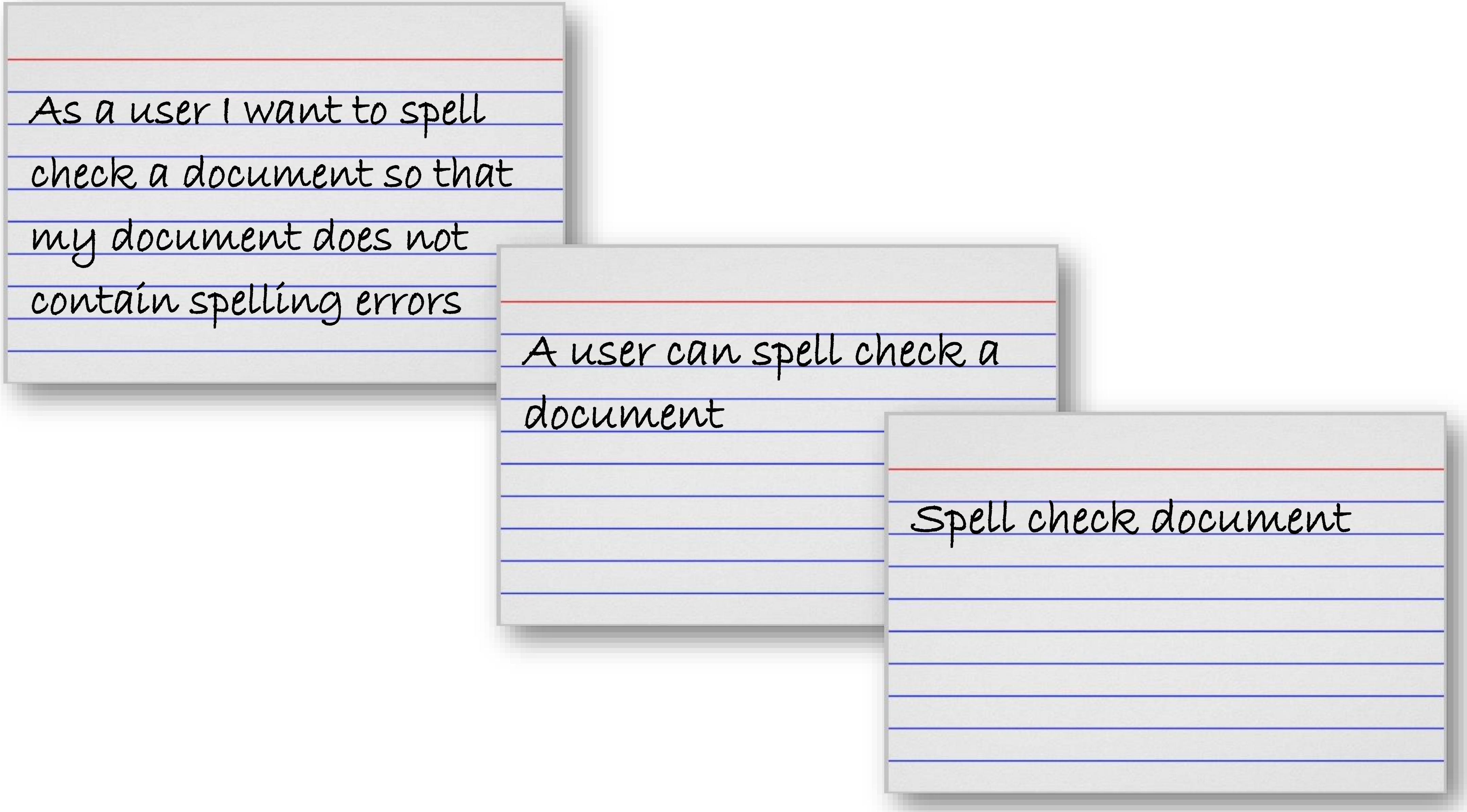
- Format:

  As a type of user,

  I {**want** / **can** / **need** / **am required to**} <some goal>

  so that <some reason>

User stories

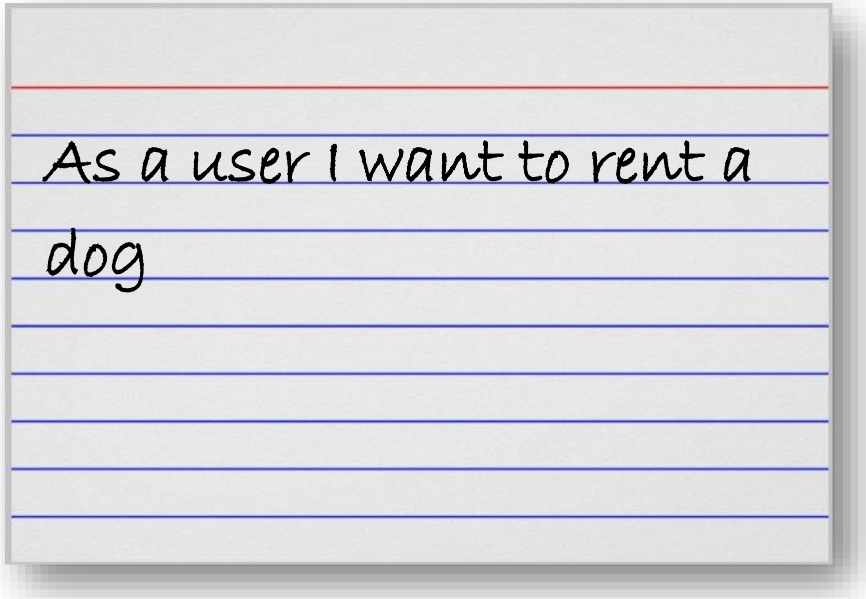


As a user I want to spell check a document so that my document does not contain spelling errors

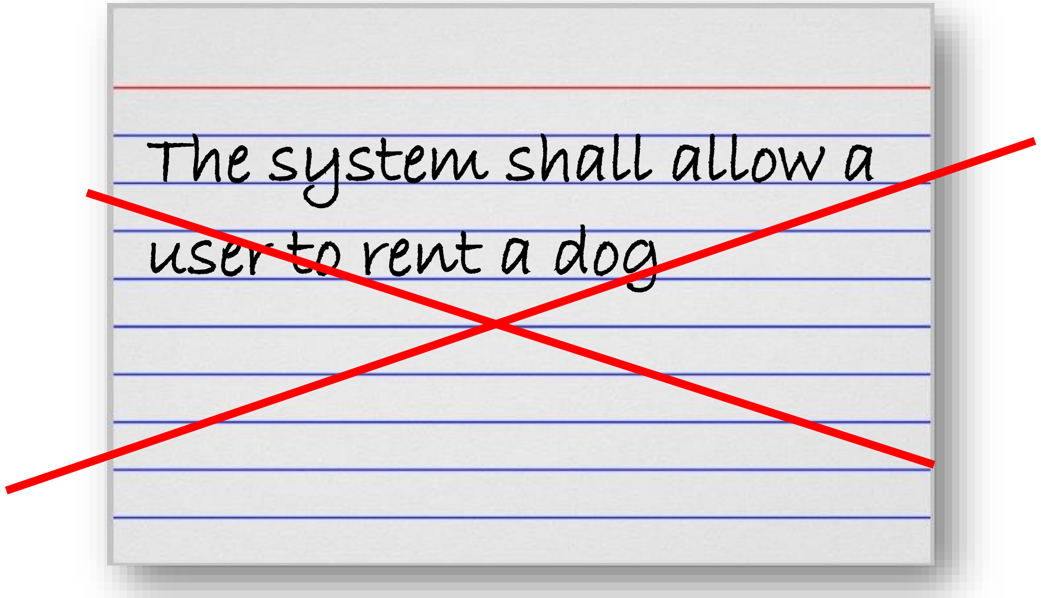
A user can spell check a document

Spell check document

User stories



As a user I want to rent a dog

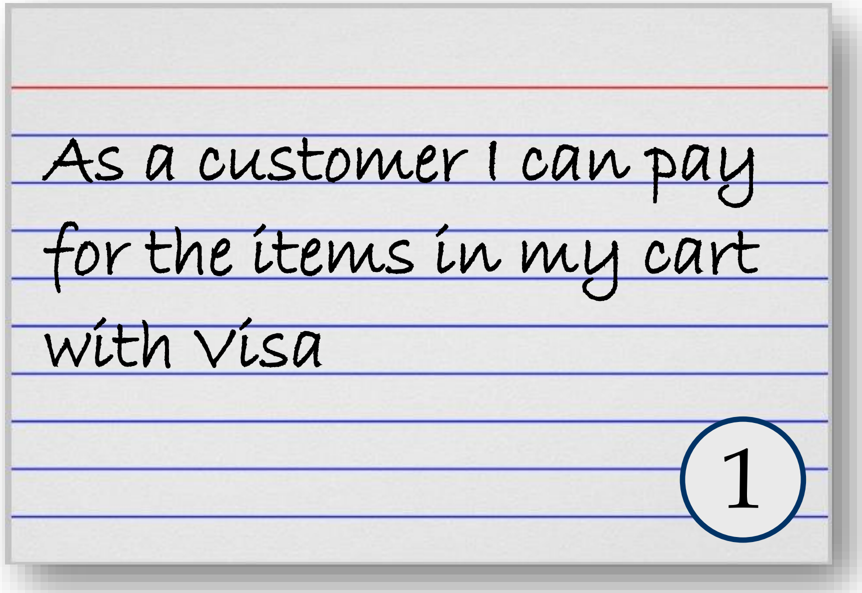


~~The system shall allow a user to rent a dog~~

INVEST mnemonic

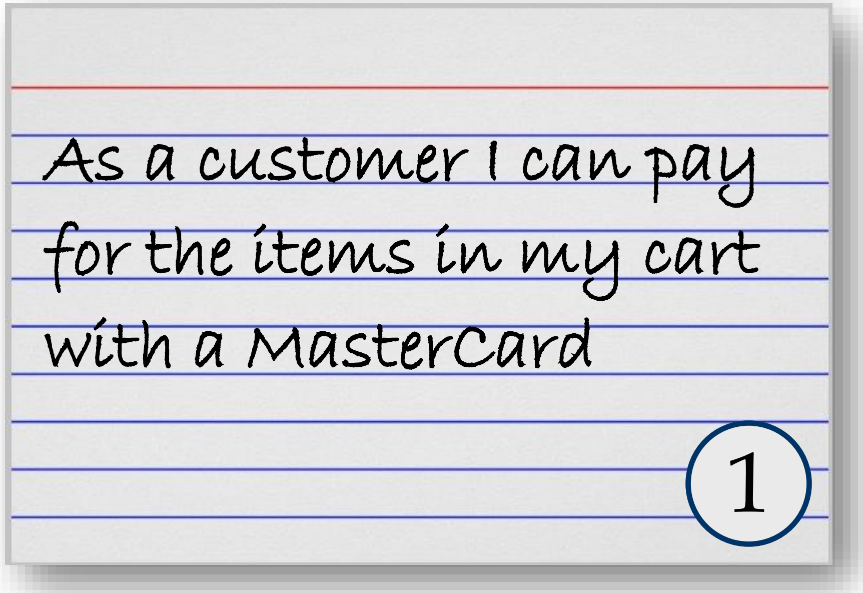
- ***I**ndependent* – allow to reprioritize in any order
- ***N**egotiable* – discuss and make tradeoffs
- ***V**aluable* – clear business benefits
- ***E**stimable* – team is able to estimate the effort
- ***S**mall* – easier to estimate and test
- ***T**estable* – how do you know when it is done?

Independent



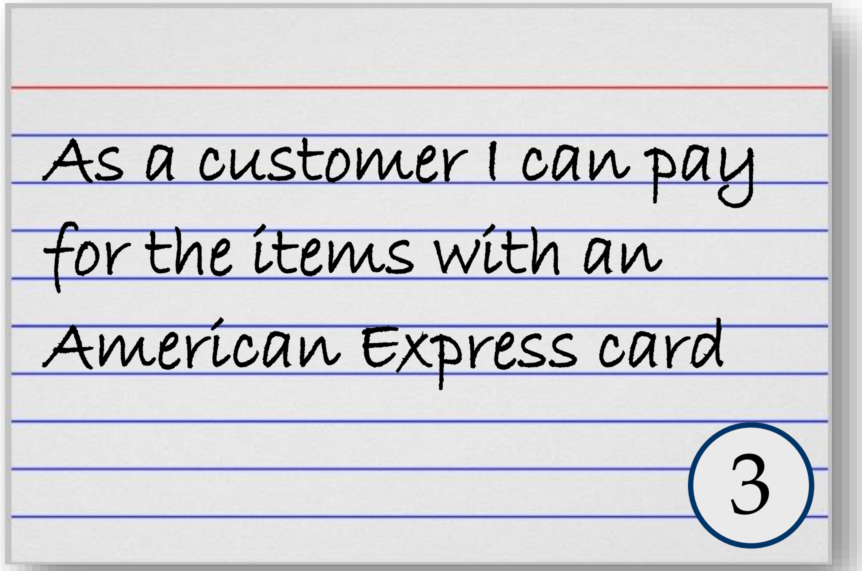
As a customer I can pay for the items in my cart with Visa

1



As a customer I can pay for the items in my cart with a MasterCard

1

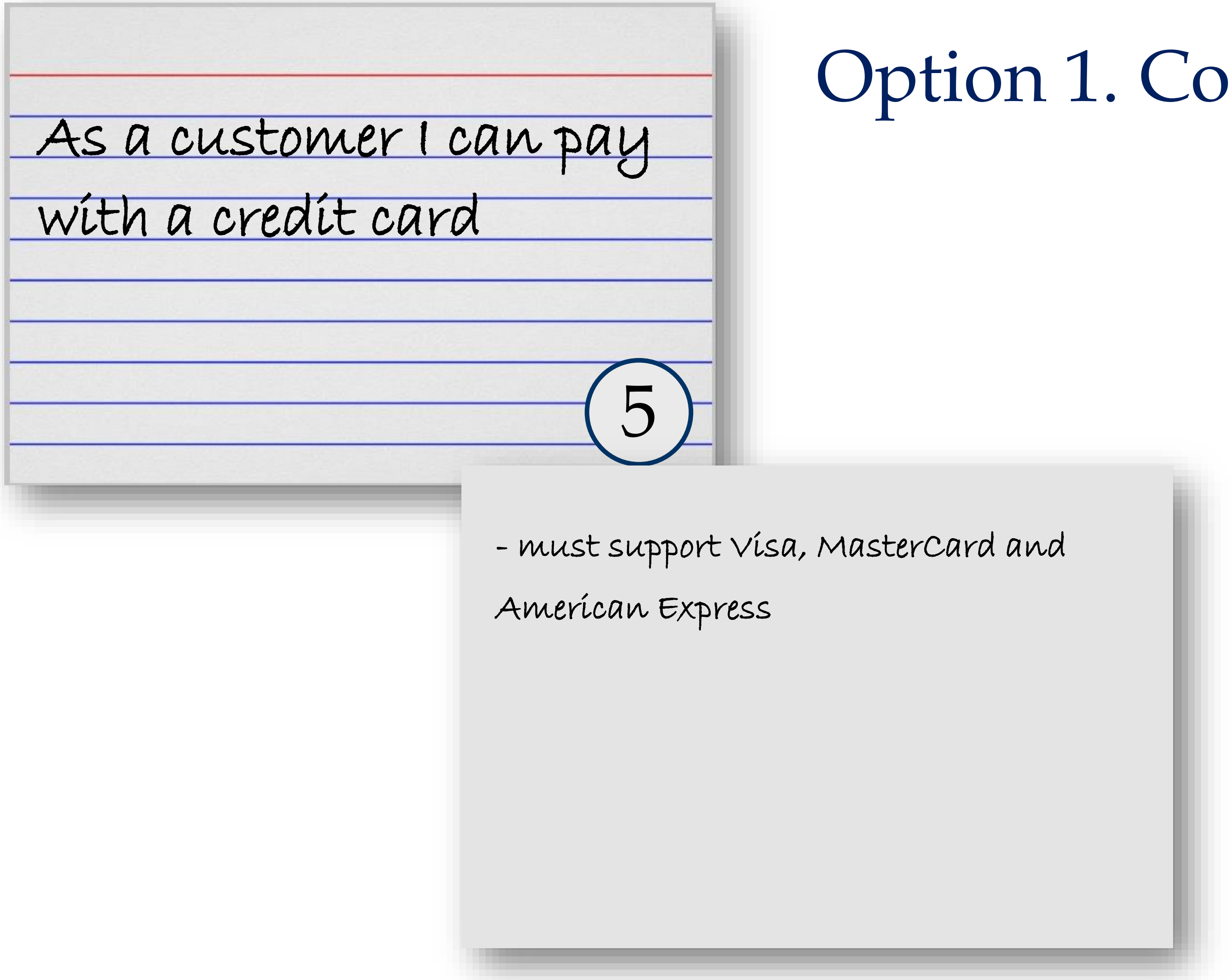


As a customer I can pay for the items with an American Express card

3

Independent

Option 1. Combine the stories

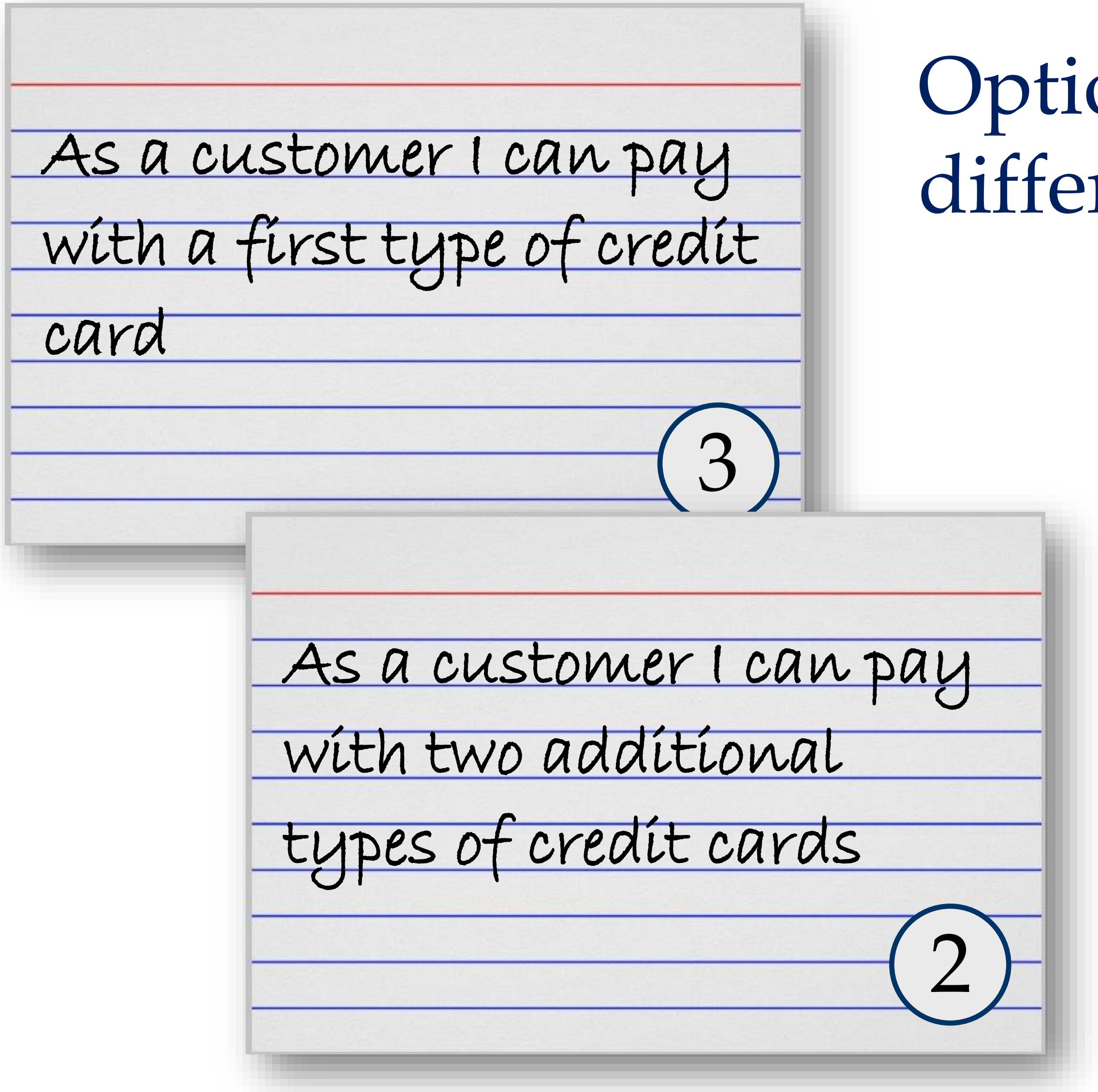


As a customer I can pay with a credit card

5

- must support Visa, MasterCard and American Express

Independent



As a customer I can pay with a first type of credit card

3

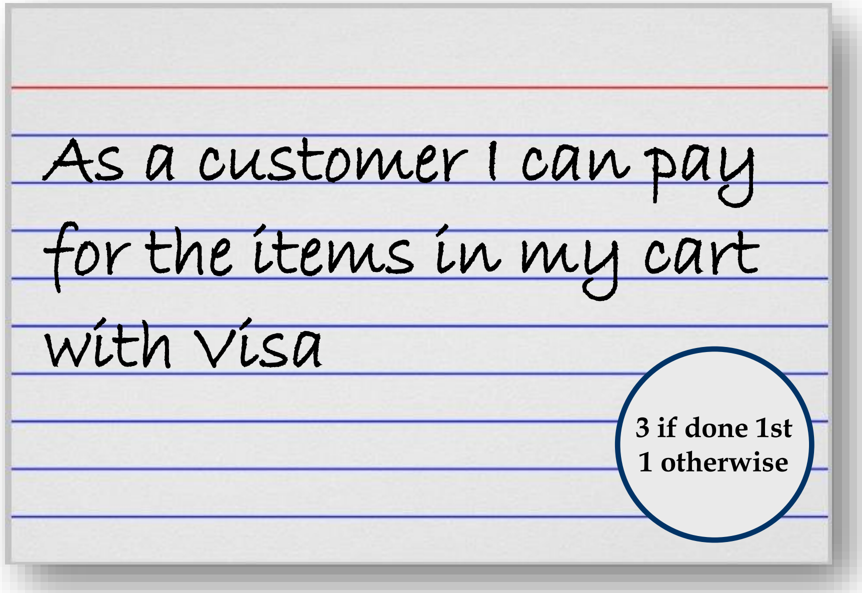
As a customer I can pay with two additional types of credit cards

2

Option 2. Split across a different dimension

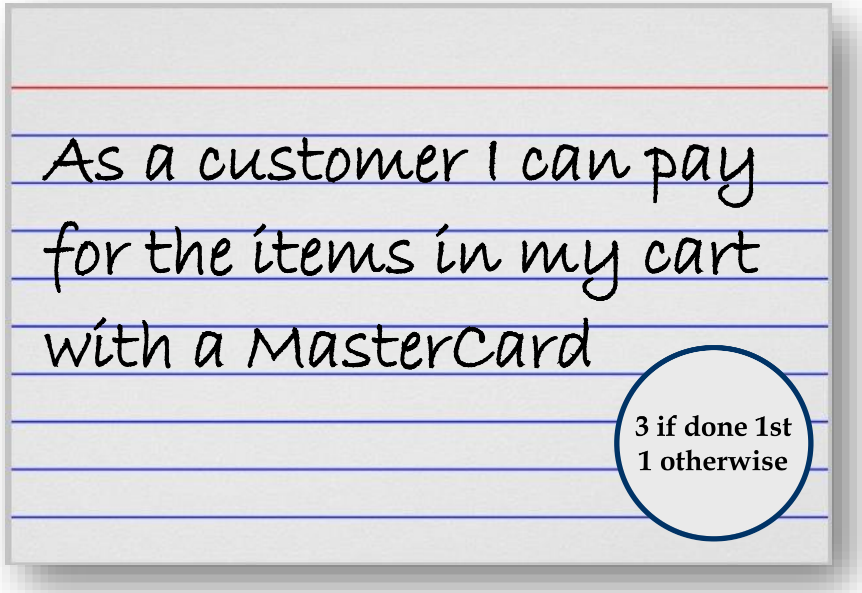
Independent

Option 3. Write two estimates



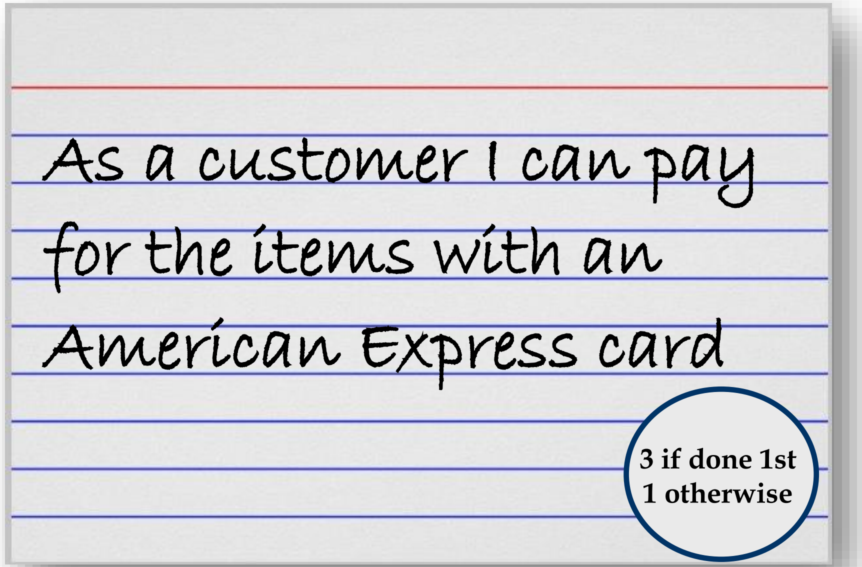
As a customer I can pay for the items in my cart with Visa

3 if done 1st
1 otherwise



As a customer I can pay for the items in my cart with a MasterCard

3 if done 1st
1 otherwise

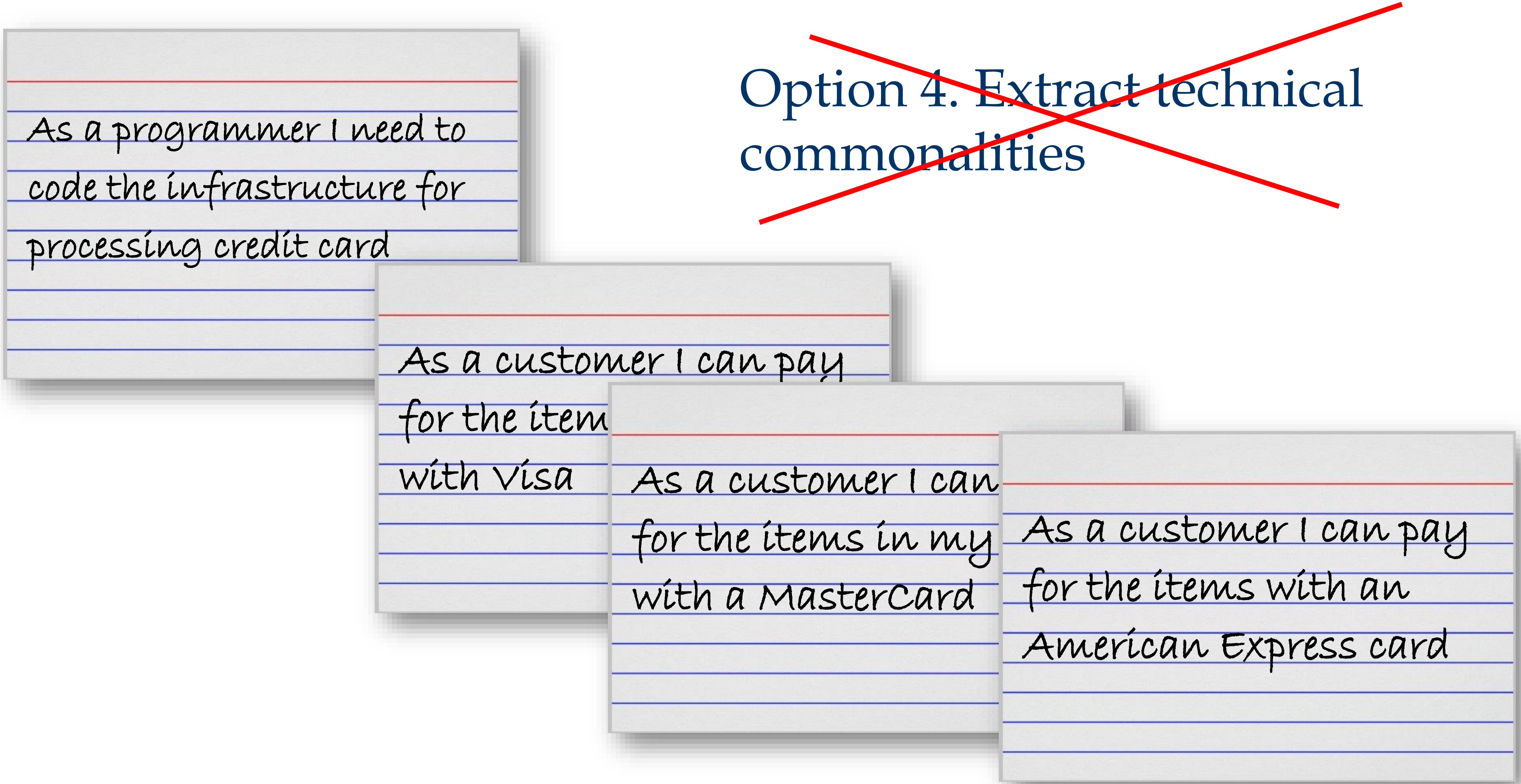


As a customer I can pay for the items with an American Express card

3 if done 1st
1 otherwise

Independent

~~Option 4. Extract technical commonalities~~



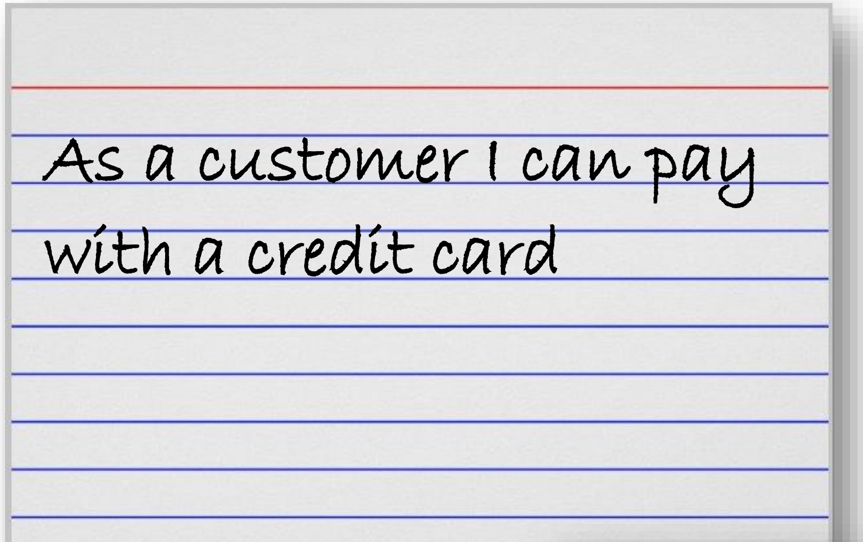
As a programmer I need to code the infrastructure for processing credit card

As a customer I can pay for the item with Visa

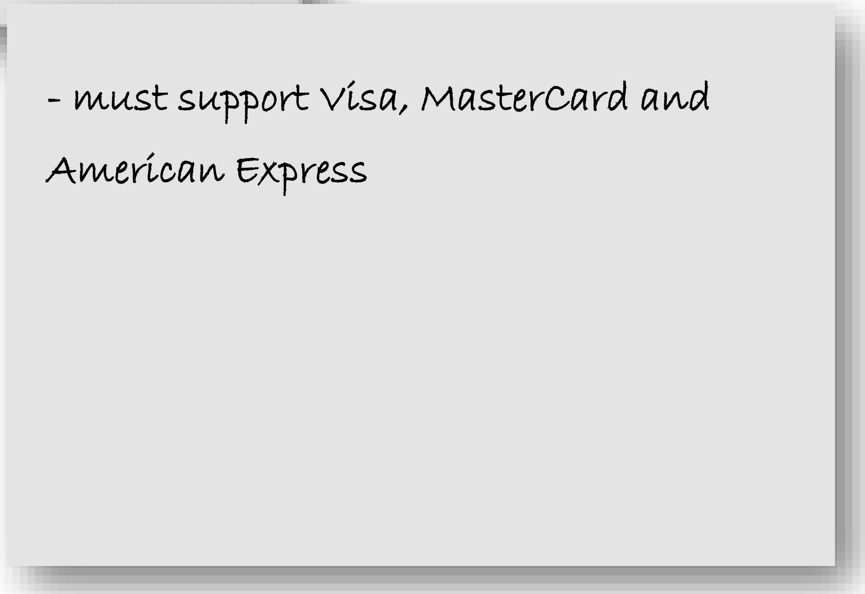
As a customer I can for the items in my with a MasterCard

As a customer I can pay for the items with an American Express card

Negotiable

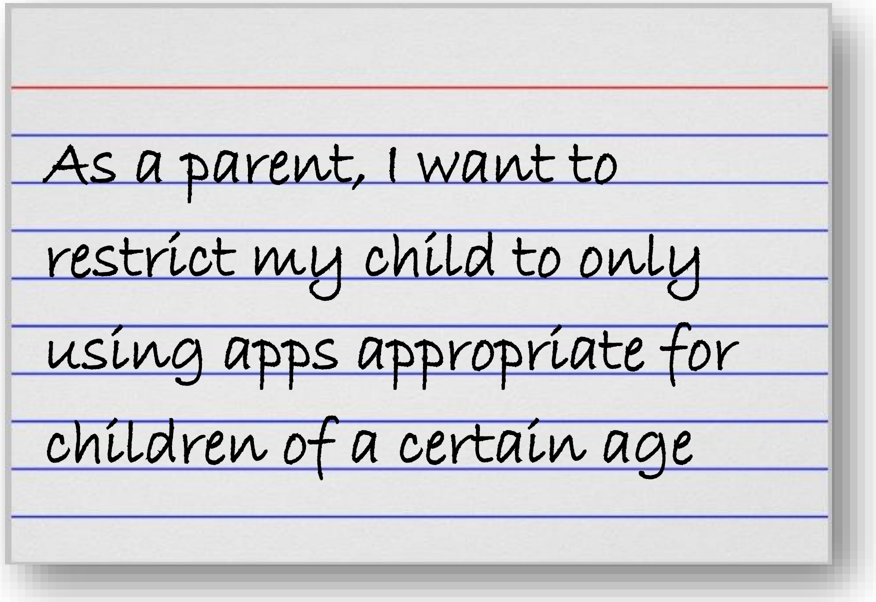


As a customer I can pay with a credit card

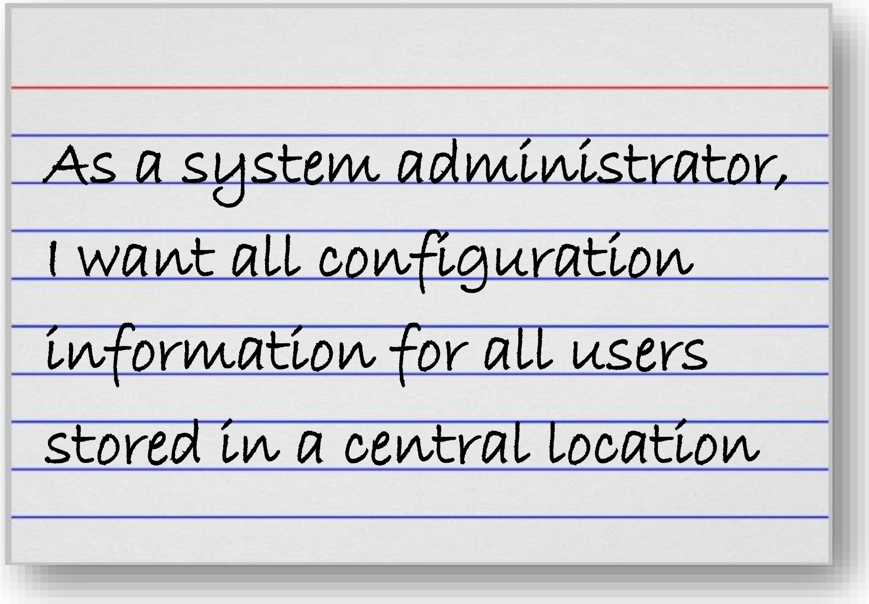


- must support Visa, MasterCard and American Express

Valuable



As a parent, I want to restrict my child to only using apps appropriate for children of a certain age

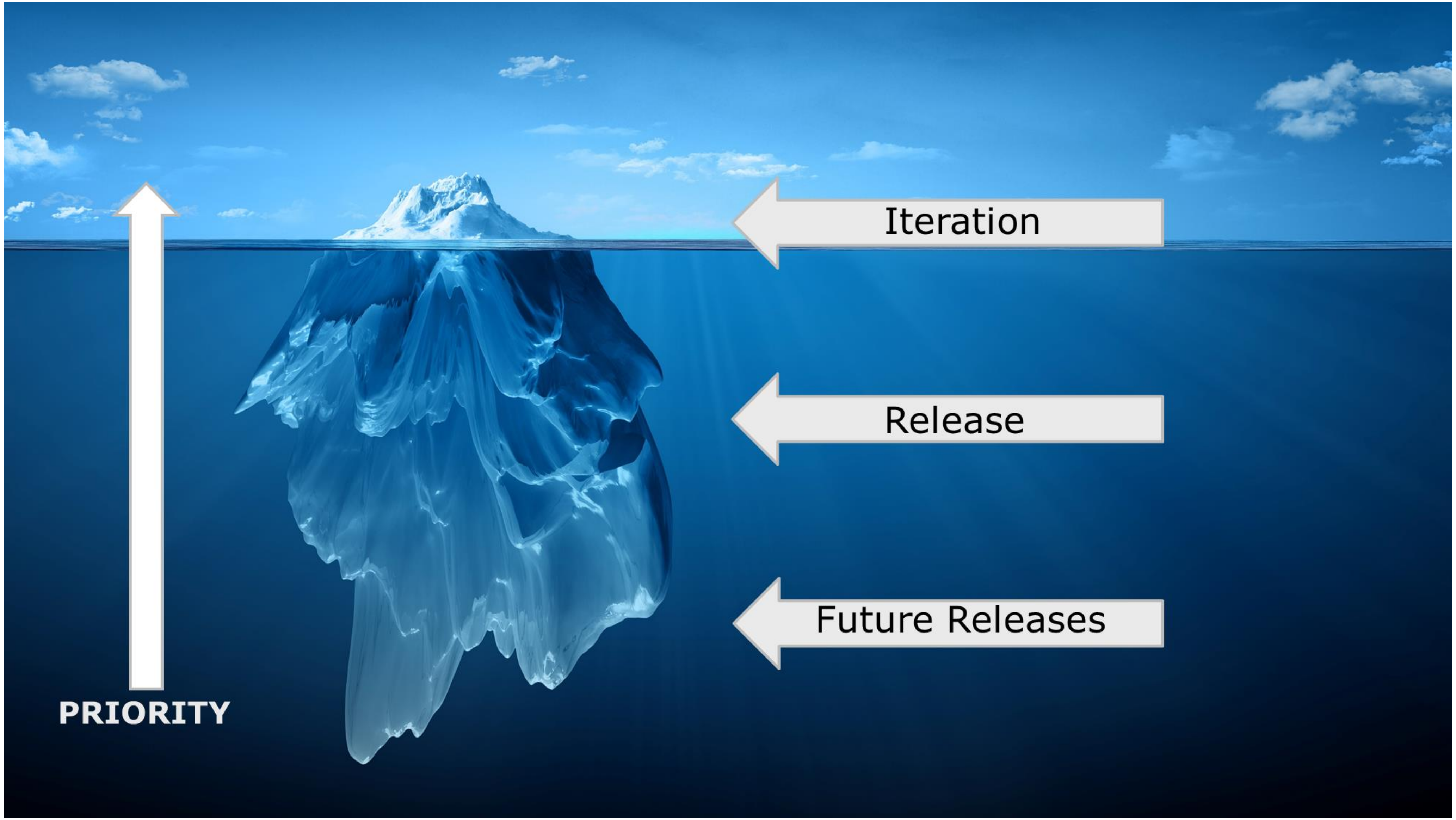


As a system administrator, I want all configuration information for all users stored in a central location

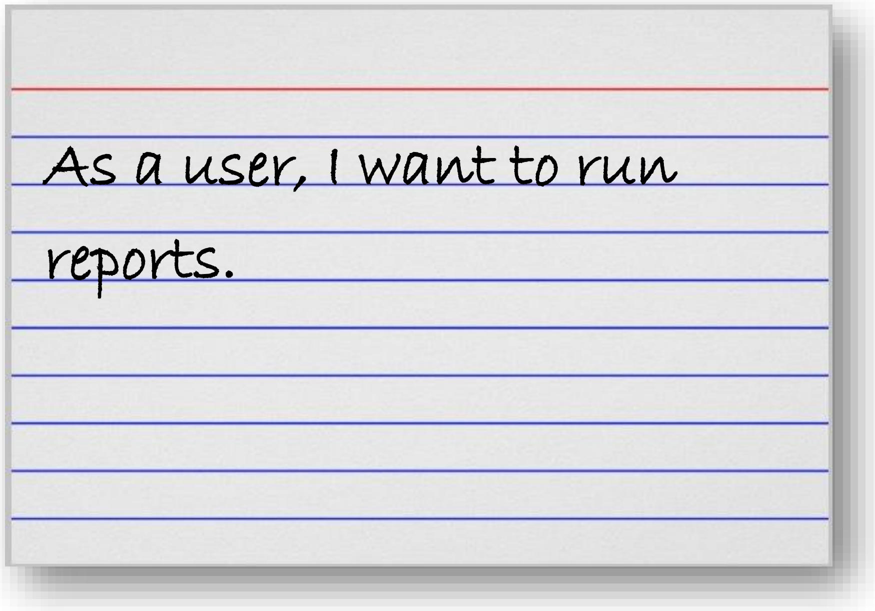
Estimable / Estimatable

- The story is too big
- Too much is unknown about the story
- Developers lack domain knowledge
- Developers lack technical knowledge

Small / Sized appropriately



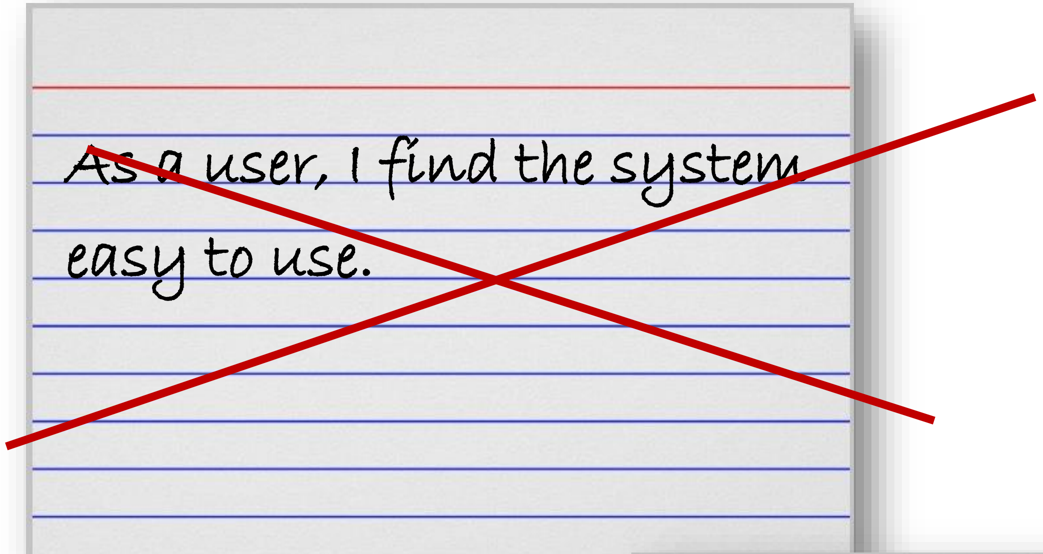
Epics



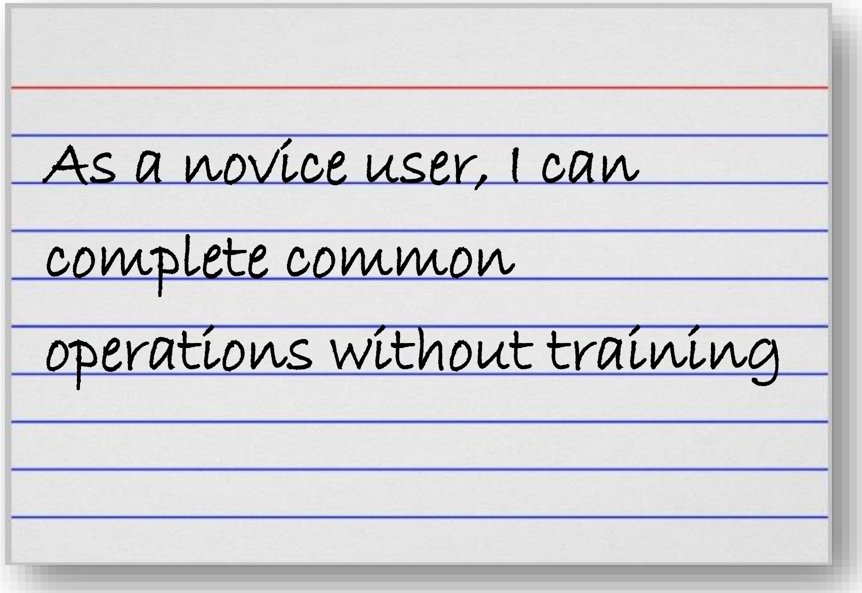
As a user, I want to run reports.

a story that is bigger than one team can do in one iteration

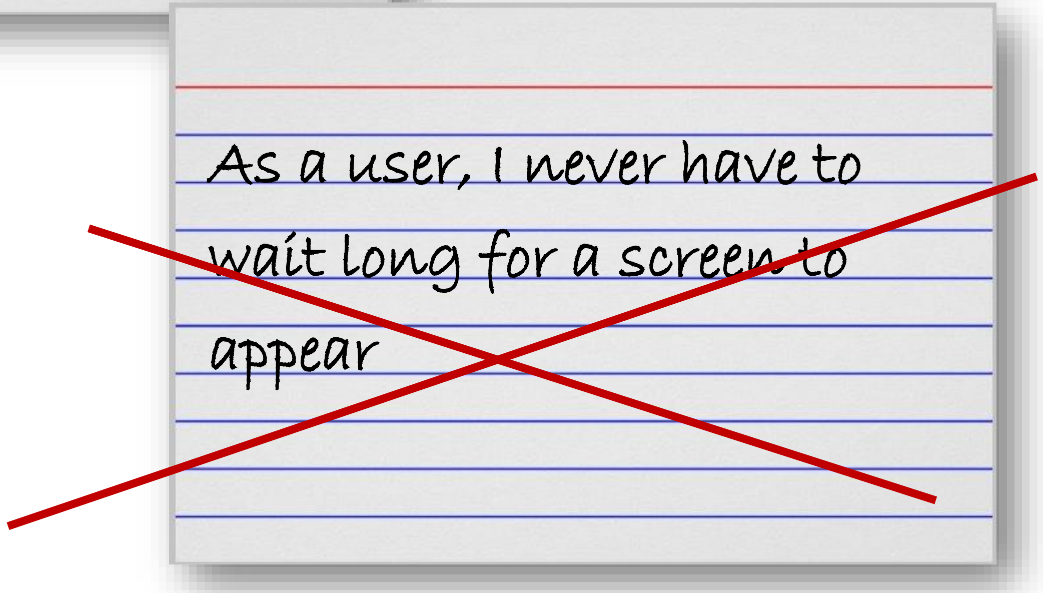
Testable



~~As a user, I find the system easy to use.~~

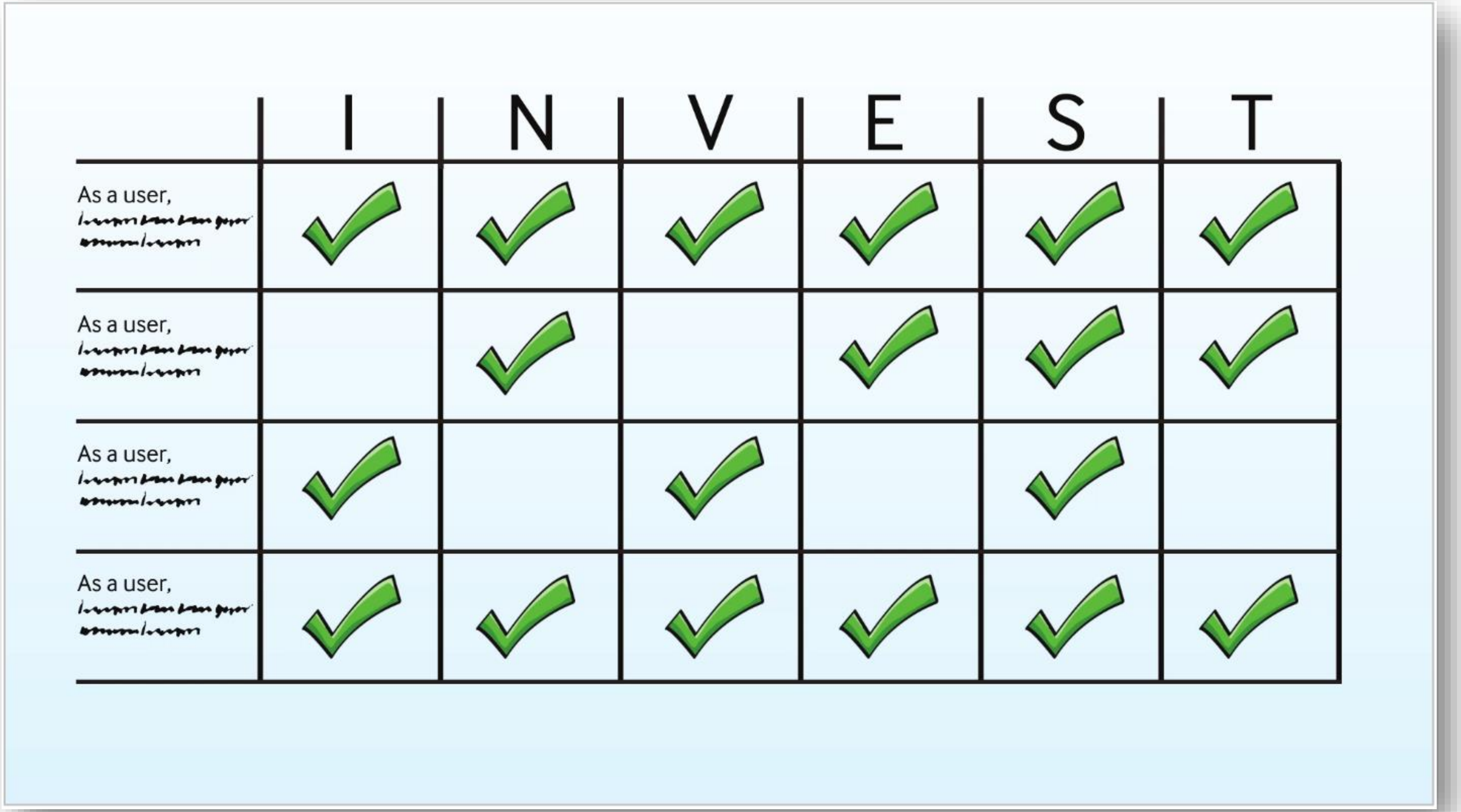


As a novice user, I can complete common operations without training



~~As a user, I never have to wait long for a screen to appear~~

INVEST mnemonic



	I	N	V	E	S	T
As a user,	✓	✓	✓	✓	✓	✓
As a user,		✓		✓	✓	✓
As a user,	✓		✓		✓	
As a user,	✓	✓	✓	✓	✓	✓

When to write user stories?

- Randomly, whenever a new idea occurs
- During sprint review meetings
- During product backlog refinement / grooming meetings,
- During story-writing workshops

When to write user stories?

- Minimum Viable Product (MVP)
  - the version of a product which allows a team to collect the maximum amount of information with the least effort

- Minimum Marketable Feature (MMF)
  - A chunk of functionality that delivers a subset of the requirements, and that is capable of returning value to the customer when released as an independent entity