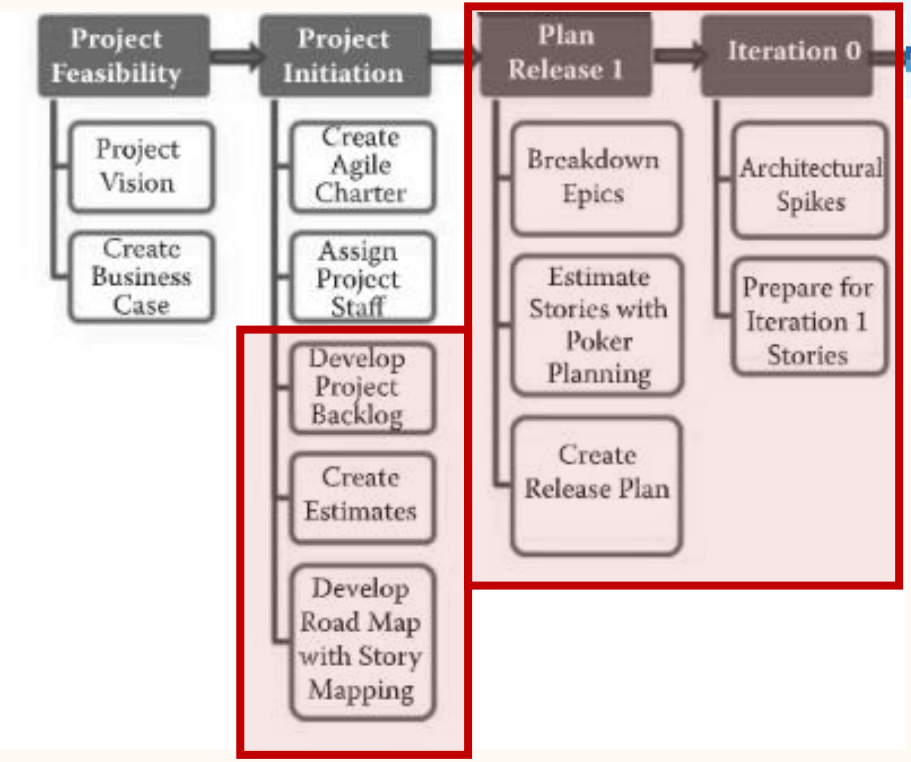
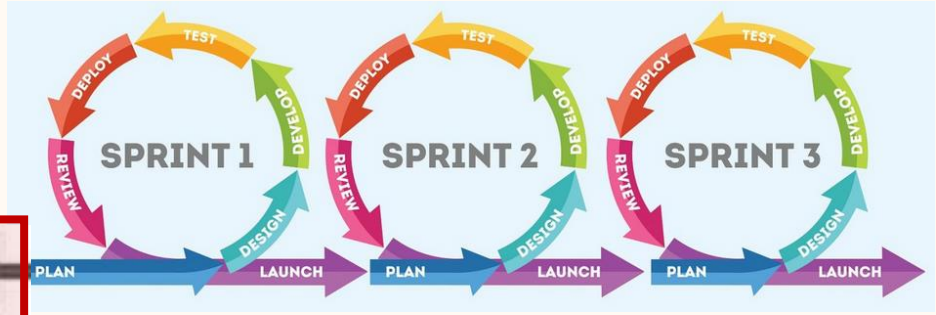
User Stories
& More Planning

Context





- Remember: Agile involves ~2-week iterations (not detailed long-term planning)
- But planning is still necessary

- Today's focus: How do we plan for Sprint 0/1?

Outline

- User personas & user stories
- Product backlog
- Release planning
- Sprint planning, estimating
- Sprint 1

User personas

- A description of a particular product **user**
- Organizing requirements around specific **user** and their needs puts more focus on…
  - Understanding the potential **value**
  - **Who** benefits from that **value**
- Identify personas as specifically as possible
  - Category of user (user roles), or a specific user
  - Model personality, specific skills, interests

Build empathy, see from user perspective

An Example

User: Fred Fish, Director of Food Services "Get me out of the office and into the kitchen"

Background: Fred is director of foodservices for Boise Controls, a mid-sized manufacturer of electronic devices used in home security systems. He uses a computer, but he's a chef by trade and not so *computer-savvy*.

Key Goals: As a manager, Fred doesn't get his hands (literally) dirty the way he used to. He stops in at all the Boise Controls sites and sticks his fingers into things once in a while to stay in touch with cooks and cooking. He wants to learn computer tools but not at the expense of managing his kitchens. A computer is just another tool for getting his administrative tasks done.

A Usage Scenario: At the start of every quarter, he meets with the head chefs and plans out the next quarter's menus. That's one of his favorite things, because each chef gets to demonstrate a new meal. The chefs spend time in the kitchen exploring each new dish. When they're done, he sends the food to his staff and his manager.

He's not a computer whiz. On a good day, he can drag in some clip art and do some formatting with fonts. Once in a while, he'll format menus with the new editor on his MacBook Pro.

Another Example

Clark Andrews

AGE	26
OCCUPATION	Software Developer
STATUS	Single
LOCATION	San Jose, CA
TIER	Experiment Hacker
ARCHETYPE	The Computer Nerd

Friendly | Clever | Go-Getter

"I feel like there's a smarter way for me to transition into a healthier lifestyle."

Motivations

- Incentive
- Fear
- Achievement
- Growth
- Power
- Social

Goals

- To cut down on unhealthy eating and drinking habits
- To measure multiple aspects of life more scientifically
- To set goals and see and make positive impacts on his life

Frustrations

- Unfamiliar with wearable technology
- Saturated tracking market
- Manual tracking is too time consuming

Bio

Aaron is a systems software developer, a "data junkie" and for the past couple years, has been very interested in tracking aspects of his health and performance. Aaron wants to track his mood, happiness, sleep quality and how his eating and exercise habits affects his well being. Although he only drinks occasionally with friends on the weekend, he would like to cut down on alcohol intake.

Personality

- Extrovert – Introvert
- Sensing – Intuition
- Thinking – Feeling
- Judging – Perceiving

Technology

- IT & Internet
- Software
- Mobile Apps
- Social Networks

Brands

Caveat about personas

- Useful for other development efforts
  - UX/UI design
  - Sales/marketing
- Good to have, so you don’t end up…
  - Developing for yourselves
  - Developing for an overly generic “user”
- But don’t spend too much time on these yourself unless you’re doing UX/UI design

User Stories

- Succinct way of defining requirements
- "Headline" Format:

  As a <role> I want to <goal> so that <benefit>

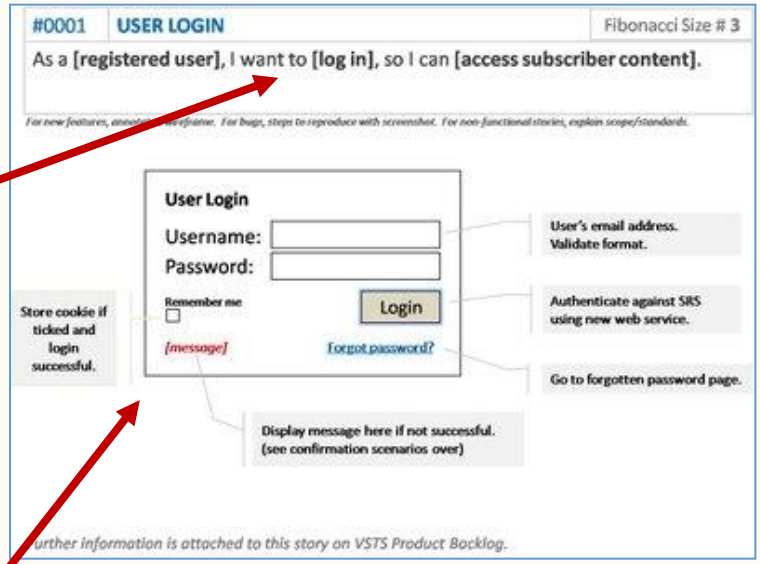
  Who? **What?** **Why?**
- Easily understood by developers and users
- Breaks requirements into small chunks of functionality

Examples (travel website)

- As a **traveler** I want to **reserve a hotel room**.
- As a **customer** I want to **cancel a reservation**.
- As a **vacation traveler** I want to **see photos of the hotel**.
- As a **frequent flyer** I want to **rebook a past trip** so that **I can save time booking trips I take often**.

User story details

- User stories usually include
  - The “headline” story
  - Ordered steps
  - Success criteria
  - Effort estimations
  - Other notes/conversations
- May include UI *sketches*, failure conditions…
- Corresponds to ***Issue*** in GitLab



Sample issue from GitLab

mdp > CoolX > Issues > #1

Open Created 2 years ago by mdp Maintainer Close issue

As an A I want to B so I can C again

Drag your designs here or click to upload.

Linked issues 0

0 0 Oldest first Show all activity Create branch

mdp @mdp changed weight to **13** 2 years ago

mdp @mdp changed milestone to %Sprint 1 (Spin-up) 2 years ago

mdp @mdp added B Feature label 2 years ago

mdp @mdp added Tank color label 2 years ago

Add a to do

0 Assignees Edit
None - assign yourself

Milestone Edit
Milestone 2 (CS619) (expired)

Time tracking
No estimate or time spent

Due date Edit
None

Labels Edit
B Feature

Weight Edit
13 - remove weight

Health status Edit
None

User story guidelines

- Use functional terms
  - Do: define expected result in “headline”
  - Don’t: tell developer how to achieve the result
- Keep it brief
  - “Placeholder for a conversation"
  - Details elaborated in face-to-face communications, notes
- Include acceptance criteria prominently

INVEST Guidelines

- Independent—we want to be able to develop in any sequence (when feasible)
- Negotiable—avoid “how” details so trade-offs can be discussed
- Valuable—identify value to users, so can compare value among stories

INVEST Guidelines

- Estimable—the team must be able to use them for planning
- Small—easier to estimate/plan
  - At sprint-level: Design, coding, and testing for story fits within **one iteration/sprint**.
- Testable—document acceptance criteria (or "done") for the story, which lead to test cases

First part of Lab 2

- Lab document
- The lab this week is in two parts
  - Part I: Personas and User Stories
    - Create a few personas for your project
    - Create a number of user stories for your project
    - Add the user stories to your GitLab project
  - Part II: Sprint Planning
- To help with generation of user stories…

User stories exercise (Part I)

- Get into groups of 3 or 4 and work on…
- Exercise sheet—at least start step 4
- When coming up with user stories:
  - Think of most vital (highest-value) functionality to the personas you identified
  - Make sure to include acceptance criteria
  - Ask questions!

Don't review other groups' results yet…

Outline

- User personas & user stories
- Product backlog
- Release planning
- Sprint planning, estimating
- Sprint 1

Product Backlog

- (Priority) queue of work to be done
- Typically consists of user stories
- Dynamic ("backlog grooming")
  - Stories continuously refined, prioritized
  - Should have enough for at least a few sprints
- Populated with increasing detail
  - Start with Epics/high-level user stories
  - Progressively elaborate to release, sprint levels

Populating the backlog

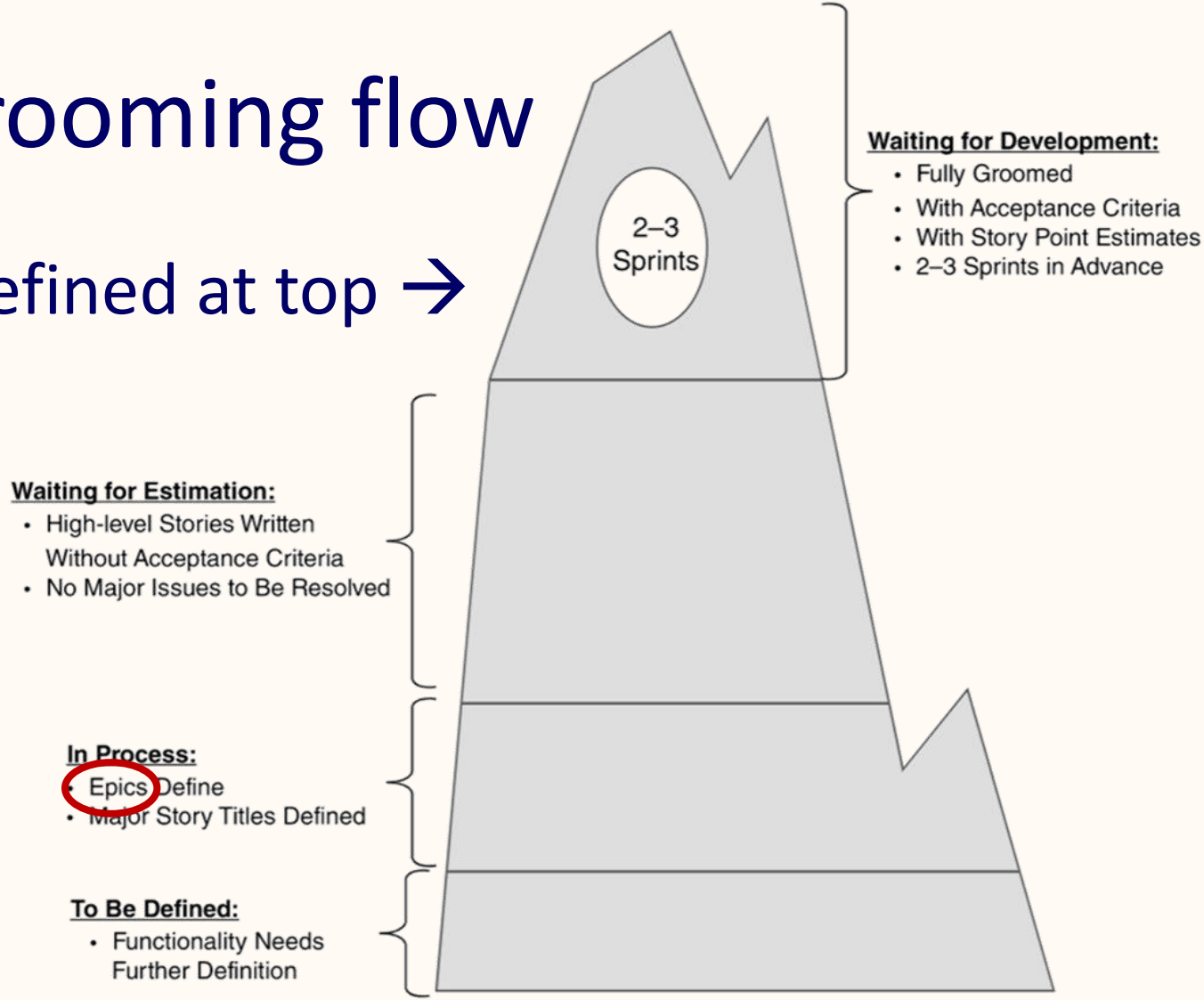
- Backlog items can come from many places
  - Executives, business people
  - Customers (feature requests, bug reports)
  - Support teams
  - Development team
- Ultimately, responsibility of **Product Owner**
  - Inputs & clarifies backlog items (developer ?'s)
  - Maintains priorities for backlog items

Grooming the backlog

- Product Owner can’t prioritize without input
- Development team…
  - Asks clarifying questions
  - Helps form success criteria, definition of “done”
  - Rejects story/item if can’t estimate time
    - Poor requirements/story (“What do you mean?”)
    - Lack of proper tools or knowledge (“Can we prepare?”)
- PO can then prioritize based on **value** & **risk**

Backlog grooming flow

Most-refined at top →



Breaking down epics for backlog

- Remember: epics group related smaller stories
- But… may start off as a large story
  - Needs to be broken down
- Example:

As an electronic banking customer, I want to be able to easily make an online deposit of a check into my bank account through my IPhone® so that I can save the time required to send a check for deposit through the mail and I can have the money immediately credited to my checking account as soon as the deposit is completed electronically.

Can decompose epic by function

As an electronic banking customer, I want to be able to scan an image of the front and back of a check into the IPhone® so that it can be deposited electronically.

Step 1

As an electronic banking customer, I want to be able to enter the deposit information associated with an electronic deposit so that the correct amount will be deposited into the correct bank account when the electronic deposit is processed.

Step 2

As an electronic banking customer, I want to be able to electronically submit a scanned check and deposit information to the bank for deposit so that I can save the time associated with sending deposits by mail.

Step 3

As an electronic banking customer, I want to be able to receive confirmation of a completed electronic deposit so that I will know that the deposit was successfully processed.

Step 4

User stories exercise (Part II)

- Get back into groups of 3 or 4
- (Might still want to look at Exercise sheet)
- Try to break down some larger stories by…
  - Pieces of functionality (steps in a process?)
  - User role/type (different functionality allowed?)
  - What could you test separately?
- Go ahead and review/score other groups

Outline

- User personas & user stories
- Product backlog
- Release planning
- Sprint planning, estimating
- Sprint 1

Software Release

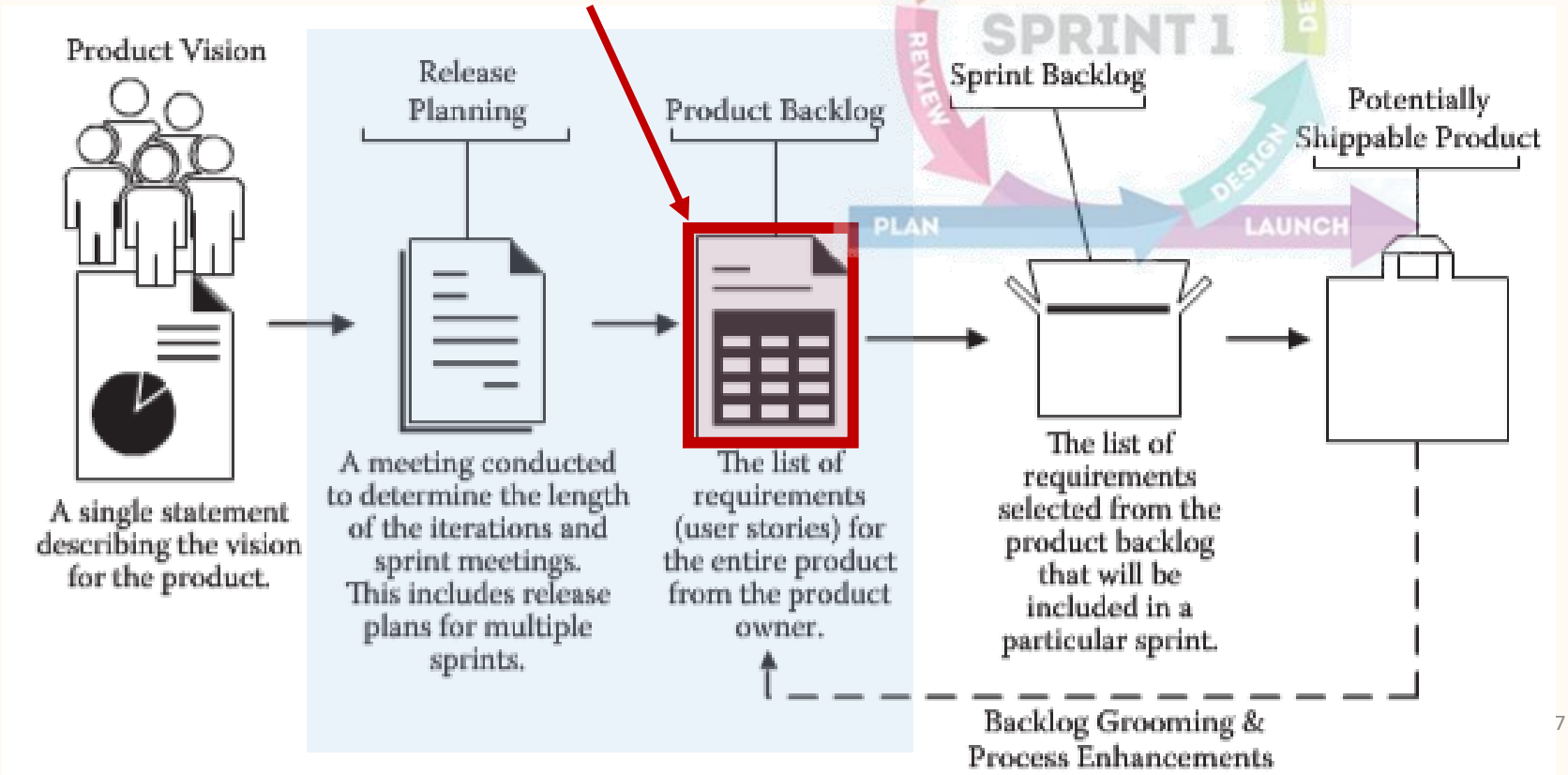
- Distribution of software product outside the agile team (to customer/end-user)
  - Major (1.0, 2.0): significant change to look/feel
  - Minor (2.1, 2.2): new/different features
  - Bug/hotfix: fixes to current functionality

version update

- Under DevOps:
  - Happens with each deployment
  - May happen to different parts at different times

Release planning

Product backlog populated



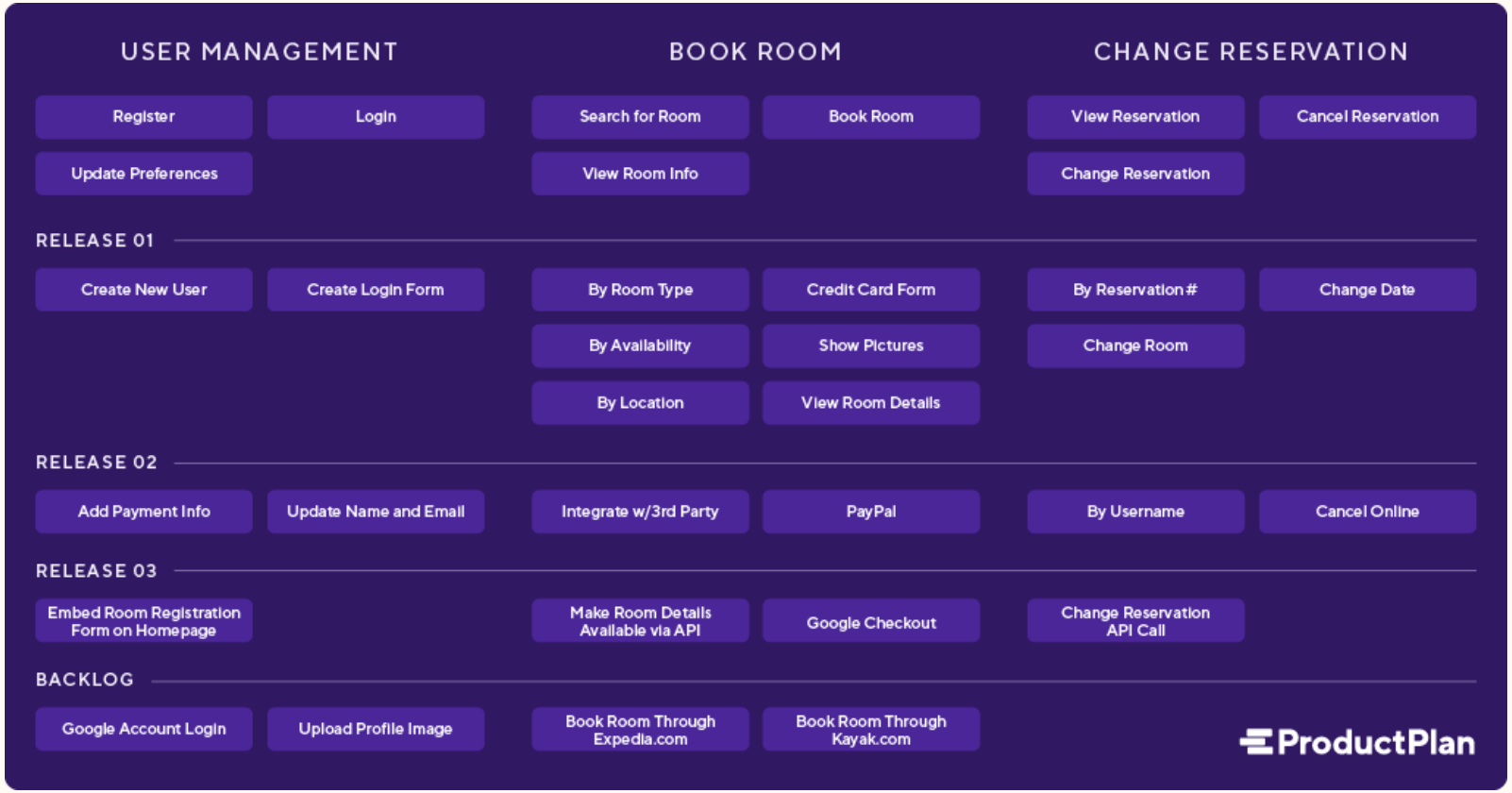
Getting to the release plan

- With backlog populated, can now…
- Estimate weights on stories (more on this soon…)
- Use **story mapping** to determine good order
- Create a **release plan**
  - What goes into release first release?
  - What might go into ***each sprint*** for the release?
    - Note: there are multiple sprints in each release
  - When is the target release date?

Story mapping

- Method for selecting & grouping features to go into a release
- On horizontal axis (X-axis)
  - Order stories by order users might do them
    - Chronological steps in “customer journey”
    - Could have epics as larger chunks, stories within them
- On vertical axis (Y-axis)
  - Order stories by priority (high=top, low=bottom)

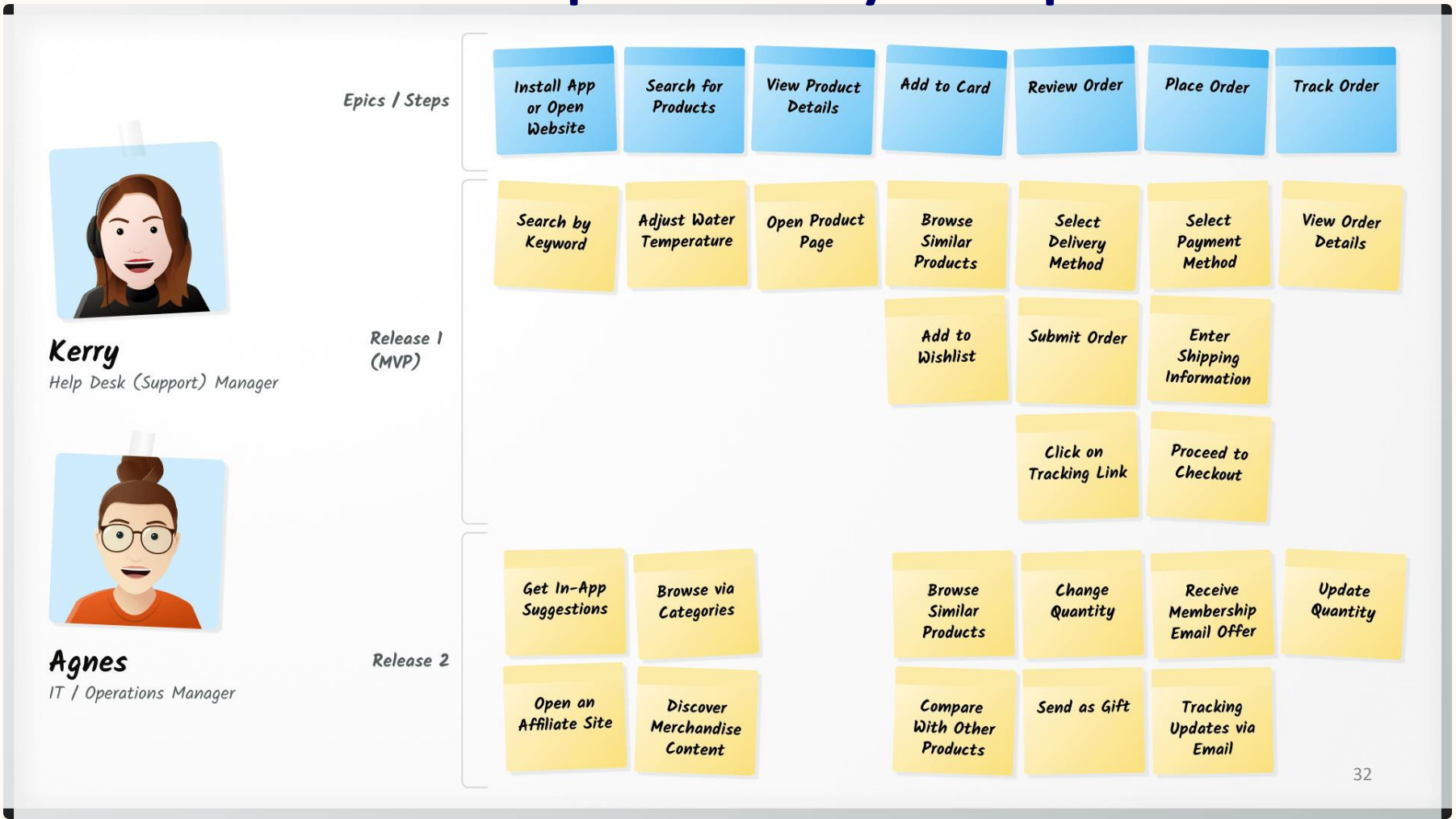
Sample story map



Story mapping details

- Often use cards/sticky-notes on wall/floor
- Helps everyone see “big picture”
  - Team sees how **persona** interacts with full system
  - Avoids high-value features users can’t get to
    - Because team sees what must be implemented first
  - Can see what is vital early, what can wait
- Releases tend to be slices across the map
- Good to update/redo regularly

Another sample story map



Create a release plan

- Determine goals and objectives per sprint
- Consider business goals, team's ***velocity***
  - # story points dev. team completes per iteration
  - Several iterations required for *reliable* velocity
- Consider priority of user stories
- Consensus must be reached

Releases for class project

- Planning is already mostly done for you
  - Initial planning/design sprint
  - Sprint 2: Data service
  - Sprint 3 (**MVP**—release 1.0?): Web app
  - Sprint 4 (release 1.1?): Your evolution of project
- Each sprint corresponds to a milestone
- After **MVP**, each sprint generally results in an **increment**—a product with additional value

Outline

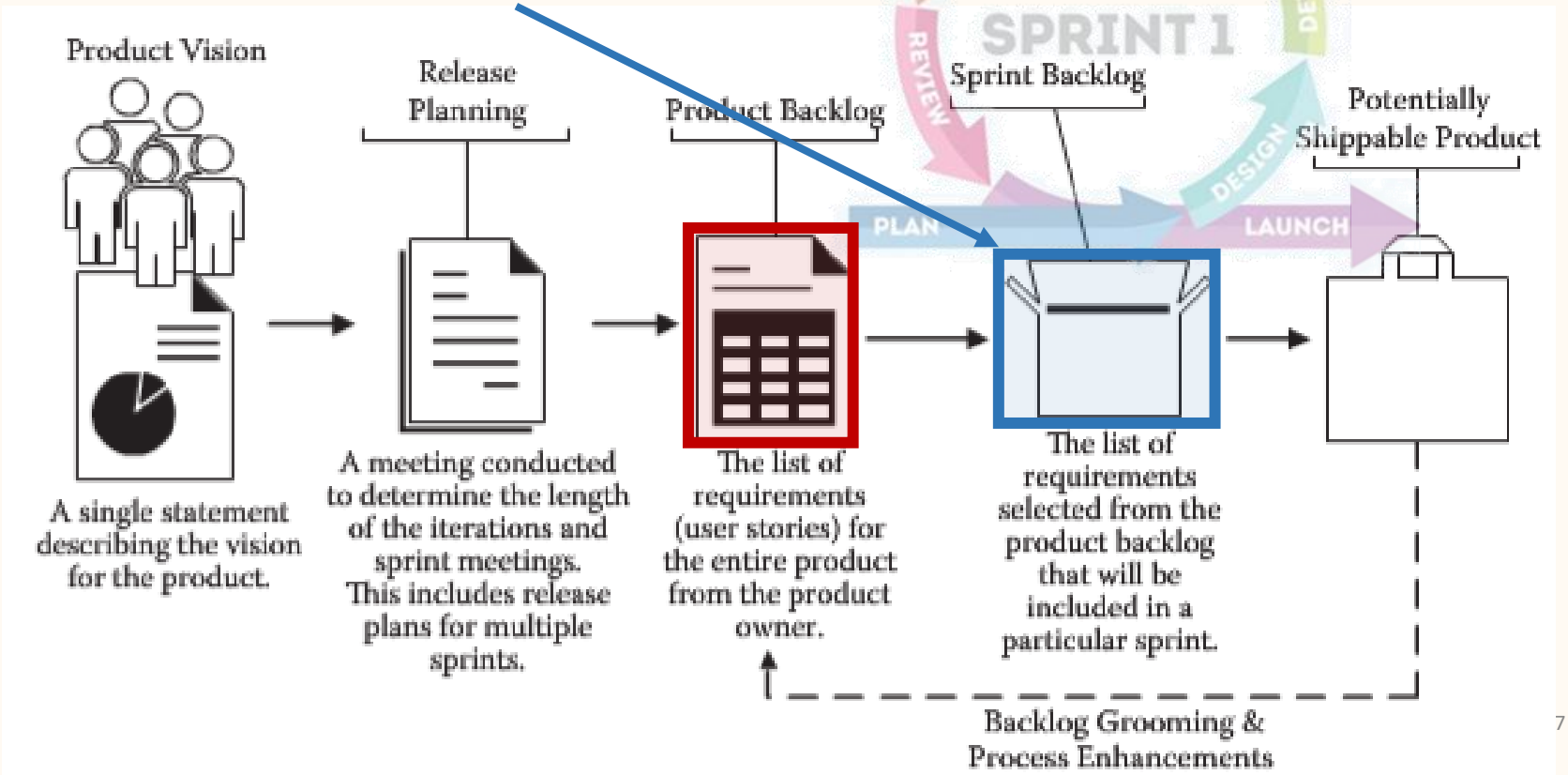
- User personas & user stories
- Product backlog
- Release planning
- Sprint planning, estimating
- Sprint 1

Reminders about sprints

- **Sprint:** a time-boxed period when team works to complete a set amount of work
- Output of each sprint after **MVP** is a working product **increment**
  - Each increment is **additive** to all prior increments
  - Each is **thoroughly verified** to ensure all functionality works together

Sprint planning

Sprint backlog populated



Sprint planning

- Sprint planning requires **user stories** from ***product*** backlog have weights
  - Weights assigned/adjusted before each sprint
- Identify the "Why?" for the sprint (goal)
- Select the "What?" (stories (weight < velocity))
- Determine "How?" for delivering increment
  - Decompose stories into **tasks** → ***sprint*** backlog
  - Assign some **tasks** to each team member to start

Assigning weights to stories

- Use **story points**
  - Relative measure of work to compare stories
  - Avoid equating to time—too unreliable
  - Calibrate by consensus, comparison to prior ones
- Lots of ways to assign story points
  - Front-burner, back-burner, fridge, freezer
  - T-shirt sizes: S, M, L, XL
  - **Fibonacci story points**

Key considerations

- When assigning story points, consider…
  - Work effort (more effort = more points)
  - Risk (riskier = more points)
  - Complexity (more complex = more points)
- Also, be sure to account for
  - Design
  - Development
  - Testing
  - Documentation…

Poker planning

- One way to reach consensus on story points
- Each team member gets own “poker” cards
  - Cards labeled with story-point score
- Product own presents a story to team
  - Each team member selects card for their estimate
  - All reveal cards at same time
  - Team discusses to reach consensus (if necessary)
  - Discussion kept to a couple of minutes

Fibonacci Story Points

- Fibonacci sequence
  - 1 1 2 3 5 8 13 21 34 55 89 …

    Small Medium Large Epic
- Exponential size categories work better (better than linear)
- Again, not tied to time, just relative size
- If answer would be ≥ 21, maybe break it up

Sprint Backlog

- Stories chosen for sprint → **sprint** backlog
- Initial discussions: How should each story be implemented/delivered?
  - Often involves breaking stories into **tasks**
    - Developer-oriented actions
    - Given time estimates (2h, 4h, 6h, …)
- Team members assigned to initial items
- Other backlog items stay for later assignment

Outline

- User personas & user stories
- Product backlog
- Release planning
- Sprint planning, estimating
- Sprint 1

Some terminology

- Iteration == sprint

Canty reading	**This course**
Iteration 0 (initial planning & designs)	Sprint 1 (initial planning & design)
Iterations 1-N	Sprints 2-N (development, testing, and deployment)
Iteration R (deployment) (done in parallel with next dev iteration)	(incorporated into sprints 3-N)

Iterations 0 & 1

- Iteration 0
  - Project is initiated ✓
  - Support and funding ✓
  - Stakeholders; team ✓
  - Agile environment ✓
- Iteration 1
  - Initial system architecture

Architectural spikes

- Iteration/sprint used to mitigate risks
  - Acquire information to mitigate risk
  - Gain understanding of a product requirement
  - Validate an estimate for a task
  - Analyze combinations of functional behavior, decompose/analyze for risk (functional spike)
  - Validate design consideration (technical spike)
- Note: don’t add customer value directly

More Iteration 0 activities

- Prepare user story details for next sprint(s)
- Use of tools and techniques:
  - Use cases
  - Activity diagrams
  - Data models
  - Sequence diagrams
  - Acceptance tests, etc....
- Team adapts as it makes progress/learns

Sprint development: key items

- Keep a journal of your development work
- Definition of “done”
  - Task: "done" when developed, tested, and verified by 1+ *other* team member
  - User story: "done" when validated by PO
- Daily Scrum (15-minute event for Dev. team)
  - Share progress toward sprint goal
  - Adapt sprint backlog as necessary (adjust plan)

Lab 2

- See lab document
- Applies some of the process to your project
  - Personas
  - User stories
  - Planning Sprint 1
    - Note that planning is restricted to design tasks for now
    - You don’t have enough yet to do more