Product Management & MVP Development

Building what matters, faster

Startup Management, Aleš Špetič, 2025

🔁 Recap from Lecture 5

Founding Team & Culture

- 👥 Defined co-founder roles and equity structures
- 💑 Explored hiring strategies and early team design
- 🌱 Built startup culture and values
- 💬 Discussed founder agreements and conflict resolution

🎯 Learning Objectives

What you'll learn today

- 🧠 Understand the MVP mindset and lean experimentation
- ⚙️ Learn key principles of agile product development
- 🔁 Explore feedback loops for continuous learning
- 🎯 Apply prioritization tools to make product decisions

What is an MVP?

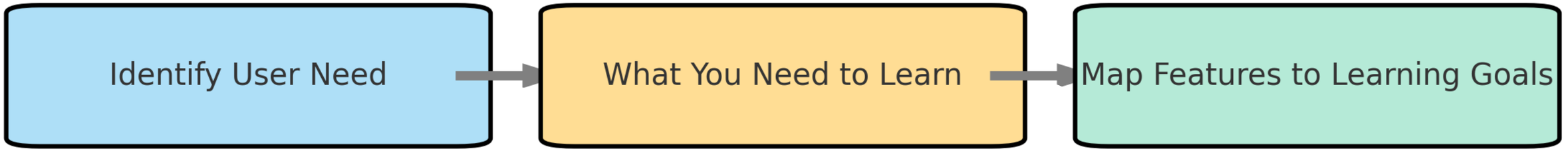
Minimum Viable Product explained

- A product with just enough features to test a hypothesis
- Built to learn quickly, not to scale
- Focus on solving the core user problem

Designing Your MVP

A practical approach to scoping

1. Identify your core user needs
2. List what you need to learn (assumptions to test)
3. Map features that support this learning



MVP Example – Dropbox

Validating the value before building

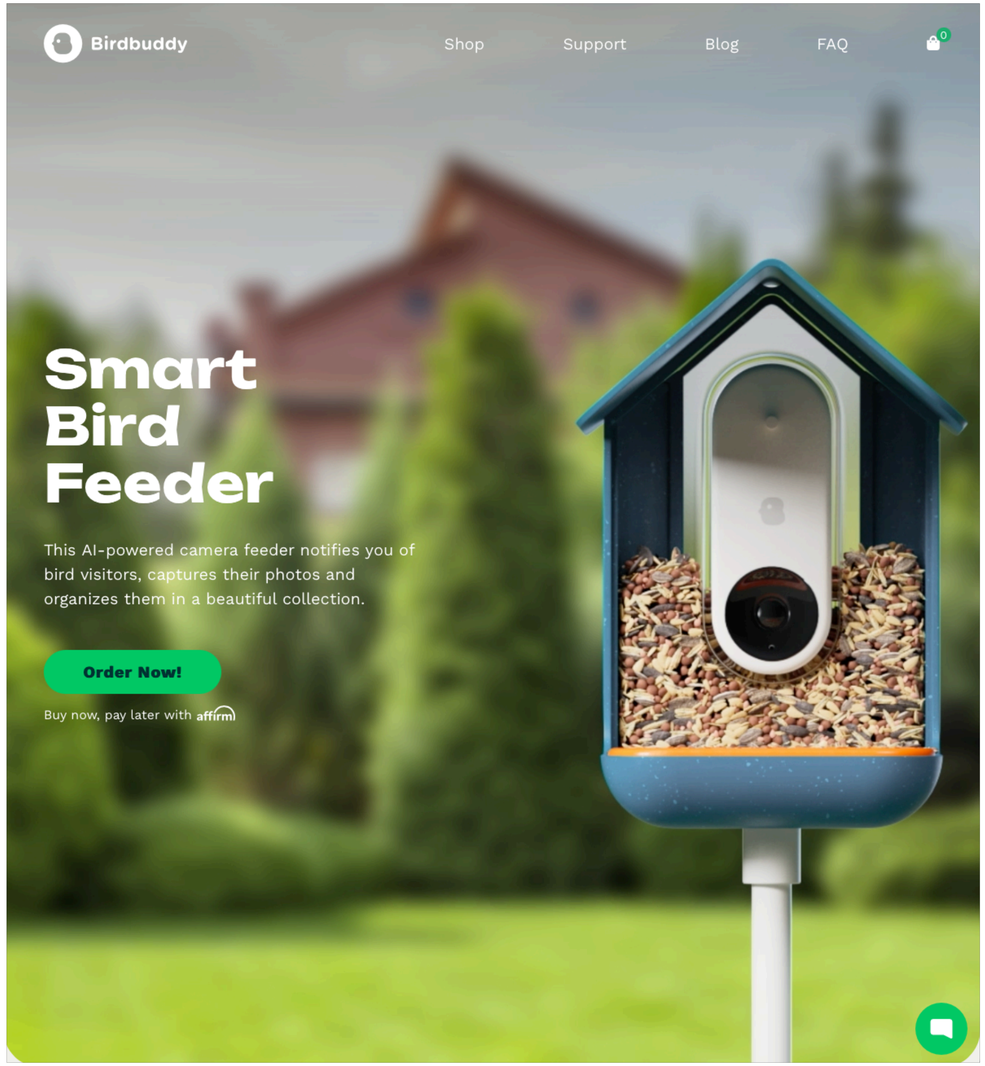
MVP Example – Zappos

Proving demand with minimal tech

- Listed shoes online without holding inventory
- Went to local stores to fulfill orders
- Validated user demand for online shoe shopping

MVP Example - BirdBuddy

Pre-product signups



MVP Myths & Mistakes

What it’s not

- MVP isn’t a beta or prototype
- Small scope doesn’t mean poor quality
- Skipping MVP wastes time and resources

MVP Testing Checklist

Before you launch, ask yourself…

- ✅ Does it solve one real user problem?
- ✅ Are your key assumptions clear?
- ✅ Can users complete the main task?
- ✅ Is the value proposition obvious?
- ✅ Are you tracking key behaviors?
- ✅ Do you have a feedback plan?

What is Agile Development?

Iterative, user-centered product building

- Agile = build small, test fast, learn often
- Emphasizes flexibility, collaboration, and speed
- Based on user feedback and continuous improvement

Agile vs. Waterfall

Two different approaches to product development

Agile:

- Iterative and incremental
- Emphasizes flexibility and user feedback
- Delivers working product frequently
- Adapts to change throughout the process

Waterfall:

- Linear and sequential
- Requires full specifications upfront
- Testing happens after development
- Difficult to change once started

Stand-Up Meetings

🕘 Keep teams aligned and accountable

- Short, daily meeting (~15 minutes)
- Each member shares:
  What I did, what I'm doing, blockers
- Keeps everyone focused and synced

Sprints

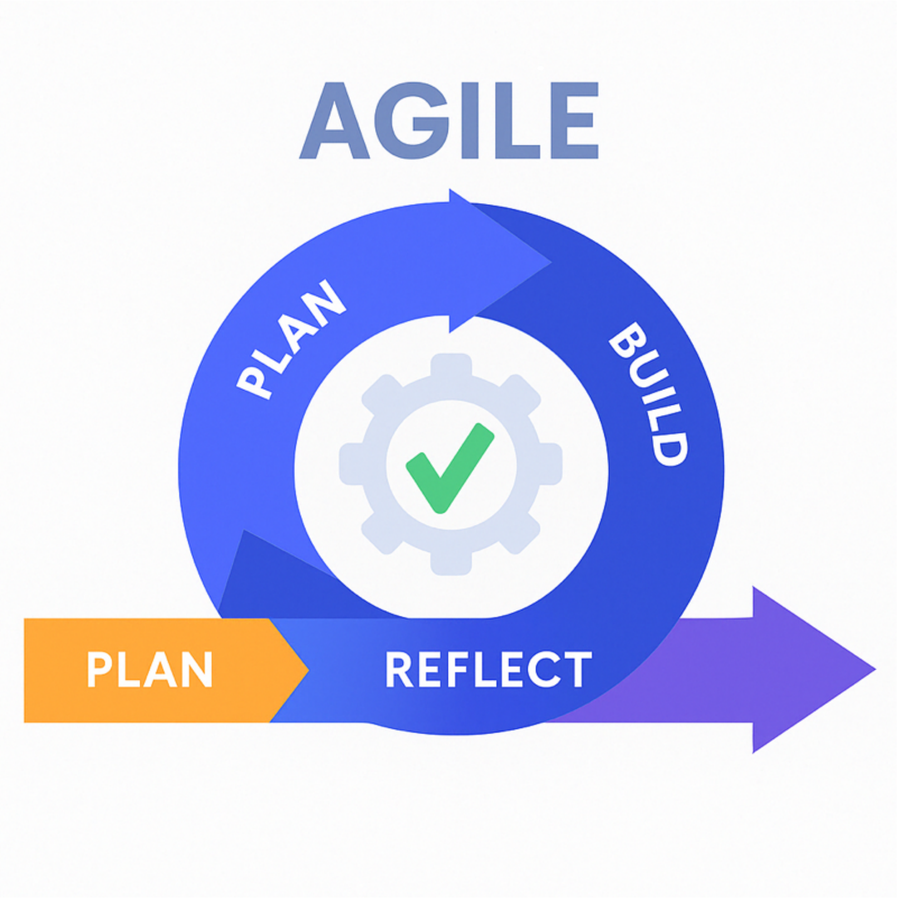
🌀 Deliver value in short cycles

- Time-boxed work cycles
  (usually 1–2 weeks)
- Clear goal and definition of done
- Ends with review and retrospective

Example Agile Sprint Cycle

From idea to feedback in one sprint

- Plan – Define sprint goal and user stories
- Build – Develop core functionality (1–2 weeks)
- Test – Share with users, gather feedback
- Reflect – Run retrospective, adjust priorities



Retrospectives

📋 Improve the team, not just the product

- End-of-sprint meeting to reflect
- What went well? What didn’t? What will we change?
- Builds continuous improvement mindset

User Feedback Loops

🔍 Learning from real users

- Build → Measure → Learn (Lean Startup Loop)
- Validate assumptions with real-world usage
- Feedback is data, not opinion

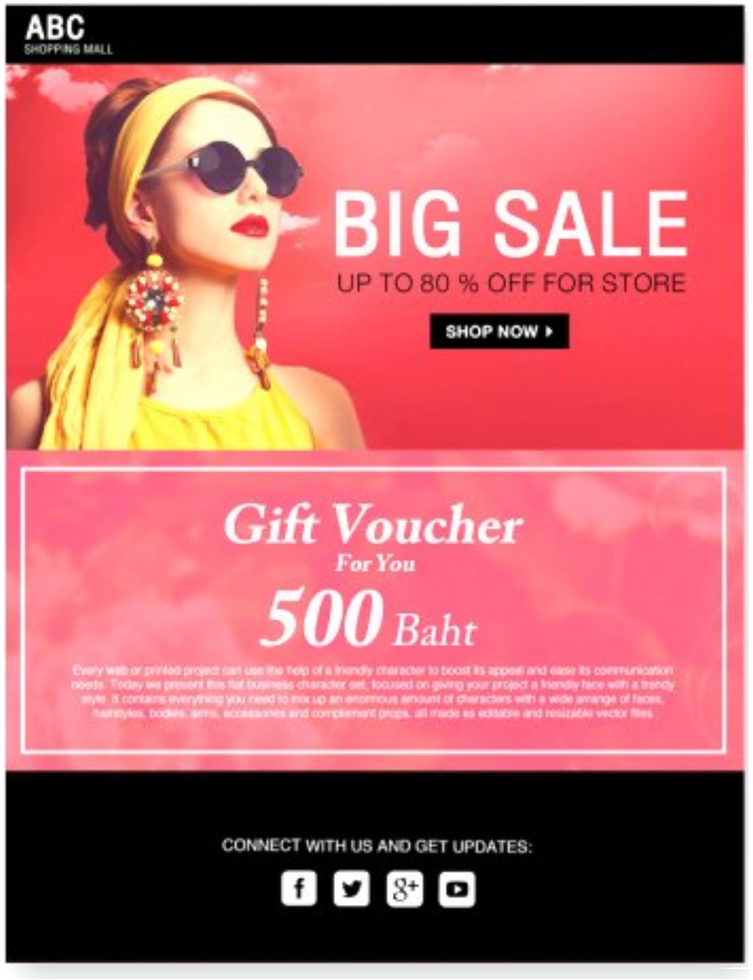
Types of Feedback

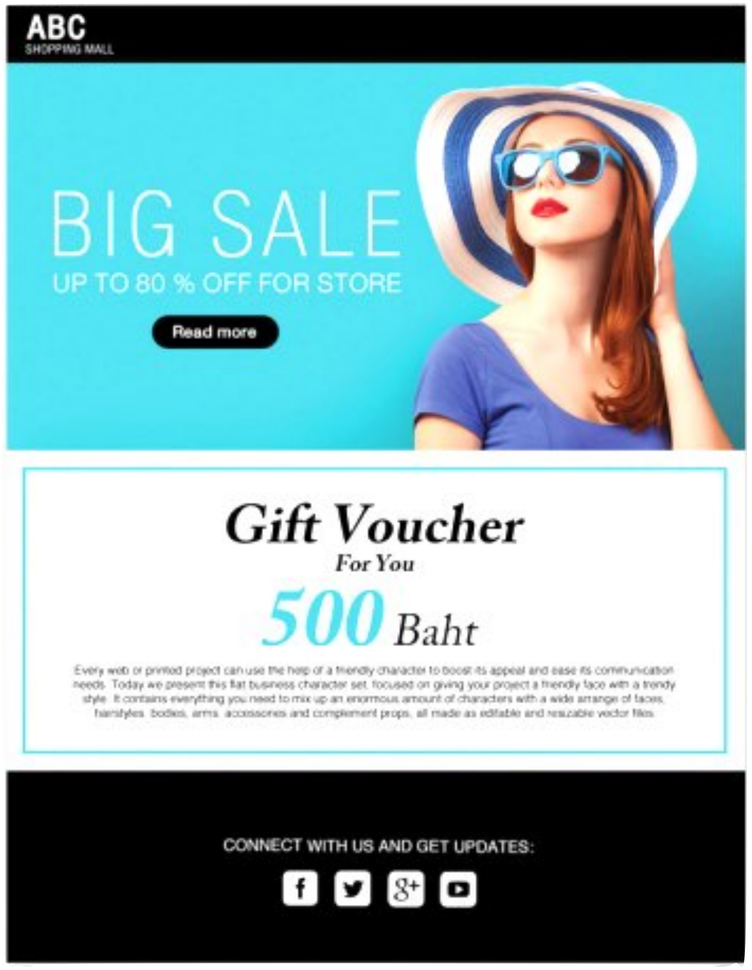
Qualitative vs Quantitative

- 🎤 Qualitative (“*Why?*”): interviews, usability testing, open-ended surveys
- 📈 Quantitative (“*What?*”): analytics, A/B testing, usage metrics
- Combine both for full picture

Example A/B Testing

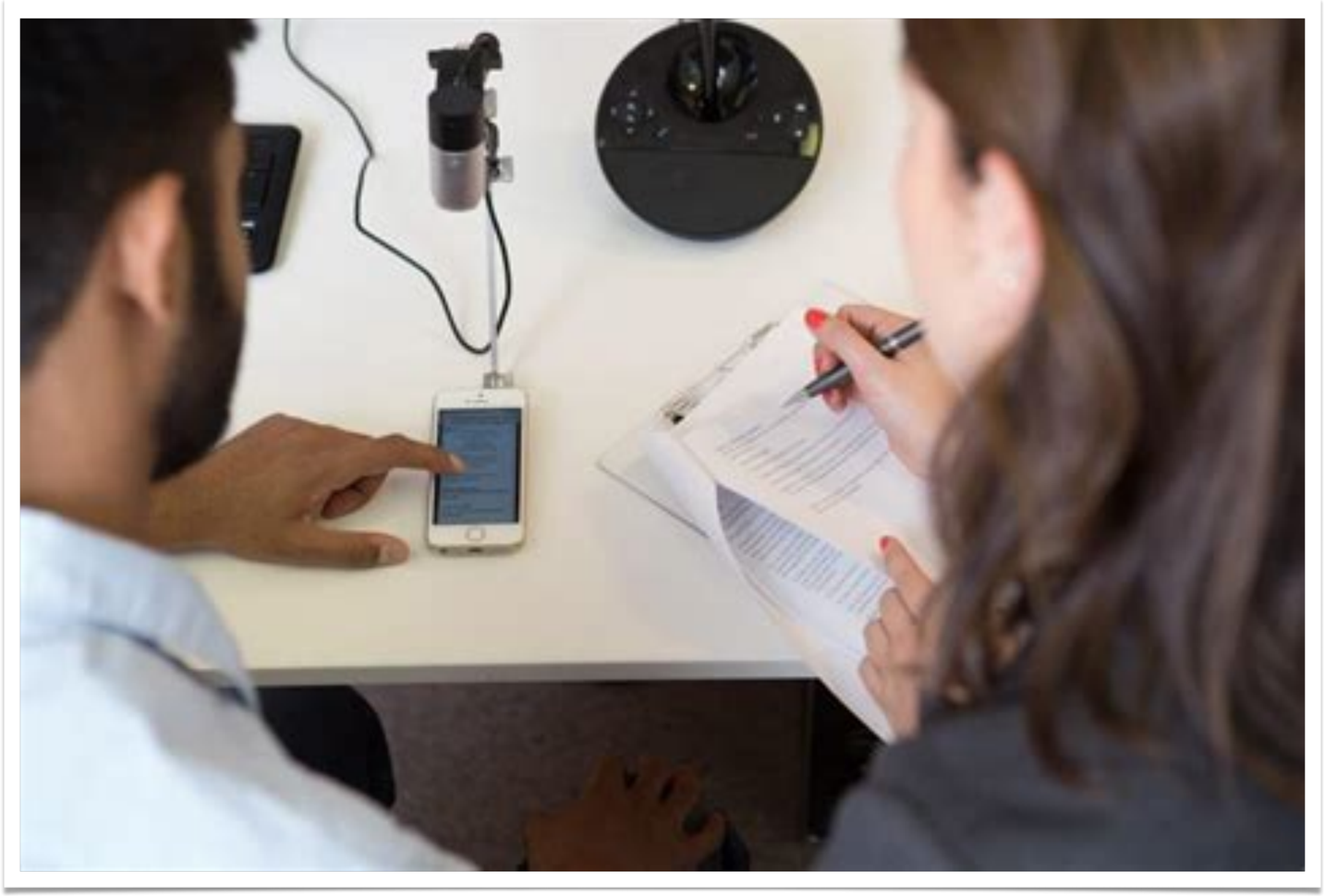
Give alternatives





Example Usability Testing

Monitor users while they use the product



Turning Feedback Into Action

From insight to iteration

- Group similar feedback into themes
- Use prioritization tools to rank improvements
- Close the loop with users

Prioritization in Product Management

Choosing what to build (and what not to)

- Founders must choose where to focus resources
- Not every request should be acted on
- Use frameworks to decide with clarity

RICE Framework

A scoring system for prioritization

- **R**each – how many users will this impact?
- **I**mpact – how strongly will it move the needle?
- **C**onfidence – how sure are you about this?
- **E**ffort – how much time or work is required?

RICE Framework

A scoring system for prioritization

- **R**each – how many users will this impact?
- **I**mpact – how strongly will it move the needle?
- **C**onfidence – how sure are you about this?
- **E**ffort – how much time or work is required?

$$\frac{R \times I \times C}{E}$$

MoSCoW Method

Categorize features by priority

- **Must have** – critical to MVP or core functionality
- **Should have** – important but not essential
- **Could have** – nice-to-have, non-blocking
- **Won’t have (now)** – intentionally excluded

Example: One Feature, Two Frameworks

Feature: "In-app referral program"

RICE Score:

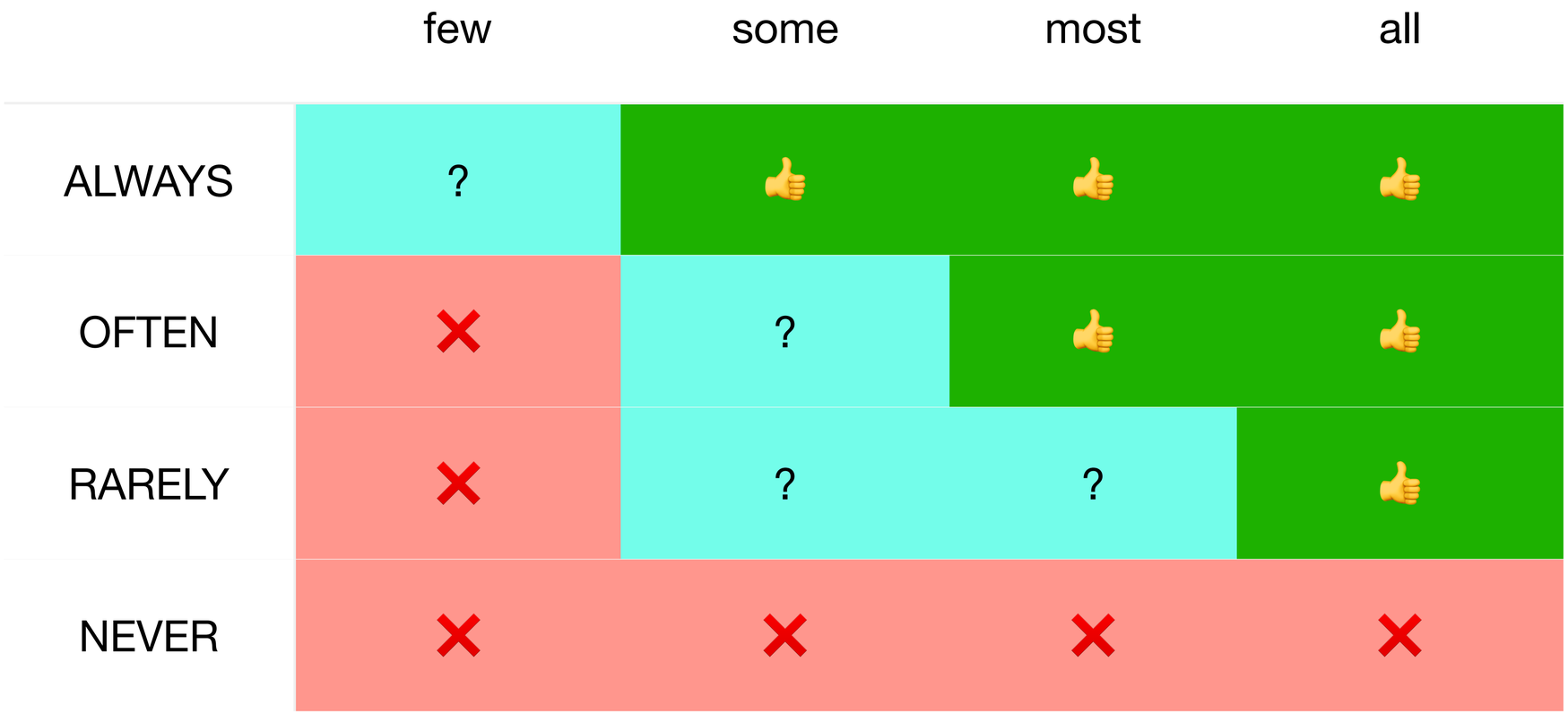
- Reach: 500 users/month
- Impact: High (8/10)
- Confidence: 80%
- Effort: Medium (2 weeks)
- **Score** = (500 × 8 × 0.8) / 2 = 1600

MoSCoW::

- Could Have – not essential for MVP, but adds long-term value

Planning - Simplified

Feature Planning / Review



Example Product Features

Sport Tickets

Feature	Description
Event Listings	Users should be able to easily browse the events..
Search Functionality	Search for upcoming local sports events, filtering by date, location, or sport type
Seat Selection	Allow users to view a seating map and select specific seats for the event, providing a clear visual representation of availability.
Buy Now	A seamless, easy-to-use checkout process with payment options such as credit card, PayPal, or digital wallets like Apple Pay.
Mobile Ticketing	Enable digital ticket delivery that users can store in their mobile wallets, allowing for QR code scanning at the event venue.
Price Transparency	Ensure that ticket pricing is transparent, with no hidden fees during checkout to build trust with users.
User Registration and Login	Simple and secure user registration and login process, allowing users to save their event preferences, payment details, and previous purchases.
Event Details	Provide clear event details, including location, time, team information, and any important updates, ensuring users have all the information they need.
Ticket Management	A section for users to manage their tickets, view past purchases, and download or resend tickets in case they lose access.
Push Notifications for Event Updates	Send users push notifications for event reminders, updates, special promotions, and last-minute tickets.
Customer Support and FAQs	Offer a support center
FAQ	FAQ section, as well as easy access to customer support for resolving ticket issues or answering common questions.

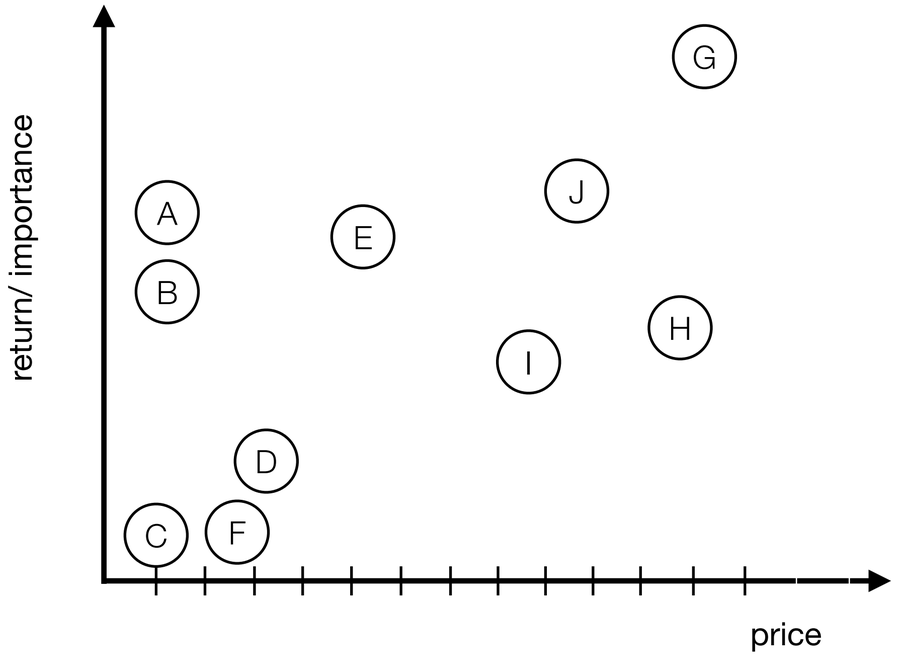
Product Features

Sport Tickets - MVP features selection

Feature	frequency	# users	selection
Event Listings	often	most	👍
Search Functionality	rarely	some	?
Seat Selection	always	all	👍
Buy Now	always	all	👍
Mobile Ticketing	always	most	👍
Price Transparency	always	all	👍
User Registration and Login	always	all	👍
Event Details	always	all	👍
Ticket Management	often	some	?
Push Notifications for Event Updates	often	most	👍
Customer Support	rarely	some	?
FAQ	often	some	?

Planning - Cost Matrix

Prioritizing Under Budget Constraints



Budget: 10

A + B + E + D

	Feature	Cost
A	Event Listings	1
B	Search Functionality	1
C	Seat Selection	1
D	Buy Now	3
E	Mobile Ticketing	5
F	Price Transparency	3
G	User Registration and Login	11
H	Event Details	11
I	Ticket Management	9
J	Push Notifications for Event Updates	10
K	Customer Support	3
L	FAQ	5

Rolling out the Product

Stages of the Rollout

Team testing: Team testing ensures that the product is free of critical bugs, functions as intended, and is ready for more extensive testing.

Company testing - Alpha: In the alpha testing phase, the product is released to a select group within your company.

Restricted Beta: The restricted beta phase expands the testing group to a limited number of external users.

Beta - The beta phase broadens the user base, making the product available to a larger, yet controlled, group.

Adding Features

Continuous Feature Development

- **Does it fit your vision?** Assess whether the proposed feature aligns with your product's long-term vision and goals.
- **Long term?** Will it remain relevant and valuable in the long run, or is it a short-term trend? Prioritize features that offer lasting value to your users.
- **For everyone?** Focus on features that benefit a significant portion of your user base.
- **Does it grow the business?** Determine if the new feature has the potential to attract more users, increase engagement, or generate additional revenue.
- **Can we support it?** Assess whether you have the resources, both in terms of development and customer support, to maintain and troubleshoot the feature.
- **Can we do it well?** Ensure that you have the expertise and technology required to implement the feature effectively.

Choosing the Right Features

Saying NO!

- **But the data looks good!** - Focus on data that aligns with your product vision and user needs.
- **It will only take a few minutes! -** While a feature may seem quick to implement, consider the broader impact on the product's complexity and user experience.
- **A customer is about to quit!** - Evaluate if the request supports your broader user base and vision.
- **We can just make it optional!** - Ensure that optional features truly enhance the user experience for a significant portion of your user base.
- **We have time to do this!** - Assess if the feature aligns with your product's immediate objectives.
- **Our competitor has it!** - Focus on differentiators and improvements that make your product stand out.

How to Say No (with Respect)

Declining feature requests diplomatically

- 🙏 **Acknowledge the idea**: "Thanks for suggesting this—it's a thoughtful addition."
- 🧭 **Align with strategy**: "Right now we're focused on [core problem or goal]."
- 🗓️ **Offer future possibility**: "It's something we'll revisit in a future phase."
- 💡 **Invite engagement**: "Can you help us better understand the pain point behind this?"

Feature Improvements

Enhancing User Experience Strategically

- **Deliberate Improvements**: Identify areas where your product can outperform competitors and invest in improvements to maintain your advantage.
- **Frequency Improvements**: Allocate resources to improve features that are crucial to fulfilling your strategic vision and driving user engagement.
- **Adoption Improvement**: Identify features that users don't frequently use but hold potential value. Develop strategies to promote these features, such as user education, guided onboarding, or highlighting their benefits.

Feature Improvements

Deliberate Improvements

A streaming service like Netflix notices that its competitor is offering better recommendation algorithms.

Improvement investment:

- invest in enhancing their machine learning algorithms to improve personalized recommendations

Feature Improvements

Frequency Improvements

Instagram notices that users engage the most with stories.

Improvement investment:

- allocate more resources to optimize stories recommendation
- add interactive stickers, polls
- include music

Feature Improvements

Adoption Improvements

Slack has a feature for integrating external project management software that users are not fully utilising.

Improvement investment

- add onboarding tutorials
- educational webinars
- highlight integration to improve adoption

Common Pitfalls & FAQs

What to avoid when applying product frameworks

- ❌ Overbuilding the MVP – simplicity is key
- ❌ Confusing Agile with chaos – it still requires planning
- ❌ Ignoring user feedback – learn early and often
- ❌ Misusing prioritization tools – frameworks are guides, not rules
- ❓ What if users give conflicting feedback?
- ❓ Can you iterate too fast?

Summary

Key takeaways from today’s session

- MVPs help you learn fast, not ship fast
- Agile development enables continuous improvement
- Feedback loops turn data into insight
- Prioritization tools keep teams focused

Assignment

Apply what you've built

- Write out your product vision and concept (list the features, key consideration, benefits, UX and UI)
- Develop an MVP
- Create product roadmap

Use Google Docs or Slides to create your pitch deck and share it with ales@spetic.si by the day before the next lecture!