



UNIVERSITATEA DIN
BUCUREȘTI
VIRTUTE ET SAPIENTIA



Course 3

Requirements Analysis and Design Definition

500 | Technology Fast 500
2019 EMEA WINNER
Deloitte.

★ | **IMPACT STAR Award**
by Deloitte. Technology
FAST 50 CENTRAL EUROPE 2020

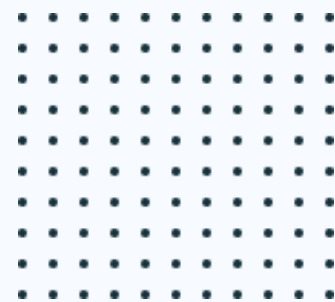
Proud Member Of
**EBRD Exclusive Blue
Ribbon Global Network Of
Best Performing SMEs**



Agenda

User Stories

Functional decomposition - a customized approach





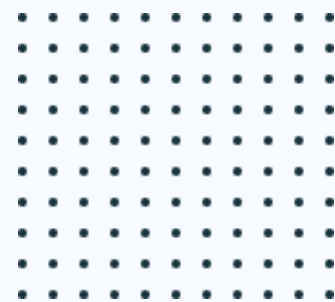
User stories

What is a user story ?

What is a story's lifecycle?

What makes a good story?

Quiz





What is a user story?

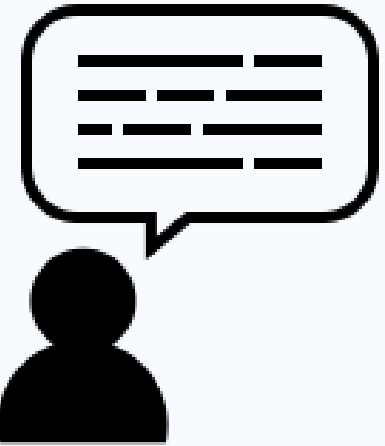
What is a user story?

Why are user stories used?

What is the anatomy of a user story?



Definition



A user story is a short, explicit expression stating **what a specific user needs** from a system in order **to achieve some goal**.

It is a way to specify requirements keeping in mind:



***who** the user is*



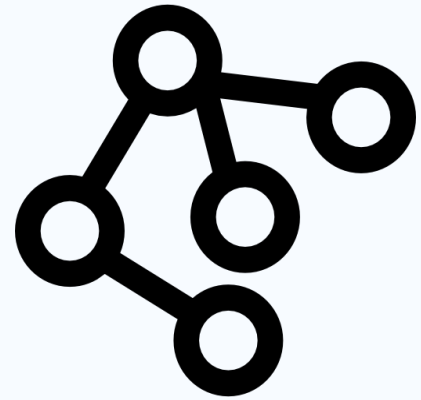
***what** the user needs*



***why** the user needs that specific system function*



The anatomy of a story



User stories are generally written using a handful of **simple patterns**.
Here is the most popular pattern.

As a *<type of user>* I want *<some feature, capability of the system>* so that *<I achieve some goal>*.

A specific type of user with a specific, identifiable job

A capability, function or feature that is required for the system to support the user

The goal the user is looking to achieve, or the value they are looking to derive using the system



A few story examples

For a company's presentation website



As a content administrator, ...

... I want to control all content that is published on the site so that we are sure of the quality & correctness of the information presented to our visitors.



As a content editor, ...

... I can add new content on the site & update existing one so that our visitors are engaged by the accuracy & freshness of the information available to them.



As website visitor, ...

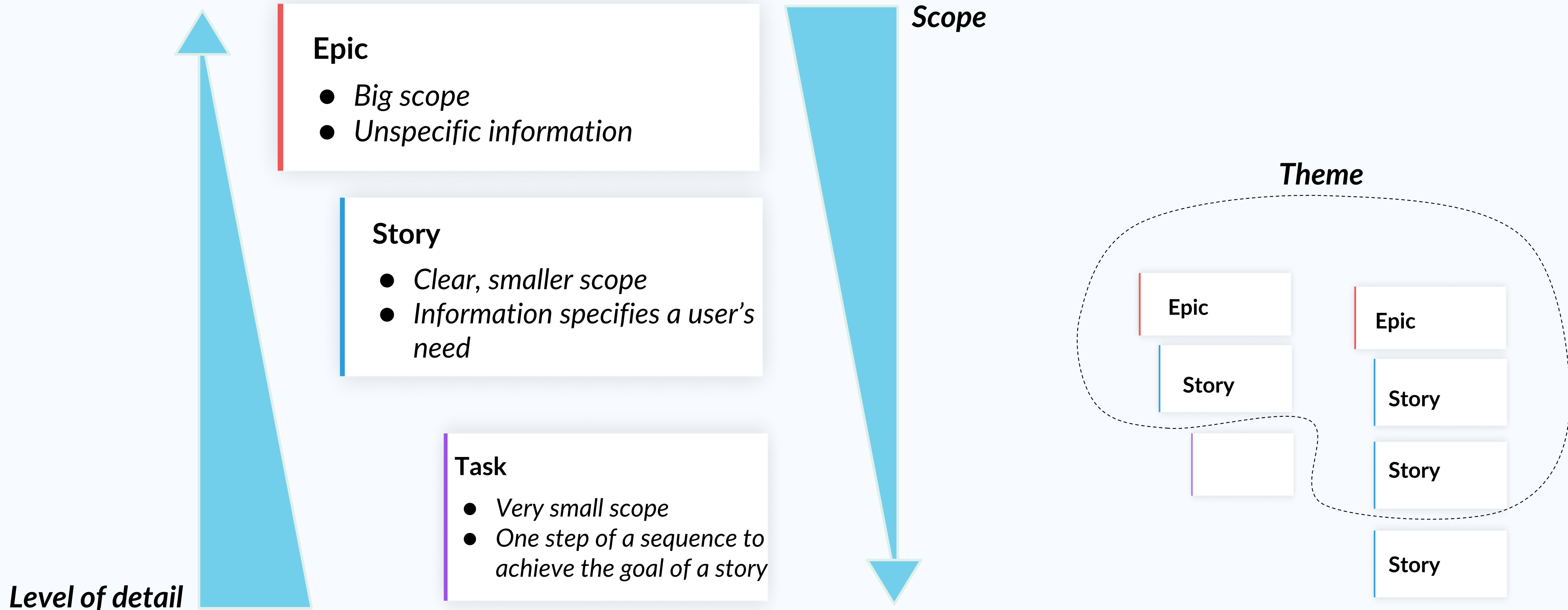
...I can quickly and easily find the specific content I am looking for, so I don't get lost in the site, get frustrated & leave soonest.



What is the lifecycle of a user story?



Themes, Epics, Stories





Epics, stories, themes - an example



As a **content administrator**, I want to **control all content** that is published on the site so that we **are sure of the quality & correctness of the information** presented to our visitors.



As a content administrator, I want to have a list of content items awaiting publication, so that I can review them.



As a content administrator, I want be notified when a new content item has been submitted for publication, so I know when I have a new review task.



As a content administrator, I want to be able to refer a content item back to the creator for revision.



As a content administrator, I publish in bulk content items that I've reviewed.



As a content administrator, I want recurring & standardized types of content to be automatically published without me having to personally review them since they present little risk.



As a content administrator, I want to be able to rate authors, so that in time, my reviewing process becomes more efficient.



From epics to stories & tasks



Product manager/Product owner/ BA
Defines epics



Product owner/Business Analyst
Drafts stories



The team, the BA & PO, system analyst
Discuss, split & detail stories - backlog refinement



The team
Implements functionality required by the stories





What makes a story a good user story?

INVEST in good user stories

Splitting user stories

Acceptance criteria



INVEST in user stories



A pidgin language is a **simplified language**, usually used for trade, that allows people who can't communicate in their native language to nonetheless work together. **User stories act like this.**

Bill Wake -INVEST in Good Stories, and SMART Tasks

I	independent	A good story is independent
N	negotiable	A good story leaves room for negotiation
V	valuable	The story clearly states what is the value to the user
E	estimable	It contains just enough detail to estimate correctly
S	small	Is small enough that it can be done in one iteration
T	testable	The resulting functionality is testable; it can be explicitly verified



Vertical vs. horizontal splitting

Vertical slice is a shorthand for “a work item that delivers a valuable change in system behavior such that you’ll probably have to touch multiple architectural layers to implement the change.” When you call the slice “done,” the system is observably more valuable to a user.

This is in contrast with a **horizontal slice**, which refers to a work item representing changes to one component or architectural layer that will need to be combined with changes to other components or layers to have observable value to users.

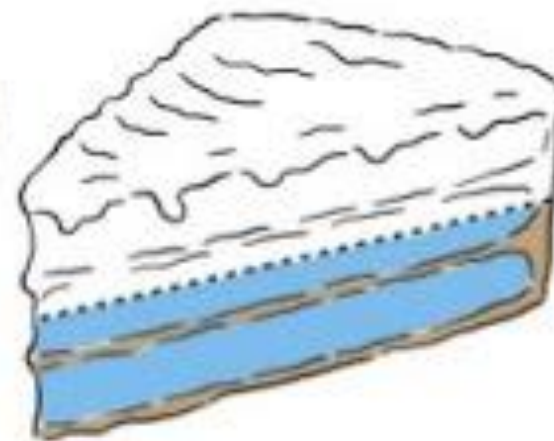
“Slicing” User Stories that are too big

Story Slicing

Product owner



Too big for a sprint



Don't slice
'horizontally'



Vertical slicing enables
PO to validate and
feedback



Splitting steps



1. Understand **why the input story is not satisfactory** in order to prepare for splitting

2. Choose and **apply a Splitting Pattern** depending on workflows/operations/ infrastructure/business rules/complexity and others

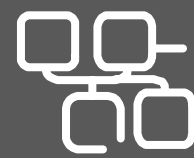
3. **Evaluate the split** to make sure all the new stories are satisfactory



How can you split?



Once you've decided your story is too big and should be split, there are a few patterns that can help you do a proper vertical slice.



WORKFLOW

Your story describes a complex workflow

1. Do a **simple version** as a **1st story** ; enhance in next stories.
2. Define the first story to cover **the beginning & end of the flow**; enhance in next stories.



OPERATIONS

Your story hides operations in wording like 'manage...', 'configure...' etc.

- Create **1 story for every action** included in 'manage':
- 'Create...',
 - 'Read...',
 - 'Update...'
 - 'Delete ...'.



INTERFACE

Your describes a complex user interface

Define the **simplest possible version of the UI** & create the 1st story; add stories to enhance the simple UI.



How can you split?

And three more patterns ...





RULES

Your story achieves one goal using different business rules

1. Group business rules; create **1 story per rule group**
2. Create **1 story per business rule**




DATA

Your story does the same kind of thing for different kinds of data

Split the story across **data boundaries** & prioritize.

Start with one data set & enhance in next stories.



PERFORMANCE

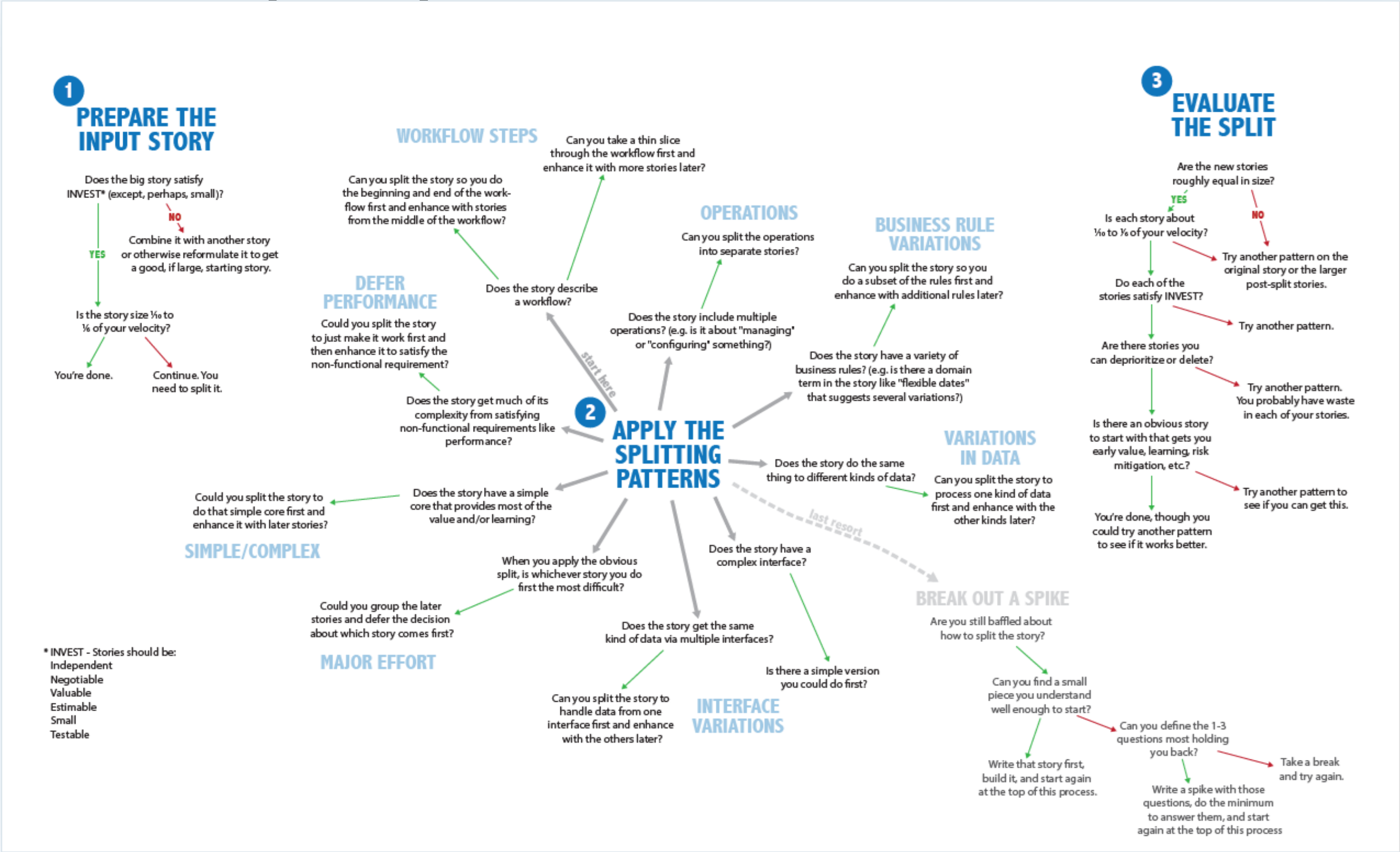
Your story requires great performance

Split into: **'make it work' & 'make it fast'** stories.



How can you split?

Richard Lawrence & Peter Green



* INVEST - Stories should be:
Independent
Negotiable
Valuable
Estimable
Small
Testable



What are acceptance criteria?



Acceptance criteria are the **conditions** that a software product or a **functionality of the product must meet in order to be accepted by the user**, or in order to bring the desired value to that user.

Acceptance criteria must:



*have clear **pass** or **fail** results*



*convey the intent but not the final solution; '**what**' not 'how'*



*be **independent** of the implementation*



*have the right granularity: provide **just enough detail** to be useful*



*describe the system's reaction to both proper (**positive**) & improper use (**negative**)*



Positive and negative acceptance criteria



Acceptance criteria must describe what the interactions between the user & system should be both when everything goes according to plan, and when something goes wrong



Positive scenarios for acceptance criteria

Positive scenarios describe positive, expected actions the user takes to achieve their goal. They are the correct actions to take as a user so that you would obtain the desired results from your interaction with the system.



Negative scenarios for acceptance criteria

Negative scenarios describe how the system should react, how it should help the user take corrective action when they use the system in unexpected or incorrect ways. For example, when the data the user provides is incomplete or incorrect.



Acceptance criteria vs. Test cases



Acceptance criteria must NOT be formulated as test cases. Be careful to describe how the system and user interact so the user is able to achieve their goal.

Acceptance criteria = **what** needs to be done to achieve the user's goal

Test cases = **how** it should be done to meet the user's expectations



Test cases are **developed later**, during the development stage.



They are **scenarios** which are **derived from acceptance criteria**.



Each acceptance criteria can have one or more acceptance tests.



Gherkin language for acceptance criteria



Gherkin is a domain specific language that provides a recognizable pattern for writing structured acceptance criteria.

Gherkin acceptance criteria are structured in 5 main elements:



*A **SCENARIO** that provides a label for the behaviour you are about to describe*



*A **GIVEN** statement that describes the initial state - the starting point for your scenario*



*A **WHEN** statement that introduces a specific action the user takes*



*A **THEN** statement that describes the result obtained by performing the action (or the final state)*



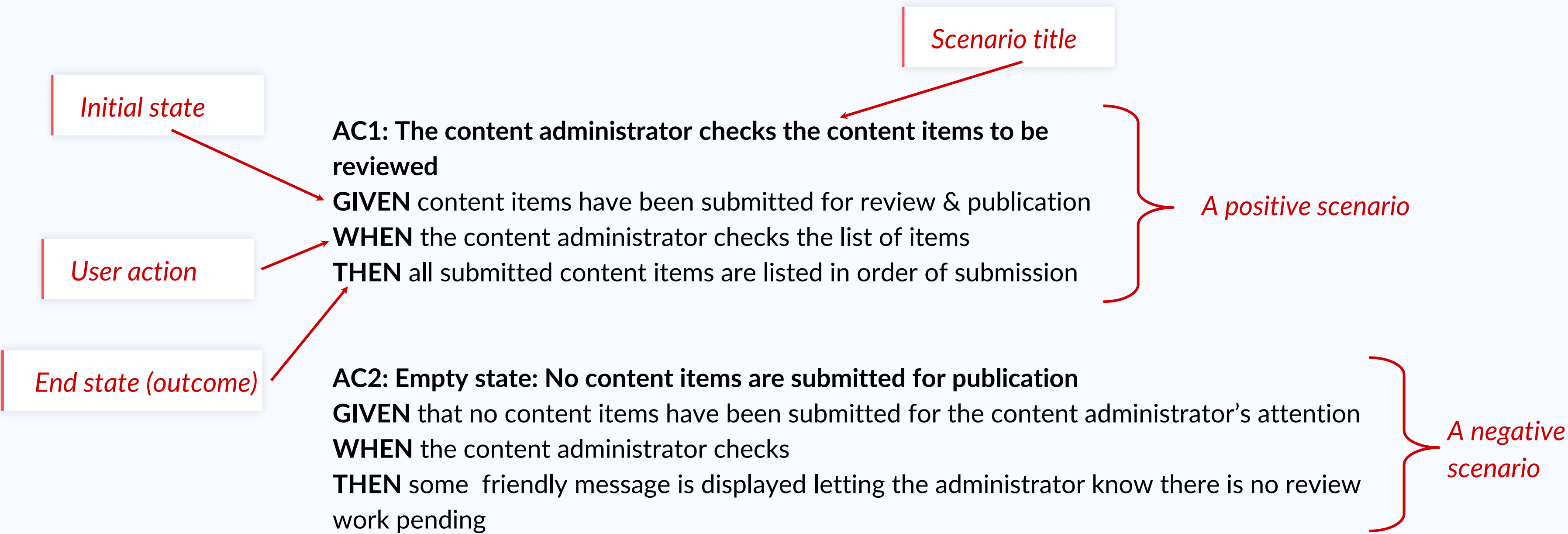
*An **AND** (or **BUT**) statement that allows you to add more conditions to describe the initial or final states*



An example of a story with acceptance criteria



As a content administrator, I want to have a list of content items awaiting publication, so that I know what I need to review and in what order.





Functional decomposition

Definition

Alternative techniques

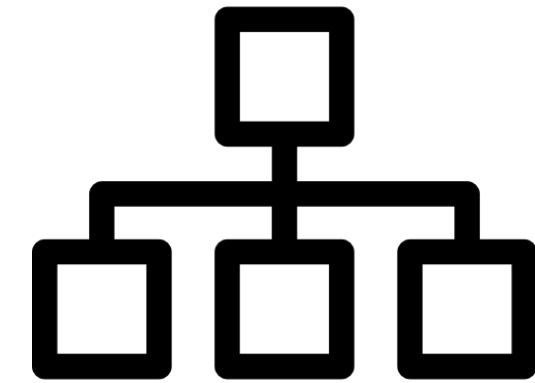
A customized approach for functional decomposition



Definition and scope of functional decomposition

Functional decomposition "helps manage complexity and reduce uncertainty by breaking down processes, systems, **functional** areas, or deliverables into their simpler constituent parts and allowing each part to be **analyzed** independently."

"A Guide to the Business Analysis Body of Knowledge (BABOK® Guide) v3."



Conceptual model of the **full** work to deliver a new business solution

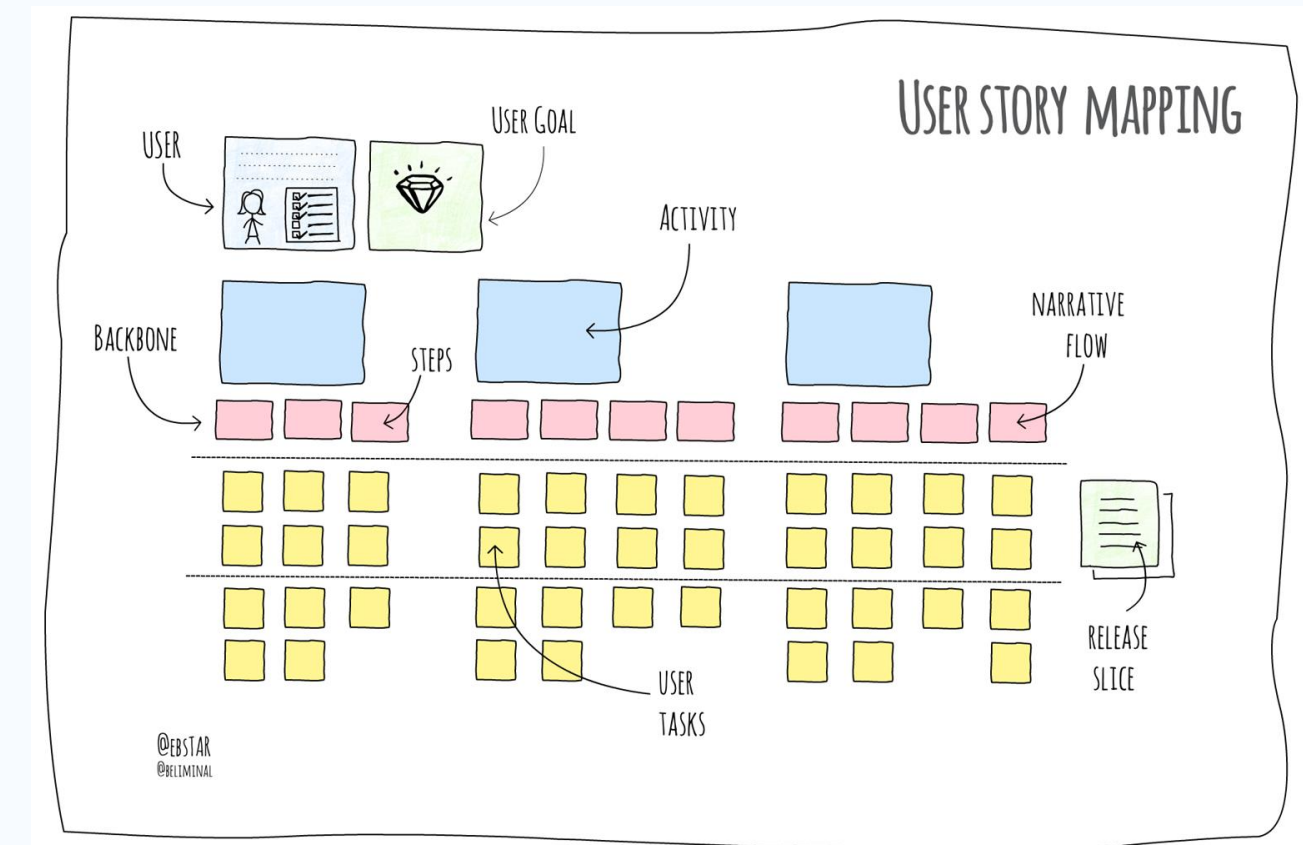
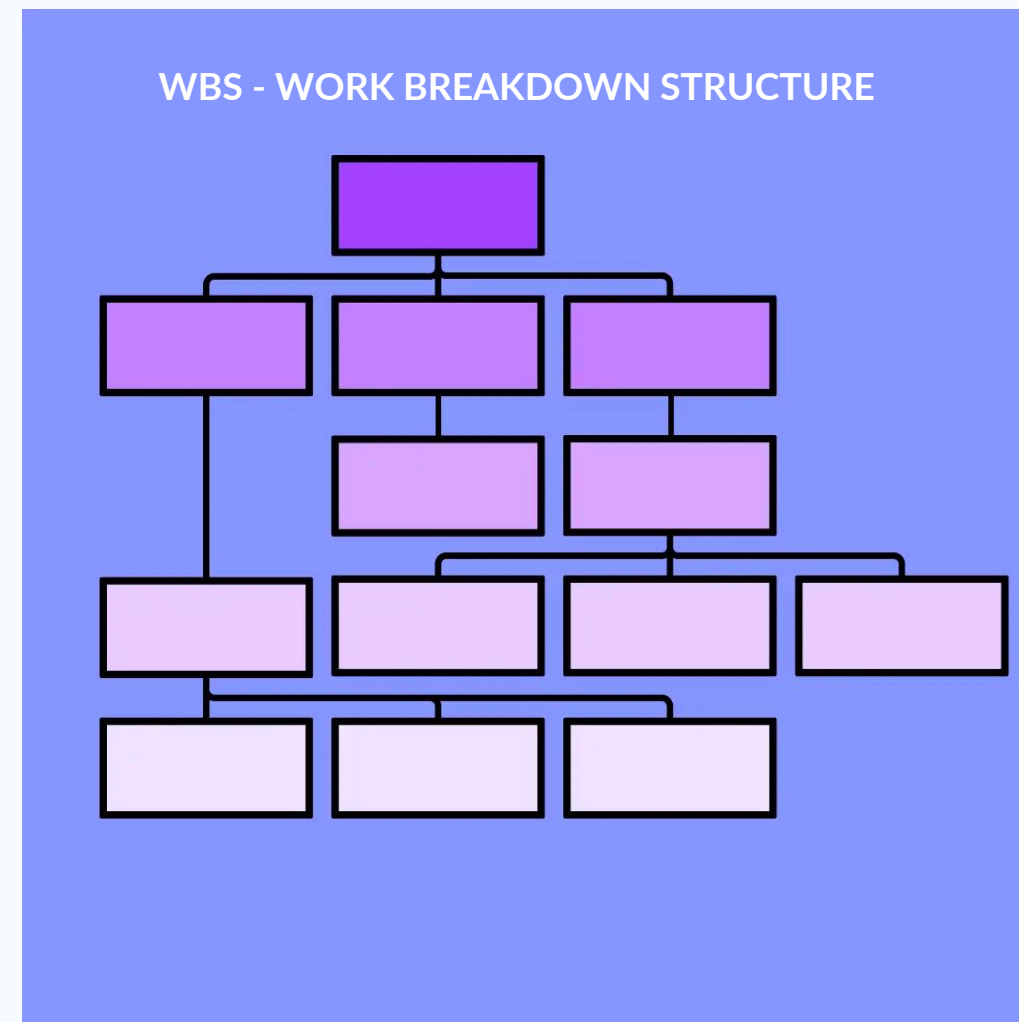
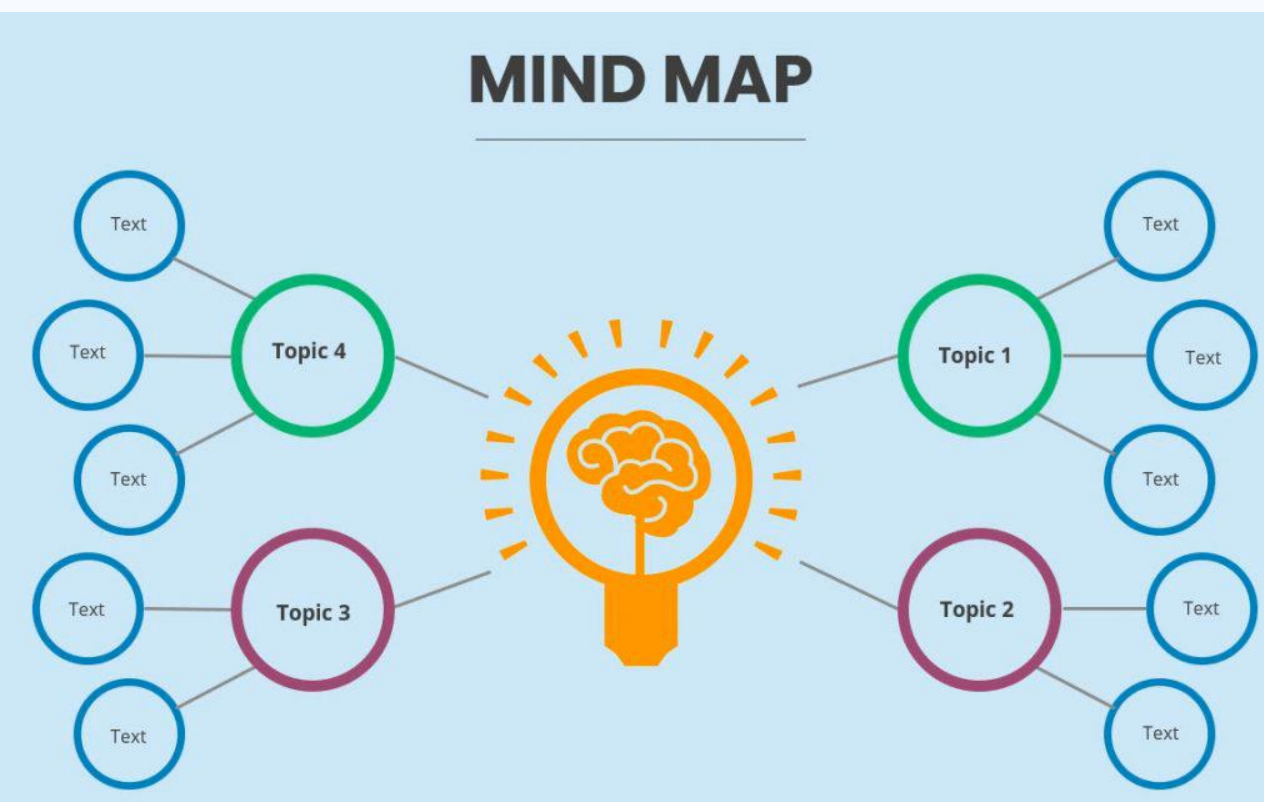
Provides a consistent view of the **functional scope**, but not necessarily the entire scope of work

Assists **estimating** in that estimates can be made for smaller chunks of work



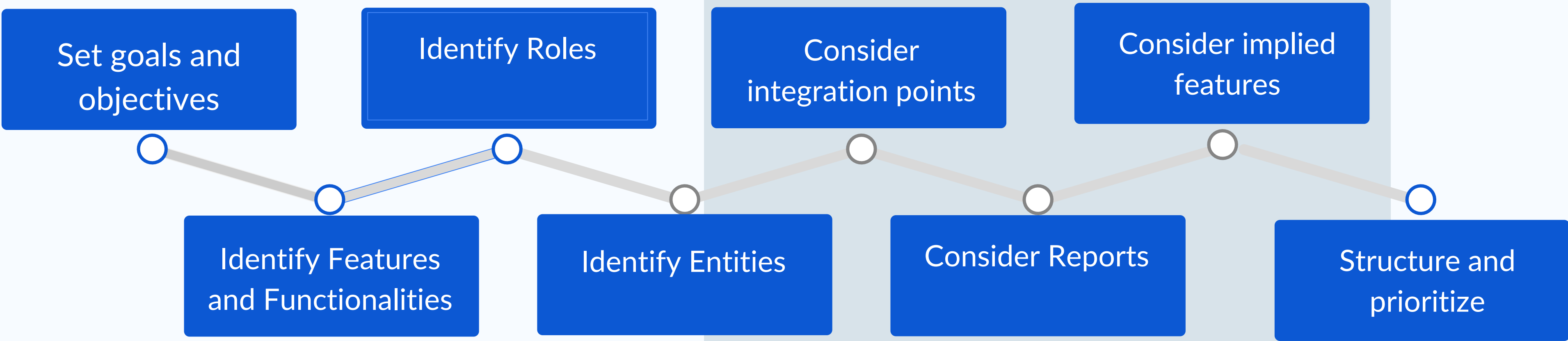
Techniques for breakdown the complexity

- Functional decomposition diagrams
- User story mapping
- Work Breakdown Structure (WBS)
- Process mapping
- ...and many more





A customized technique for functional decomposition





Set the goals and objectives

The first thing to do is set the goals and objectives:



identify and set up what are the **expected goals and the objectives.**



align the audience expectations



focus the effort for functional breakdown and building the roadmap towards these goals



Usually, these are stated as opportunities or problems for the company:

Problem: it's hard for Romanian workers to find jobs abroad

Goal: publish job openings across Europe in a manner that makes them easy to access & comprehend



Set the goals and objectives

Sample questions to get you started in setting the goals or defining the product vision

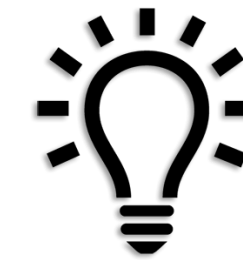
- What do you expect to achieve with this product/project/app?/ What problem does the product aim to solve for the users?
- Who is the intended audience or user base? (private persons, business users, categories etc)
- How will the product benefit the users or the business?
- How do you measure the success?
- What are the key features or functionalities we envision?
- Do you have some specific requirements for this app, specific capabilities or characteristics other apps don't have?
- What do you want your customers to remember the app for?
- What is the timeline or roadmap for the product?/ What is the desired launch date of the first version?
- What devices do you want to support for your users?



Identify the features and functionalities

Features are the “tools” used within a system, by the users to enable them to complete a set of tasks or actions, that provides value to them

Functionalities are how those features actually work to provide you with a desired outcome, what can do in terms of operations and interactions



Example

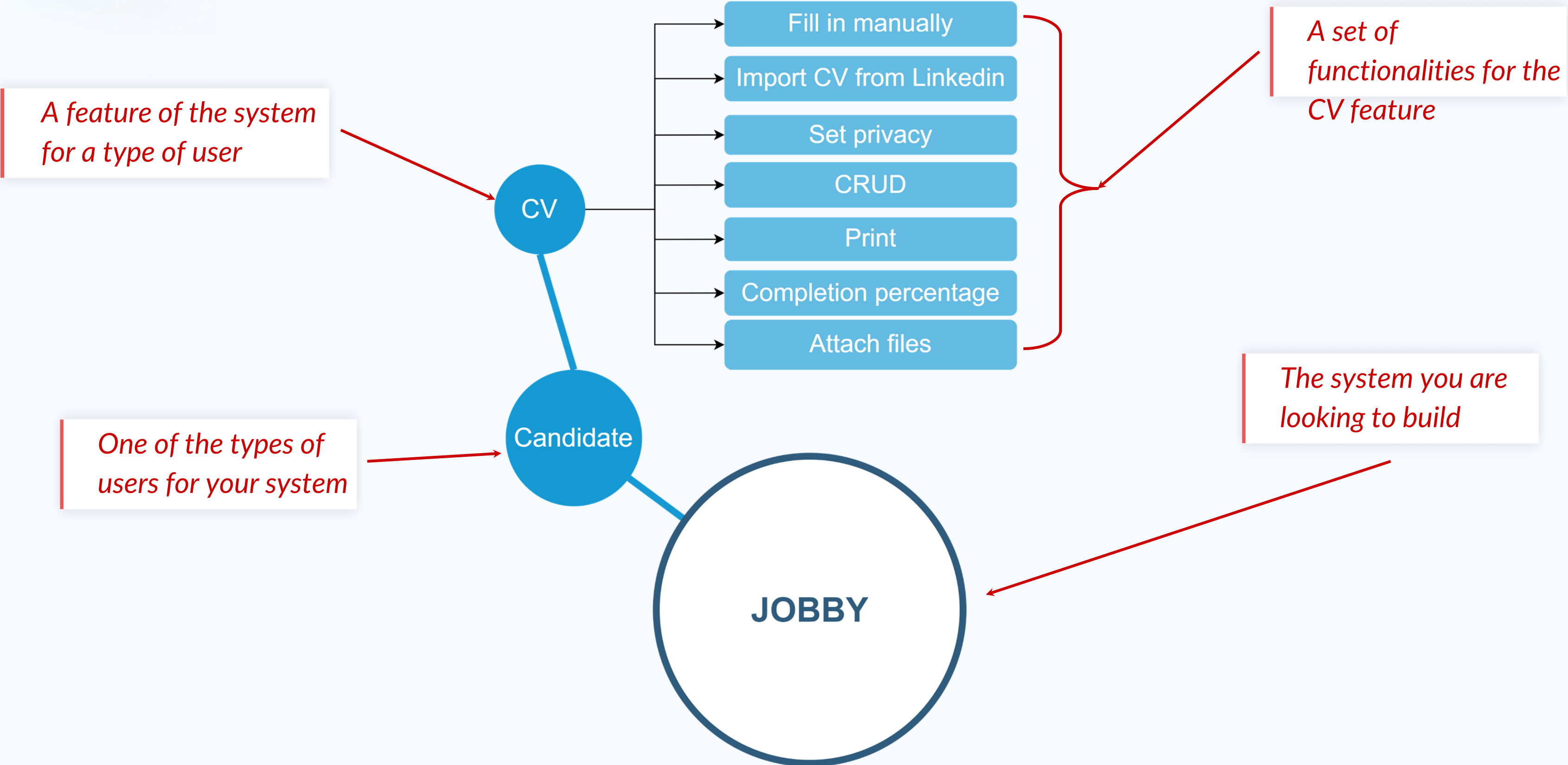
Feature: Product Search

Functionalities:

- Keyword Search
- Filters and Sorting
- Autocomplete and Suggestions
- Search Results Display
- Product Thumbnails
- Pagination
- Faceted Search
- Search History
- Voice Search



Feature & functionalities example





Discover the users and their roles

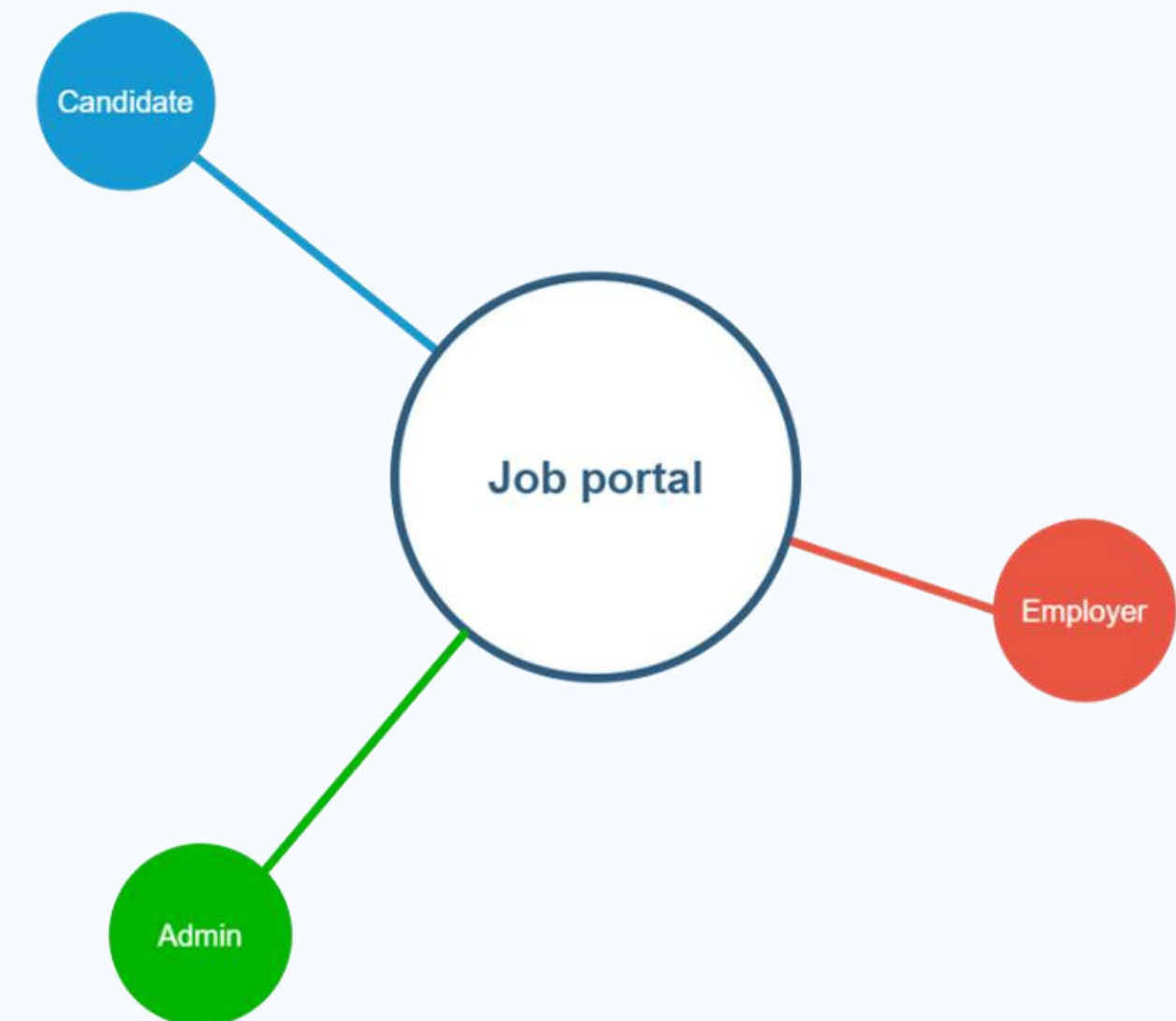
The **roles, permissions and end users** that will interact with the system determine to a large extent the level of complexity that will govern the system you are building.



It's important to know from the beginning an estimation for:

- *Number of users*
- *Geographical position*
- *Working hours*

- *Different roles*
- *Different personas*
- *Permissions in the system*





Determine the entities

Any singular, identifiable and separate object in your system's domain is considered an entity of that system. While working on functional decomposition, consider the types of people (users), places, things and concepts, attributes and relationships, important for business.



The CRUD pattern can help you identify your entities:



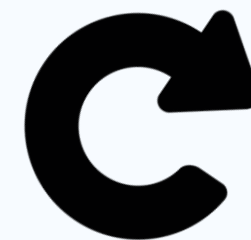
CREATE

C



READ

R



UPDATE

U



DELETE

D



Determine the entities

CREATE

From external systems
Manually

READ

*Each user may have
different views of
data*

UPDATE

*Flow updates, entity
lifecycle*
Automatic updates

DELETE

Logical delete
Physical delete

	Create	Read	Update	Delete
Department	Manually	List view Detail view Organizational chart	Manually	Physical
Employee	Who and How?	Who and How?	Who and How?	Who and How?
Project	Who and How?	Who and How?	Who and How?	Who and How?



Identify the integration points

Integration points are points at which information or documents are exchanged between your system and other systems.

Here are some clues to identify integration points:



Internal systems that the client is currently using and might be communicating with the new one



External systems



Different types of data sources that might be integrated using e.g. an API, excel/csv file; an initial database that should be incorporated into the system, feeds, etc.



Discuss required reports

Represent aggregation of data in different shapes.

- Does the client need reports?
- If yes, how many and what's their complexity?
- Do you have all the data of the reports captured in data model?

Consider the following when determining reports complexity:

- Do they imply data aggregation or are just a simple export?
- Number of records present in database
- What is the representation form of the reports?





The outputs of functional decomposition

The main outputs of this process might be:



Functionalities list (mind maps, diagrams, tables)



Assumptions and limitations

- To align expectations
- For more accurate estimates

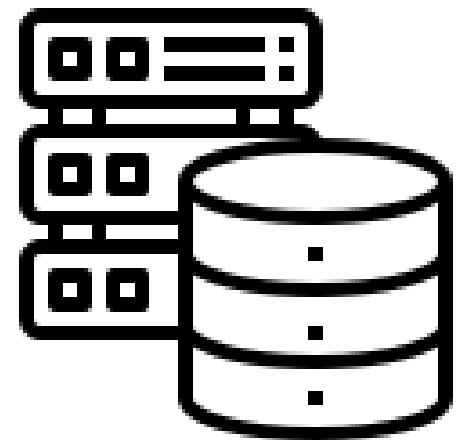
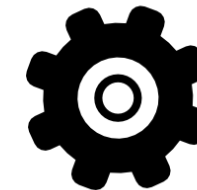
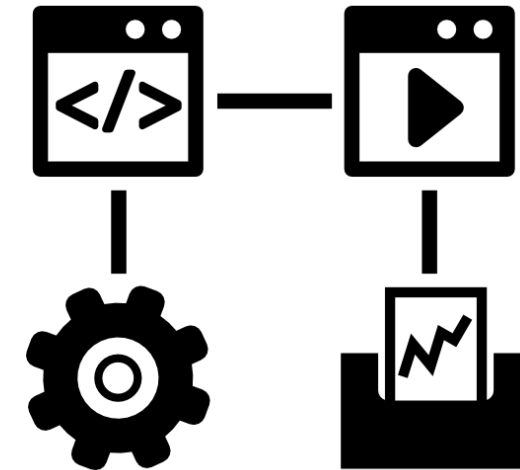
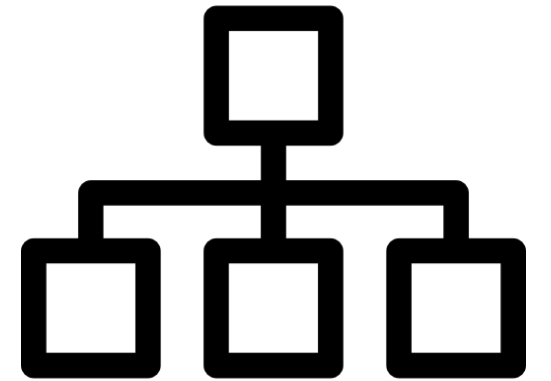




Table view sample

Company	Manage account	In order to post jobs and search for candidates, a company needs to create an account.	<u>Create account</u> An account can be created by using an email address.	Phase 1
			<u>Fill in company details</u> The employer can add some details about his company so that the candidates can see it as part of the posted jobs. Details to be filled in: <ul style="list-style-type: none">- Description- Locations- Industries- No. of employees	Phase 1
			<u>Add multiple users</u>	Phase 2
			<u>View/Update/Delete profile</u>	Phase 1
			<u>Link account to a Facebook page</u>	non-MVP
			<u>Link account to a Linkedin account</u>	non-MVP
	Job	One of the main entities in the portal is the Vacancy . An employer adds a job by filling in the job information, so that he can receive from the platform suggestions for candidates	<u>Add</u> An employer can add a job by filling in the job information and posting it on the portal. Also, the information can be saved during the process and posted later. <u>Upload file</u>	Phase 2 Phase 2

