



MOBILE APP DESIGN
AND DEVELOPMENT
498-02

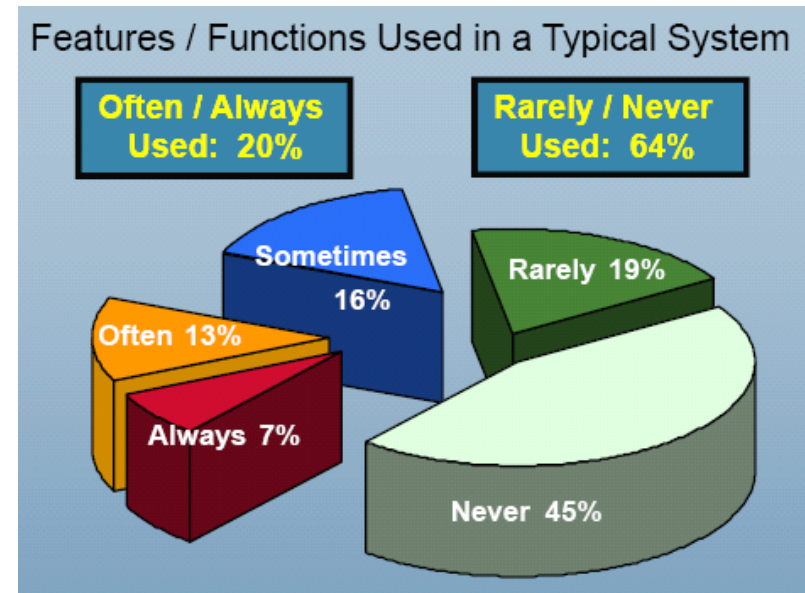
Story Map Scoping

Which features?

64% of features in a typical system is wasted effort

Want to build a product with the smallest set of features that captures our **value proposition**

Also, to help us **scope** your project!



How to go about finding this **smallest set of features?**

[Johnson, Standish Group Study; Cayenne Apps, *5 Steps to Building Minimum Viable Product with Story Mapping*]

Story Mapping

Map out **features** as a **2D map**, backing up each feature with its **acceptance criterion**

Example: Starting an Online Bookseller in 1992

Step 1: define your **value proposition**

Example: **Buy book online, 24/7**

Step 2: define the **User eXperience (UX)** or flow of control for your app (we'll assume only one flow)

What are the steps users take to achieve your value proposition?

Step 2: User eXperience

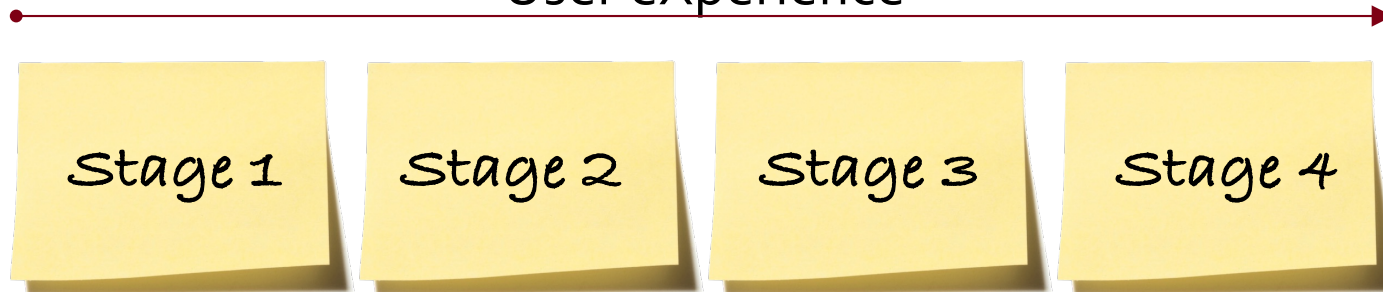
What are the **stages** most users would follow in using your product?

Focus on the **usage flow** here, not on the features



Value proposition: Buy book online, 24/7

User eXperience



Step 2: User eXperience

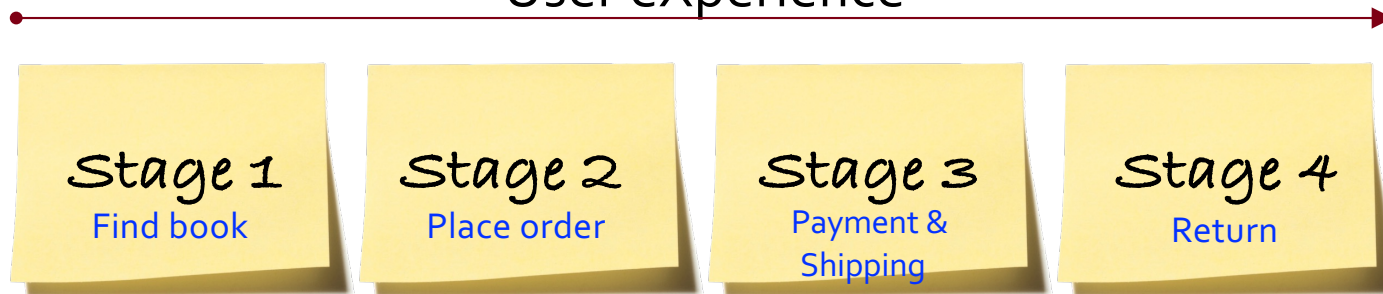
What are the **stages** most users would follow in using your product?

Focus on the **usage flow** here, not on the features



Value proposition: Buy book online, 24/7

User eXperience



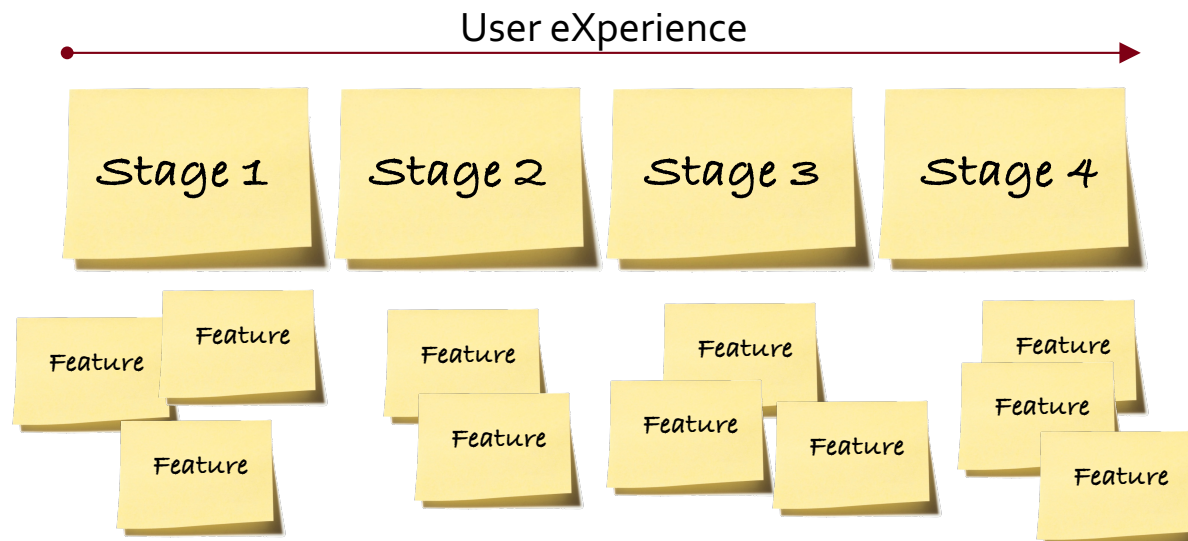
Step 3: List of Features per Stage

List all features that should be part of a stage

Brainstorm, be creative, list features unordered



Value proposition: _____



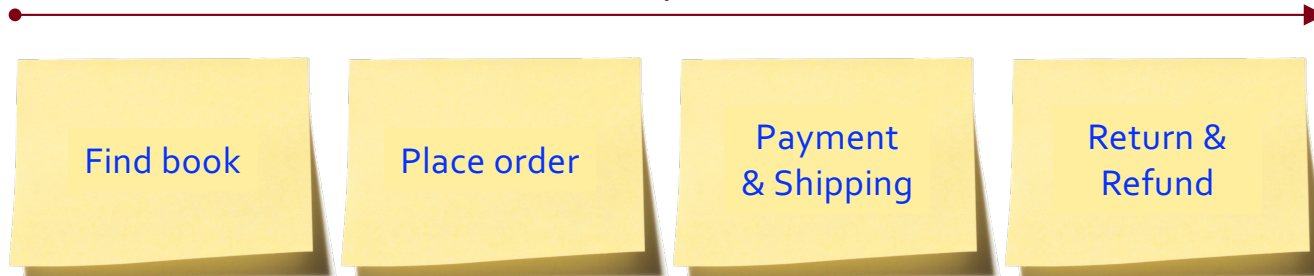
Step 3: List of Features per Stage



Value proposition:

Buy book online, 24/7

User eXperience



Step 3: List of Features per Stage



Value proposition: Buy book online, 24/7

User eXperience



[Cayenne Apps, 5 Steps to Building Minimum Viable Product with Story Mapping]

Acceptance Criteria

When is a **feature** considered done?

What conditions must a **feature** satisfy to be acceptable to its user
("user" could be a function)

Acceptance criterion is **per feature**,

NOT per stage, **NOT** per app 🙅 COMMON ERROR

This is part of your **grading rubric**: how should the grader determine
when a **feature** is done and to be awarded full credit, 85% credit,
70%, 50%, 0%?

Given-When-Then

A way to write acceptance criteria

Written in the Gherkin syntax:

- GIVEN some **context** or **state** in the UX flow
 - AND some additional context (optional)
- WHEN an **event**, such as a **user action** or function call, **is triggered**
 - **can only be ONE, no AND!**
- THEN the **feature responds** with **certain action** with some **expected outcome**
 - AND some additional outcome (optional)

} name
Storymap
feature

Step 4: Prioritize Features

Rank order the features in each stage according to user input/interviews

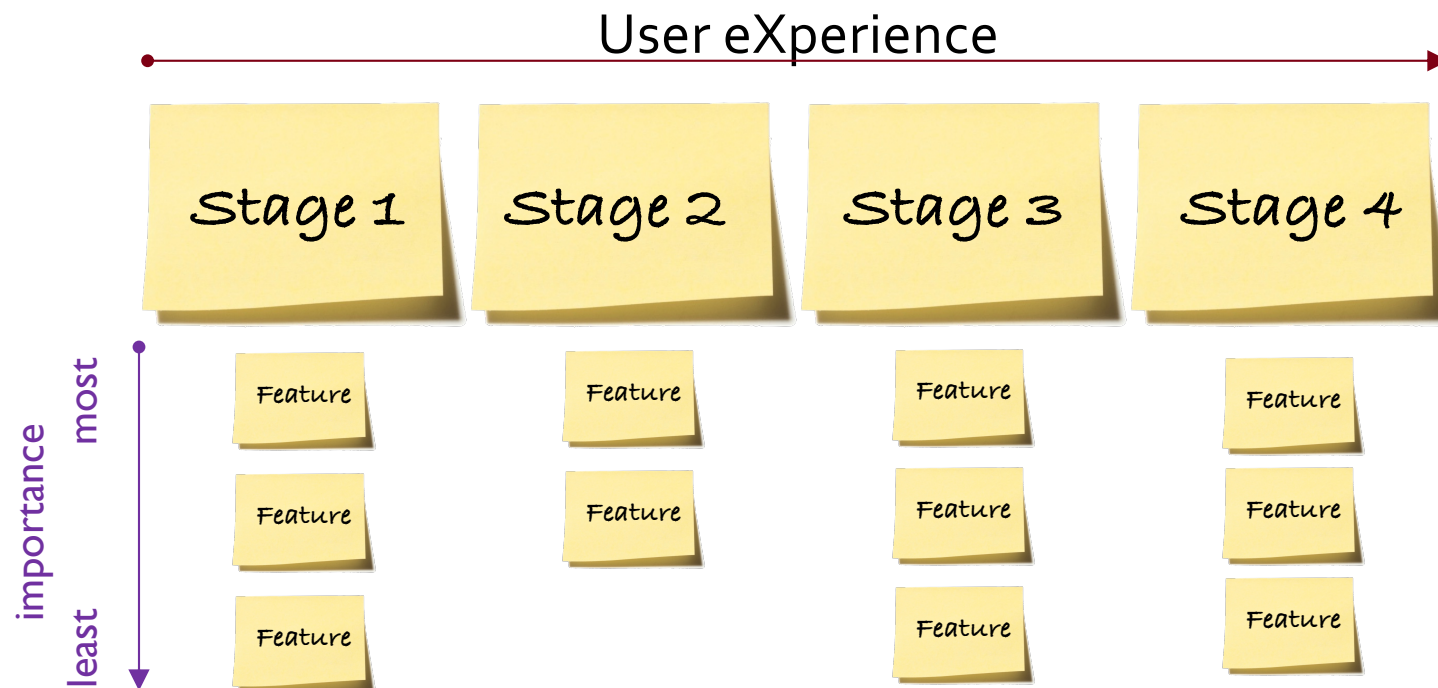
Consider:

- How **important** is the feature to complete the process?
- How **often** will the feature be used?
- How **many users** will use the feature?
- How **much gain/pain relieve** will the feature bring to user?
- How **“risky”** is the feature?
 - how soon to ready
 - how long to done
 - latent benefits

Step 4: Prioritize Features



Value proposition: _____

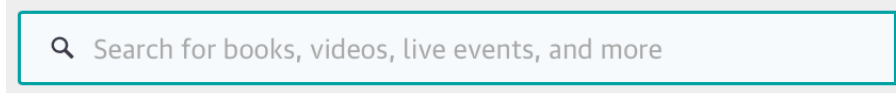


[Cayenne Apps, 5 Steps to Building Minimum Viable Product with Story Mapping]

Features vs. UI vs. Implementation

Feature: search books

UI:

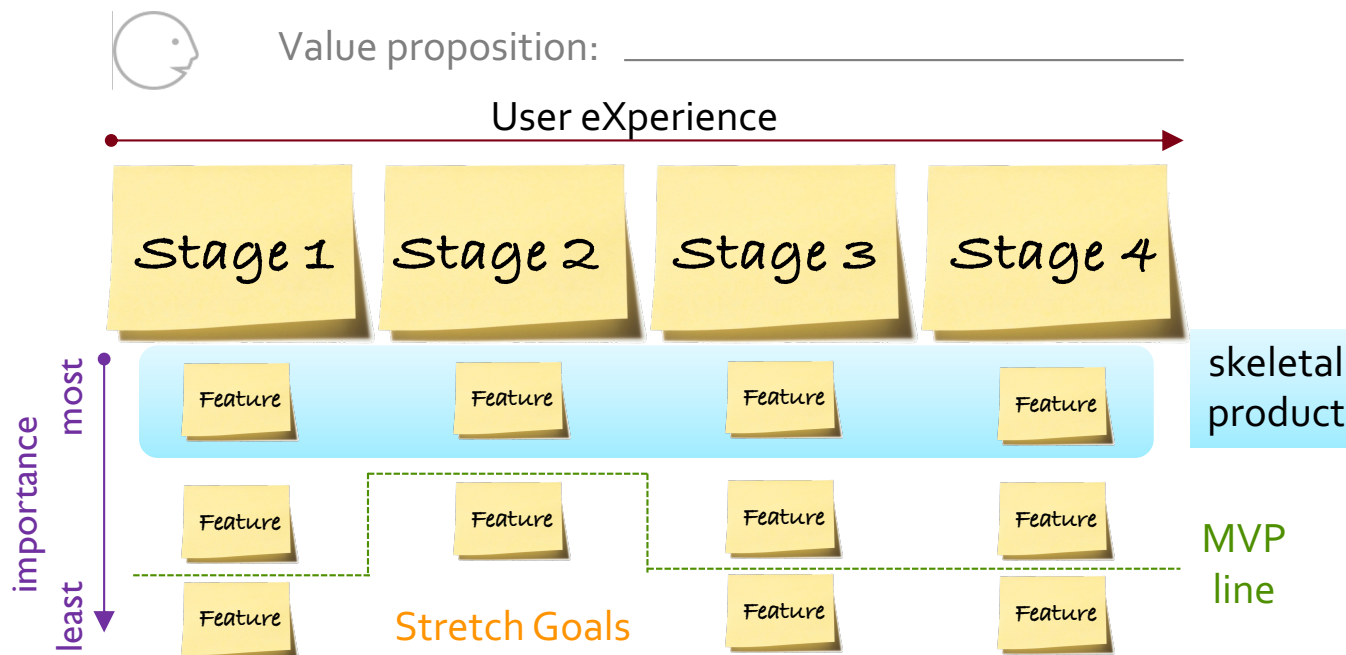
A rectangular search input field with a light blue border and a light blue background. Inside the field, on the left, is a small magnifying glass icon. To the right of the icon is the placeholder text "Search for books, videos, live events, and more" in a light gray font.

UI/UX design, not
part of Storymap

Implementation:

- define database schema
- create index
- search by author, title, date, etc.
- sort search results
- search external database(s)
(alternate flow)

Story Map Step 5: Define MVP



Skeletal product: barely functioning, built first

MVP: must have vs. **Stretch Goals:** nice-to-have features

Using EGM to Construct Story Map

Project Scoping Alternatives DUE Thu, 10/01, 2 pm

Story Map DUE Thu, 10/08, 6 pm

Peanuts and Hay

Elephant (Data Parser):

- Parses raw transcripts and slide deck
- Outputs pain points ordered by frequency/sentiment (Exhibit E-A)
- Outputs desired features ordered by preference (Exhibit E-B)
- Only those related to value proposition

Goldfish 1 (PM Critic):

- Acts as a Senior Product Manager
- Critiques extraction accuracy and relation to value proposition (Exhibits G-A, G-B)
- Triggers a loop back to the Elephant until critiques are minor

Feature List [Exhibits C&D]

Elephant (Ideation):

- Proposes mobile app concepts addressing all points in E-A and E-B
- Outputs initial feature list (Exhibit E-C)

Goldfish 2 (Context Check):

- Reads E-A,B,C to ensure E-C makes sense without external context
- Forces Elephant to add missing context if it fails the blind test

Goldfish 3 (PM Critic):

- Critiques the app concept (Exhibit G-C)
- Triggers a loop back to the Elephant to finalize the feature list (Exhibit E-D)

Algorithmic Scoping

Elephant (Extraction):

- Groups E-A into isolated workflows delivering value proposition (Story Arcs)
- Identifies Rigorous Features/Pivotal Nodes (no CRUD allowed)
- Outputs a strict Python dictionary

Python (Execution):

- Applies constraint math: $\text{Score} = W / (1 + |R - T|)$
- Outputs Top 3 Base Scopes
- Calculates Stretch Goals 1 and 2 using Bipartite Projection

Architectural Verification

Goldfish 4 (System Architecture):

- Audits the Top 3 Base Scopes (Exhibit E-E)
- Verifies Pivotal Nodes are truly rigorous and hunts for missing dependencies (Exhibit G-D)
- Loops to Elephant/Python until the architecture is structurally sound

Goldfish 5 (Engineering Manager):

- Audits the Stretch Goals (Exhibits E-F, E-G)
- Critiques architectural leverage and cascading state risks (Exhibit G-E)
- Loops to Elephant to finalize the expansion pathway

Scoping 1-1

Team Pitch:

- Team selects a Base Scope from the validated E-E menu and defends it in person

Project Scoping Alternatives DUE Thu, 10/01, 2 pm

1-1: Fri, 10/02

- Evaluate scope, may choose one of the Stretch scopes instead