

What is a PM?



LECTURE
SUMMARY

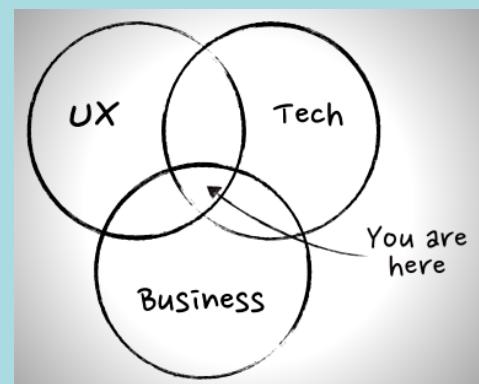
Covered in this lecture:

The product manager role and
their responsibilities

Taught by:



- ▶ "The ambiguity of the product management role is near to its essence."
- ▶ The role of the PM and their exact responsibilities change across different industries and different companies
- You are not a manager of anybody
- You are a communicator that puts all the pieces together by getting feedback from everyone else
- The PM is a communications hub, a prioritizer, a researcher, a presenter, but most importantly, they're responsible for the ultimate success of the product



Dan Schmidt

See you next lecture!

What is a product?



LECTURE
SUMMARY

Covered in this lecture:

The definition of a product;
products vs. features

Taught by:



- A product could be anything
- The PM is not always in charge of an entire product, but of certain sections of it
- A specific feature can be assigned to an entire feature team to develop it
- Example: Facebook's newsfeed has more PMs working on different parts of it
- PMs can also be split up by platform

See you next lecture!

PM
it

I'm bloated



ACTIVITY:
Find my features

Let's take a quick tour

So what is Twitter, exactly?

Twitter is an online social networking service that enables users to send and read short 140-character messages called "tweets". Registered users can read and post tweets, but those who are unregistered can only read them

Interesting, how does it work?

Users create a free account, add personal details, & photos. Afterwards they can select any number of other Twitter users to "Follow". "Following" a user subscribes you to receive any tweet they send out.

The profile screen looks like this:

Evan Kimbrell
@evankimbrell
Joined February 2009

TWEETS 16 FOLLOWING 49 FOLLOWERS 5,088 LIKES 6

You Retweeted
Cole Mercer @colemercier · Mar 18
YES!!! Big, big things coming here at @SoundCloud

SoundCloud @SoundCloud
It's official: our partnership with @SonyMusicGlobal is official. Get all the details: bit.ly/22so2TH

Evan Kimbrell @evankimbrell · 21h
@careercareerfoundry If you need a quote, DM me. Sorry, fell asleep at the Twitter wheel

CareerFoundry @careercareerfoundry
@evankimbrell We are about to post a new blog about #webdev on our site. Would you like to contribute a quote?

Who to follow - Refresh · View all

- joachim @joachimroncin Follow
- coroflot @coroflot Follow
- Smashing Hub @smashing... Follow

Find friends

Trends - Change

- #ThePassionLive 153K Tweets
- #TheWalkingDead 155K Tweets

When a user decides to tweet something, it will show up in their own personalized timeline feed shown above.

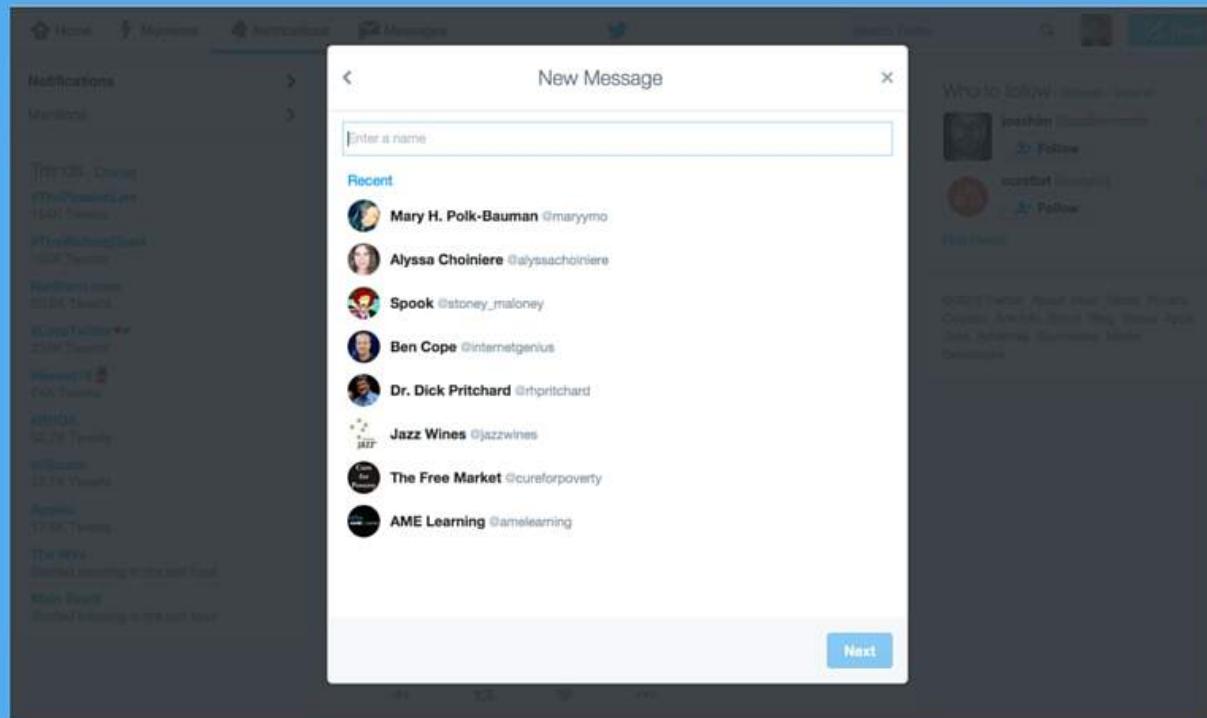
To follow someone the user just has to navigate to the profile page of another user and click "Follow"

The screenshot shows the Twitter profile page for the account @Udemy. The header features the Udemy logo and the tagline 'Learn anything. Anywhere.' Below the header, there's a large green 'U' icon. The profile summary states: 'The world's largest destination for online courses.' It also shows location as San Francisco, CA, and joined August 2009. The stats section indicates 10.3K tweets, 7,766 following, 95.9K followers, 310 likes, and 1 list. The 'Tweets' tab is selected, showing two recent tweets from the Udemy account. The right sidebar is titled 'Who to follow' and lists three users: joachim, coroflot, and Smashing Hub, each with a 'Follow' button. A 'Find friends' link is also present.

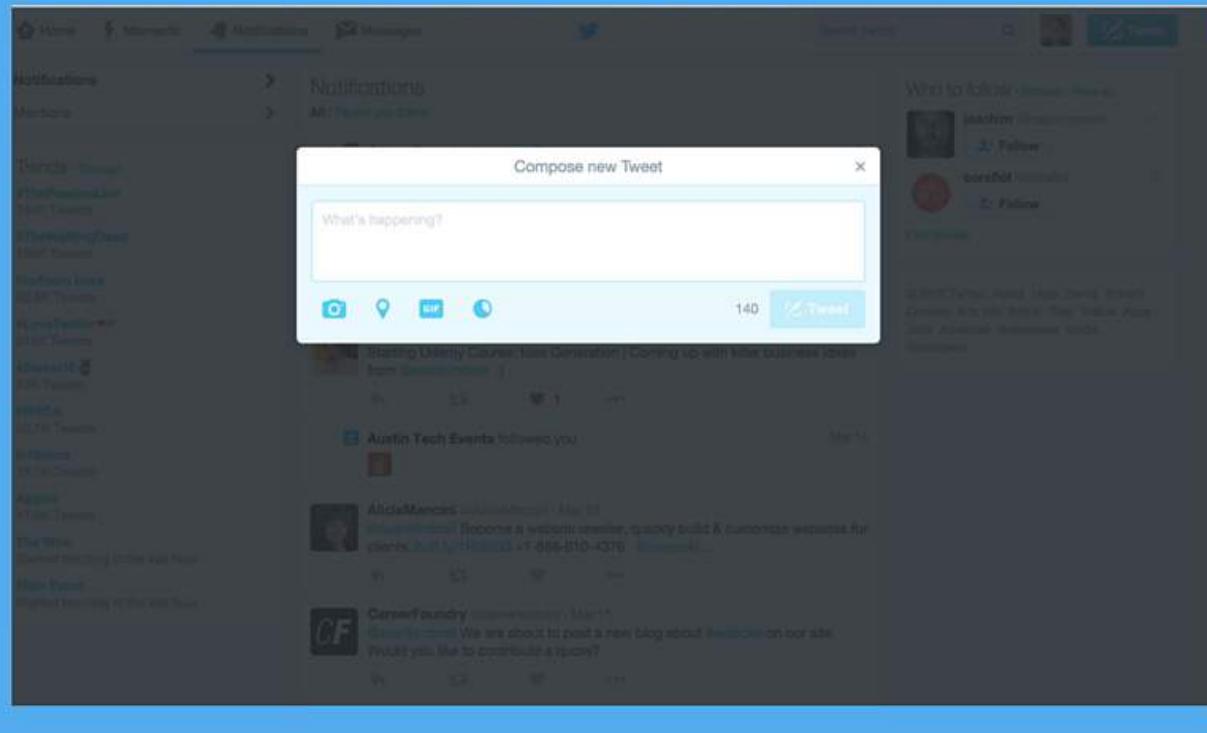
To see what the aggregate of your Following list is tweeting, you use the Home screen that houses your main timeline.

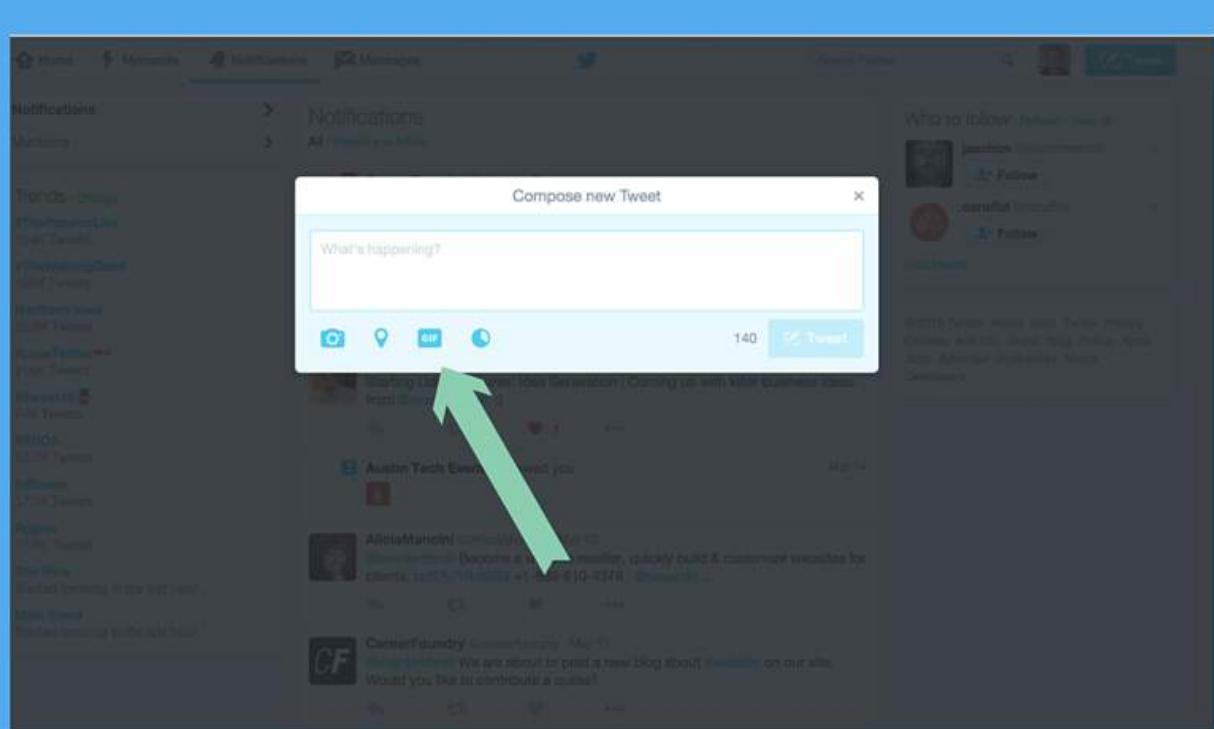
The screenshot shows the Twitter Home screen. At the top, there's a search bar and navigation links for Home, Moments, Notifications, and Messages. The main feed displays several tweets from followed users. One tweet from Evan Kimbrell (@evankimbrell) is highlighted, showing 16 tweets, 49 following, and 5,088 followers. Another tweet from Product Hunt (@ProductHunt) is also visible, featuring a photo of a MacBook with a glowing logo. The right sidebar contains a 'Who to follow' section with the same three users as the previous screenshot, along with a 'Find friends' link. The bottom of the screen includes a footer with links to Twitter's terms of service, privacy policy, and other legal information.

If you want to message someone privately, you have the option of sending a Direct Message (assuming they follow your profile)



When you're ready to tweet yourself, you click Tweet in the right hand corner and get this:





There are several features baked into the Tweet button

- Location*
- Add Photo*
- Add a GIF*
- Add a Poll*

Now you can interact with other people on Twitter publicly. You do this by:

- Liking their tweets*
- Re-tweeting their tweets*
- Replying to their tweets*

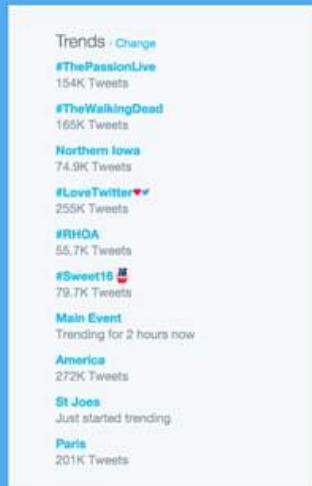
Let's talk about the feature called "Hashtagging"



When you tweet and you add the symbol # before a word, it creates a hashtag out of that word. When someone then searches for # plus any word, they'll see a timeline of any tweets that included the hashtag

A screenshot of the Twitter search interface showing results for the hashtag #AppleEvent. The search bar at the top is highlighted with a green arrow. The results page has a blue header with the search term "#AppleEvent". Below the header are tabs for "Top", "Live", "News", "Photos", "Videos", and "More options". On the left sidebar, there are sections for "Related searches" (including #apple and #iphonese), "Who to follow" (with Khan Academy and Treehouse listed), "Find friends", "Trends" (including #ThePassionLive, #TheWalkingDead, Northern Iowa, and #LoveTwitter), and "Top news" (listing "5 things to watch for at Apple's next event" by Newsday). The main content area shows a tweet from Newsday with a photo of the Apple logo on the side of a building. The tweet has been liked 24 times and was posted on March 20, 2016.

From the Hashtags, Twitter can then display a list of the most popular at any given time.



Last stop on our tour. Woot woot. Twitter has one last primary feature: Moments

Home Moments Notifications Messages Search Twitter Tweet

Today | Election 2016 | March Madness | News | Sports | Entertainment | Fun

Rihanna couldn't quite handle this

The Passion: People had questions and praise for Tyler Perry's musical

Eugene leveled up on The Walking Dead

R.I.P. Anthony Davis' 2015-2016 season

Moments allows you to see a curated list of the largest "moments" happening across Twitter. Mainly curated from celebrities or accounts that are high profile / news related

Alright, now your turn

Find a "product" in Twitter that we didn't cover and share it with the group

I'll give you a hint. We barely scratched the surface of what you can do and what needs to get done on Twitter.

Scour the photos, create an account and browse.

What parts do you think are complicated enough that a Product Manager might be assigned to them?

Twitter has 300m active users so nothing is trivial.

Remember: Whoever gets the most obscure, the most hidden product and posts it will get a prize

Types of PMs



Covered in this lecture:

The differences between the
3 types of PMs

Taught by:



- ▶ There are 3 types of product managers
- ▶ The difference between them is that they have different stakeholders
- ▶ Stakeholders are the people who have input into what you are building

Examples: users, shareholders, lawyers, marketing

- #1 Internal product manager
 - builds tools for other people inside the company
- #2 Business to business product manager
 - (SAAS - Software as a service product manager)
 - the clients are other companies
 - the PM interacts with the sales people of the company

- #3 Business to consumer product manager

- the most common
- the client is the average consumer
- it requires a lot of vision and creativity
- the PM has to get feedback from users and analyze the data in order to adapt the product



See you next lecture!

What type of PM do you want to be?



Covered in this lecture:

What personality types and what levels of experience each of the three PM types is best suited for

Taught by:



● Internal PM

- great for starting out
- you learn a lot about technology
- integrate with other systems
- you do a lot of project management
- less risk
- small number of users

● Business to business PM

- good introduction role
- small number of users
- you can be flexible and creative
- priorities weighed by \$\$\$
- tight deadlines
- one or very few platforms

● Business to consumer PM

- uncertainty and pressure
- millions of users
- multiple platforms
- high risk, the company can lose money
- you have to do extensive user testing
- you learn a lot



See you next lecture!

INTERNAL PM

THE INTERNAL PM

BUILDING PRODUCTS FOR YOUR OWN COMPANY

Your products solve problems, create efficiency, and have your coworkers looking at you like you're Santa Claus.



Ability to prioritize

People are going to want one thing plus 20 more, so you need to figure out what stays and goes.



Organized

Your to-do list is going to be galactic.



Ability to juggle

Yeah, like, actually juggling. That's what we mean.



Decisive

You won't be able to make everyone happy and that's ok.



Diplomatic

People are going to...disagree about which direction to head in. Put out the fires and carry on.



Self-Driven

You will be given a lot of independence. What you do with it is up to you, padawan.

B2C PM

THE CONSUMER PM

BUILDING PRODUCTS FOR CONSUMERS

There are over 7 billion consumers in this world and your job is to figure out which ones to make happy. You're the person who was like, "Hey let's make emojiis because it will make everyone's lives better."



Visionary

You have to rely on your intuition to come up with ideas for your product.



Detail-oriented

Sometimes the make or break is as small as moving the button from left to right.



Analytical

Be able to make judgement calls based on data, even if it conflicts with your initial vision.



Empathetic

Understand who you're designing for in order to create the best experience possible.



Strategic

Your vision, the consumer vision, and the company vision need to be combined into an actionable plan. Connect the dots.



Confident

There will always be multiple solutions; have confidence in the one that you created.

B2B PM

THE B2B PM

BUILDING PRODUCTS FOR OTHER COMPANIES

You build products for companies who sell to other companies. The stakes are high in this game, as what you create affects the Hamptons' housing market.



Communicative

You will be working with a lot of people and will need to be able to get your point across.



Attentive

Anticipate what's next for the company, not just the here and now.



Dedicated

These companies are going to want results, and fast.



Collaborative

You should be able to navigate through the stormy sea of opinions.



Problem Solver

The company might not give you a lot of information to work with, but you'll make it work, eventually.



Confident

You have got to hold your own among a lot of uber-confident people.

Product vs. Project management



LECTURE
SUMMARY

Covered in this lecture:

The differences between
product management and
project management

Taught by:



- ▶ Product managers are responsible for the success of the product
 - Success is defined by KPI or metrics
 - The method by which they reach their goals is not defined, it's up to them and their team
- ▶ Project managers are responsible for accomplishing a project, not a goal
 - A project usually has a timeline and a budget as a constraint
 - As a PRODUCT manager, you have to have the skill of PROJECT management in order to manage the execution of the product

See you next lecture!

A day in the life of a PM



Covered in this lecture:

Cole's work schedule on
typical and atypical days

Taught by:



▶ 9:00 AM

Check emails & technology news

Keeping up with the industry is important as a PM

▶ 10:00 AM

Metrics dashboard

We try to track down the reason why the metrics have been affected

▶ 11:00 AM

Team standup meeting

Updates on the metrics

▶ 11:15 AM

Meet with designers and talk about problems and feedback

We think about new solutions or work on current ones

► **12:00 PM**

Write specs, tickets, user stories for upcoming features

► **1:00 PM**

Test latest mobile release

Testing on Android and iOS and report bugs

► **2:00 PM**

From now on, things change depending on the day

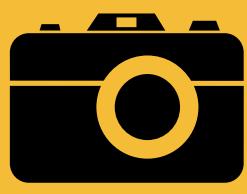
- Monday: Meet with team to do sprint planning or retrospective
- Most days: Meet with other PMs that are stakeholders
- Random other things:
 - Meet with data science team and look at exploratory data from user behavior
 - User tests of new features
 - Present metrics milestones to executives, investors, or other PMs
 - Present a recent accomplishment on Fridays
 - Open houses
 - Meet with new employers for orientation

► **As a PM, the days are always dynamic**



See you next lecture!

Need Product Inspiration?



Blogs & Profiles to follow

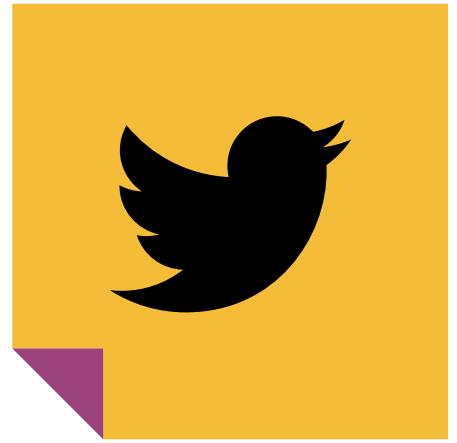
In the next pages, we have a curated list of both blogs and Twitter profiles that we suggest you follow. These people are not only some of the pre-eminent thought leaders in product management but they also create all around great content. Keep up to date with these people so you can stay up to date with what's happening

BLOGS



- blog.intercom.io
- steveblank.com
- firstround.com/review
- medium.com/product-love
- productbeautiful.com
- svpg.com
- nirandfar.com
- mironov.com
- pmblog.quora.com
- slackhq.com
- nerds.airbnb.com
- productmanagerclub.com
- goodproductmanager.com
- mindtheproduct.com
- medium.com/earnest-product-management

Twitter



- Hunter Walk (hunterwalk)
- Ryan Hoover (rrhoover)
- Josh Elman (joshelman)
- Ryan Singer (rjs)
- Expa (expa)
- Jason Evanish (Evanish)
- Sean Rose (@sean_a_rose)
- Kenton Kivestu (kivestu)
- Ben Chestnut (benchestnut)
- Ian McAllister (ianmcall)
- Joe Zadeh (joebot)
- Pete Davies (pdavies)
- Dan Olsen (danolsen)
- Matt Schlicht (MattPRD)
- Ariel Seidman (aseidman)



- Ken Norton (kennethn)
- Sachin Rekhi (sachinrekhi)
- Julie Zhuo (joulee)
- Braden Kowitz (kowitz)
- Chris Dixon (cdixon)
- Matte Scheinker (matte)
- Andrew Chen (andrewchen)
- Garrett Camp (gmc)
- dustin curtis (dcurtis)
- Kevin Rose (kevinrose)
- Jason Shellen (shellen)
- Intercom (intercom)
- Sriram Krishnan (sriramk)
- Michael S Galpert (msg)
- Delighted (delighted)
- Rebekah Cox (artypapers)
- Ellen Chisa (ellenchisa)
- David Sacks (DavidSacks)
- Brian Lovin (brian_lovin)

1 Day MVP



Enroll for free!

Use the coupon:
becomeapm



THE PRODUCT MANAGER HANDBOOK

Compiled by Carl Shan
Designed by Brittany Cheng

INTRODUCTION

“What in the world is Product Management?”

It was the above question, and my burning desire to learn the answer, that sparked the creation of this handbook.

You see, the inspiration for this handbook came when I was hired as an intern Product Manager in a large education technology company. At that point, although I had successfully impressed the interviewers with my background and passion for education to secure the job, I still had relatively little idea what it really meant to be a Product Manager.

Worried that I wouldn’t be able to excel in my role, I decided to spend the few months until my internship connecting with, interviewing, and learning from some of the best Product Managers in the field.

Fortunately, I was able to get in touch with some of the most brilliant, thoughtful and helpful individuals working in Product Management. Hailing from companies like Google, Facebook and Microsoft, these Product Managers not only agreed to share their insights with me, but they also generously gave permission for their thoughts to be included in this handbook to be distributed publicly with the entire world.

What you have in front of you are the distilled and polished gems of wisdom that were unearthed during the course of all these conversations.

This handbook provides invaluable insight for anyone interested in working as a Product Manager or who simply wants to learn about what it takes to build an excellent product. In reading the conversations contained here, you will find career advice, product advice and even life advice.

My dream is that the insights contained within this handbook will serve as inspiration for people everywhere to create amazing products that improve the world.

Enjoy.

Carl Shan

CONTENTS

HOW TO GET A JOB AS A PM	4
INTERVIEWS	8
JEREMY CARR	8
CLEARSLIDE	
JASON SHAH	13
YAMMER	
LUKE SEGARS	17
GOOGLE	
LILY HE	21
WORK MARKET	
SUNIL SAHA	25
PERKVILLE	
SEAN GABRIEL	29
MICROSOFT	
DAVID SHEIN	34
FACEBOOK	
PAUL ROSANIA	37
TWITTER	
LAYLA AMJADI	45
FACEBOOK	
AVICHAL GARG	52
FACEBOOK	
ACKNOWLEDGMENTS	58
ABOUT CARL & BRITTANY	59

HOW TO GET A JOB AS A PRODUCT MANAGER

Gayle Laakmann McDowell and Jackie Bavaro

People go to medical school to become a doctor and law school to become a lawyer, but what do they do to become a product manager? Business school is one option, but there are many others. Product management jobs are within reach of new graduates.

How do you get into Product Management straight out of college?

Big tech companies like Google, Microsoft, and Facebook are always hiring new grad product managers¹. Write up your résumé and head to your school's career fair to chat with the recruiters.

If these companies aren't recruiting at your school, you'll need to network. Find friends who can connect you with a recruiter, or try to connect with employees from the company using Twitter, Quora, LinkedIn, or their blogs. Many employees are happy to refer people who have shown a genuine interest in the company and have a strong resume.

Some startups will also hire fresh college graduates into Product Manager roles, but this is more unusual. Typically, to land such a role, you'll have to really stand out as a PM candidate and get your foot in through your personal / professional network.

¹ Note: The name of this role might differ from company to company. Microsoft hires many new grads for Program Manager roles, which is the equivalent of other companies' product manager roles. Microsoft also has a Product Manager role, but this is more of a marketing function and is usually not entry level. Google has an entry-level role called a Associate Product Manager and a more senior role called a Product Manager.

How do you get a Product Management internship?

Product Manager internships are obtained the same way that full-time PM roles are: through career fairs and networking. The big tech companies tend to have PM internships, but the smaller companies do not.

What if you can't get a PM internship?

If you can't get a PM internship but desperately want to be a PM, never fear! You can still get a lot of relevant experience that will help you in your path to be a PM. Consider the following paths.

Option 1: Do a software development internship.

Companies would ideally like their PMs to have strong technical skills, so a software developer internship is a good time to boost your skills here. During your internship, look for ways to show leadership. Can you volunteer to write up the spec for a new feature? Analyze data that you've gotten from customers? Maybe run a few meetings? Doing these things will help you demonstrate PM talent.

But should you go for a startup or a big company role? Both can be good paths.

A big company will stick an excellent name on your resume, and give you an "in" with a recruiter at that company. That could be very useful when you look for a PM role the following year.

On the other hand, startups often have less defined roles – and lots of work to be done. They are moving fast and the upcoming features may not be fully fleshed out. Guess who gets to define them? The programmers. In this situation, you aren't a programmer; you're a "programmer++." You have the opportunity to take on PM-like responsibilities even as a software developer intern.

Option 2: Build a side project

Just because you're a student doesn't mean you can't be an entrepreneur — at least on your own side project.

If you have coding skills, you can build your own web or mobile application. This means that you're developing your technical skills and your leadership and analytical skills. You are acting as a developer and a PM.

Need money for your summer work? No problem. You can do software development consulting by taking on projects from Elance and oDesk.

If you don't have coding skills, you could use your summer to learn to code, you could partner with an engineer, you could (if you have the money) outsource development on oDesk

HOW TO GET A JOB AS A PM

or Elance, or you could launch something that doesn't require programming. There is a lot of off-the-shelf software to help companies in specific niches.

Building a side project is an excellent path for freshmen and sophomores who might otherwise have trouble obtaining an internship. Give your project a snazzy name and you might even be able to list this under your resume's employment section, with you as Founder / CEO.

What do recruiters look for in PM candidates?

The background of the "perfect" PM varies across companies and even teams, but usually has the following attributes:

- Leadership
- Analytical & Data Skills
- Technical Skills
- Initiative
- Product Design Skills & Customer Focus
- Strong Work Ethic

Note that this is the perfect PM. Even many experienced industry PMs will be missing some of these attributes.

This can be a useful framework to approach your experience and resume from. How can you demonstrate that you have these skills? If you don't yet have these skills (or haven't yet done something to demonstrate that you do), how can you develop these skills?

For example, a student from a strong school with a major in Computer Science and a strong GPA might get a phone screen just by handing in her resume. Her major shows technical skills and her GPA is a signal of work ethic. However, her resume would be even stronger if she had launched a programming contest on campus. That shows initiative.

Although some of these attributes sound "fluffy," they can all be demonstrated through concrete actions.

- Leadership? Become a president of a club or lead an organization.
- Analytical / data skills? Quantitative coursework (computer science, math, physics, economics, etc) can demonstrate you know your stuff here.
- Technical skills? A Computer Science major or minor will do the trick. Or you can learn to code and list some projects you've done on your resume. Or, even if you don't know how to code, you can at least maintain your own website.
- Initiative? Do some side projects for fun. Launch a club. Organize a school-wide volunteer effort.
- Product Design skills / customer focus? Focus on creating a beautiful application — and provide screenshots on your resume. If the aesthetics of application design aren't your

HOW TO GET A JOB AS A PM

thing, get a friend to help you out with it, while you focus on getting a feature set that really addresses your user's needs.

- Work ethic? A good GPA, a bunch of projects, or basically anything difficult that you've been successful in shows work ethic. One student listed on his resume that he "completed 62 miles of a 100 mile ultra-marathon, after getting injuring ankle on mile 30." This might not have been the most medically sound decision, but it did show perseverance. (Yes, his interviewers asked about this!) He's now working his butt off as a PM for Apple.

Demonstrating these doesn't mean demonstrating them all separately. In fact, a single side project could show all five of these aspects.

Once you've made good progress with some of these aspects, add what you did to your resume and apply for a PM job. As you meet with people, talk about what you did and the choices you made. You've just created PM experience for yourself!

***Cracking the PM Interview* is now available on Amazon! Get it [here](#) today.**

This advice to aspiring Product Managers was kindly contributed by Gayle Laakmann McDowell, author of "Cracking the Coding Interview" (and ex-Google, Microsoft, and Apple engineer), and Jackie Bavaro, a Product Manager at Asana (and ex-Google and Microsoft PM). You can learn more about their thoughts on how to land and excel in a Product or Program Manager job and interview through their new book "Cracking the PM Interview," which is available on Amazon [here](#).

JEREMY CARR

Director of Product at ClearSlide

JEREMY'S BACKGROUND

Jeremy is the Director of Product at ClearSlide, where he focuses on data and analytic products. ClearSlide is a sales engagement platform that helps to close more deals faster. Previously, he held product leadership roles at Palantir, Stylemob (acquired by Glam), and Videoegg (now Say Media).

He earned an MS in management science and engineering from Stanford and BA in computer science and economics from Carleton College.

SUMMARY OF JEREMY'S INTERVIEW

In his following interview, Jeremy shares:

- The thought experiment Product Managers can use to figure out what skills to build
- The 5 top career metrics he uses to track his own success
- Specific product design skills to learn as a Product Manager
- His thoughts on the potential conflict between ambition and contentment
- And more...

Read on to learn more from Jeremy!

JEREMY'S ANSWERS

In your opinion, what are the goals and purpose of Product Managers? And as someone who is not coming from the engineering side of things, what are some of the most valuable skills I could develop in a 3-month internship as a Product Management intern?

In terms of understanding the goals and purposes of a PM, I would suggest reading Ben Horowitz's piece on '[Good Product Managers, Bad Product Managers](#)'.

How will you contribute?

As for skills, I would recommend to start developing yourself on the design side of product (ability to prototype), and/or the technical side of product (substantively know what you're talking about; be able to estimate how long things will take and think through details).

A useful heuristic to use in thinking about this is to imagine there just being two people on the product team: you and a developer. How would you contribute? This thought experiment distills more clearly the skills you might be interested in developing to build a solid foundation, whether you stay working in startups or not. Another approach is to think of a team that is comprised of yourself, five developers, one designer, and one QA person. Now how will you contribute? Communication, organization, GTD, and good project management become more relevant.

Specific design skills that are worthwhile to pick up would be to look into wireframing (e.g., using tools like Visio, Balsamiq), fundamental UX principles and design (higher-resolution mockups; concepts of user flow), graphic design (Photoshop or Illustrator), and potentially learning JavaScript (AJAX and jQuery are pretty important), etc.

Additionally if you're looking for principles on a life well lived, you should check out Ray Dalio's "Principles". He's someone who's done a lot of deep thinking on this. Another book I would recommend is Peter Bevelin's "Seeking Wisdom".

What could an intern or newly hired Product Manager do to add value as quickly as possible?

Domain: within first few weeks, try to find a project with measurable impact you want to deliver on over the course of your internship. Then make sure that you execute on it.

Skills: each product management opportunity is different, but functionally, they're typically design (information architecture, UX, or graphic design), development/spec'ing (detail-orientation, learning to work with engineering, etc), or project management (ticketing, prioritization, etc). Make sure that you leave the internship having built some of these skills.

Personally, I also think you should use this to assess what you enjoy doing on a day to day

basis. It's hard to maintain focus and discipline if you're working on a part of the stack that you don't find engaging.

What mental models or criteria do you use to judge the success of a product?

On the business side, there are many books on KPI's. Familiarize yourself with those concepts. Be clear about how the feature/product is contributing to the overall success of the company. Ask yourself "if this were wildly successful, what would that look like?" Important to consider whether the feature/product you're working on is an experiment or an incremental improvement; will drastically change how you evaluate its success.

Assessing the technical quality of a product is tricky from a product standpoint. You certainly need to ensure the user flows make sense, and that no "bugs" that are actually unintended consequences make it into your designs. Beyond that, a lot of the technical merit of your product will rely on the technical team you work with, and an engineering manager. Concepts like regression testing, test driven design, uptime, dev - staging - launch process...are all useful technical concepts to stay on top of.

Ask yourself "if this were wildly successful, what would that look like?"

What are the metrics you use in measuring your own success? Which ones are the highest priority, and why?

1. Product: Am I effectively balancing the interests of shareholders, customers, and employees? Can I build a roadmap that captures the vision of the executive team/customers? Can I get consensus and/or buy-in for that? Can I deliver on time with good quality?
2. Leadership: Can I motivate my team? Can I point folks in the right direction? Can I remove roadblocks from them quickly and effectively?

Three additional things I will say are good rules of thumb to follow in tracking your own career are:

1. Focus on the quality of people you're working with. That's why I joined Palantir and am still invested in the alumni network, as well as ClearSlide. The people I work with are amazing, and I'm sure will be doing some incredible things in the future. With good company, no road is long.
2. Look at the individual contributions you can make. What skills are you building? For example, as a PM who may not be coming from an entirely technical background, it would be valuable to deeply understand marketing and distribution channels.
3. Have a personal narrative that you feel comfortable sharing with others, as well as internalizing. You should be able to tie together a cohesive story. The things you pursue in the future should in some way be compounding upon the things you've done in the past. Look at the amount of opportunities you receive to learn new things.

For me personally, I try to stick to thinking about the short and medium term. I look at things in the 3 to 5-year range. I find it hard to plan beyond that, and consider life an endless sequence of emergent effects. For some people, they have a life goal such as becoming VP of Product at Amazon. And that may work for them. But I think there's a lot of serendipity in life, and I'm very much open to possibilities that I can't imagine right now.

Focus on the quality of the people you're working with.

entails, try to make it something you're genuinely enjoying.

I think if you're engaged day to day in your role, things are more likely than not to turn out for the better. I am not a fan of living a deferred life plan, which is the notion of putting off today what you really want to do in favor of some things you think you "need" to do. So whatever it is that your career

I have friends who tell me they're uncomfortable deeply valuing happiness and contentment in their own lives as they are concerned it will stifle ambition. Do you have any thoughts here, especially in the context of you've just mentioned about enjoying every day?

Is happiness and contentment really at odds with ambition? I think people frequently confuse contentment and happiness for its more malicious relative — complacency. Complacency leads to the stifling stagnation that results in dissatisfaction down the road. Fundamentally, ambition is a strong desire to achieve something; from a scientific mindset it's as simple as wanting to solve problems and answer questions.

There are also some companies that I think we can agree are very ambitious companies, all while maintaining a sense of play. I think some of the most successful companies in the Valley are great examples of successfully implementing this type of culture. Facebook and Palantir are two examples. Learning from these examples can help us to better understand why happiness doesn't necessarily conflict with ambition. I embrace goal-oriented environments, which can be hard charging but I ultimately find fulfilling.

When I speak with older professionals or more successful professionals, one thing I notice is that they've largely figured out their work/life balance. There are two types of hard work that entrepreneurs (or really anyone) tend to be engaged in:

1. Sustainable hard work
2. Unsustainable hard work

You may be able to maintain the unsustainable kind for months, or even years. But I'd urge such a person to at least note its impact on one's life — it probably looks like living a shitty life right now. So I'd encourage folks to find sustainable work. Keep in mind that, with all of these things (startups, success, life), it's a marathon, not a sprint.

I guess I have a different perspective on this than many Silicon Valley technologists because I travel a lot and have seen a lot of alternative ways of living that are outside of the Silicon Valley bubble.

I've often heard the advice that I should do a "technical" job first for a few years before jumping into PM, because it's hard to switch back to a "technical" job (such as programming, designing) after doing a PM role. What do you think of this advice? Is it a good idea to jump right into the PM role after graduation, or is it better to do a regular role first and move in to the PM role?

There are different types of PMs. At a small company, Product Managers need to be able to contribute to real product creation. At larger companies, not necessarily. But generally, the more technical you are, the better (it gives you a leg up against the army of MBA's that also want to be PMs). How will you differentiate yourself if you do a PM role immediately after graduation?

JASON SHAH

Product Manager at Yammer

JASON'S BACKGROUND

Jason is a Product Manager at Yammer, where he focuses on building out new features for the leading enterprise social network. He is also the founder of HeatData, a mobile web heatmap and analytics tool.

Prior to Yammer, Jason founded INeedAPencil.com, which provides free online education tools to students from low-income families. After growing INeedAPencil.com to more than 50,000 users with an average score increase of 202 points, Jason sold INeedAPencil.com to CK12 in 2011.

Jason holds an AB with honors from Harvard College with degrees in sociology and computer science. He blogs regularly at blog.jasonshah.org and tweets shorter thoughts at [@jasonyogeshshah](https://twitter.com/jasonyogeshshah).

SUMMARY OF JASON'S INTERVIEW

In his following interview, Jason elaborates on:

- A story that illustrates the goals of Product Managers
- The 3 important skills for interns to learn
- How a Product Manager could compensate for a non-technical background
- And more...

Read on to learn more from Jason!

JASON'S ANSWERS

In your opinion, what are the goals and purpose of Product Managers? What are some of the most valuable skills that a young intern aspiring to be a Product Manager could develop?

The goal of Product Managers is to help a team build the best product possible - through prioritizing what to work on, helping design beautiful user experiences, guiding engineering to avoid roadblocks while leaving them autonomous, and working with all other parts of the organization to provide transparency and input into the product development process.

One way to look at it is to see that part of the role is to make engineering go faster.

Overall, you want to empower the engineering team, the designers, the analytics team, etc. Your job isn't as concrete as other disciplines, so you need to be good at supporting those other roles.

A story that illustrates the points made above relates to something that I worked on recently at Yammer. This feature allowed users to mention Pending Users. Pending Users are people who began to sign up but haven't activated their accounts yet. Now, just thinking about this, it's not a particularly inspiring feature. The name of the feature is quite boring, admittedly. But it's a crucial one, and you need to, as Product Manager, be able to have the vision to see its power and explain why it's such a valuable part of the product.

For example, I framed it in terms of the larger vision of connecting employees and accelerating collaboration. It's important to be able to communicate with (via mentioning) people in your company, whether or not they are registered for Yammer, and we can never replace email until we nail that. With that vision, people could easily see the value in what we were building.

As an intern, it's going to be especially difficult to be a successful Product Manager, because you're in a ramp-up period of three to six months before you really get the hang of things. So for valuable skills, I would say:

1. **Prioritization:** This is hard as a PM. With limited resources and a backlog of projects, you need to make sure you identify the highest-impact projects to work on. Your choice of prioritization impacts everything: what gets built, how the rest of the product is affected, the morale of the teams working on the feature. You should be using data, user research, product vision, and an understanding of engineering costs to help prioritize features.
2. **Vision:** Anyone can come in and suggest we move a pixel here and a pixel there. Where do you see the product going in three to five years? What about the product today is being missed by our users? How do we turn a moderately successful app into a sensation?
3. **Analytics:** PMs today need to be data-informed. You should be looking at how people use your product and generating ideas there (in addition to more green-field brainstorming).

You want to empower the engineering team.

You should be measuring the performance of features rigorously and only releasing what does well and killing what doesn't.

I've heard that it's heavily preferred for Product Managers to have technical backgrounds (e.g., they've previously been engineers). What advice would you give to someone who may not have as much of a technical background but is still aspiring to work as a Product Manager?

Your choice of prioritization impacts everything.

The reason it's valuable for PMs to be technical is because it allows them to make good tradeoffs. Technical PMs understand engineering costs. As a result, these PMs can make better decisions about what will return the greatest benefits for the least costs and how to build true MVPs (minimum viable products).

So while you need not be completely technical, you have to be able to build credibility and then complement it with actual skills. So the fact that you have some rudimentary understanding of computer science will help you avoid any large faux pas.

Ideally, you want to position yourself to be in a company working with engineers who also have some product sense so that you don't need to dictate every last detail.

If you are not technical, try to pick up some understanding of basic concepts: performance, frontend and backend, APIs, etc. There is no way around this.

Sit with your engineering team to understand how they build and what their key constraints are. Try to unblock them. When you have a feature idea, get engineering input. Over time you will learn what is expensive and what is cheap (in engineering terms). When something is expensive, you want to aim to build a minimum viable product to validate your idea early or have an extremely good justification of the engineering investment.

The point is that you should be making intelligent decisions that others will respect, regardless of your level of technical background.

As an intern, I'm concerned about the level of impact I can have in just three months at a company. I want to learn a lot, and think I will, but I also want to get a chance to actually make an impact. How did you spend your first three months in your first PM role, and what would you do differently if you could redo it?

My first three months were spent largely in shadowing other Product Managers, studying what's good and bad about our product and competitors', and understanding how our product development process works.

If you want to maximize your learning and growth as someone who's aspiring to be a Product Manager, I believe you should shadow other folks on the product team — PMs, designers, even engineers. Understand how people work. It will help you figure out how to be effective.

Another thing I'd say is to research what the product has been in the past: try to understand how the team got to where it is today. What was the MVP version of this product? It will show you what has been prioritized in the past and how the team is thinking today. Check out past A/B tests that they ran to better understand what's been changed and how certain hypotheses worked or failed.

What are some mental models you use to view the quality of a product? In other words, what are the criteria you use for judging how successful a product is?

At a high level, I ask myself: "What is the user's goal?" and "How hard does this product make it to accomplish that goal?" It shouldn't be that hard to get done what you need to get done as a user. If it's Airbnb, it should be easy to book a good place to stay. If you're Uber, it should be easy to get a black car fast. If you're Google Search, it should be easy to find the right information.

Deeply engage with and listen to the senior product people on the team.

From a metrics standpoint, you can also measure a product by the following:

1. **Retention:** But also understanding the nuances within retention (e.g. one day vs. seven days vs. six months).
2. **Engagement:** Level of interaction and the durations that people are on Yammer.
3. **Virality:** How many new users are our current users inviting and converting.

LUKE SEGARS

Product Manager at Google

LUKE'S BACKGROUND

Luke got his bachelors and masters degrees in computer science before stumbling upon a Product Management internship at Google. He spent a summer working on search ads and decided that he really enjoyed the work and wanted to continue. He's now at Google full time and has worked on search and YouTube products.

SUMMARY OF LUKE'S INTERVIEW

In his following interview, Luke raises many thoughtful points, including:

- The top 3 most important objectives are as a Product Manager
- Whether a technical background is necessary for success at Product Management
- The most valuable skills to learn as an aspiring Product Manager
- And more!

Read on to learn from Luke!

LUKE'S ANSWERS

In your opinion, what are the goals and purpose of Product Managers? What are some of the most valuable skills that a young aspiring Product Manager could develop?

The job of a Product Manager is to identify real world problems and to come up with ways to solve them. It's an inherently social job because (1) the best solutions rarely come from one person and (2) successful products aren't developed in a vacuum. It also requires a degree of scrappiness — a lot of things go into making a successful product and there isn't always an obvious person to do everything. The PM can often pick that stuff up.

The most valuable skills that you can learn in an internship are going to vary a lot. For me:

1. What goes into developing a product outside of coming up with an idea and building it?
Hint: there are lots of answers.
2. How to pitch an idea, and how to tell a good idea from a bad one. What makes an idea stick? What makes a project succeed or fail aside from the capabilities of the individuals that are involved?
3. Learn how to structure your thinking. Many creative people are able to explode forth with creative ideas. Coming up with a meaningful and communicable structure for those ideas is at least as important as coming up with them in the first place.

Identify real world problems and come up with ways to solve them.

I've heard that it's heavily preferred for Product Managers to have technical backgrounds (e.g., they've previously been engineers). What advice would you give to someone who may not have as much of a technical background but is still aspiring to work as a Product Manager?

Talk with engineers. Find people who are willing to spend time explaining things and soak it all in. Having technical experience is important but having an interest in and appreciation for development is really important as well. I wouldn't try to fake it, but just make it clear that you're interested and I bet you'll be able to find people who can teach you.

What advice would you give to a young Product Manager just starting in his or her role? How could they allocate their time and energy so as to make a meaningful impact in the first few months of the job? How did you spend your first few months in your first PM role, and what would you do differently if you could redo it?

I created a new ad format for Google search, specifically targeted at music labels. There was a

ton to learn and, at least in my case, a ton to do. One of the things that I did well was pushing beyond my comfort zone to do things that needed to be done and make fast progress on my project.

One trap I've fallen into since I've come back is accepting too much work. Here is how I would rank your objectives when you start if you're actually interested in product management as a career:

1. Learn.
2. Do high quality work.
3. Do lots of things.

Unfortunately it's really, really hard to do all three well. Product Management is an extremely flexible field with a lot of different required skills, so investing explicit effort into improving some of those skills is likely to make the rest of your life much, much better.

What are some mental models you use to view the quality of a product? In other words, what are the criteria you use for judging how successful a product is?

How useful is the product vs the next best thing?

That depends a lot on the product. My favorite criteria: how useful is this product vs the next best thing? A product that helps rural farmers in underdeveloped regions find places to sell their crops can have a tremendous impact compared to a watch that lets you check text messages two seconds faster than you would on your phone, though the latter is shinier and likely to make more money.

On the other hand, money isn't something to be discounted. Currency, I suppose, is one of the signals we have to measure the quality of a product in the public eye, though I think that's a simplified view of the scene in many (potentially dangerous) ways.

Who are Product Managers in your field that you admire? What traits do you admire about them?

There are a number of people that I've met at Google that I admire. Some of the ones that stand out work as PM's and lead engineers. One of the most admirable qualities that I've encountered is when these people have a very strong sense of what direction a product should take and are able to articulate that reasoning clearly. This is an important skill and in my opinion a pretty tough one as well.

What are the metrics you use in measuring your own success? Which ones are the highest priority, and why?

1. Quality of relationships with peers.

2. Regularity of high-quality product releases.
3. General organization on a day to day basis.

I'd say I evaluate myself primarily along those criteria, and roughly in that order. This one probably varies more than any of your other questions depending on the company you work for and particular product within that company.

I've often heard the advice that I should do a “technical” job first for a few years before jumping into PM, because it's hard to switch back to a “technical” job (such as programming, designing) after doing a PM role. Do you think it's a better idea to jump right into the PM role after graduation, or should I do a regular role first and move in to the PM role?

I don't have any experience to base it on but I got the same advice. I've found my technical knowledge to be adequate for getting by with a very technical team. It undoubtedly would have helped to do some actual engineering for a while, but it's not clear that it'd be more useful than a year or two of product experience. I think the underlying critical bit is to BE CURIOUS and CONTINUE TO LEARN. If you do that then I don't think skipping the engineering step is a huge mistake.

Investing explicit effort is likely to make the rest of your life much, much easier.

LILY HE

Product Manager at Work Market

LILY'S BACKGROUND

Lily graduated from MIT with bachelor degrees in math and finance. After graduation, she worked in trading and consulting before transitioning into product management at Work Market, where she works currently.

SUMMARY OF LILY'S INTERVIEW

In her following interview, Lily shares her experience on:

- Her atypical background and what led her to Product Management
- What's unique about being a PM at a startup
- The techniques she used to switch careers into Product Management
- And more!

Read on to learn more from Lily!

LILY'S ANSWERS

Can you talk a little bit about your background and what led you to product management?

My path to product management is atypical. With a background in finance and math, I had worked in trading and consulting before joining the product management team at Work Market. While being on the trading floor is often exciting and nerve-racking, I have always been interested to be part of a growing business that is constantly innovating and building products that customers will love. I first transitioned from trading into consulting — you may call it a “safe” switch — in order to build my business analytical skills and intuition. However, I quickly realized that I wanted to be more proactive, rather than just advising clients on solutions and ideas, and become more actively involved with the product, tactics, and strategy of single company; thus, finding my way to Work Market.

What is Work Market?

I wanted to be more proactive and more actively involved.

Work Market is a cloud-based contractor management platform that is connected to an online marketplace of professionals seeking work assignments. Built to allow scalability, increase productivity, and enhance the quality of both contractors and the work that they fulfill, Work Market enables

enterprise organizations to do more than take control of skyrocketing operating costs — it empowers them to actually reduce costs and drive efficiency into their business.

If you were standing in a first-grade classroom and had to explain to the students what a Product Manager is, what would you tell them?

A Product Manager is someone who takes an idea and brings it to reality. They use all the tools they have at their disposal to turn their idea into a product that their customers would love and need.

What are some roles and responsibilities a Product Manager has at a company?

First and foremost, a Product Manager needs to be passionate about the product that they are building. Their typical tasks include: writing specifications, prioritizing features, conducting user research, analyzing data, coordinating communications, amassing executives buyin. A Product Manager needs to be able to break a complex project into manageable tasks and prioritize their executions by working closely with the engineering, marketing, and sales team.

Before going into Product Management, did you have an engineering background?

Although I did not have software engineering background, I do have a strong analytical background, especially studying in math and working both finance and consulting. Having a strong quantitative skills is becoming increasingly important as we rely more on data analysis to make our product decisions.

What do you believe is unique to being a Product Manager at a startup instead of a large company?

When you are on the product team of a small startup, everyone is a generalist because the product is too young for specialization. For me, this generalization has been the best part about working at Work Market. Not only do I write specifications, I also focus on user design, data analysis and some front-end coding as well. Eventually, as the company and the product team grow, it will become necessary to have a team hierarchy and for each PM to own a specific vertical of the product.

When you are on the product team of a small startup, everyone is a generalist.

What advice would you give someone who is coming from a non-engineering background and is interested in product management?

Switching careers in general can be a challenging task. First, network as much as possible in the industry or field, product management here, that you are interested in by talking to friends, friends of friends, alumni. Usually, you'll find your job opportunities through one of those connections. But if not, you should browse job boards and VC websites because most VCs list job openings at their portfolio companies. Personally, I've found the VC listings very helpful. Most importantly, you should know your story well. Why are you interested in Product Management? What skills can you leverage from your existing experience? How would you contribute to the team right off from the start? Knowing your story will be key to convincing your interviewers that they should hire you over someone who has existing experience in PM.

Do you think there is anything particularly unique about being a woman in Product Management?

While engineering is in fact a male-dominated field, I have never encountered any gender-related issues — a caveat, my experience is limited to Work Market and thus, I don't think I have enough experience to speak generally about this topic.

Do you have any advice you would give to young individuals who are aspiring to work as Product Managers?

If possible, work in both small and big organizations.

If you are still in college, I would recommend doing an internship in product management. Product management roles and responsibilities can be very different depending on the size of the organization. Thus, if possible, work in both small and big organizations to figure out what type of product management that you would like to do.

Also, take classes that will help build the skills that you'll need in product management such as design, user interface, and statistical analysis.

Lastly, don't be afraid to reach out to firms that you are interested. Use your alumni connection if it exists.

What are some specific things you've done to develop the skills you believe a successful Product Manager needs to have?

For me, I focused on learning everything about user interface and user design first since I did not have a product background. I've also become pretty proficient with HTML, CSS and a little bit of jQuery. As a small product team, we try to help out our engineering team as much as possible. So we frequently dig into the code and make small design tweaks and interface changes.

Additionally, data analysis skills (using either SQL, Excel, Python or R) are a must so that you can analyze your data to help you make better product decisions or run occasional ad-hoc analysis. Conducting A/B testing and user research is also key to making good decisions.

Lastly, the ability to take feedback critically and effectively coordinating among various internal teams are among some of the other skills that I've developed on the job.

SUNIL SAHA

CEO of Perkville

SUNIL'S BACKGROUND

Sunil graduated with a bachelor's in human biology from Stanford. He then went on to join a medical device company before moving into a market research firm (which also specialized in medical devices). Afterwards, he joined Neoforma, an internet company, where he fell in love with the internet space and began his first foray into Product Management. From Neoforma, he moved on to being a senior Product Manager at both Yahoo! and LinkedIn.

Now he is the CEO of Perkville, a startup he co-founded in 2010 that is trying to eliminate loyalty cards.

SUMMARY OF SUNIL'S INTERVIEW

In his following interview, Sunil touches upon:

- What Product Managers should focus on early in their roles
- The top four metrics he uses to measure success
- Valuable things to do as a young Product Management intern
- And more!

Read on to learn more from Sunil!

SUNIL'S ANSWERS

In your opinion, what are the goals and purpose of Product Managers? What are some of the most valuable skills that a young aspiring Product Manager could develop?

The main goal of a Product Manager is to improve the business. Whatever the bottom line may be, your goal is to move it in a positive direction.

If you're joining the team in a short term role such as an internship, then it'll be difficult to make a big impact. If I were doing an internship in product management, I would focus on one project and release it from start to finish. Try to get a feature out that has a meaningful impact on the business. This experience will teach you how to lead a team.

Typically, as a Product Manager, no one officially reports to you, but you still need to muster the charisma and resources to lead.

Doing something like owning a product is valuable during a stint as a Product Manager because it's very results-oriented and actionable. It'll teach you leadership and how to scope down to a feature to the bare minimum and hopefully release it within a few weeks. A lot of PMs are not good at scoping things down to a small chunk.

Whatever product you're working on, make sure it's measurable.

And when you go through the internship, whatever product you're working on, make sure it's measurable. A/B test new features for increased retention, engagement, and whatever other metrics you're testing against.

I've heard that it's heavily preferred for Product Managers to have technical backgrounds (e.g., they've previously been engineers). What advice would you give to someone who may not have as much of a technical background but is still aspiring to work as a Product Manager?

I actually don't think you need a super technical background to be a good PM. However, you do need to understand data. You do also need to be able to do some basic querying, such as via SQL. That way you can speed up the analytical process because you can run the queries yourself without having to go through someone else to get information.

But as long as you have a good mind for numbers and statistics, you should be in good shape.

Remember that the engineers are there to do the coding. You, as the PM, are there to drive the business forward.

What would be some valuable things for Product Managers to focus on early in their role?

One thing I would be is extremely skeptical about the feature ideas that people have. Eighty percent of ideas that people come up with probably won't yield good results. As a Product Manager, start with this assumption — be extremely skeptical about the features you work on.

Of the 80% of ideas that flop, I would say 80% of them flop because they are just plain bad ideas. 20% of them flop because of execution. So focus on ideas, yet still be skeptical of them.

Analyze the data, understand the customers (e.g., look at customer support tickets), make an effort to interview customers to look at the problems people are having.

Focus on ideas and yet still be skeptical of them.

As a new Product Manager, it would be valuable to first look at the product's historical metrics to see how they've been growing over time. Look at past projects, what's been successful, what hasn't. Doing these things will yield valuable frameworks upon which you can build your own experience.

What are some mental models you use to view the quality of a product? In other words, what are the criteria you use for judging how successful a product is?

The top 3 criteria that are used, almost de facto, at any company are:

1. Revenue — pretty self-evident
2. Usage levels and frequency (MAU, DAU)
3. Growth of product (number of users)

For an education product that you may do an internship on, there's probably more specific metrics such as whether it's improving test scores.

What is the most unique skill you've seen a PM have that made them exceptional?

I would say the ability to develop the simplest solution to a problem possible as described here: [The One Cost Engineers and Product Managers Don't Consider](#).

Complexity adds development, maintenance, implementation, training, etc cost so keeping it simple is very important.

One example of a Product Manager who did an exceptional job of this is Marissa Mayer, who kept the Google home page simple and clean despite what I'm sure was immense pressure to leverage its traffic for other Google properties.

What are the metrics you use in measuring your own success? Which ones are the highest priority, and why?

Now as CEO, I've realized that this is the best training to be a better Product Manager. And to be honest, Product Management is great training for becoming a CEO. As CEO, I use look at several metrics, but most important in my current startup are ...

1. Growth in or toward profitability as a function of ...

- a. Revenue growth
- b. Gross margin improvement
- c. Customer acquisition cost
- d. Churn reduction

2. Runway for the company as a function of ...

- a. Cash in the bank
- b. Monthly burn rate

3. Market share

4. Customer support satisfaction

SEAN GABRIEL

Program Manager at Microsoft

SEAN'S BACKGROUND

Sean is a Program Manager on the Xbox Music, Video and Ads team. Starting from a hobbyist background in web development, he interned as a PM on the Microsoft FrontPage team, returning to the Office division for a full-time PM position after graduating. He was a PM for several teams in Office, contributing to areas like human workflow and classroom education, before joining the Xbox division to build multimedia experiences. On the side, he's also moonlighted for a startup focused on structured debates. Sean graduated with a B.S. degree in Electrical Engineering and Computer Science from UC Berkeley and has been a PM at Microsoft for almost seven years.

SUMMARY OF SEAN'S INTERVIEW

In his interview, Sean covers:

- How Microsoft's PM role differs from other companies
- What he's learned from the mistakes he's made
- Whether you should jump into a PM role immediately after graduation
- And more!

Read on to learn more from Sean!

SEAN'S ANSWERS

Can you explain the PM role at Microsoft? Why is the PM role at Microsoft known as the “Program Manager” and how do you think it compares to similar roles at other companies?

Here at Microsoft, we separate the roles of Program Manager and Product Manager. Product Manager is a role within marketing and it's usually a less technical role focused on strategy, vision, delivering products to market, and sustaining business growth long-term.

There are elements of this in the Program Manager role, but another part of it is collaborating with the team that is actually building the product and shepherding them toward their mission. There's a mix of high-level vision as well as day-to-day work with developers and testers to build the product. Here at Microsoft, Program Managers get to own the use scenarios, defining what a feature in the software should do. We get a lot of space and freedom to own the feature, figure out what needs to be done, and make mistakes along the way.

The big deliverable for a Program Manager at Microsoft is the functional spec, defining everything that the feature needs to do to satisfy the user's needs, how it will work, how it will be built, and justifications for all decisions.

Why do you think the Program Manager is important to a team?

The PM is often the jack-of-all-trades, master of none.

being competitive? Are we relevant to the customer? We have PMs on almost every team at Microsoft to ensure that the precious time we spend coding features is what the market wants and needs.

The PM is often the jack-of-all-trades, master of none. We do a lot of talking to customers, either through customer interviews, internal dogfooding, and/or collaboration with user researchers. I once went to Oklahoma City to visit a SharePoint customer. The company was an energy company, and we learned how SharePoint was critical to getting energy to a lot of people. At the site visit, I got to see how our customers were really using our technology and how we could improve our technology to make people's jobs work better.

A team without a PM is going to be very heavily focused on execution but may not know if they are going in the right direction. The big value a PM provides is being a strong customer advocate, helping people gut check: Are we going in the right direction? Are we addressing the market? Are we

What is your experience with making mistakes as a PM?

I make mistakes everyday. Now that Microsoft is beginning to move more quickly, on a more

agile development process, I have the flexibility to fail every day. Because we're doing everything in bite-sized chunks, we can more quickly adapt to whatever is changing around us. That's not a specific answer, but I think it's important. We used to make much larger mistakes and paid a lot for them in the past. Now, as a company moving more quickly, we know it's okay to make mistakes, but we should recover from them more quickly. It's better to make a bite-sized mistake that will only sting a little, rather than chewing your foot off and taking a long time to recover.

How has your role as a PM been changing over the last 7 years?

Some things are constant, such as the position you're in and the way you interact with people; the PM is often known as the "glue" that holds the team together. What has changed is the types of people I work with, the disciplines they're from, and how many of them are there. This changes depending on the team I'm on, the part of the product cycle we're in, and other factors. The particulars of my role have changed, but the general idea of what I do hasn't changed much. And as you become more senior on teams, your scope of responsibility increases, where you become responsible for larger features. PMs at a higher level of the company do the same thing as individual contributor PMs, just at a much larger scale.

The big value a PM provides is being a strong customer advocate.

What specific skills do you think make a good PM and how do you develop these skills?

I break down the PM role into pillars of design, communication, and execution.

1. **Communication:** This is tricky. Anyone can tell whether or not it's working, but it's hard to explain why or how. This is the hardest one to teach. You have to like people. That's very important for a PM. This doesn't necessarily mean you have to be outgoing or sociable. But you do have to be able to orchestrate people and rally them around your cause.
2. **Execution:** Attention to detail. The buck stops with you. Be aware of your accountabilities, who is responsible for what, and don't be afraid to speak up when something needs to be done.
3. **Design:** There are usually rules and constraints to get you started with design. It's very important here to have user empathy, being able to channel your customers and understand their needs. This ends with delivering a functional spec, stating your design and how it will work. In the end, it always comes back to the customer. You have to prove that what you're doing is better for the customer. You are on the team as the customer's advocate.

What advice do you have for pushing your ideas forward as a young PM or even as an intern?

Interns are my favorite, because interns have an immunity card. Interns are immune to politics, to the pressures of the schedule beyond your 12-week internship. If you're a young PM, don't be afraid of your lack of experience. People hired you for a reason, because you show promise, and they want to see you put that to the test. You're not entrenched in office politics, so you should feel free to speak their mind.

The challenge for interns while they're here is to be able to make a big impact, not just on your own team. There are a lot of resources and people here to talk to and toss ideas around with, so make the most of it. How can you make everything happen in just twelve weeks? It starts with being confident and not being afraid to speak up and get stuff done.

What criteria do you use to measure your own success as a PM?

[The PM is] the “servant leader” of the team.

I firmly believe in the concept of the PM as the “servant leader” of the team. You're given the mantelpiece to drive design and requirements, but at the end of the day you need to go convince other people who you are not the boss of to go do the things you want them to do. So every day, I look around

me and ask, “Are people around me happy?” If they're not, especially regarding the things in my control, I might not be doing my job well. Are my developers happy? Are my testers happy? Are my leads happy? If everyone seems like they don't need me, then that's an ideal situation. It means I've done a good enough job selling the idea to them, and they will just run with it and work it out.

Long term, I measure success in terms of the scope of what I'm working on. Beyond just promotions and levels, I'm excited about being able to work on the size of problems and types of problems I'm interested in. Now that I'm working in Xbox on consumer technology, I feel like I'm in the place of my own personal passion, and I think that's a good place to be.

Would you recommend jumping into a PM role immediately after graduation or beginning with another role such as developer or designer first?

This is an age-old question: Who's the better PM, the one who started as a PM or the one who did something else first? In my experience, I've seen both be wildly successful. There's a certain skillset that makes good PMs, and if you have it, then that's all you need to make it here.

I would say it's important to understand other disciplines so that your team can understand that you've been in their shoes. It helps if you have a technical background (basic scripting is sufficient programming experience for most PMs) so that you have a sense of what the team is talking about when you're staring at their screens figuring out an issue. I don't think you

need to have done a ton of programming before being a PM, but it's good to have a little bit of exposure.

Diversity of experience among PMs is important, too, because you can bring your area of expertise, whether it's technical, business, or design, to the team. This creates a wealth of opinions, which means more and better ideas.

Imagine you're at an elementary school on Career Day. How do you explain to the kids what a Program Manager does?

I would tell people that I'm a Jedi knight. Being a PM means seeing your ideas and creations take a life of their own. A lot of what the PM does is getting the ball rolling and seeing everyone that you work with take the ball and carry it over the finish line. It's kind of like Jedi mind tricks, getting people to do cool things without forcing them to do it. Almost like inception.

The work of a PM is kind of like doing Jedi mind tricks.

DAVID SHEIN

Product Manager at Facebook

DAVID'S BACKGROUND

David graduated in 2009 with an undergraduate degree in quantitative economics. He then spent two years at the Kansas Federal Reserve doing macroeconomic research. Most recently, David has spent the past two years working in corporate finance at McKinsey and Company. Although he has a deep interest in technology, he does not have any direct coding experience. He just transitioned from his role at McKinsey to work as a Product Manager at Facebook this August in their RPM program (rotation PM program).

SUMMARY OF DAVID'S INTERVIEW

In his following interview, David discusses:

- Why he transitioned from consulting to Product Management
- How he landed a job as a PM at Facebook without a technical background
- How to excel as a non-technical PM
- And more!

Read on to learn more from David!

DAVID'S ANSWERS

What made you decide on transitioning from consulting to Product Management?

I came about this role in a rather serendipitous way. I wasn't actively looking into becoming a Product Manager, but Facebook had reached out to McKinsey about consultants that might be interested in a potential role at Facebook.

This surprised me because most Product Managers tend to have backgrounds as engineers, designers, CEOs, or COOs. I was able to fortunately secure a position and was hired by Facebook to become a Product Manager.

In your opinion, what are the goals and purpose of Product Managers?

Understand the product that you're dealing with at a very intimate level.

language and be able to engage intelligently with your team members who are from different backgrounds.

In addition to the above, two more points that I would say are heavily important are:

1. Understand the product that you're dealing with at a very intimate level. That means understanding how people interact with your product to a very deep degree. You need to understand how they're understanding the product and their goals and frustrations.
2. Learn how to communicate effectively. You're going to have to come in and rely upon the engineers to do the coding, so you need to make sure you can easily explain what you're trying to get done and what you need to get done.

Ultimately, I think that the primary goal would be to actually ship a product.

As someone who doesn't come from a background of engineering, what are your thoughts on working as a Product Manager at a heavily technical company?

I don't think that my lack of a technical background will hinder my ability to excel. When I was learning more about the potential of the role, I asked myself: Could I really succeed without a technical background? I heard from Facebook that they actually had confidence that I could.

When speaking with other people without a technical background who had joined the program the prior year and were successful the key was not about understanding how the indi-

I think that Product Management roles are about aligning resources, the user experience, a deep understanding of the customer, and the market. That generally means that even if you don't have to have a technical background, you should still understand the

vidual lines of code are written, but learning to speak the language of engineers and designers as well as effectively allocating and leveraging resources across the organization

One thing that was suggested was to quickly learn how to ask the right questions and speak the right lingo.

For example, if Facebook is looking to release a new feature, as a Product Manager, you need to be able to understand when people you're working with start explaining the implementation and design choices behind various components of features. I don't think you necessarily need to understand the detailed technical implementation, but you do need to be up to par in the language and terminology used in the discussion so you can understand and contribute.

None of this is to say that it still isn't great to have a technical background going in. It's just meant to show that it may not be completely necessary to have been an engineer or designer prior to stepping into a Product Management role.

What are some examples of strategies you prepared for securing a role as a Product Manager at technology companies?

Broadly speaking, it was three points. First off, I spent a lot of time learning a lot about the product. I delved into the product both at the front end and back end (e.g., when looking at Facebook Places, what could be added to it and how would it need to grow to help deliver a new and differentiated impact to people). Secondly, I sat back and thought strategically about how I would actually use this product (e.g., how do I use Graph Search on a daily basis). Thirdly and finally, I took a macro-level view and thought about the scope and scale of the product (e.g., what products does Facebook currently not have in the market and what would I add to their product portfolio if hired tomorrow).

Thinking about a product at these three levels, from the highest view to the lowest, will help you understand the product at a deep enough level to make some reasoned and intelligent judgments about the product you're working with.

Quickly learn how to ask the right questions and speak the right lingo.

PAUL ROSANIA

Senior Product Manager at Twitter

PAUL'S BACKGROUND

Paul graduated CS from Dartmouth in 2005. He did consulting for a few years, before starting a business called CollegeJobConnect. It ended up failing, but through the process he got a call from a notable Silicon Valley founder, who had heard of Paul and wanted to see if he would be a good fit to join their startup as the first Growth Hacker.

Paul said yes, moved out to SV in 2011 and spent 1 year working there.

He then got a call from Twitter; he had been able to 10x growth at the startup, doing both engineering and product strategy, but realized that he wanted to move to more of a pure product-related role, and to try his skills on a much bigger stage. So he accepted Twitter's offer and has now been at Twitter about 10 months.

You can follow Paul on Twitter at [@ptr](#).

SUMMARY OF PAUL'S INTERVIEW

In this following interview, Paul shares his insights on:

- How Product Managers are like sports coaches
- What it means to be an advocate for the user
- The 3 most fundamental skills a success PM should have
- The most important thing he's learned as a PM at Twitter
- And more!

Read on to learn more from Paul!

PAUL'S ANSWERS

You've had a very interesting background up to your role now at Twitter. Can you talk a little bit about how you got to where you are now?

I wandered a bit, early on in my career. Prior to Twitter, I worked as an engineer at two startups, founded one, and also spent three years doing management consulting. At the most recent startup, I was the first person to spend 100% of my time focused on growth. As a result, I split my time between product management — brainstorming, roadmapping, prioritizing, etc. — and engineering.

Coming to Twitter, I chose to go all-in on product management. Twitter's full of world-class engineers, and I felt I had more to add on the product side. I'm a Computer Science major by education, but my passion has always been in products, and designing and shipping them by any means necessary. Going into Twitter, I wanted to play to those strengths, particularly product leadership, long-term planning, and doing what it takes to get things shipped. So I decided to do PM at Twitter, initially in Growth. My role at Twitter has evolved over the past few months, and rather than being concentrated on Growth I'm starting to spread out and work on other things around the company.

In your opinion, what are the goals and purpose of Product Managers? What are some of the most valuable skills I could develop in a short-term internship as a Product Management Intern?

Someone needs to spend every waking hour laser-focused on the customer.

ing the two. With project management, your number one goal is to make sure that the thing you're working on actually ships. With product management, your goal is to make sure you're building the right product, with the right set of priorities, in a manner consistent with company goals, while constantly advocating for the user.

At Twitter, we collaborate with Technical Program Managers (TPMs), who are interdisciplinary, but function somewhat like project managers. On our biggest and most complex projects, I collaborate with a TPM. On others projects, I take on project management responsibility directly.

I think this reflects on an important aspect of success as a PM: doing whatever jobs need doing to get your project into your customers' hands. A big part of my job is doing whatever dirty work is necessary so the real work can get done. Often, that means writing email, or repre-

The exact role of Product Management varies from company to company. For starters, some companies combine product and project management into one role. Others split them apart. Those that split them have different ways of balanc-

senting the team in meetings. Whatever makes the process of building software as smooth as possible for my team.

Every once in awhile someone I hear someone ask if PMs are really necessary. After all, I work with a lot of engineers who have a strong intuitive sense for what a good product looks like. But if an engineer asks me, I tend to ask in return, “Do you want to spend your time building? Or do you want to spend your time doing all the planning and coordinating?” A lot goes into shipping products, beyond writing code and having an eye for quality.

When you’re engrossed in building software, you don’t always get direct exposure to users themselves. Part of a PM’s job is to remain connected to the people who live with the product you ship. At Twitter this is particularly challenging — with 200 million active users, it’s really hard to infer what people want, just by using your gut or conversations with a few friends. It’s just not a representative sample.

Make sure that you’re building the right product and advocating for the user.

So it takes some time to develop the right mental frame to start to design our products in the right way, and this frame is a critical aspect of a PM’s value. I think you could fire all the PM’s at Twitter, or at most companies, and not notice anything wrong for a couple months. And then you’d notice what you’re building is losing traction in the market, or you’re shipping features your users don’t care about. Someone needs to spend every waking hour laser-focused on the customer. PM’s keep you honest on that front.

One last thing on being a great PM, by way of analogy. A PM is like a sports coach. You don’t manage the players directly or negotiate their contracts. All of your influence is indirect. Your success, and ultimately your team’s success, lies in your ability to lead. If the players don’t respect you, your job becomes impossible.

Great PMs guide their teams to success, without ever setting foot on the metaphorical playing field. They are the glue that sticks the team together, empowers them to succeed, and keeps them focused on the goal.

As for skills to focus on, I think there are three fundamentals: organization, communication, and thought leadership.

Organization is the key to success. It’s trite, but true. I mentioned earlier that the PM does whatever it takes to ship product, and part of doing that is not forgetting what needs to be done. Figure out a way to organize your thoughts, questions, and to-dos. I’ve spent a lot of time creating a system that works for me, and I refine it constantly. Ultimately, this means my manager can trust me to get things done without oversight, and my team knows when they pass things to me they don’t have to worry about whether I’ll follow up.

Communication helps indirectly. To be effective as a PM, you need to be perceived as a reliable hub for your product. Otherwise, people will go directly to members of your team, distracting them and slowing your project. At a minimum this means prompt, clear and succinct responses to questions. (And a lot of email!) At the next level, this means laying out and evangelizing the medium and long term goals and roadmap for your project. When people know

what you're planning to do and why, they're more likely to leave you alone and let you and your team execute your vision.

Thought leadership means knowing your product and industry cold, and having a clear and passionate vision for where you want your product to go. This might seem like the most important thing, but without organization and communication, you're faced with shoddy thought leadership, or a beautiful roadmap no one knows about or respects.

What do you mean by being an 'advocate for the user'?

Don't lose the broader customer experience.

Let me give you a specific example from Twitter. Think about our signup process for a second. Everyone who uses Twitter can explain intuitively why you need to have a username. Without one, no one could find your profile to follow you, and there would be no way to @-mention you. But if you talk to potential users, you encounter people who have no idea why the signup process is the way it is. What's a username? Why would I need one?

Eventually, you start to identify patterns in user feedback, where your customer doesn't have the same understanding of your product that you think they do. As a PM, it's your job to identify these patterns. Over time, the patterns form into broad intuitions about how users will approach your product, and you surprise yourself by guessing user feedback before you hear it. This is what I mean by advocating for the user. It's my job to make sure that we don't lose the broader customer experience.

As a new PM, your intuition will be raw, but you can overcome that by talking to customers. At first, the patterns you find will be simple: people struggle with passwords. But over time, you start to detect broader heuristics: web forms can never be too simple or straightforward. You start to connect these heuristics to each other, and generalize. Through repetition and generalization, these heuristics build a foundation for intuitive product sense.

What do you do as a PM at Twitter to make sure you're consistently in tune and empathizing with the end users?

There are many tactics, but only one strategy: talk to your users.

We're a little spoiled at Twitter, because we have a dedicated user research team keeping us connected to our audience. I immerse myself in their results whenever I can. This luxury is dangerous though, since it's indirect. As a PM, it's easy to interpret research results abstractly, and start to think of users in terms of cohorts and percentages.

Quantitative data is critical, but it's also critical to stay connected to real humans who use your software. I also volunteer occasionally, helping people with computer-related issues, and there are often questions about Twitter. It's fascinating to watch people do even the simplest things, like logging in. Many people will do basic things like repeatedly type their password

wrong. After a few tries, they might even conclude that there's a bug with Twitter instead of realizing that they've just been typing their password wrong the whole time.

These are legitimate user problems, and it helps to be there with them to witness it first hand. Otherwise it feels too abstract. We're humans after all.

So I think empathy is the right word, but at the same time I don't think you can identify all of these things just by being empathetic.

I envy teams whose customers arrive with credit cards in hand. Paying customers are either happy or angrily calling demanding to deliver feedback to you. It's painful to hear it sometimes, but at least they're giving explicit feedback. I don't ever get that, and I can't just call up a handful of our 200+ million users hoping to get an accurate representation of the user base. So I have a particular challenge understanding the 'average' user. This is where our user research team really saves the day. They conduct studies of all sizes, drilling in on all kinds of problems in a way that accurately represents our massive user base.

What would you say is one of the most important things you've learned as a Product Manager at Twitter?

I used to catch myself asking other people questions when I already knew what they were going to say. Or I would have a tingling sense in the back of my brain that something was off with a deliverable. For example, someone might ask me what the next steps for a project are. I'd write a few bullets into an email, and as I hit "send" I would already know exactly the first question they were going to ask. But I wouldn't write it in, maybe out of laziness. Without fail, they would ask that follow-up question. And all you can think about to yourself in that moment is, "How did I know and why didn't I answer it?"

I encountered this a lot in my early career, but once I became a PM, I knew I had to fix it fundamentally. For PMs, it's a critical failure. You really need to guard against it. You're expected to be the person who notices incomplete work, when you do it and when other people do it. Otherwise you will ship an A- product. You'll know in your heart it's an A- and it will end up being an A-. As a PM, that's unacceptable. The buck stops with you.

The buck stops with you.

You can't count on someone else calling you out if your product is subpar. You were hired to be the arbiter of that, and other people won't even necessarily know the right questions to ask.

What are some of your responsibilities as a Product Manager at Twitter?

I mentioned earlier that organization, communication, and thought leadership are great skills to develop. These are some of the cornerstone responsibilities of a PM at Twitter.

Diligence is another responsibility of every PM. The product you build is a reflection of you and the effort you put into it. That means being honest (and sometimes critical) with yourself,

your team and your work. A good PM wants to do what's right, even when it hurts. That means doing homework and proving to yourself and others that your product is as good as you think it is, and your vision is 'correct'.

For me, this level of diligence was unnatural at first. There's so much stuff that happens each day that your memory gets overloaded easily. I quickly realized I had to stop trying to keep everything in my head. I had a couple instances where I forgot to do something I'd been asked to do, or where I stepped into a meeting and realized I hadn't planned out what I was going to say as much as I thought.

Nowadays I write everything down immediately, even if I plan to do it right away. For simple tasks, I use Things.app on my Mac. When I meet with someone, I jot down notes in Evernote as we talk. As a result, I never forget to do things, and I never forget conversations with team members. When I have product ideas I jot them into a running Google Doc for the project. This system frees my mind from remembering things, and lets me concentrate on organizing and planning without wasted cycles trying to remember things.

I also find that writing helps me keep myself honest: sometimes an idea that seems crisp in your head is hard to explain in prose. That's a sign your thinking isn't yet complete.

A good PM wants to do what's right, not boss people around.

One last thing I want to say is that as a PM, you don't have a ton of control or flexibility over your own time. You're often at the beck-and-call of meetings, email and questions from other people. It's tempting to fight back, but these obstacles are actually a part of the job. Be honest about whether each meeting is important, but also realize that by having distraction-filled days, you're clearing distractions from your team. (And sometimes these meetings help you collect information that keeps you alters your roadmap and keeps you focused on the right goals, which saves time in the long run.)

If you're feeling overwhelmed, don't be afraid to block off time to do long-term thinking. I put one-to-two hour blocks on my calendar a few days a week when I'm busy, to make sure I have time to think. You need time to synthesize your thoughts, and if you don't reserve that time other people will take it from you.

I can program in Python and R, but don't have as concrete of development experience as I've heard many PMs have. As someone who has somewhat of a technical background, what are some things I can do to compensate for this?

Having a strong engineering background helps you as a PM in two ways.

First, it helps you build a strong rapport with your engineers. If you can walk the walk, they're more likely to trust you to make judgment calls on their behalf, and not to overpromise when they're out of earshot. You're also less likely to put your foot in your mouth talking about the details of something you're asking them to build.

Second, having a strong engineering background helps you estimate, which in turn allows you to better coach your team by sniffing out opportunities to push them to new levels of per-

formance. When you work with a really great team, and you know when to push, everyone grows and your team, your company, and you all benefit.

So to compensate for a lack of concrete experience, brainstorm ways to accomplish these goals. Find opportunities to hack on projects, ideally by writing code alongside your team. At Twitter we have quarterly Hack Weeks, which are a great chance for me to stretch myself and prove my mettle to the team I count on.

I'm concerned about the level of impact I can have in just 3 months at a company. I want to learn a lot, and think I will, but I also want to get a chance to actually make an impact. How did you spend your first 3 months in your first PM role, and what would you do differently if you could redo it?

I got my hands dirty right away, and I recommend the same. At Twitter, we always look for intern projects that are substantial in size, allow the intern to really “own” something they can drive to completion on their own (with guidance), and are shippable in the 3 months they’re on site. Oh, and always, always real projects. No practice projects or busywork.

I got my hands dirty right away.

Hopefully your company lays out the same opportunities for you, but if they don’t do it instinctively, speak up! You’re right that there’s a chance you won’t make an impact. But you definitely can, I’ve seen big impacts time and again from interns here. Make sure you’re working on a real thing that’s likely to ship before you leave, that you can point to when you’re gone.

What are some mental models you use to view the quality of an educational product? In other words, what are the criteria you use for judging how successful a product is?

I don’t have specific background in educational software, but I can speak generally: Build a product users love. Talk to them, listen to their feedback. State the problems you are trying to solve clearly, up front, and always from the user’s perspective. Break things down. Focus on high leverage projects first. Ship.

I highly recommend reading *Inspired* by Marty Cagan. It’s a quick read, and it will help you develop a framework for approaching product development.

I've often heard the advice that I should do a “technical” job first for a few years before jumping into PM, because it's hard to switch back to a “technical” job (such as programming, designing) after doing a PM role. What do you think of this advice? Is it a good idea to jump right into the PM role after graduation, or is it better to do a regular role first and move in to the PM role?

The best career advice I ever received was, “you can’t do what you want by doing something

PAUL ROSANIA

else." Do what you love. Worry less about a hypothetical career switch. If your passion really is in product management, you won't leave it.

(Oh, and the world is clamoring for talented designers and engineers. If you need to make that switch later, you'll find work.)

LAYLA AMJADI

Product Manager at Facebook

LAYLA'S BACKGROUND

Layla was most recently a Senior Associate Consultant at The Bridgespan Group, a management consulting firm that serves the nonprofit sector. At Bridgespan she built significant expertise in philanthropy and served as the Product Manager on a groundbreaking philanthropy video series titled "Conversations with Remarkable Givers." The series features original interviews with 60 impressive philanthropists, including David Rubenstein, Pierre Omidyar, and Melinda Gates. Clients from her casework included organizations, such as the United Nations Foundation and Facing History and Ourselves.

Layla has also interned at Procter & Gamble in marketing and at the US Department of State in the Bureau of Democracy, Human Rights, and Labor. She dedicated the majority of her undergraduate career to serving on the national managing committee of STAND, the student-led division of the Genocide Intervention Network, which organizes hundreds of high school and college chapters in to stop and prevent genocide. She was elected as the organization's Executive Director in her senior year.

Layla graduated from Harvard University with a B.A. in Government in 2010 and speaks Farsi and Spanish. She can be followed on Twitter at [@LaylaAmjadi](#).

SUMMARY OF LAYLA'S INTERVIEW

In her following interview, Layla shares her thoughts on:

- How to influence her team without authority
- The most valuable skill all PM's should know
- What is unique about being a female Product Manager
- And more!

Read on to learn more from Layla!

LAYLA'S ANSWERS

Can you talk a little bit about your background and what led you to product management?

I graduated from Harvard in 2010 and had spent the majority of my undergrad working with a non-profit organization called STAND (The Student-Led Division of the Genocide Intervention Network). STAND mobilizes hundreds of college and high school chapters on genocide prevention advocacy.

While at STAND, I served as its National Programming Director in my junior year and the Executive Director in my senior year. Throughout it all, I had the opportunity to work on campaigns, websites, and events in a cross-functional role. I really enjoyed being able to work with policy experts and grassroots organizers on one day, and web developers the next for any given initiative. It was challenging, exciting, and I was always learning from my teammates.

I came to really love the PM role as I had come to understand it.

So when it came time for my internship before my senior year, I went looking for a role that would emulate that cross-functional, hub-of-the-wheel experience, which is why I pursued brand management at Procter & Gamble. At P&G I got to work with sales, R&D, PR, agency partners, business analysts, and point-of-purchase experts, so I

definitely got the cross-functional experience I wanted, but something didn't click for me. Part of it was that I wasn't necessarily connecting to my product (cleaners) or "her" (40 year old moms), and another part was that P&G was a hierarchical environment that was a bit slow moving and risk averse. I was the only undergraduate intern in the program, so I had plenty of MBA intern mentors over that summer, and they recommended that I look into management consulting to expose myself to different types of organizations and really build up my quantitative and strategic background. I took their advice and ended up going to work for The Bridgespan Group, a non-profit consulting firm that was incubated at Bain & Company and co-founded by Bain's former Managing Director. Interestingly, it was this consulting job that opened the opportunity for me to work in a product management role.

Bridgespan was given a large, multi-year grant from a prominent foundation to develop the Give Smart initiative, an effort to support philanthropists to give effectively and accountably. Part one of this initiative was the Give Smart book. Part two was an original video series showcasing the wisdom of 60 high-net-worth philanthropists from around the country. We developed a TED Talks-like content platform with about 1,500 video clips, featuring 1-3 minute sound bites from philanthropists like Melinda Gates, Michael J. Fox, Julian Robertson, Tom Steyer, and David Rubenstein on a range of topics. I was more-or-less a Product Manager on the video project.

So ultimately, through helping to build and populate this platform over the past 1.5 years at Bridgespan, I came to really love the PM role as I had come to understand it. Now I'll be

transitioning onwards to starting as a Product Manager at Facebook in August.

How did this transition happen?

As an organization grows, it can begin to take more risks on the types of people it hires for any given role. I saw this in my consulting cases. I can imagine as Facebook has grown there's been room to hire more people from different backgrounds for the PM role. I can see how having a management consulting background as a PM would be helpful, because I think, at its core, it's a strategic role.

If you were standing in a first-grade classroom and had to explain to the students what a Product Manager is, what would you tell them?

A PM works with a team to decide what a puzzle will assemble to look like when it's completed.

I would use the example of a puzzle and say that a Product Manager works with a team to decide what a puzzle will assemble to look like when it is completed. Then the Product Manager works with the team to break it apart and jumps in whenever necessary to help teammates with their pieces. The PM then focuses

on working with the team to put the puzzle back together in a timely and efficient manner. From my experience, as a Product Manager, you help your team decide the best way to spend their time and resources on a project and keep everyone pumped along the way.

From your experience at Bridgespan, what were some of the most valuable things you learned about succeeding as a Product Manager?

The three major takeaways from my first PM experience were:

1. **Avoid “nice-to-haves”:** As a PM, part of your job to make sure that the team is and feels like they are using their limited time efficiently. This means avoiding workstreams that are “nice-to-have’s” that won’t necessarily change the the “answer” or the direction of the product. Understanding what is a “nice-to-have” requires going through the hypotheticals. If a proposed piece of work could lead to “X” or “Y” results, but we also know that regardless of “X” or “Y” result, we’d still do “Z”, then that research might not be worth the time. It doesn’t change the answer. But you have to weigh the pros and cons and balance efficiency with efficacy. Sometimes a workstream might not immediately impact a product in the short run, but could build knowledge that will be useful down-the-line, or it could be a way to satisfy the team’s intellectual curiosity. So its not always cut and dry. There are tradeoffs.
2. **Influence without authority:** This is a key part of being a Product Manager, and is admittedly a very difficult thing to learn how to do well. As a PM you are coordinating a team of peers. You are working with other cross-functionals in the organization who might not

even have a time allocation for your product. You're not the boss. But, you are responsible for executing the vision. So how do you navigate this? When I was working on the Give Smart video project, this happened very naturally (it's important to be genuine) but I built very strong, one-on-one relationships with each of my teammates on the project and throughout the organization. I came to understand how important the project was for the person given all the other work they have going on. You have to remember, just because you are working 100% on something, doesn't mean everyone else is. So you have to understand the full set of projects someone has on their plate so you can help them prioritize. I came to understand their likes, dislikes, what motivates them, what frustrates them. Having a really comprehensive understanding of the "John the whole person / professional" not just "John the PR expert I need something from" is the only way to be successful in "Influencing Without Authority." You can't just think about what's in it for you all the time. What's in it for them? What do they need? What do they want to learn? Having this mindset is key to being successful in a working relationship where the person you need something from doesn't have to do it.

-
3. **Overinvest in communication:** Whenever I join a new team, I overinvest in communication at the beginning. This isn't because I'm trying to micromanage the situation. It's because I'm trying to sync up our brains. I want to understand my team member's thought processes, biases, values, and expertise. I want to get to a point where I can anticipate what each team member's stance would be on a particular decision or what area of the decision they would definitely want to weigh in on. I get to this point by constantly asking "Why?" This way, I can understand their points-of-view and if they happen to not be in the room at some decision-point, I can still represent their thoughts and advocate for them. This does two things -- it builds trust between me and my teammates, and at the same time it increases efficiency on the backend. Trust amongst teammates, I believe, leads to faster, better results. To illustrate, if John is going into a meeting with Jen about X decision, and I know John knows how I think and the calls I'd make, I don't need to be in the meeting, and I can repurpose my time in a way that would better benefit the team's needs.

You're not the boss. But you are responsible for executing the vision.

What are some things you would say are in common with what you did at a non-profit like BridgeSpan and what Product Managers at technology companies might do?

At Bridgespan, I learned a very valuable skill that I think all Product Managers need to learn to do: how to carefully say "No." This was very difficult for me to get comfortable with, especially since I was the most junior person on the team and managing-up four senior leadership members.

One story that illustrates this is when a partner wanted a very specific video-feature for the website. He was very adamant that having this feature was a crucial part of the experience.

But my team and I didn't believe that we had the capacity to execute on the newly proposed feature, and we didn't think that it would dramatically change the user experience. It wouldn't

"change the answer." So through bringing the facts to the table (i.e., this is how many resources it would take and this is how many resources we have available), we were able to objectively say 'no.'

I think this story shows that as a Product Manager, you're likely in a high-energy place and there are probably many people around you who are highly-skilled and have many great ideas. But as a Product Manager, one of your jobs is to promote and feed this excited energy, but channel it effectively, which will most likely require saying 'No' to some of these ideas.

Yet there's nothing less empowering than saying 'No' to people all the time. To counteract this, one of the most valuable things you can do at the beginning of any project is to work with your team to develop a set of project priorities. As a team, you can evaluate new ideas against this framework and decide in an objective fashion which ideas should be pursued. You give everyone the "saying no" responsibility in this way.

All PMs need to learn how to carefully say "No."

Thinking about this from the other side, I also learned how to make it clear that I was keeping track of everyone's ideas. We had a team whiteboard that contained an idea bank that everyone contributed to. Having this be a public, visible bank of ideas showed my team that I hadn't forgotten what they had mentioned, and that I was actively looking for opportunities (i.e., time and resources) to make their ideas happen. You need to show people that you are listening to their ideas, even if you have to say 'no' for the time being.

Do you think there is anything particularly unique about being a woman in Product Management? Do you have any advice you would give to women who aspired to work as Product Managers?

Honestly, I've never felt evaluated by my peers on the axis of 'Woman' v. 'Non-Woman.' I think the bias that I felt a lot more was more along the lines of 'Technical v. Non-Technical.'

In terms of advice I would give to others, I think that being a woman should be embraced as an advantage. As a woman, you have a unique set of traits and abilities. You're naturally detail oriented and zoned-in on people's thought processes. These are strengths to be leveraged, and I've never felt disadvantaged because I'm a woman. Finally, many technical products have user bases that are > 50% female. Having a woman's opinion in the room is important and valued.

You mentioned that you feel pressure along the 'Technical v. Non-Technical' axis. As someone who doesn't have a technical background, what are your thoughts on doing Product Management at a very engineering-driven company like Facebook?

I think that whatever job you're in, you should always play to your strengths and try to surround yourself with those who can educate you and compensate for your weaknesses.

My strength is that I can coordinate the execution of a project with a timeline, and I've been told that I can keep people excited along the way, motivating them to do their best work. As a choreographer, dancer, and amateur interior designer, photographer, and artist, I have also developed a good gut for design and flow. So I'm confident and comfortable in these areas. Facebook has such strong teams of engineers that I'm not worried about not having a technical background. Plus, as a PM you're not telling engineers how to do their job, you are unlocking their knowledge and bringing it to bear when its time to make decisions.

What advice would you give to someone looking to work as a Product Manager coming from a non-technical perspective?

Do your homework; be a sponge. Go to extra professional development opportunities to get up to speed and make sure to become as knowledgeable as possible. Be a learner and respectful. Tell the team you work with that you want to learn from them and you want to grow from working with them.

Keep in mind that you're the coordinator, not the boss of the people you're working with. Go into your role with confidence and humility.

Given that you'll be starting at Facebook in August, what have you done, and what will you be doing to prepare for your new role?

I think that it's good to have terminology down for someone who's not as deep in the tech world. So keeping up to date on the lingo by reading tech blogs and magazines is really important. I've been also deepening my knowledge about the tech ecosystem and technologies that are used.

Finally, I've also been an avid user of the product. I've been building my gut intuition as a user and having specific anchor references in my head that I can pull up in real-time during meetings.

Surround yourself with those who can educate you and compensate for your weaknesses.

So that means I've been spending a lot of time on Google+ and Path and various different messenger apps to survey the field and being cognizant of the designs that these apps chose compared to the choices Facebook made. It's very helpful in orienting yourself to understand the context these other apps are set in and the tensions they face.

Are there any final thoughts you'd like to share?

Ultimately, my long-term goal is to bring technology and innovation to the problems the non-profit sector works to solve, because so far scaling nonprofit solutions to social problems has proven to be very expensive. Technology can provide that platform for scale. And I also just

have a hard time believing that scaling nonprofits, one-by-one, to \$20-30M revenue ceilings is going to change the world. All sectors (public, for profit, nonprofit) need to bring their unique assets to bear on a problem. For example, CVS is a network of brick and mortar stores in a range of communities across the country. CVS is a platform for scale. They rolled out Minute Clinics faster than any community health organization could have rolled out their clinics.

There aren't enough technical people in nonprofits.

And they are doing their part to increase access to lower cost options for health care. And it helps their bottom line. Win, win, win. What attracted me to Facebook is that it's also a huge platform for scale, they only difference is that users get to decide what they'd find most beneficial to roll through the network.

There aren't enough technical people in nonprofits. And there aren't enough people who are willing to take risks and do something totally out-of-the-box. If more people get into nonprofits with a technology background, I believe that more solutions will reach more people in increasingly cost-effective and innovative ways.

AVICHAL GARG

Product Manager at Facebook

AVICHAL'S BACKGROUND

Avichal is a product manager on Facebook's Platform team working on a new, undisclosed product and managing the teams responsible for user facing platform products such as Facebook Login, social plugins (the Like button on 3rd party sites), and Timeline Collections. Previously, he was co-founder and CEO of Spool (acquired by Facebook in 2012), co-founder and CTO of PrepMe (acquired in 2011 by the Daily Mail Group, LON: DMGT), and was a product manager at Google in Search Quality and Ads Quality as well as started Google Transit, one of Google Map's core differentiating features. He has an M.S. and B.S. from Stanford University.

SUMMARY OF AVICHAL'S INTERVIEW

In his following interview, Avichal goes over:

- What it means for a PM to be analytically skilled
- The most unique skill that he has seen a PM have
- How to develop great product sense
- And more!

Read on to learn more from Avichal!

AVICHAL'S ANSWERS

In your opinion, what are the goals and purpose of product managers? What are some of the most valuable skills that a I could develop in a three-month period as a Product Management Intern?

I think that Product Managers play an editorial role on the team. There's a lot going on when you're trying to get a product out the door, and someone needs to play the editor. Given this perspective, I think one of hardest things about the role is figuring out, given your time, what are you getting done and why?

I've heard from many of the other PM's that I've spoken with as well that another important role of a PM is motivating team members. Is this something that strikes you as true for you?

You do have to motivate people in addition as well. In my opinion, the vehicle for motivation is the product vision that you're pitching. You know that quote about leading people to build a ship?¹ I think that really applies here for Product Managers and pitching a product vision to the team.

In terms of most valuable skills, here's an exercise I would suggest doing: first, enumerate what skills you need (technical understanding, ability to evaluate tradeoffs, analytical skills, design, persuasion). Then, you should think about all of these and go over all what you feel like you lack and what you can learn in three months. I would say that the easiest skill to pick up would be the analytical bits. It's hard to become much more persuasive or a better designer in three months. It's also harder to become a way better engineer in three months.

Enumerate all of your skills and figure out what to double down on.

So to recap: enumerate all of your skills and figure out what to double down on and what you're interested in improving.

In your opinion, what does it mean for a PM to be analytically skilled?

Given a problem, can they (a) understand the problem? Can they (b) ask questions that have simple data answers that lead you down the path of being able to solve that problem? (That includes asking for simple metrics, quick metrics.) And finally, can they (c) generate a clear hypothesis beforehand?

¹ The quote Avichal is referencing here is by Antoine de Saint Exupéry — “If you want to build a ship, don’t drum up people to collect wood and don’t assign them tasks and work, but rather teach them to long for the endless immensity of the sea.”

I can program in Python and R, but don't have as concrete of development experience as I've heard many PMs have. As someone who has somewhat of a technical background, what are some things I can do to compensate for this?

I think you should try to become more technical while you can in school. Generally, you become less technical over time, so hedge against this. Being technical is not a hard prerequisite but it is incredibly valuable.

The other direction you could go is to become more design-oriented. If you're a great designer, you can also become a great PM. In general, you tend to see people who are successful in one of those two roles.

Neither design nor engineering has an end state.

Something to keep in mind is that design doesn't have an end state. In other words, there is no point at which you just stop learning design. It's a continual iterative process that you have to go through to learn how to become a good designer — so start early! What you tend to see is that designers tend

to get better as they get older. This is just as true for engineering.

A lot of people don't think of skills such as design or programming as lifelong skills; they think of things as thresholds. "If I'm 'good enough,' then I can stop" is the thought process. But really, you have to keep on getting better and better at all of these things.

Another thing to note is that design tends to be easier than engineering to do outside of school. So my advice would be to study engineering in school, and on the design end you can be self-driven to maintain.

What are some mental models or patterns that you've seen that make a high-quality product?

If I had to mention one thing, it'd be that I think that high-quality products don't try to do too much. They do one particular thing, and they do it exceptionally well. And that's the core of their value proposition. And I think a lot of times, if you're not absolutely amazing at one thing at a product, but instead spread across many different domains, then you tend to be replaceable and not have lasting value.

Look at Google, for example. Google at its core is still a search company. It took them many years to get other products out the door, and that's something to take note of.

What is the most unique skill you've seen a PM have that made him or her exceptional?

I think if I had to try to pinpoint it, it would boil down to respect. When you're really good, it's really easy to develop a big ego. But the best PMs I've worked with don't have big egos. They're not trying to be smarter than everyone in the room or put anyone down. For example,

it's easy to blow junior people off when you're an executive at a big company, but the best product leaders treat everyone with respect and humility. That's something that just resonates with people and makes them want to work for you that much more.

One trait I've heard mentioned before as being important to a Product Management position is having product sense. In your opinion, how does one develop great product sense?

It's a very hard thing to learn. I'm not even 100% sure if you can effectively learn it. Everyone who has had this skill seems to have it innately. If you can synthesize human motivation and human desire into a tangible thing (e.g., where the buttons go, what you name the product) then you're on the right track. Maybe you can develop this through an extreme attention to detail. Dissect what makes something good and what makes something not good. You can learn from that process.

The best product leaders treat everyone with respect and humility.

I guess as a whole, people who are good at noticing the little details tend to be able to pattern match (e.g., what makes many other products great will probably make this one great) and develop the product intuition necessary to be a great PM.

What are the metrics you use in measuring your own success? Which ones are the highest priority and why?

The most important metric for me is freedom: "Where can I have the most long-term freedom to learn, be happy, and have an impact?" For me, that's turned out to be having a couple of startups before my current role at Facebook. This path has given me both financial freedom and a skillset that gives me tremendous freedom inside a company that values innovative, creative problem solving (which Facebook values tremendously).

How much freedom do you feel you have in your Product Management role?

I think that the PM position is pretty high in terms of the amount of freedom you get.

Keep in mind that different companies also give you different flavors of PM.

Facebook has a bias towards PMs who are strong in design or analytics. If you look at the backgrounds of people who work at FB, there are a lot of people who have diverse backgrounds, such as in design or data analysis. Facebook is extremely data-driven and as a result, even a design-driven PM here would probably need to have more of a background in analytics.

It's also important to note that in terms of the difference between the top 0.1% of PM's, you would find that the best of the best are highly skilled in both of these domains.

I've often heard the advice that I should do a "technical" job first for a few years before jumping into PM, because it's hard to switch back to a "technical" job (such as programming, designing) after doing a PM role. Do you think it's a better idea to jump right into the PM role after graduation, or should I do a regular role first and move in to the PM role?

I think it's fine either way. It's really a matter of whether you think you have the requisite background in technical abilities. It's okay to jump into PM directly if you feel comfortable with your background as either a technical individual or a designer. Honestly, a lot of times, people coming out of school tend not to have the requisite skillset yet. So if this is true for you, you should really go build some of that skillset and get good at those areas before going into a Product Management role.

For students who are interested in Product Management, but who might also feel like they are lacking some important skills, here's what I would suggest: if you can get a PM job at a tier-one organization (e.g., IDEO, Apple, FB, Google) to synthesize product work: go do it.

If you can't, try to enumerate what gaps you have in your skills and fill those in. Maybe it's engineering, or design, or something else. Build these skills with some time, and then take another shot at it. That's not to say that being a PM at a tier-one large company is an end-state, but it's a way of benchmarking yourself. It's a litmus test of whether you have these skills.

It's probably the closest you can get to an SAT. So that might be a good proxy for whether you should go off to develop deeper skillsets or not.

The most important metric for me is freedom.

ACKNOWLEDGMENTS

CARL has been fortunate to receive so much help from the numerous people who assisted in the creation of *The Product Manager Handbook*. The feedback many of his close friends gave ended up being invaluable in guiding the direction this handbook took. He would especially like to thank Chloe Lim for thoroughly editing the handbook and Gerson Abesamis for giving thoughtful feedback on initial designs. He would also like to thank Gayle Laakmann McDowell and Jackie Bavaro for contributing their thoughtful advice and feedback.

He would like to thank Brittany Cheng for agreeing to help with his wacky idea of compiling a Product Manager Handbook. Without her sharp design skills and patient feedback, this handbook would never have seen the light of day.

Finally, this handbook would not at all have been possible if it wasn't for the generosity and thoughtfulness by each and every of the successful Product Managers he interviewed. So to them, Carl would like to express his sincere appreciation.

BRITTANY would like to thank the many individuals who took time out of their busy PM schedules to speak to her and Carl about the work of a PM. She would also like to thank Edward Cheng for his design feedback during the initial development of the handbook layout. Finally, she would like to thank Carl Shan for his initiative, organization, persistence, proactiveness, and daily onslaught of emails.

ABOUT CARL & BRITTANY

CARL SHAN is a lifelong student, mentor, teacher and aspiring Product Manager. With Brittany's help, he compiled this handbook to share what he learned through his conversations about Product Management with as many people as possible. He is also the co-founder of CompassPoint Mentorship, a national not-for-profit educational venture that matches high-school students up with college students for peer-to-peer mentoring.

This summer, Carl worked as a Mobile Product Manager at Pearson in New York City. His work led him to being nominated & selected by company executives as Pearson's 'Intern of the Year.'

He currently studies Statistics at UC Berkeley.

You can get in touch with him at carl@carlshan.com

BRITTANY CHENG is passionate about technology, design, and education. Since taking her first graphic design class in high school, she has not stopped designing. The PM role excites her because it combines her love for design and her interest in technology. She is a co-director of Berkeley Innovation, a human-centered design student organization at UC Berkeley, and manages up to seven student design projects each semester.

This summer, Brittany was a Program Manager Intern at Microsoft. She worked with designers and developers to design, spec, prototype, and implement new features and experiences for Movie Moments, a Windows 8.1 app.

She currently studies Electrical Engineering & Computer Science at UC Berkeley, with an emphasis in human-centered design.

You can get in touch with her at bcheng42@gmail.com



Softwares for all

You tools you need can get expensive. This helps

It's unavoidable that you're going to need to use some tools to be effective at Product Management. While individual softwares don't break the bank, they will when you add them up. Attached are links to get extended trials for a variety of popular tools used everyday by Product Managers



inVision

Used for: Prototyping, Mockups

Deal: Free 3 months

Get

Sprint.ly

Used for: Agile Project management

Deal: Free 3 months

Get



Marvel

Used for: Prototyping

Deal: Free 2 months

Get

proto.io

Proto.io

Used for: Mobile prototyping

Deal: Free 3 months

Get

Crazyegg

Used for: Heatmapping

Deal: Free 3 months

Get



pidoco
POWERFUL PROTOTYPING

Pidoco

Used for:

Wireframing & Prototyping

Deal: 50% off lifetime

Get

Intercom on Product Management



Intercom

Intercom on Product Management

Some helpful thoughts on the challenges of building software products from the team at Intercom.

Our software enables internet businesses to see who is using their product and makes it easy to communicate with them through email and in-app messages

www.intercom.io

We also regularly share our thoughts on product management, design, startups and the business of software.

blog.intercom.io

Cover and chapter illustrations: Quentin Vijoux

Hooked images in Chapter 1 are courtesy of: Nir Eyal,

Author of “Hooked: How to Build Habit-Forming Products” Blog at: NirAndFar.com

Readers on subway photo in Chapter 3 by Alfred Lui on Flickr

© 2015 Intercom Inc.

ISBN 978-0-9861392-0-8

We'll spare you the legal mumbo jumbo. But please don't share this book or rip off any content or imagery in it without giving us appropriate credit and a link.

TABLE OF CONTENTS

Introduction

Evaluate your product

When to say no to new features

Which new features to build

Getting that feature used

INTRODUCTION

At Intercom we believe a great product should be the first focus of every startup.

Product management - a job which has elements of engineering, marketing, research and project management - is key to building great products. It requires a relentless focus on cohesion; ensuring that how a product is designed, engineered, named, branded, and marketed are all united under a single vision. It is still a young and rather ill-defined discipline, due to the constantly changing nature of software products. Some argue it's not even a distinct discipline these days. In the startup world a lot of people become PMs by default rather than design.

Regardless of what route you came by, or what your current role is, if you are a product person we compiled these articles for you.

We've been writing about our product lessons and experiences on the [Inside Intercom](#) blog for just over three years. This short book connects the most relevant posts on product design and management from that body of writing so that others can hopefully fast track their development. Regular readers will note that we added a few sections to connect ideas or bridge gaps between articles.

We'll start by evaluating your current product, looking at what parts of it are being used and what areas are ripe for improvement. In Chapter 2 we'll look at the hard choices that surround building new features and why "no" is the most important word in the product manager's vocabulary. Chapter 3 will give you a checklist that every new feature has to pass and discusses how to roll them out. We'll finish by looking at how you get those new features used by your customers. Because at the end of the day shipping code means nothing unless it is used and valued.

Let's jump in.

CHAPTER 1

Evaluate your product



Let's start at the beginning. You're a product manager? A startup founder? CEO? CTO? Whatever your job title let's assume you're making the decisions about your product. What features to develop, when to develop them, how to get users to start using them - that kind of thing.

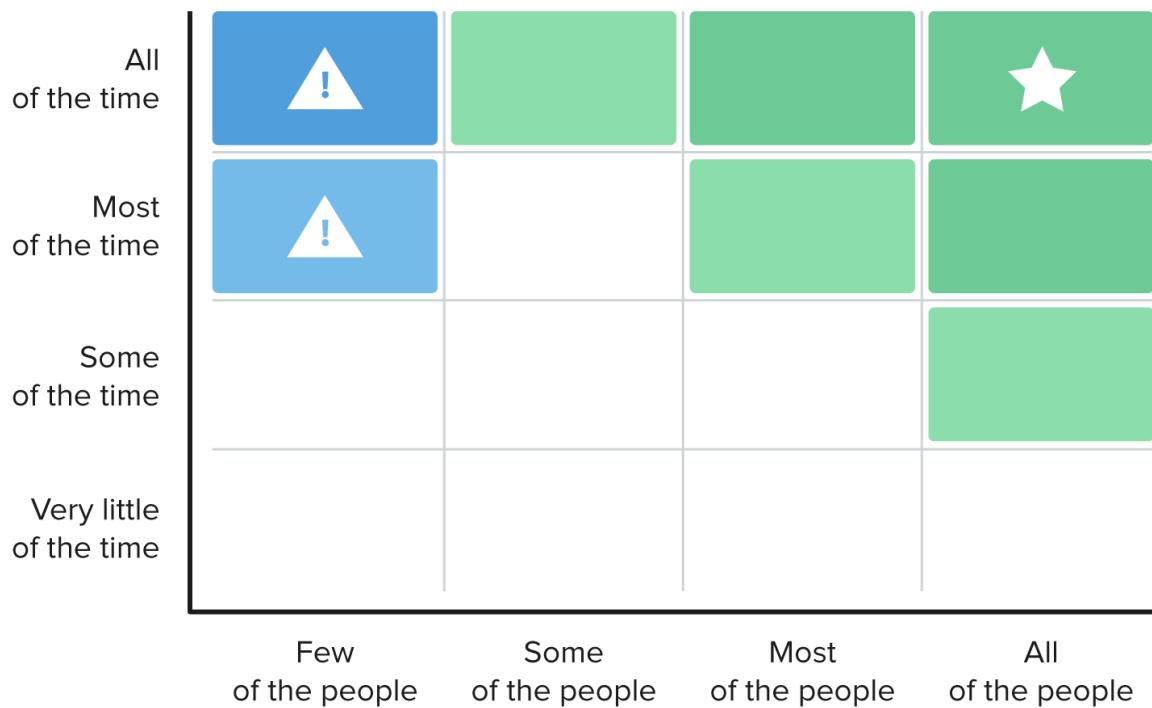
As product people we all have great hopes and dreams for our products and how they are going to fare once they hit the big bad world. You're probably filling notebooks full of ideas that will make you the next Uber, Slack or Facebook. But before you start creating a product roadmap that's going to lead to greatness it's time to take stock and see what's currently going on in your product.

What are people actually doing in your product?

Are all users using all the features in your product? Of course they're not. Let's start by talking about that.

When planning your roadmap, and where your team spend their time, it's useful to ask "how many people are actually using each of our product's features?". This is product management 101 and will probably take you a few minutes of SQL, or a couple of seconds of [Intercom](#). A simple way to visualize feature usage is to plot out all your features on two axes: how many people use a feature, and how often. You'll most likely see trends like this one.

THE TYPICAL FEATURE AUDIT



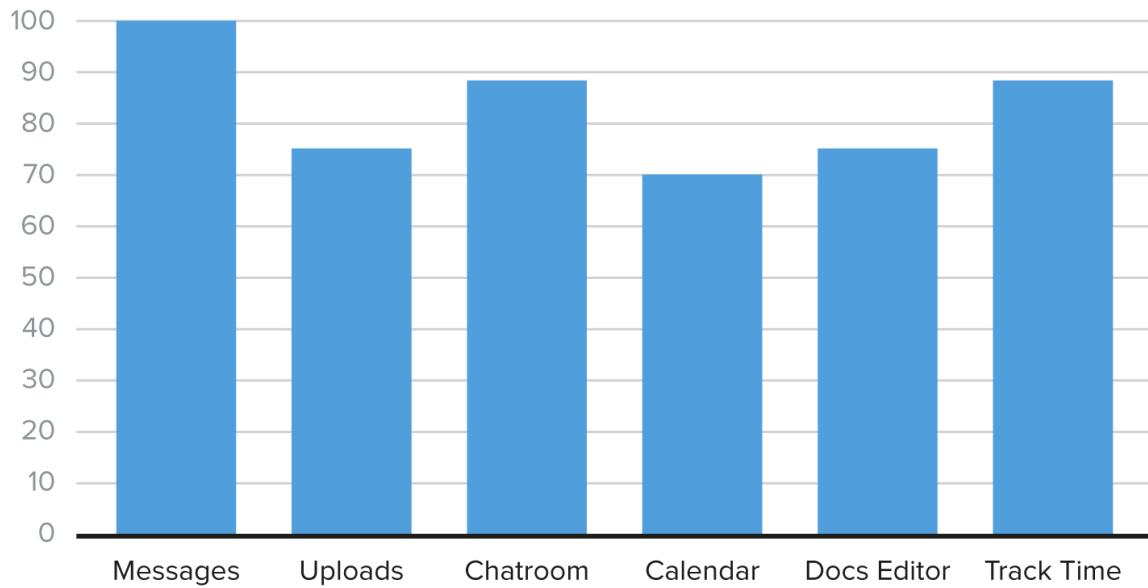
The core value of your product is in the top right area, up where the star is, because that's what people are actually using your product for.

Sidenote: Exclude administrative features like account creation, password reset, etc. from this exercise. They're not relevant here. Also, exclude features that only certain users (e.g. enterprise customers) can access. They should be evaluated separately.

If you have features in the top left it's a sign of features with poor adoption. In other words, there are a small amount of customers depending on this feature, but most rarely touch it.

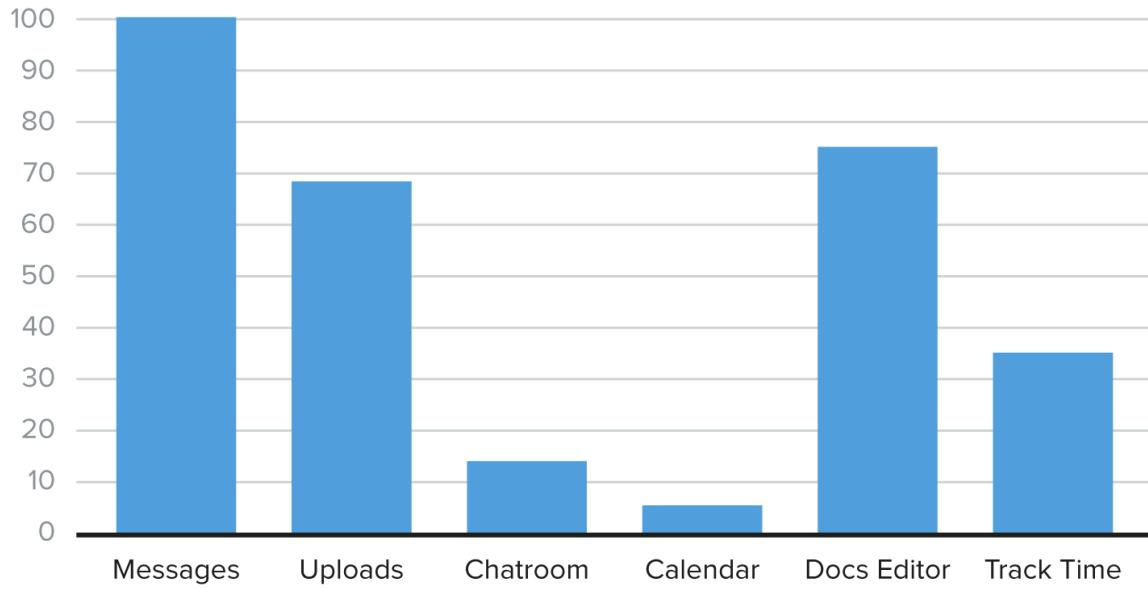
An even simpler way to think about it is this: What percentage of your customers or users have adopted each feature? You can do this with simple bar charts. Shown here is a dream product. The one we all think we're creating when we have Photoshop open. All my users are going to use and love all these features, right?

THE IDEAL FEATURE USAGE



But as every product manager discovers, the messy reality of a product looks a lot more like this.

THE TYPICAL FEATURE USAGE



How did you get here? You built a product that had solid messaging, files and document editing features. These key features were highlighted on your marketing site, documented clearly with great screenshots, included in your product tour, and every new sign-up loved them.

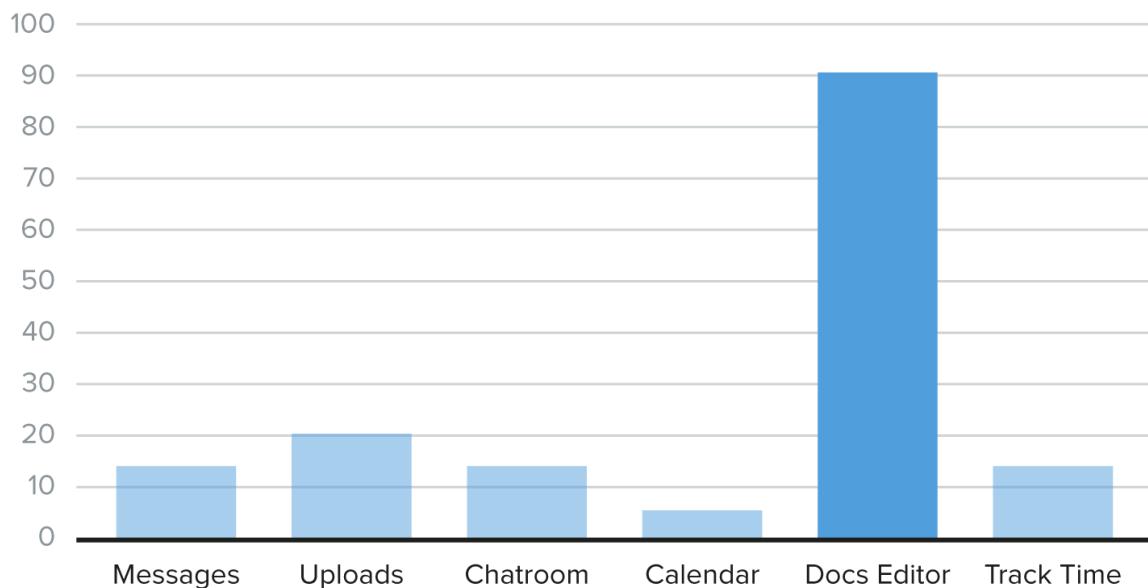
But success can be a lousy teacher. You believed you could do no wrong and that you could build lots more.

So you added a chat room, but it didn't go so well, you kinda botched the launch of it. Then you added a calendar and that went worse. No one created more than one event and it's still not even mentioned on your marketing site. Now you've built this time tracking features that's kind of popular, but only with a certain type of user.

This is your product, you need to fix this.

A sidenote on disruption

USAGE THAT SIGNALS POSSIBLE DISRUPTION



If you're looking at this sort of feature breakdown, you have an excellent product for one precise workflow, but you've added all these other pieces no one

uses. Think of Hangouts inside Google+ as an example here, or primitive document editing (i.e. **BUI**) inside Microsoft Word.

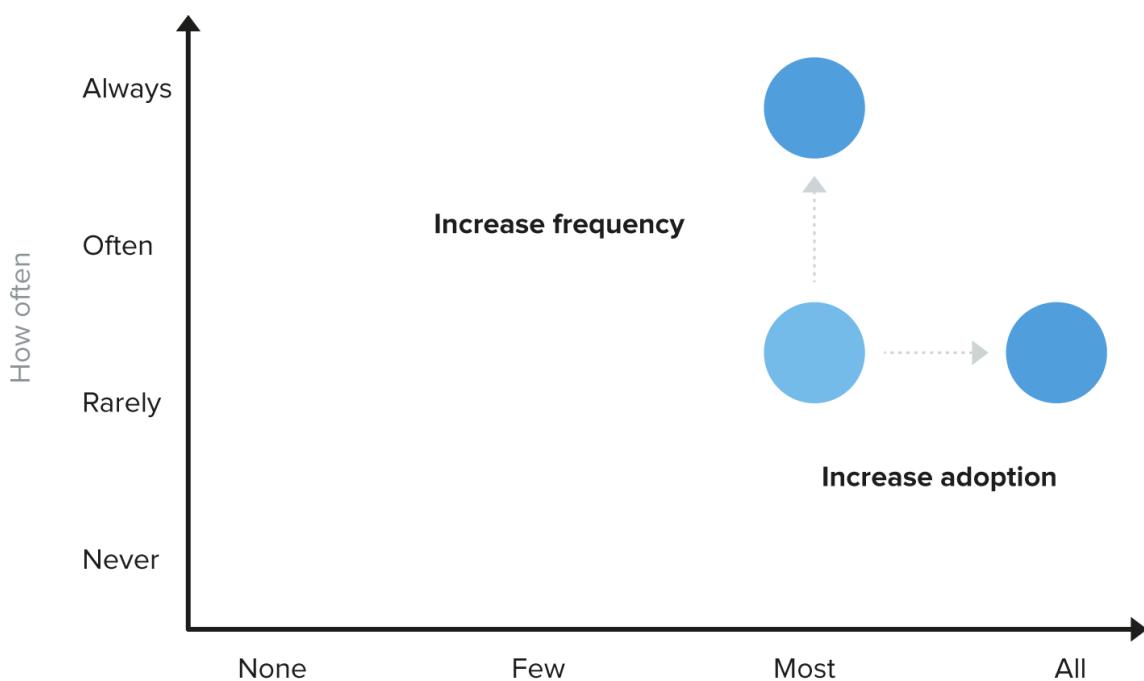
If you are looking at a chart like this you are vulnerable to disruption, in the true Clay Christensen sense of the phrase. Someone can build a simple product, focussing on that one key feature that's superior in just one way (cheaper, faster, collaborative, easier to use, mobile etc.), and you'll struggle to compete, because you're carrying all those other junk features around too.

What do you do with your feature audit?

For any given feature with limited adoption, you have four choices:

- Kill it: admit defeat, and start to remove it from your product.
- Increase the adoption rate: get more people to use it.
- Increase the frequency: get people to use it more often.
- Deliberately improve it: make it quantifiably better for those who use it.

Roughly, you can visualize it like this:



How to improve your features

Kaizen is the philosophy of continuous improvement. Web businesses searching to find product/market fit all follow some variation of *Kaizen* whether they know it or not.

Shipping code doesn't mean that you're improving anything. Similarly, you can make undeniable improvements to parts of your product and get no response or appreciation for it. It all comes down to the type of improvements you're making.

The two most popular ways to improve a product are to add new features, or to improve existing ones. First we are going to look at ways to improve existing features before moving on to new features in the next chapter.

Improving existing features

You can improve an existing feature in three different ways:

- You can make it better (**deliberate improvement**).
- You can change it so customers use it more often (**frequency improvement**).
- Or you can change it so more people can use it (**adoption improvement**).

1. DELIBERATE IMPROVEMENTS

This is when you know why customers use an existing feature and what they appreciate about it. A deliberate improvement seeks only to make it better in ways that will be appreciated by the current users. For example making it faster or easier to use, or improving the design.

Use deliberate improvements when: there is a feature that all your customers use and like, and you see opportunity to add significant value to it.

It's worth noting that deliberately improving a well-adopted frequently used feature is high risk high reward. As an example think about improving the editor in a blogging platform. Get it right, and every single user gets the bene-

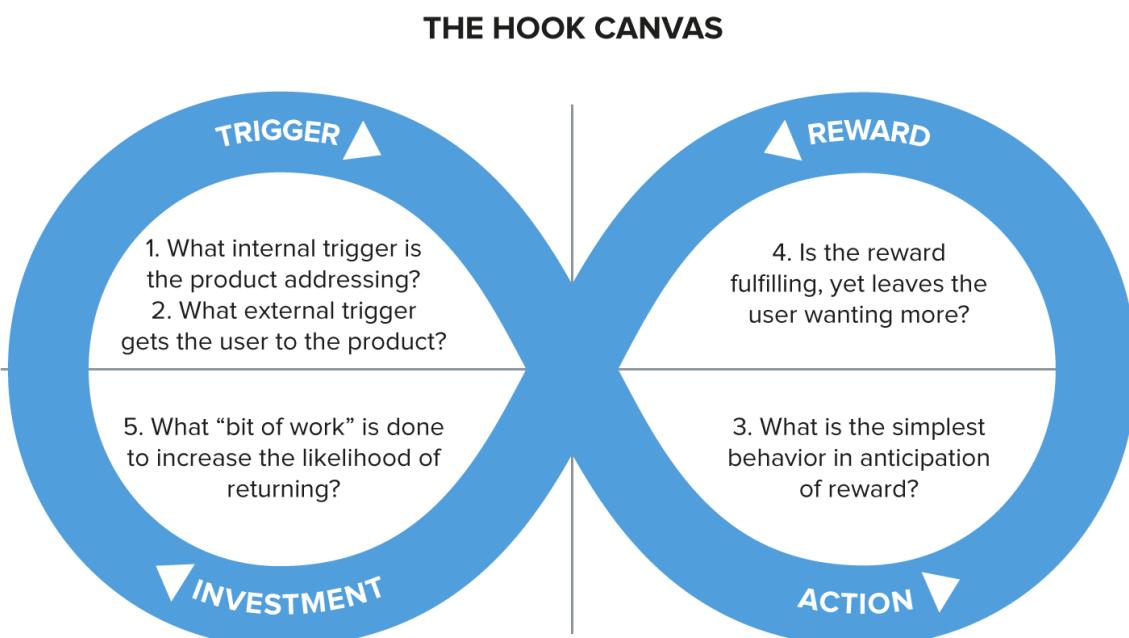
fit every time they use it. Get it wrong and you've broken the workflow of your entire userbase. High risk, high reward.

2. FREQUENCY IMPROVEMENTS

These are improvements that hope to get a customer to use the feature more often. Adding more items to an activity feed, or more options to a search tool means that people read it more often, or use it for more tasks each day. This type of improvement can turn a once-a-week feature into an every day feature.

LinkedIn endorsements do this quite well. They added in these one-click multi-directional endorsements where you can easily (dare I say, accidentally) endorse four of your friends for skills they don't have, and accidentally connect to four more people as a result, which in turn creates more endorsements, more logins, and more single click broadcasts. A vicious circle.

What they're following is a pattern explained by [Nir Eyal](#), author of Hooked, who says habits are formed from a repeated pattern with four key elements...



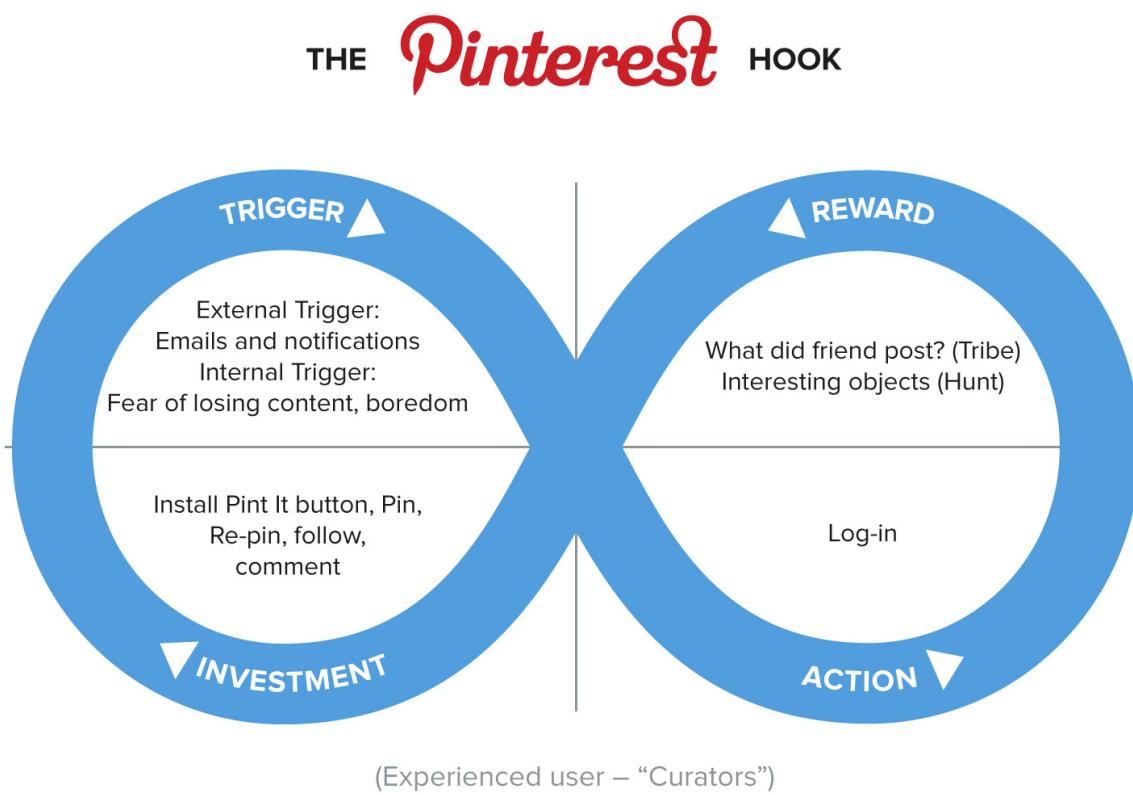
Trigger: the reason the user goes to the product (e.g. you received an email to say a contact had endorsed you for a skill).

Action: they take in anticipation of the reward (e.g. scroll, search, browse, etc).

Reward: the user gets from taking their action (e.g. seeing a beautiful Pinterest board).

Investment: the user makes which will plant the seed for more triggers (e.g. subscribe, pin, like, connect, etc).

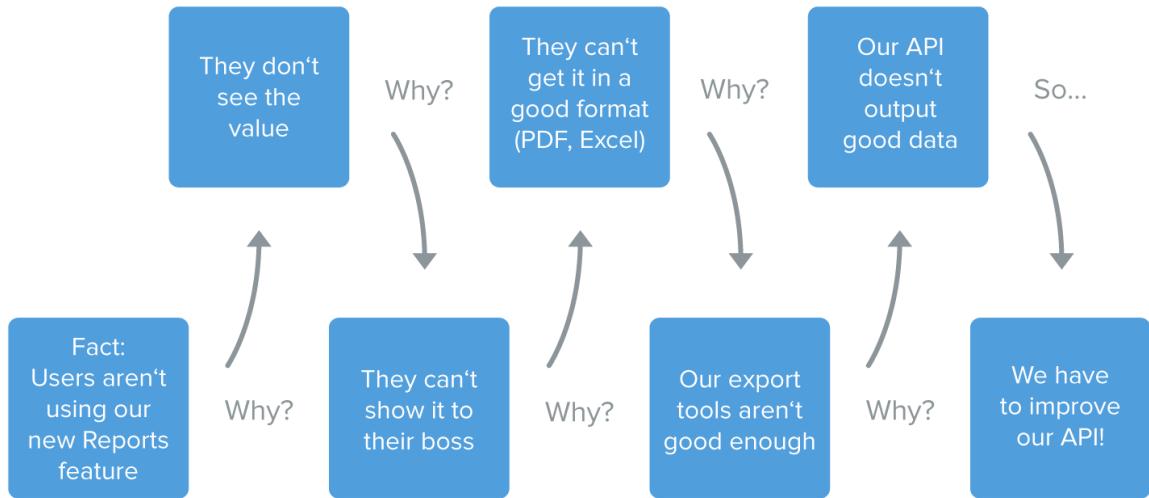
As an example, Nir points to Pinterest's hook as the following:



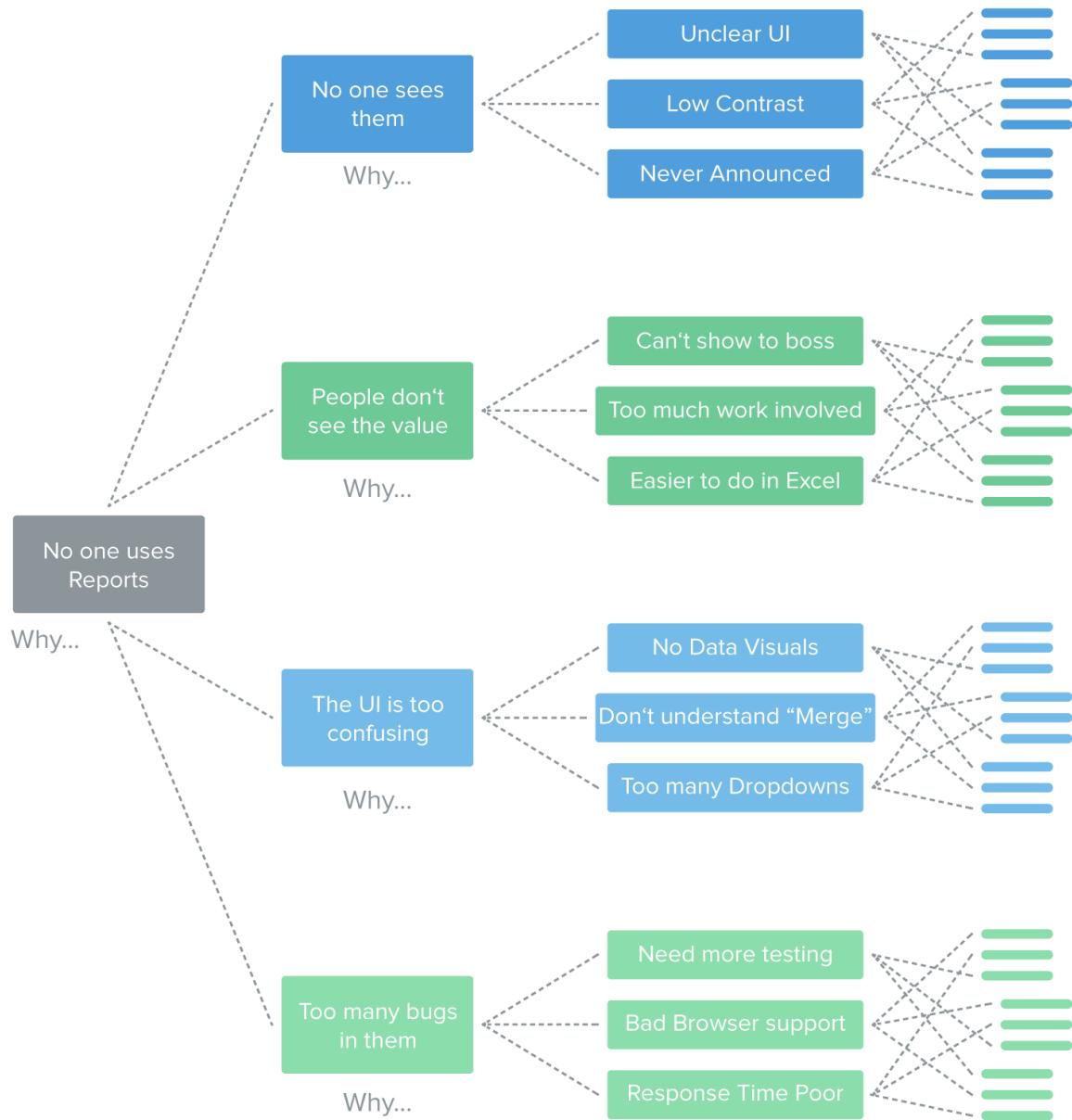
Use frequency improvements when: there is a feature that the majority of your customers use infrequently, and you believe that using it more would be of benefit to them. It's worth considering how your business will profit from this increased frequency. For example if Basecamp improve the frequency at which users create projects, they'll naturally profit, as that's how their pricing works. But there are lots of easy ways to gamify/hack feature usage that have no net benefit, or in some cases actually damage the core product offering. For example, LinkedIn's metrics for endorsements might look much better as a result,

but it's worth asking have they paid the price in credibility?

3. ADOPTION IMPROVEMENTS



Adoption improvements target those who don't use a feature. To get more people using it, rank and resolve the issues that are stopping them from using it. This is where the five whys technique is genuinely useful. You might have a situation around users not using your reports feature. Why? They don't see the value. Why? They can't show it to their boss. Why? They can't get it into a suitable good format. Why? Because our export tools aren't good enough? Why? Because our API doesn't produce good data. If you ask why enough times, eventually you'll work it out and get to the root cause.



Don't just talk to one customer because invariably things are more complicated than that. You'll find these blocking patterns over and over again. You can then resolve the key issues, the fundamental things that are actually blocking people from using all of your product.

Use adoption improvements when: there is an important feature that a good chunk of your users have yet to adopt, and you see some obvious integrations or changes that will make it easier for them to get on board.

When planning adoption improvements, always consider improvements

outside of the software too. Sometimes it's not about how the feature is designed or built, it's about how it's explained. Often users just need to know *why* or *how* to use a reporting feature. In those cases better product marketing and customer communication is how you solve it, not product tweaks.

Continuous improvements

In their early stages startups have advantages over the incumbents. They move quicker and adapt faster, without much technical debt, legacy features, compatibility issues, or high value customers restricting their movement. Sometimes this speed and agility can cause startups to pivot like headless chickens, rather than focusing on improving their product in meaningful ways. The product manager's challenge is two-fold; firstly, finding improvements that will benefit the business and its customers, and secondly, ensuring that these improvements don't get lost on a whiteboard somewhere, and actually make it out the door. Because if there's one thing that's true for startup web products, it's this: if you're not shipping, you're dead.

CHAPTER 2

When to say no to new features



There is a finite amount of improvement you can put into your existing product and feature set. At some point you are going to have to consider what new features make sense for your product and how you are going to introduce them. Facebook famously faced down a revolt from some of its earliest and most loyal college users when it introduced the News Feed. But would it ever have seen the kind of global growth it enjoyed if it hadn't faced down those users and shown them the value of the new features?

In this chapter you'll learn about creating a product roadmap, with a formula for predicting where your efforts are best placed. But perhaps most importantly for product managers you'll learn why "No" is the most important word in your vocabulary and why you should hear yourself saying it a lot.

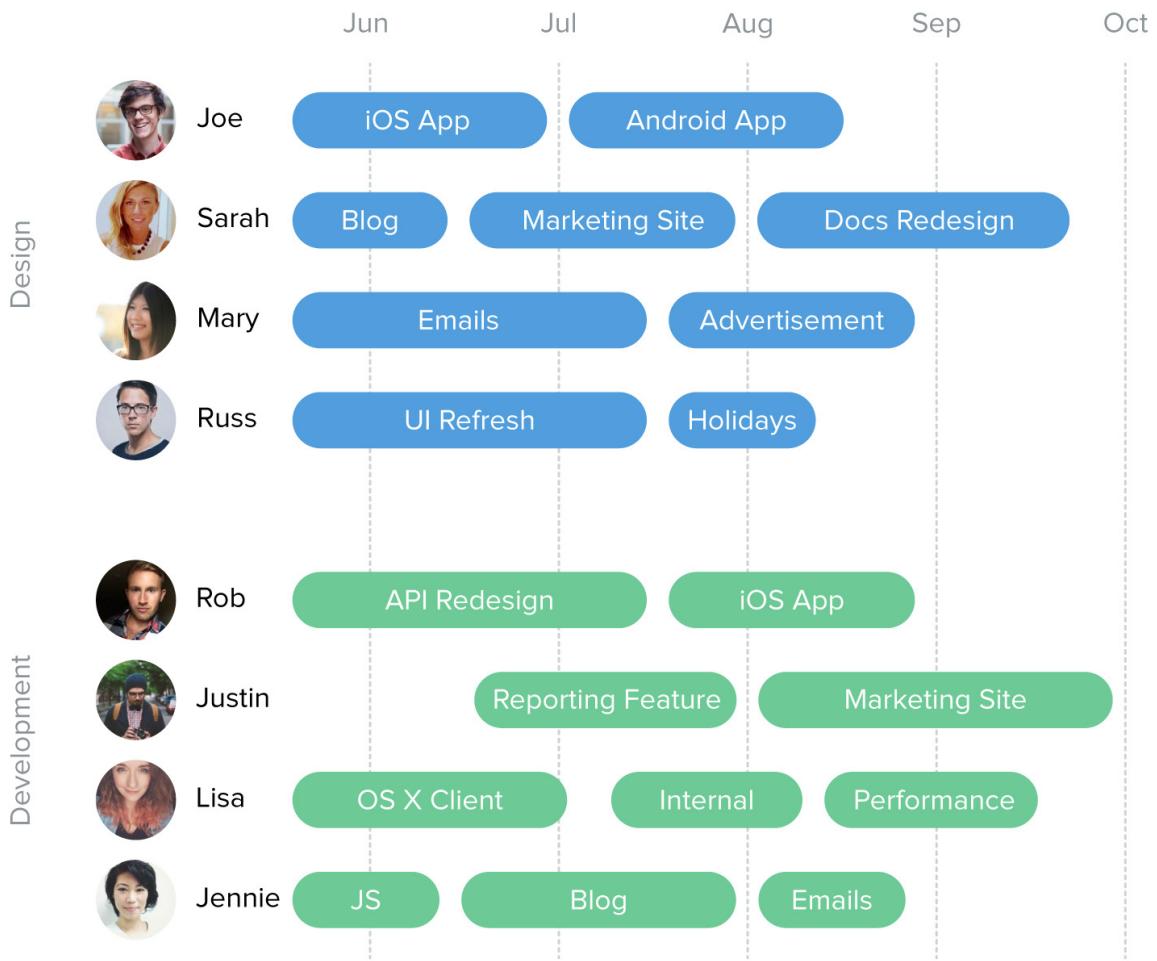
Adding New Features

New features expand the scope of the product, often making a big marketing splash, getting a version bump, and result in some press coverage. Often the fanfare attracts new customers and new use-cases for the product. Typically, new features are the only improvements that outsiders (i.e. non-customers) will ever hear about.

New features are risky. You have to be very confident they will be valued, as they're like children; you have to support them no matter what.

Ask your customers "Would you like a (Calendar/TimeTracker/Gantt Chart)?" and they'll reply "Yes". It's a one-way "something for nothing" offer; why wouldn't they? They haven't had to make a trade-off between competing priorities. This leads to customers saying they want stuff that they don't really want.

Asking your customers "Would you rather that we made the product much faster, or that we added more labelling features?" and you'll get a different answer. Everyone values speed. So when planning new features it's important to understand the trade-offs at play.



Where do you suck? Where does it matter?

If you are going to build new features then you need to get them on your product roadmap - maybe yours looks something like the example above. A roadmap is built out of hard decisions. The bugs you must fix will fight with the features you must finish, the features your customers want will compete with the ones you know they need.

If you focus only on new features you'll build a product that is miles wide and inches deep. And if you focus only on repairs you'll never innovate, thus becoming irrelevant. Hard decisions indeed.

When you are focusing on improving your product, a great question to ask is

“Where do we suck, and where does it matter?”

To improve a product you focus on the parts that are both important and disappointing to customers. Both are required otherwise you'll end up working on areas that no one cares about, or over-serving areas where you're already more than good enough.

In his popular article, [Turning Customer Input into Innovation](#), Anthony Ullwick proposes an opportunity algorithm which offers a practical way to plan a roadmap, taking importance and satisfaction into account.

Where the opportunities lie?

	Imp.	Sat.	Opp.		Imp.	Sat.	Opp.
Be aware of any problems with any of my projects	9	7	11	Keep our clients aware of what we're working on.	10	7	13
Find what my team have been working on recently	6	8	6	Review an individual's contributions to a project	5	8	5
Understand how accurate our projects estimates are	4	1	7	Record project decisions in a clear formal manner	9	5	13
Find contact details for a client quickly	8	2	14				

$$\text{OPPORTUNITY} = \text{IMPORTANCE} + (\text{IMPORTANCE} - \text{SATISFACTION})$$

Ullwick's simple opportunity algorithm cleanly identifies the shortcomings in a product, by getting customers to rank the jobs they need to do by how important they are, and how satisfied they are currently. The opportunity is then simply expressed as:

$$importance + (importance - satisfaction)$$

Sidenote: The bracketed element bottoms out at zero, over-serving a job doesn't reduce its opportunity.

Aside from the simplicity of the formula, another nice trait is how it often highlights opportunities that would otherwise have gone unnoticed. Regularly the biggest opportunities lie in areas the product manager regards as being "complete", "bug free", "good enough" etc. Put simply: a minor improvement on an important task is almost always a larger opportunity than a big improvement on an ancillary one.

Here is where the 80/20 school of thought can lead product managers astray. The idea that 20% of the features will get you 80% of the value may well be correct, but it also means that on important tasks, you're giving customers a B-grade experience where it matters most. This point was made by Mark Zuckerberg talking about the early days of Facebook...

“

You can't just 80/20 everything. There have to be certain things that you just are the best at. Things where you go **way further** than anyone else does, to establish this quality bar and have your product be the best thing that's out there.



MARK ZUCKERBERG
CEO of Facebook

There's no right way to prioritize a roadmap, but there are plenty of wrong ones. If there are opportunities in existing product areas, and your roadmap ignores them in favour of new features, then you'll soon be that jack-of-all-jobs product.

Address your product's shortcomings, or someone else will.

Why you don't add new features

Product managers, or founders fulfilling that role, have to be great at saying no. Not “maybe” or “later”. The only word is “No”.

Building a great product isn’t about creating tons of tactically useful features which are tangentially related. It’s about delivering a cohesive product within well defined parameters.

When your product gets traction, you’ll find yourself inundated with good ideas for features. These will come from your customers, your colleagues, and yourself. Because they’re good ideas, there will always be lots of reasons to say yes to them. Here’s 12 arguments that are commonly used to sneak features into a product:

1. But the data looks good

“We’ve tried this feature with a small group and engagement is off the charts.” Often this approach suffers from selective data analysis. Products are complex systems. What appears to be an increase in engagement is really just pushing numbers around from place to place. You might consider your labels feature a success based on its usage, but did you notice that that your tags feature is suddenly no longer popular? Even when the data is solid, and the increase in engagement is good, you still have to question whether it fits within the purview of the product. Add Tetris to your product and you’ll probably see a boost in engagement, but does that mean your product is better?

2. But it’ll only take a few minutes

The main problem with this argument is that the scope of work should never be a reason to include a feature in a product. Maybe it’s a reason to bump it up the roadmap as a quick win, but that’s a roadmap decision, not a product one.

Lots of bad ideas can be built quickly. Don’t be seduced. There are [no small changes](#). Even the tiniest additions add hidden complexity that isn’t accounted

for in the “but it’s just 5 minutes” estimate.

3. But this customer is about to quit

This is feature blackmail. No customer can be more important than a good product. The road to consulting-ware is signposted “just this once for just this customer”. It leads to the perfect product, for just one customer, and even then their loyalty now depends on you doing what they say. Delivering extra value to one customer comes at the cost of taking value away from many others.

4. But we can just make it optional

This leads to death by preferences. Making features optional hides the complexity from the default screens in the interface, but it still surfaces everywhere else. The visible cost of this is a messy interface with lots of conditional design and heaps of configuration. The hidden cost is that every optional feature weakens your product definition. You become “a time tracker that can also send invoices and, sorta, do payment reconciliation. But not reporting. Yet...I think...I don’t know.”

5. But my cousin’s neighbour said...

This is the “appeal to the anecdote”. It is rife in consumer products, and in SaaS companies that can’t decide what precise jobs they do. Extrapolating from a tiny sample is an easy way to bypass years of experience, research, data, and behavior to make a statement that sounds reasonable. Saying “*my brother’s company use Google Analytics, they all use advanced segments*” is an easy way to make a case for advanced segments, bypassing key questions like:

- what your product actually does.
- whether your brother’s company are a good target customer.
- whether they actually use it or just say they do.
- and whether advanced segments are actually the right solution for what your customers are trying to do.

6. But we've nothing else planned

The devil makes work for idle product teams, and boy does he come up with some shitty features. The problem here is someone sees one or more engineers sitting idle and immediately rushes through a new feature to “keep ‘em busy”. Decisions are rushed and designs are cobbled together all in the name of avoiding idle time. This is a bad way to “improve” a product.

Instead of adding to technical debt here, there’s an opportunity to pay some off. As anyone who has worked in a professional kitchen knows: “if you’ve time to lean, you’ve time to clean”. Idle time is best used fixing bugs, cleaning up test suites, refactoring, etc. rather than derailing a product vision just to “keep the team productive”.

7. But we’re supposed to be allowed to work on whatever we want

This argument appeals to cultural pride. There are many big name companies that promise engineers they can build whatever they want and ship it. Usually this promise has one of two outcomes:

1. It’s a lie told to attract engineers. This gets noticed quickly and falls apart. [You can’t fake culture](#).
2. It’s true. The end result is a one-size-fits-none product full of half-baked ideas.

There’s a difference between encouraging engineers to build things internally (a good thing) and letting people add features to a product bypassing product management (a bad thing).

8. But 713,000 people want it

Always beware when someone falls back to raw numbers to justify something. Any product with any amount of traction can make an emotive claim using numbers. e.g. “You could fill Dolores Park with people who have asked for Excel integration.” Such a claim forces you to take off your product design hat, and be

one of the “people”. Are you really going to say no to all those faces?

You have to. Because the majority of your users will suffer otherwise. The question isn’t “could we fill Dolores Park with people who want this feature?”, it’s “is this a valuable feature, within our purview, that all our customers will use?”.

9. But our competitors already have it

That doesn’t mean it’s a good idea. It could be something they’re trying out. It could be a shit idea. It could be something they’re planning on killing. It’s a mistake to assume that your competitors are in any way smarter or more tactical than you. Obsessing about your competitor’s features relegates you to permanently delivering “yesterday’s technology tomorrow”.

10. But if we don’t build it, someone else will

That doesn’t mean it should be in your product. If someone else builds it, do customers no longer need your product? Will they all switch over? Simply saying “someone else will” sounds good, but means nothing. We’ve all caught ourselves saying it. Often it’s the logic used to expand a product because you’re not willing to admit your product stops somewhere. You’re afraid to [draw the line](#).

Here’s an example: A typical date might involve a movie, dinner, and a lift home. If a cinema owner is constantly worried about what other businesses will build, and hungry to capture more value, they’ll put a restaurant into their cinema and start a cab company. Then they’ll be weak at all three. Then restaurants start screening movies...

11. But the boss really wants it

If the boss is also the product manager, and has the necessary time and insight to make smart holistic decisions, then this is fine. However, if someone is trying to earn brownie points by focusing on pet projects that their manager has a penchant for, this inevitably leads to trouble.

12. But this could be “the one”

This is a classic “Appeal to the Unknown”. [Editing a product](#) requires some hard decisions about what to build. You can speculate that any unbuilt feature could transform your product. But speculation is all it is, nothing more. When you’re afraid to make hard decisions, you fall back on appealing to the unknown, and therefore building everything. You end up with a repository of features, not a product.

Why is “no” important?

The thing is, no one keeps crap ideas in their roadmap. Identifying and eliminating the bad ideas is the easy bit. Real product decisions aren’t easy. They require you to look at a proposal and say “This is a really great idea, I can see why our customers would like it. Well done. But we’re not going to build it. Instead, here’s what we’re doing.”

There are no small changes

Still not convinced that there are no small changes? Let’s play this out with a simple example.

“We want to limit the length of a text review in the product to 140 characters, because we may want to use SMS to send them out at some stage. That’s a small change, right?”

Wrong.

There are no small changes when you’re committed to delivering quality software. Let’s look at the above case. A naïve programmer may well get this coded in three minutes – after all it’s just an if-statement.

But product management is never that simple. Before you embark on any “small changes” to your product you need to ask some questions. Sticking to

our review length example from above, let's start with some easy ones.

What happens when the review is above 140 characters? Do we crop the string, or display an error message to the user? If we display an error, where does it appear? What does it say? Who is going to write the error message? How do we explain to the user why we're limiting them to 140 characters? How will these errors look? Do we have a style defined? If not, who is designing it?

But wait, there's more...

In the unlikely event that we have answers to hand for all of those initial concerns, we're still not finished. Just doing this server-side is a messy way to handle an error. We should do this client-side. But if we're going to do client-side validation then that throws up a few more questions...

Who's writing the JavaScript? Does the JavaScript display the same type of error as the server-side code? If not, what's the new style? How does it behave without JavaScript? How do we ensure that any future update to the 140 character requirement impacts both client-side and server-side validation?

We're still not done. Look at this from a user's point of view. They're already frustrated by having to limit a review to 140 characters for a bizarre reason they won't understand, and now we're asking them to guess how long their message is? There must be a better way. Let's give them a character counter. Oh, well that raises a few more questions...

Nearly there...

Who is going to write this character counter? If we're using one we found on the net, then who wants to test it in our target browsers (i.e. not just Chrome 37 and beyond)?

Also, where is the count of letters displayed on the screen? What does the count look like? Of course, the style should change as the user approaches zero characters, and should definitely look erroneous when they've used more than 140 characters—or should it stop accepting input at that point? If so, what happens when they paste something in? Should we let them edit it down, or alert them?

When we've implemented the character counter, styled all the errors, implemented the server-side validations, and checked it in all of our supported browsers then it's just a case of writing tests for it and then deploying it. Assuming your time to production is solid, this bit will be straightforward.

All of this happily ignores the fact that users will wonder why someone wrote an eighty word review just before them and now they're only allowed write a 140 character one. Obviously we'll need to keep support in the loop on this, and update our documentation, API, iPhone, and Android apps. Also what do we do with all the previous reviews? Should we crop them? Or leave them as is? Don't get me started on how we're gonna deal with all the emoji that people use these days... good luck sending them in a text message. We'll probably need to sanitize the input string of rogue characters, and this means new error messages, new server-side code... the list goes on.

Once you get through all of this you will have your feature in place, and this is just for a character count. Now try something that's more complex than an if-statement. There are no tiny features when you're doing things properly. This is why as a product manager you need a good understanding of what it takes to implement a feature before you nod your head and add it to the roadmap.

The big picture...

Often what seems like a two minute job can often turn into a two hour job when the bigger picture isn't considered. Features that seemed like "good value" at a two minute estimate are rightfully out of scope at two hours.

Key point: Scope grows in minutes, not months. Look after the minutes, and the months take care of themselves.

Agreeing to features is deceptively easy. Coding them rarely is. Maintaining them can be a nightmare. When you're striving for quality, there are no small changes.

Nothing comes for free

Often in software you'll hear that you get something for free e.g. "the framework just gives us this for free", or "if we use X service provider, we get Y for free", or your developers will tell you "we were building this bit anyway so we get this extra piece for free".

Nothing is free in software.

It's not free because you're sitting there talking about it, you're debating if you should do it. That's costing you time for starters. It's not free in the doing of it, or the verifying it was done right, that it works, that all edge cases are workable. QA is never free.

It's not free in the communication of doing it. If you add a feature, you have to tell your team, to tell customer support, to then tell your customers. You have to tell your customers, otherwise it's definitely pointless. Communication is never free.

It's not not free to carry it forward. Once it's there you'll have to support queries from customers, you'll have docs, how-tos, videos, all needing to be updated. It's not free to undo. Maintenance is never free.

Think of these features like house pets. Craigslist is full of free pets, yet you're not stocking up your house for a very simple reason. There's a big difference between the retail price and cost of ownership.

CHAPTER 3

Which new features to build



When it comes to product strategy at Intercom we find the [Jobs-to-be-Done Framework](#) is extremely useful in helping us to decide what we should build. Popularised by Harvard Business School professor Clay Christensen, it is a way of looking at the motivations of customers in using a product, rather than the traditional marketing techniques of slicing and dicing customers according to demographics.

People don't buy a product because they fall into a particular group e.g. female, mid 30s, living in suburbia, working part-time. But they do buy a product to solve a problem. If you understand the "job" that customers are hiring your product to do, then you can make sure that you have a razor sharp focus on helping them achieve the desired result. The features you choose to build should be the ones that will help them do the job that needs to be done.



Making things people want

The problems people encounter in their lives rarely change from generation to

generation. The products they hire to solve these problems change all the time.

When you're building or managing a new product, you have to believe you can create a better solution that people will want to use because it delivers a better outcome for them. A strong understanding of the outcome customers want, and how they currently get it, is essential for you to succeed in product management.

Maybe your customers want to be entertained, or spend more time with their friends, or understand what projects teammates are working on, or maybe they want to project growth for their business. If the desired outcome is real then they are already achieving it through some product in some way. Your job is to improve upon that.

Sidenote: If you can't find what product they're currently using, the chances are that it's a fictitious outcome ("Wouldn't it be cool if I could get all my social media files in one place?") or an aspirational one ("Of course I want to lose weight"). Espoused behavior never reflects reality.

Focusing on outcome, rather than category, industry, or product type, lets you understand your real competitors. The second a company focuses on "the industry it's in" rather than the "outcome it delivers", it loses touch, and shortly after, loses customers.

Newspapers, for example, believed they were in the "Newspaper Industry", and as such struggled to work out why bored commuters had stopped buying their product. They would look left and right at their competitors and wonder which newspaper had stolen their customers. They would experiment with new formats, new layouts, lower prices, sharper headlines, but they couldn't stop the rot. Had they instead focussed on the outcome they deliver (bored commuters want to be entertained for short bursts of time with bite-sized articles), then their competitors (Twitter, Facebook, news apps) wouldn't have been so oblique to them.

Let's look at some jobs that, like boredom during a commute, have stuck around for years, through hundreds of technological advances.

Age old jobs, new solutions

People, particularly students and young people, wanted to pass notes and messages, without fear of other people seeing them...

People still want this so today they use Snapchat.

People wanted to store photos in a safe place, like the shoe box under the spare bed...

People still want this so today they use Dropbox.

People wanted to put their favorite photos in a prominent place, like their mantelpiece, so everyone could see them...

People still want this so today they use Facebook.

People wanted to collect scrapbooks of ideas, for home renovations or other projects...

People still want this so today they use Pinterest.

People wanted to leave nice reviews, and tips for other travellers in physical guest books...

People still want this, so today they use Foursquare.

Doing a better job

There are literally hundreds of examples like the ones above and there's a common trend in all of them. Making things people want involves understanding a long standing human or business need and then using technology to:

- take out steps.
- make it possible for more people.
- make it possible in more situations.

The first approach, removing steps, is the most common for start-ups. Pick a need where the existing solutions are old, complex, and bloated, and find the simplest smallest set of steps possible to deliver the same outcome. Arranging a taxi in a city used to involve calling many numbers until you found a company with availability, then a lengthy dialogue about your location, destination and

required arrival time.

Today you press one button and a car shows up.

The second approach usually involves reducing the cost (in time or money), or barriers to using a product so that more people can use it, thus expanding the market. Fourteen years ago, if someone wanted to get their writing online they had to rent a Linux server, download a .tar.gz file containing the source code of a blogging engine, upload it, run a series of weird commands to unpack it and give it write access, and then configure it.

Today this is two clicks in Medium.

The third approach involves removing common situational limitations on a workflow. Accepting payment used to involve bulky machines with rolls of thermal paper, faxing paperwork to banks, ISDN lines, and batch transaction transfers run nightly.

Today you swipe a card through a phone and you're done.

Jeff Bezos is famous for saying “focus on the things that don’t change”. The problems that people and businesses encounter don’t change often. The ways they can be solved changes almost yearly. So it stands to reason that making things people want should start with the “what people want” bit, and not the more tempting “things we can make”.

Remember: It’s easier to make things people want, than it is to make people want things.

An acid test for new features

Here’s a simple set of Yes/No questions that you can quickly answer before you add another feature to your product roadmap.

In the previous chapter we wrote about how product strategy means saying no, but a list of reasons to reject a feature isn’t as immediately useful as a test that any new feature must pass. So here’s a list of questions your new feature must

score straight yes's on.

1. Does it fit your vision?

What do you believe that no one else does? What do you know about this problem that no one else does? How do you believe it should be solved? Anyone can pull the data, run the focus groups, read the Gartner reports, get out of the building, but only you have your vision.

Product decisions based on vision alone sometimes seem irrational, because they're tough decisions. Most people understand that their bug tracker doesn't also monitor server uptime. That's rarely debated, it's not a marginal call. But should a project management tool have a reporting feature? Not so clear.

The more nuanced decisions are the ones where you meet resistance. Colleagues, customers, even other product managers and founders will push back. Here's some examples:

- Apple refused to ship netbooks at a time when netbooks were the most popular style of PC. Every analyst [demanded them](#).
- Basecamp refused to add Gantt Charts to their product. For that they were labelled [blind ideologists](#).
- Developer Garrett Dimon [refuses to add an “on-hold” state](#) to his issue tracker, Sifter, as he simply doesn't believe it's the right way to manage issues.

Worse than being a hard decision, you'll never truly know if you got it right. There is no right. There is no wrong. This is art, not science. It's just you and your vision.

2. Will it still matter in 5 years?

It's a hard and boring question to ask but will this still deliver value in five years time? You'll feel like the curmudgeon of product planning. But that app that was so hot in 2013 was gone by 2014. Those slide, fade, and fold effects that everyone loved last year look tacky as hell this year. If you're going to spend the best years of your life on a product eschew the trendy,

and focus on the meaningful.

3. Will everyone benefit from it?

Beware the “fre-cently” bias. You never doubt the things you hear frequently or recently should be road-mapped. It’s a natural reaction caused by your inbuilt empathy for customers. It’s not nice to tell people “no” repeatedly and hear the same responses over and over, so when possible “fre-cently” is used as an excuse to reward yourself. “Sure we’ll build that, I’ve heard it twice today already,” says the founder with 4,800 daily active users, to the unbridled joy of 0.0625% of her customer base.

The danger of the fre-cently bias is that it gives you the illusion of analysis, the sense that you’ve done your homework and this is a rational conclusion you’ve come to. In reality you’ve made a lazy decision to satisfy the whims of a small sample of vocal users without taking a second to investigate how many people really want or need the feature.

4. Will it improve, complement or innovate on the existing workflow?

Adding a whole new workflow to your product should be a rare occurrence. The majority of your time should be invested in improving, complementing, or innovating upon existing ones, and for any given project you should know which of these you are doing.

If you’re improving the current solution, your metric will be customer satisfaction and/or maybe a decrease in [tool time](#), or support requests.

If you’re complementing it, your metric will be an increased throughput for the workflow, as it now works in more cases or can be used in more circumstances.

Innovation is the trickiest. You’re shooting for the mythical “whole new way”, which carries so much risk but offers so much reward. Your measure will likely be new customers acquired though often they come indirectly, as a result of the PR, marketing or awareness created.

Redesigns are fun, but you can spin your wheels on them. A good way to cut through the bullshit is to simply ask “Will more people use it? Will people use it more? If neither, then will it definitely be better for those who do use it?”

5. Does it grow the business?

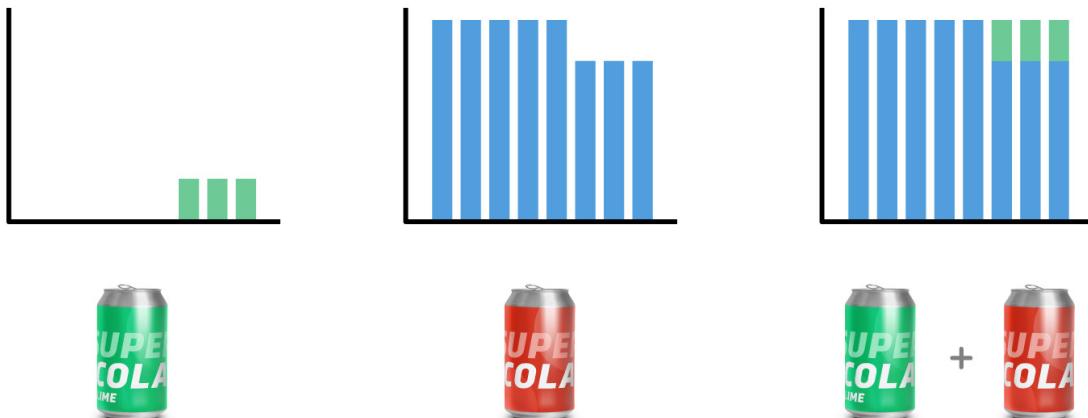
Can you join the dots between the impact this new feature will have and new revenue? e.g. a project management software company might make the case for project templates, the argument being templates can be used in more cases, which should increase the number of projects customers have, which in turn increases upgrades, and thus revenue.

Note that reducing churn also grows the business; dollars don't care where they come from. Often a feature is added to ensure stickiness of customers, or to widen the moat around a product. The key point in all cases is to understand how any work affects growth, after all [everyone works on growth](#).

6. Will it generate new meaningful engagement?

Most metrics ignore the system and focus on the isolated feature being added. Put a new button in your product and people will click it. Get enough clicks and you can call that an increase in engagement. But that's nonsense. A counter-metric is “have people stopped doing anything else?”. So if you add a metric to track one area of your product, you must also analyse the other areas that are likely to be impacted.

WATCH OUT FOR SIDE EFFECTS



A real world metaphor is the effect of “Diet Coke with Lime” on Diet Coke sales. We see how launching a whole new workflow affects sales. If you’re the PM on Diet Coke with Lime, you could call your launch a success, but if you’re the CEO you’ll realise you complicated your product line, bought all these limes, all these juicers, and all that advertising, and have no new revenue to show for it. Not exactly a success, no matter what metrics that PM can show you.

7. If it succeeds, can we support and afford it?

One fallacy of “quick wins” and “easy hacks”, is they usually only evaluate effort required before shipping e.g. someone surprises you with a JavaScript bookmarklet for your app, or an agency produces a Windows Phone app in record time and low cost, and because the effort was relatively minimal, and the idea makes sense, you ship it. Success!

Then the support requests come in, and it turns out none of your team know XAML, so you’re stuck with a broken build live, and a few hundred customers complaining to support about it. Of course it’s never that obvious. Good engineers rarely say they don’t know something. They’ll hack at it some evenings, display some initial competency, and then estimate how long until they fix the bug. This is all time that you didn’t plan on spending when you shipped this app. And that estimate is wrong. And then some.

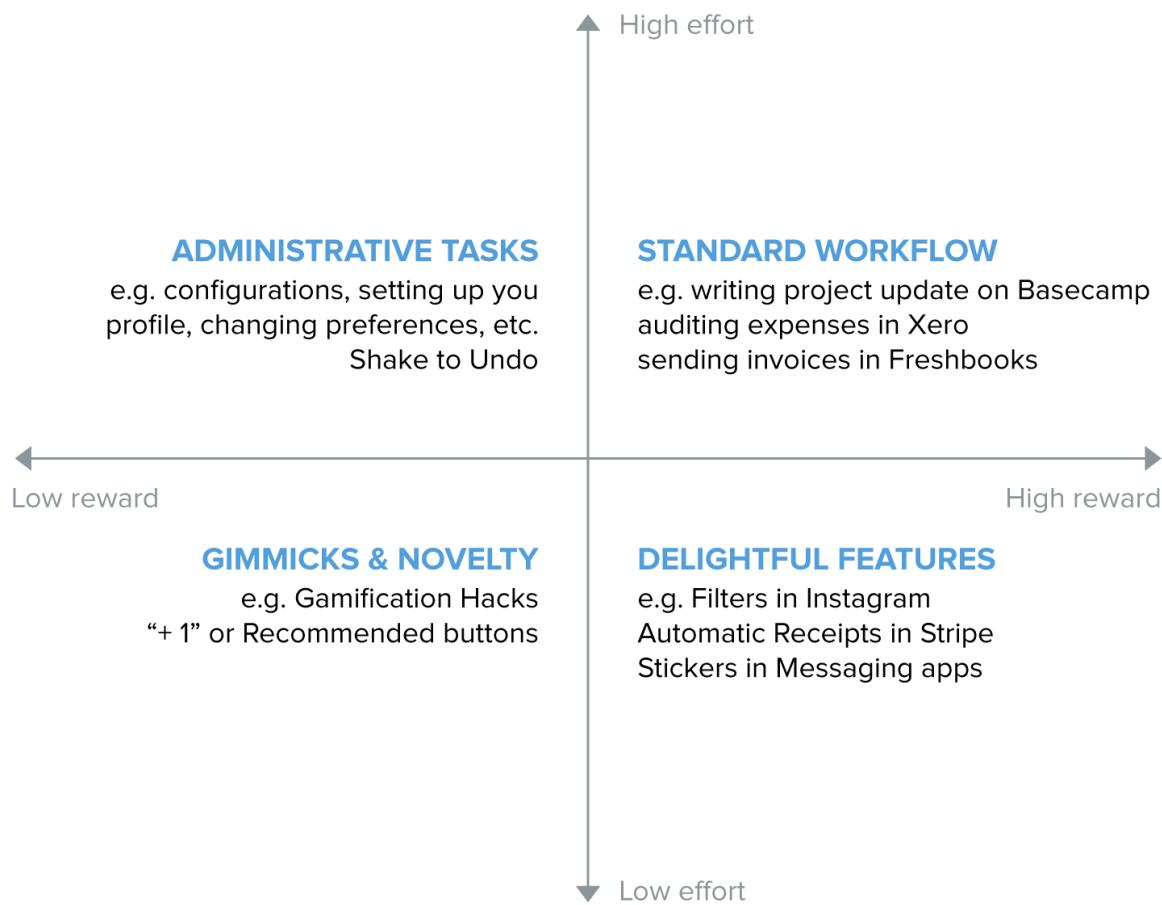
Another area that can bite is offering incentives, or initiatives that you can't justify. If you have a CMS product, and you offer free homepage designs, you'll get more sign-ups. It makes sense to do this early on, when you really need to entice customers to try an as yet unproven product, but it won't work long term. It goes without saying that doing things that don't scale is a great strategy, until you need to scale.

8. Can we design it so that reward is greater than effort?

As my colleague Paul Adams [previously wrote](#), for any feature to be used the perceived benefit has to be greater than the perceived effort. End users understood the benefit Google Plus could bring, but the overhead of dragging and dropping many copies of your contacts into various boxes, on a regular basis, was simply not worth it. I feel that [check-splitting apps](#) habitually fall into this category. Splitting a check can be a pain point, but any solution seen thus far still costs too much, and we're not talking about price. We're talking about time, overhead, social capital, etc.

Product design is about cost-benefit analysis. How useful is something vs how hard is it to do. Here's how I think of it...

THE COST-BENEFIT ANALYSIS OF FEATURES



It's important to know which quadrant you're building in and if you're wise you'll steer clear of the upper left for all but the most essential of features.

9. Can we do it well?

Every product has its neglected corners, usually areas where the PM dabbled but conceded defeat. Conceding defeat all too rarely results in removing a feature, it usually just means ignoring it, leaving a messy trail and a confused offering.

The problem here is when product teams tackle areas they don't fully understand. This is often the case when a team moves beyond [self-design](#), that is, past the point where they are their customer. At this point their design process breaks, and they need a new one. Examples include:

- teams that don't track time adding a time-tracking feature.
- people who don't manage calendars designing a calendar management option.
- designers who don't close issues building an issue tracking system.

Note that none of the above are inherently wrong, what's wrong is the design process. It needs to move beyond “works fine for me” into something much more activity focussed. To build a feature well, you have to understand the job it does intimately. As a corollary to the old saying: if you can't do it well, it ain't worth doing.

10. Can we scope it well?

Starting with the [cupcake release](#) for a feature is essential. Early usable releases provide the feedback necessary for an idea to flourish. A good sign that a feature isn't well scoped is when it lacks specifics, e.g. “Make it work for bigger companies”, or that it's feature based e.g. “reports”, as opposed to job-based e.g. “Let sales managers see their team performance”.

It's always easy to agree on truisms in product meetings. Someone says “It should be easier for bigger teams”, everyone nods, and a Post-it gets stuck on a whiteboard. It's only in the specifics that you'll understand the scope e.g. “we should let people group their colleagues by department” vs “we just need an enterprise mode”.

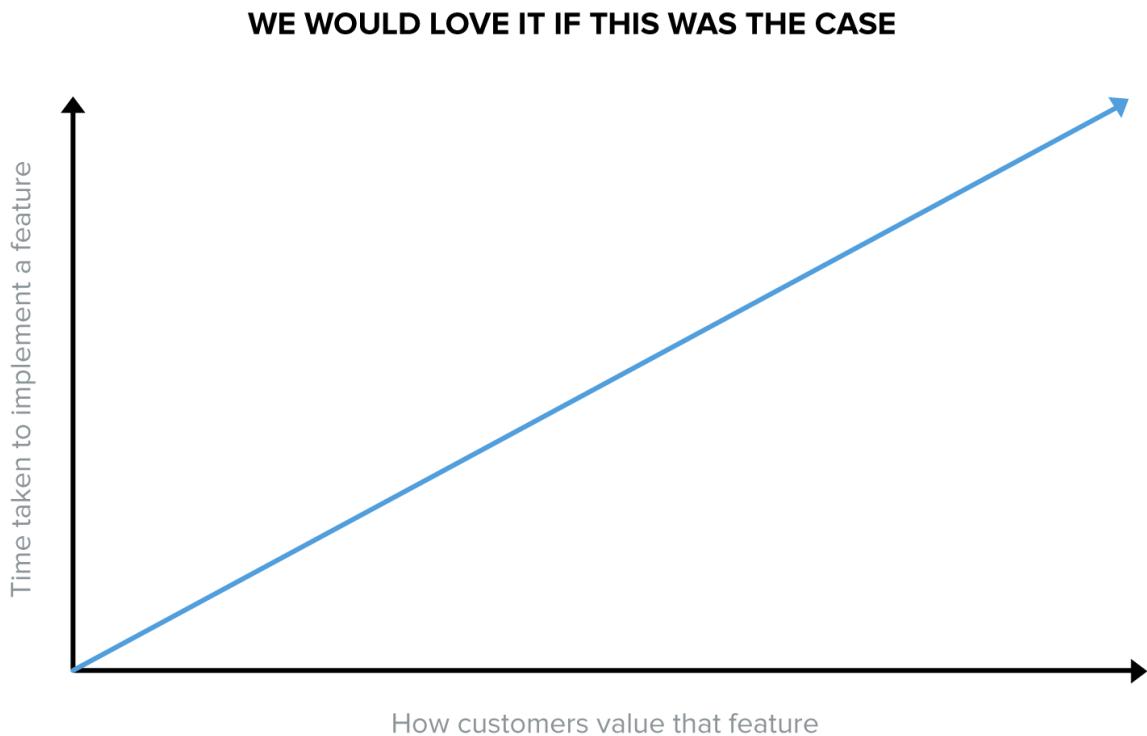
Badly scoped features meet no one's requirements, ship late, if at all, and add nothing to the product but confusion.

Slippery slopes & marginal calls

It's tempting to excuse occasional violations of this list, assuming that “as long as it's right most of the time”, it'll be fine. Maybe that's true in theory, but this is software. Reality bats last here. This is why we say that product strategy means saying no. Roadmaps are incredibly hard and require agonising trade-offs, but regardless, every good product manager needs a firm checklist for when they say yes, and when they say no. And they don't make exceptions.

Products get bloated one lazy decision at a time. There is no single choice you'll make where some light turns on saying "Your Product is Now Too Complex", and you'll never get an email saying "Last Thursday at 2:03pm Your Product was Disrupted". Bloat, complexity, and disruption can only be seen in the rear view mirror, so always be mindful of the risk of saying yes.

Features & physics envy



Physics envy is a term that can be used to describe the desire of designers, developers, and product managers to enforce laws on a system that simply doesn't obey them. We would love it if the return on investment was always perfectly proportional to the time spent. It would make product planning easy. The problem is that you can spend five weeks working on a killer feature, only to see it go ignored by your users. Conversely you can add one sound effect, have your cute logo say "Hi", or make your favicon blink, and all of a sudden everyone is talking about you. It's not rational, but then what is?

Rather than try to stare enviously at those who don't work with such irrationality, let's see how we can use this to our benefit. When you're planning your next few months work, plot your features on this chart...



Anything in the lower right corner is special. These are features where the return is disproportionate to the effort. Some examples I've seen are:

- rewrite the app copy to make it more personal, funny, precise or useful.
- add a couple of keyboard shortcuts to let power users be more productive on the main screen.
- remember the last active project and bring the user back there when they next log in.
- provide extra information and links to actions in email notifications.

When to use quick wins

Customers won't value all the development you do on a project. Some necessary tasks such as back-ups, re-factoring, optimization, improving infrastructure offer little immediate reward. This doesn't mean you can ignore those issues. The trick is to plan your roadmap so there's never long periods where customers feel the application has been abandoned.

This is precisely where quick wins are useful. You can schedule these quick wins to cover time periods where it would otherwise appear that nothing is happening. This way you're constantly delivering value to your customers, whilst still tackling the larger features or back end problems. Remember the only thing worse than an app in constant flux, is one where users can see cobwebs forming.

Rolling out new features

Roll outs are make or break and they're damn hard to get right. Doubly so if you're changing your user's workflows. For a lot of business to business products, it's someone's job to use it. There are thousands of people whose entire job is to use [Intercom](#). So when we are considering changing a behavior, or adding a new one, we don't do it lightly.

We follow a series of steps to progressively release it using feature flags. Your mileage may vary, but we've had good success with this approach.

Team testing

The first step is to deploy it live with real data, but only visible to the team that built it. This is important because it avoids too many people reporting the same issues but also permits a release before every single step is perfect. The team themselves know how far they've gone, so they alone can tell what's a bug versus what's simply incomplete.

Company testing

When a team believes their feature is complete, it is released it to the entire company who immediately use it in their workflow. There are no elaborate explanations, tutorials, or excuses. It's simply shipped, just like we intend to ship it to customers. This catches any confusions we might have created; if life-long Intercomrades don't understand it, what chance do our customers have?

Restricted beta

The first public release is to a single digit percentage of users. We maintain a “Trusted Testers” group for this purpose. We communicate it as a beta, and specifically ask for feedback later, once they have used it. That final point is subtle, but it’s worth remembering. Feedback on what it’s like to use a feature can only come from people who have used it. Not people who tested it or played around with it. Jake Knapp wrote that “[Reactions > Feedback](#)”, citing how when people are trying to be helpful they’ll make up things in an effort to offer you improvements. Speculation and espoused behavior have no place in product design.

What you’re looking for here is:

- **Discoverability** - are people finding this feature?
- **Engagement** - are people using this feature?
- **Adoption** - is it now being used as part of a workflow?
- **Use Cases** - how is it being used? what use-cases are popular?
- **Barriers** - Who isn’t using it? why? what’s preventing them?

During restricted beta you can ensure it’s discoverable for all users, refine your marketing to focus on pitching the true value as your customers see it, and you can resolve barriers to usage and adoption.

Full roll out

This is when it’s available to everyone who is eligible for it and you need to think about how you are going to tell them. Typically not all features are available to all users - there’s some basic price discrimination at play - so at a minimum you split your message into the “haves” and “have-nots”.

Those eligible for the feature should be sold on it. Feature announcements are often very inward looking, focusing much more on “what we did”, rather than “what you can do”. If you want people to use your product, encourage them by showing them what they can use it to achieve.

Those ineligible should be sold on the feature as a reason to upgrade. Often you

can achieve this with a free trial (e.g. Try our premium plan for one month free).

Message schedule for the feature

Once a feature is launched it's easy to forget about it and move on to the fun of designing the next one. But features without engagement are the seeds of bloatware. For any significant feature, you need to have plans for those who use it, and those who don't.

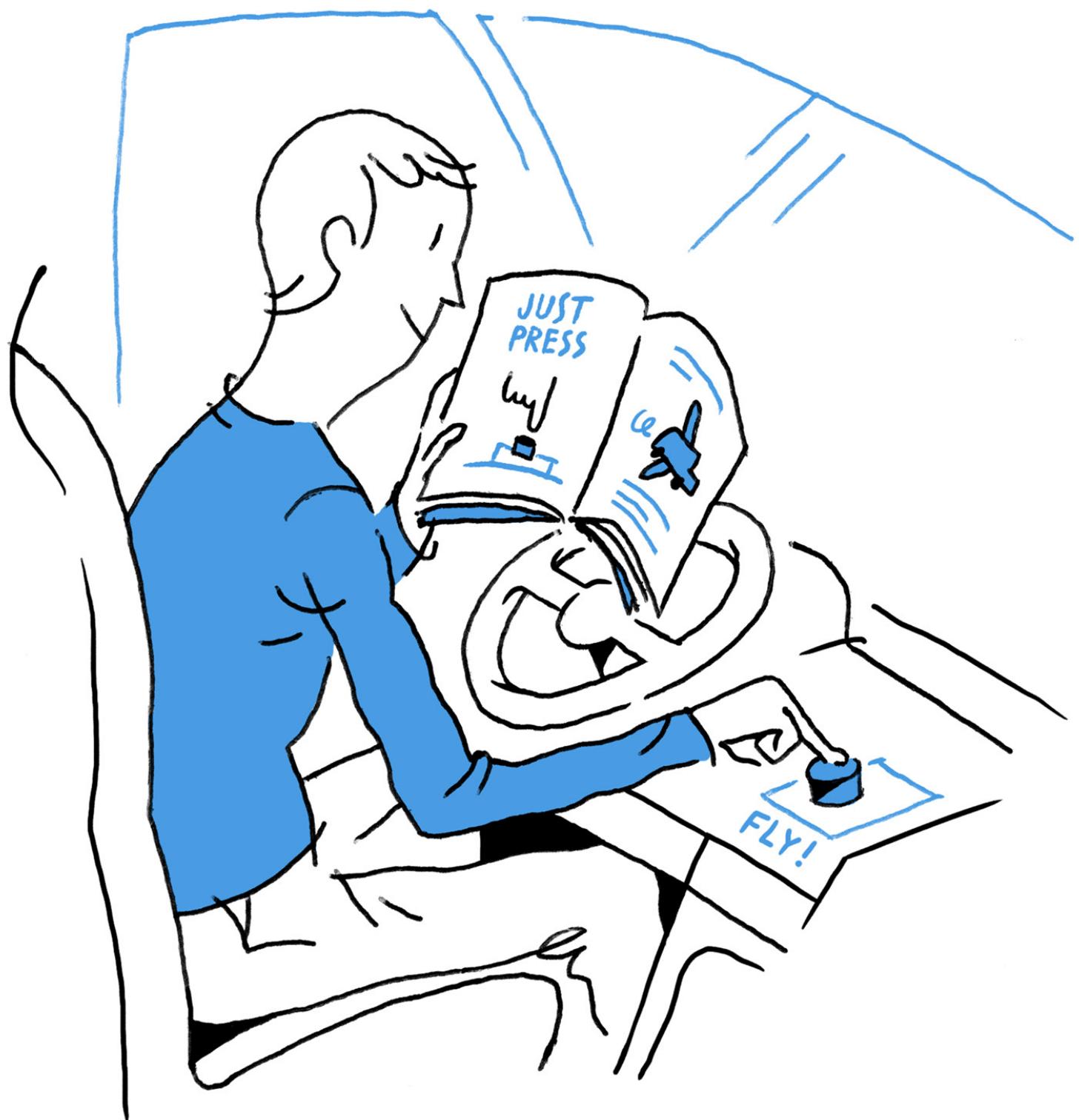
When people are using it well, you want to gather use cases, and marketing collateral from them for future use. Depending on your plans you may also want feedback.

When people aren't using it, you want to encourage them, educate them, and engage them, the only acceptable outcome here is they'll start using it, or they'll tell you why not.

There is no pride in having a “big” product with lots of features, only a highly engaged one. The more surface area your product has, the more important it is that you chase engagement and learn from your mistakes. This is most important when you're young and scrappy. You're capable of shipping a new feature every week, but you should focus that energy on growing usage of your product, not growing your product.

CHAPTER 4

Getting that feature used



Deciding what features you need to build and getting them built is really only the start of the process. New features are just code that's gathering virtual dust unless they are being used by your customers. As discussed in Chapter 1, if a feature is not being used it's time to make hard decisions about killing it or making it better. Fail to make a decision one way or another and you risk losing customers to a nimbler competitor with a more focused product that does the job more efficiently.

Launching a successful feature demands the same skills as launching a successful product. The difference is that you also have to navigate around all of your legacy decisions, and appease current customers too. It's tricky.

The majority of new features flop. You just don't notice it happening. And that's the point. "New improvements" sit unappreciated, unused, and are eventually cast aside. Welcome to software. Improvement is hard.

You do months of research. You pull demographics, metrics, ethnographics, analytics, psychographics, you name it. You meet with customers for days, weeks, months. You meet their parents, you feed their dogs, you feed their parent's dogs. You do it all to understand what customers really need. Not what they say they need. Not what they think they want, or what they're asking for. What they really need. And you go and build that!

And it still flops. This is a tough racket.

Why new features usually flop

The thing is, unless customers use a feature it may as well not exist. This is often forgotten in the rush to ship fast. It's not just about shipping code to servers, or checking boxes on a roadmap, it's about getting your software used. So before you ship that next feature, ask yourself these questions...

1. Will everyone see and understand it?

Fun fact: when Microsoft asked their users what they wanted added to Office,

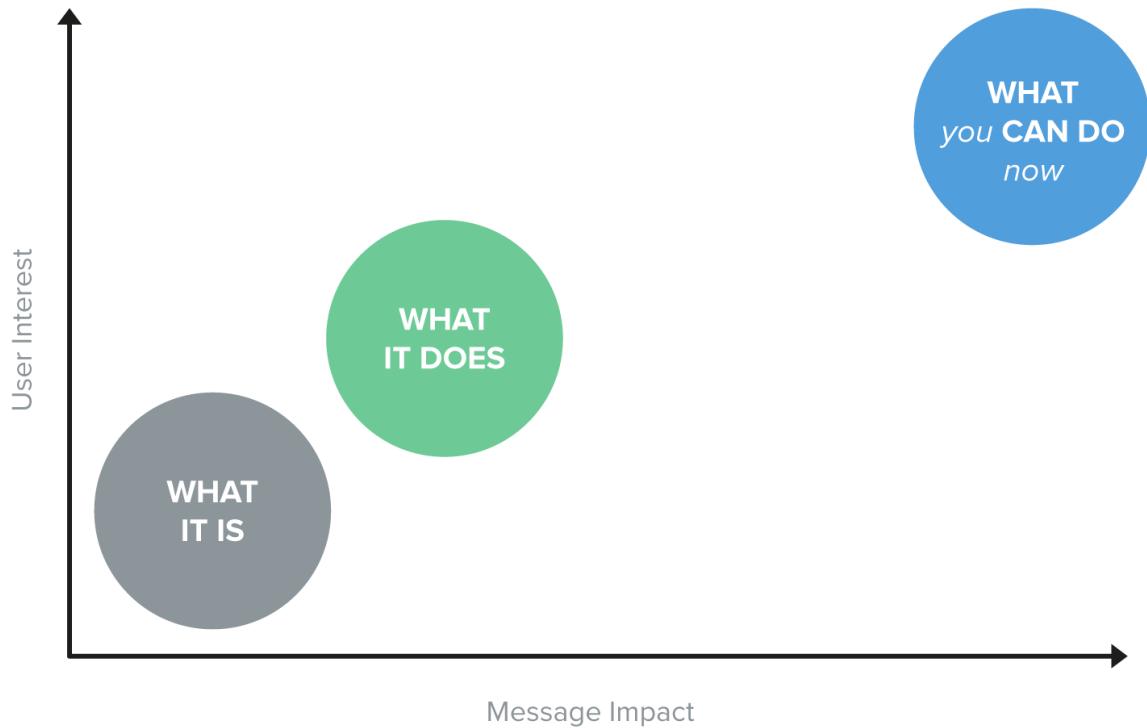
they found 90% of the requested features were already there. Microsoft assumed this was an awareness challenge, hence the launch of the ribbon toolbar which highlights all features, and therefore highlights nothing. Like I said, this is tricky.

When you design the first version of a product, you want it to look complete. So you don't necessarily plan for expansion, or leave space for future features. As you add them you make space for them, hastily, causing discoverability problems for your users. If you find yourself constantly fielding questions about where a feature is, then you have a discoverability problem and all new features will flop, or at least require lots of training to find and adopt.

2. Are you showing users what you did, or what they can do?

Telling your customers something is a “ground up rewrite”, “HTML5 based”, “responsive” or anything like that will miss the mark unless you’re selling to developers. No one cares what you did, or often even how you did it. Your customers care about what they can do.

THE IMPACT OF FEATURE ANNOUNCEMENTS



Focus your message on what your users can now achieve with this feature and you'll get their attention.

3. Are you announcing it in context?

New features, especially small additions, land in products without much context. Your goal should never be to “just get it launched”. The goal should be to “just get it used”. Email is the wrong medium for these announcements: it’s often overkill, and usually arrives at the wrong time and in the wrong context. The right time to promote an improvement is when someone is in your product in a position to use it i.e. an in-app announcement that is triggered according to rules and events.

4. How will tomorrow's sign ups hear about it?

As a product grows, it expands beyond what's learnable in one sitting. That's not a problem, not every feature makes sense up front. Features are only rele-

vant when they solve problems. Put simply you don't need to tag files until they are uploaded. You don't need to merge tags until you've got lots of different ones. So it stands to reason that telling new users how to merge tags is a badly timed message.

There's a right time to introduce features to customers, and this is where targeted messaging matters. We promote saved replies in Intercom once a few replies have been sent. We promote keyboard shortcuts after a user has used the product long enough to care about them. This is how you make messages matter.

5. Do you plan to follow up with users & non-users?

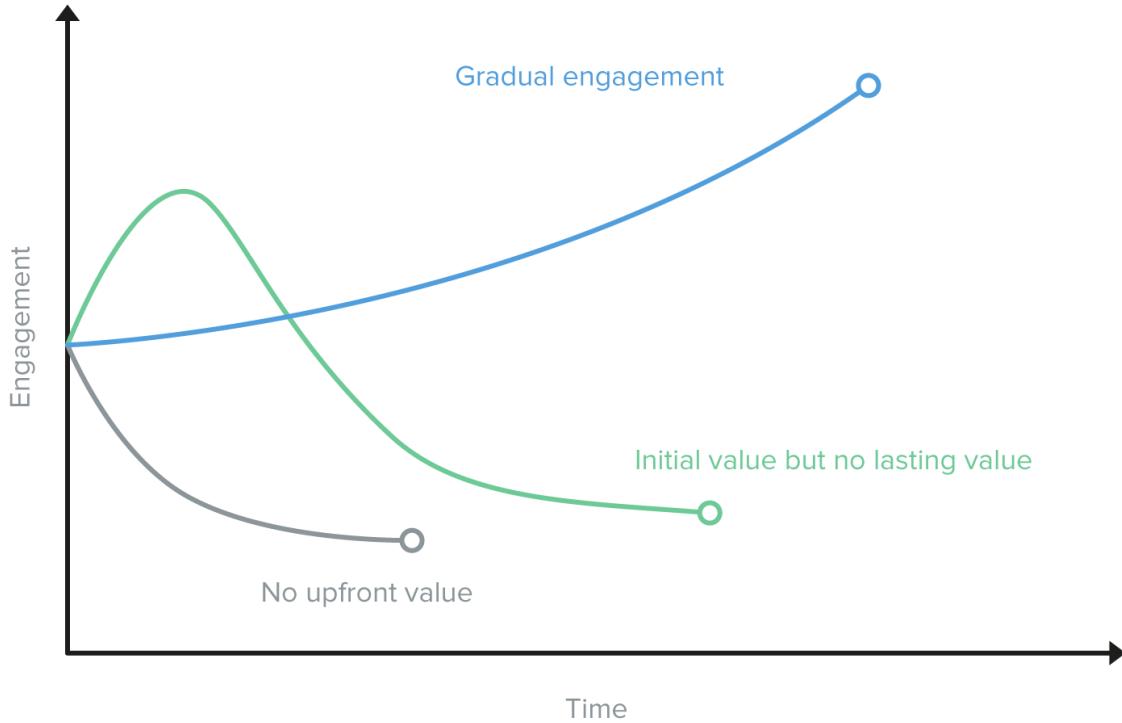
Once a feature is live, you should follow up with those who use it to understand how, why, and when it's used. Look for ways to increase usage of it. Here's some questions I've asked customers about a new feature:

- When did you notice it? What did you think? Did you use it straight away? What attracted you to it?
- Did you need to read the documentation? Was it sufficient?
- Were there any barriers to you using it?
- Are there times in your work day that you want to use this but can't?
- Have you told anyone about what you use it for? What did you say?

It's equally, if not more, important to follow up with those who don't use it and understand what's stopping them. Often you'll find barriers that are easy to break down e.g. "I keep forgetting to check back on it", "I don't know if anyone else in the company uses it", "I need to be able to get a CSV of the data", etc. These are all resolvable problems once you understand them.

Your design process must acknowledge that you'll get things wrong. So many roadmaps and sprints ignore this. Designers and product people get things wrong all the time and when they do, they're faced with two choices: improve it, or ignore it and jump onto the next feature. Do the latter enough times and you too will be telling the world that 90% of your feature requests were already in your app. Hope you like ribbons!

Increase user engagement



The definition of an “engaged user” varies from product to product. For a to-do app an engaged user should be logging in every day to add and complete items whereas for an invoicing app an engaged user might only log in once per month. There is no consistent quantifiable definition of engagement across different products.

Unlike page views, visitors, returning visitors, and conversions, there’s also no analytics app that can instantly tell you what you need to know. But ignore engagement at your peril.

Google+ claims 170,000,000 users, which gets them a few news stories, but ignores a very real problem. Almost none of their users are engaged. They’re just people who clicked a link titled “OK” when faced with Google+. It’s similar to newspapers goosing up their page views using hacks like slideshows or multi-page articles. At some point you have to wonder who you’re really fooling.

Engagement is just one piece of the puzzle, but it is so frequently ignored that there's lots of quick wins to be had. Here's three ways to increase user engagement.

1. Make a strong first impression

Every day a potential customer is seeing your interface for the very first time. This can be forgotten within teams that are always designing for themselves. The first screen your users see has three important jobs:

- Explain how your application works.
- Motivate your users to get started.
- Let your users know how to get help, if and when they need it.

Web applications that do none of the above get the exact behavior that they designed for. The users will log out and most likely never return.

There are lots of great ways to welcome a user to your app and show them around. There are blank slates, tutorials, dummy data, etc. A good first step that we encourage Intercom customers to do is to post a welcome message to greet each new sign up personally. This works great for starting conversations, which in turn increases conversions.

Simply communicating with your users is a great way to encourage them to ask questions, try out features, and stick around. By starting a dialogue they're far more likely to say things like "How can I email inactive users" or "How should I use tags?".

At best you'll win yourself more customers, and at worst you learn what's missing or misunderstood in your application.

2. Gradually expose the depth of your product

Every worthwhile application has features that aren't immediately apparent or useful. These can include quick-wins such as email notifications and alerts, third-party integrations, export features, and even small optimizations like keyboard shortcuts.

These deeper benefits can't be called out immediately. After all, who cares about data export before they have data, or keyboard shortcuts before they've used any features?

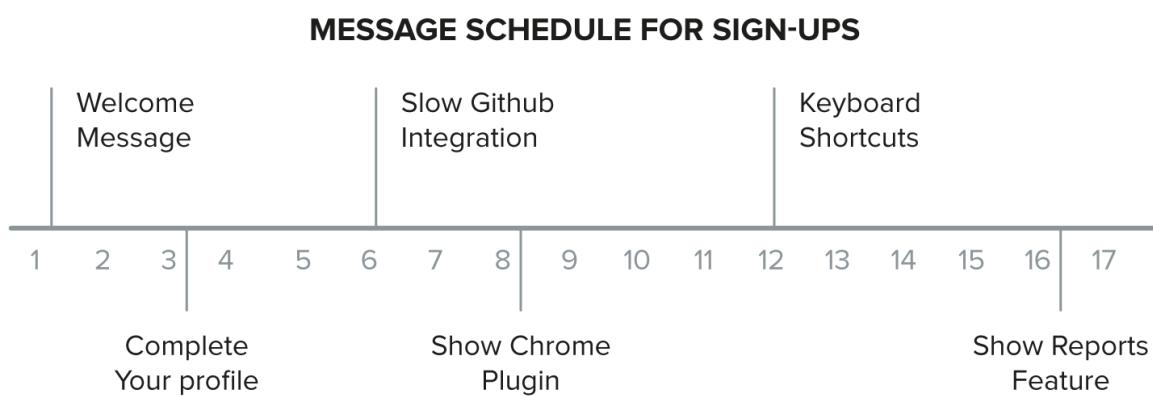
Most product owners tend to try to expose these features either through badly timed email blasts, documentation, or an FAQ. None of these approaches work well.

Emails arrive out of context, out of time, and are more likely to interrupt a user than to interest them. Their response is to archive your message and stop reading future emails.

Leaving features to be explained and promoted in your FAQ or help sections means their only chance of exposure is when a user has experienced an issue with a part of your product. This isn't the ideal time to distract them with other features.

We advise our customers to create message schedules to gradually promote certain features. When you have good insight into your userbase you can work out which secondary features delight users and at what point they're useful. Once you've done that it's just a matter of timely communication. Intercom lets you stagger messages over a series of sessions. This means each time a user logs in you're showing them more and more reasons to stick around.

Here's an example engagement routine from one of our customers:



Always end each message by letting your customers know that you're there to help if they have any questions. This is key to getting feedback, which helps you tweak your engagement routine.

3. Announce features and improvements in-app

Users don't notice when your product development slows down. They're logging in to use your product, not monitor your development progress. However, if things go quiet for long enough, they'll be easily distracted when a competitor releases a new feature, whether valuable or frivolous.

The only time your users care about features or improvements is in your application, so that's exactly where you should be announcing them.

We see a 10x increase in communications (as do our customers) from in-app messages over email announcements, and there are other, softer, benefits too. For example when we announced the new map sharing features in Intercom, our users clicked straight through, were immediately impressed and started sharing the location of their customers around the world. I can't imagine an email achieving a similar effect.

Develop your message schedule

Scenario: You're a product manager who wants to get your customers engaging with your product more often, when they're outside the office. To achieve this you've developed a mobile version of your app. Now you need to tell your customers about it.

The zero points method here is a blanket email shot, to all registered users, right now, with a link to the mobile app. What happens next?

Some customers will receive it while at their desk. Some receive it while using the website on their phone. Some will receive it who haven't used your product in years. Some get it at 4AM, others at 10AM.

Your reporting tool tells you that you had a 4.43% click-through rate and you'll be disappointed as this was a huge stressful effort. The worst response is to say "5% clicked? That's great, Just sent 19 more of those mails and we're golden."

Can't we do better than this?

Alternative approaches

You could split the customers into two groups: those who've previously logged in to the product from their phone, and those who haven't. That would let you target a better message to each group.

You could create a permanent message in the app for anyone using it on a mobile device so they know what they're missing. You could email everyone after they next log out, so you know that they've recently used the app, and it will be fresh in their minds. You could write a hilarious message, poking fun at competitors, and include a funny graphic. That'll get people talking which helps spread the word.

You could take a lesson from Slack who make their release notes in the App Store funny and enjoyable to read. e.g.

"Fixed: We've fixed the loophole in the last release that allowed Gif to be pronounced "Jif". As you were.

Fixed: When the keyboard was dismissed by swiping down, the chat window would then partially scroll back up, which was interesting, but not in a good way."

All of these are options that go beyond the default "Let's just mail everyone loads" approach that most companies pick.

Be comfortable killing a feature

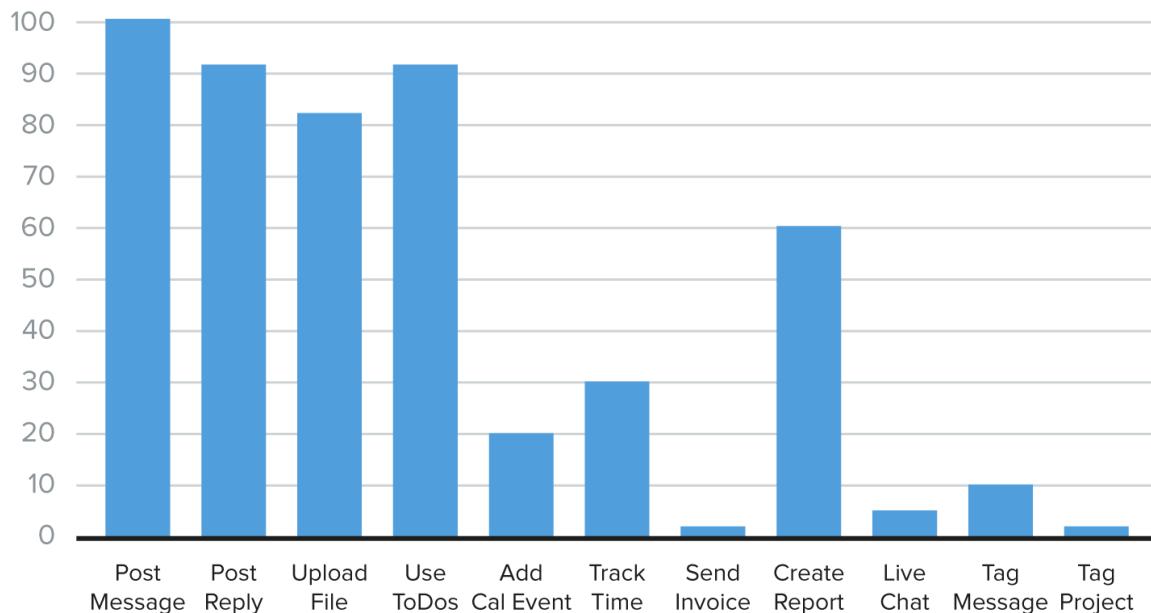
When you think about feature creep and bloated products what comes to mind? Endless tabs, toolbars, settings, and preferences, right?

For a five year period you couldn't sit through a talk on UX or product strategy without seeing some variation of the infamous Microsoft Word screenshot, with all the toolbar options turned on, leaving a tiny space at the bottom of the screen for the user to type anything. It's the perfect example of the need for simplicity.

Poor user interface like that highlights the cost of feature creep, which is why you need to be comfortable killing a feature. There is simply too much stuff for users to comprehend. The problem is that horrible screens like that Microsoft Word one don't inform the product manager about what to do next. Sure there's too much shit on the screen, but what do you do about it?

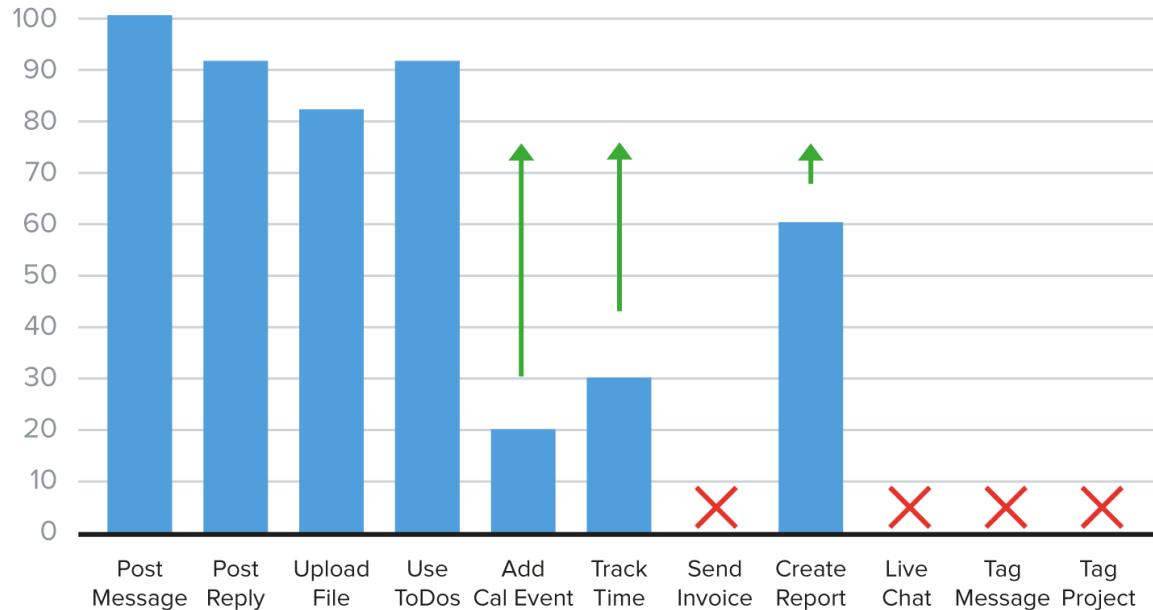
To solve feature creep you need to identify which features are being adopted by everyone, and which ones are struggling to gain traction. It's time again to run the feature audit, explained in Chapter 1.

RUN YOUR FEATURE AUDIT AGAIN



When you have any type of long tail of features that have little usage, you've gone off course, or your strategy is "build everything that everyone wants". The latter might sound crazy, but products like Microsoft Word have very done well with that plan. But you're probably not building Microsoft Word. If you want a well defined product, you must be comfortable killing features.

Fish or cut bait



Features struggle for adoption for lots of reasons, users don't see the value in using it, they find it too complicated, it's hidden somewhere clever in your product, they're the wrong type of users, and dozens more. This is why it's important to say no. When you're faced with a feature that only 8% of your user base are using, you have to make a call: Kill it or Keep it.

Killing a feature can take many forms, you can hide it from new users, message current users to explain your decision and give them time to prepare. Or you can just tear it out of the UI and never mention it again. Both work, with varying amounts of overhead. The bigger the product i.e. the more users you have, the less you get away with dramatic moves.

Keeping a feature that has no adoption requires design and communication. You need to target the users who are using it, and work out why. Target some users who aren't, and work out why. Obviously this is where a tool like Intercom shines as you can target the users and start the conversations in seconds.

Often you'll find simply promoting the feature through in-app messaging works. We've worked with customers who have driven up to 30% increases in

feature adoption simply by messaging the right type of users at the right point in their lifecycle. Similarly we've seen customers get enough raw feedback from customers to make it clear that a feature simply isn't valued, which makes its removal painless.

Good product owners let in very few dud features. Great ones kill them on sight.

Conclusion: A work in progress

We had to make difficult calls about what points and pieces to include in this compilation. We tried to zero in on the posts that would be useful for someone either early in their career or who has recently transitioned to product management. Our thinking on how to run, manage and create products is a work in progress, and this is our first stab at compiling it.

If you found this useful check out our blog [Inside Intercom](#) where we regularly post our thoughts on product management and related topics such as design, the business of startups and customer success.

We don't have all the answers, but we're growing a healthy list of questions.

Thank you for reading this book

We'd love if you shared your newfound wisdom with friends:

[Facebook](#) | [Twitter](#) | [Linkedin](#)

Resources - Section 2

- <https://air.mozilla.org/the-role-of-a-product-manager-everything-and-nothing/>
- <http://www.cision.com/us/2015/12/10-new-facebook-features-for-2016/>
- <http://www.cision.com/us/2016/01/8-new-features-on-twitter-in-2016/>
- <https://medium.com/earnest-product-management/3-types-of-product-management-dec4b2d77271>
- <https://hbr.org/2014/03/five-questions-to-identify-key-stakeholders>
- <https://www.oracle.com/index.html>
- <https://www.salesforce.com/>
- http://www.huffingtonpost.com/brian-de-haaff/the-product-manager-vs-pr_b_8040402.html
- <http://www.instructables.com/id/How-to-build-a-card-house/>
- <https://news.ycombinator.com/>
- <https://www.producthunt.com/>
- <https://www.launchticker.com/>
- <https://www.techmeme.com/>
- <https://becomeaproductmanager.signup.team/>
- <http://www.evansperks.com/>

Want to ask a question and have us answer it? Film your question (with anything) and throw the file in this balloon page:

<https://balloon.io/becomeaproductmanager>

The 4 major phases of the Product Lifecycle



Covered in this lecture:

What the four phases of the product lifecycle are and how they look like

Taught by:



- ▶ Once a company has developed a new product, the product goes through four main stages
- #1 Introduction
 - the company introduces the product to the market
 - there is little to no competition
 - the business typically loses money on the item
- #2 Growth
 - the product has been accepted by the marketplace
 - the sales start to rise
 - the company starts improving the product
 - there are still few competitors

● #3 Maturity

- sales will reach their peak
- more competitors enter the market
- the company finds it difficult to compete

● #4 Decline

- the company reaches its saturation point
- the sales begin to diminish
- products are phased out from the marketplace
- the product is deemed old or irrelevant



See you next lecture!



6 Products 4 Phases

In the following pages we're going to present you with a product and a brief description. It's up to you to determine what phase of the business you think they're in.

Ready? OK!



GrouponTM Getaways

GROUPON GETAWAYS is a branch of the discount coupon site, but instead of ten dollar mani-pedis, you're offered deeply discounted vacation packages.

Groupon launched in 2008 and has since been under fire for things like causing businesses to go bankrupt, a controversial Superbowl ad, and eventually an ousted CEO.

INTRO

GROWTH

MATURITY

DECLINE



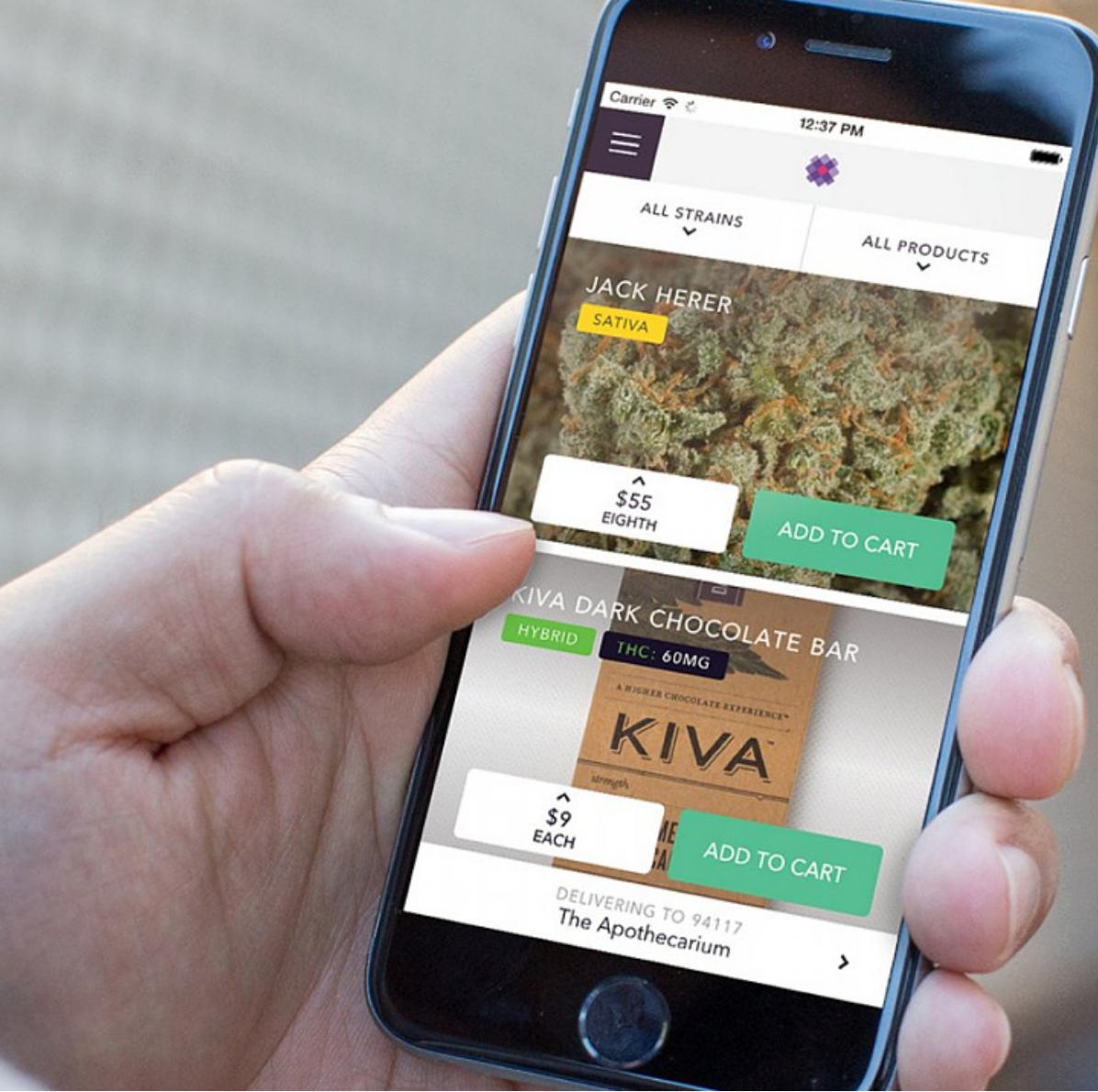
Amazon's DASH BUTTONS are small, adhesive Wi-Fi-connected buttons that allow you to re-order household products...with the push of a button. When they were introduced in March 2015, people thought they were an early April Fool's joke. Make sure to keep out of reach of toddlers and jerk friends or you might find yourself with 1,000 rolls of toilet paper.

INTRO

GROWTH

MATURITY

DECLINE



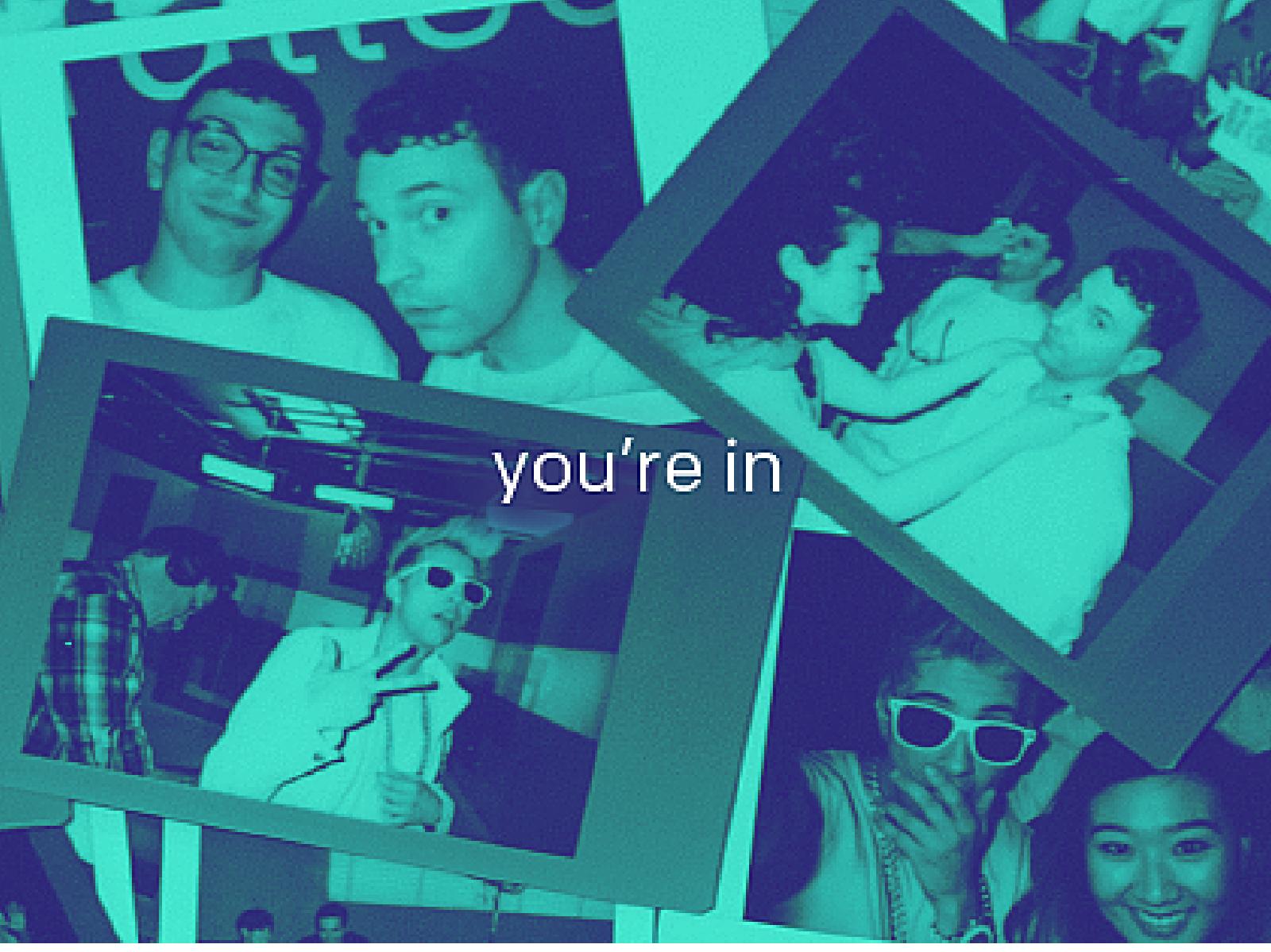
MEADOW is a cannabis delivery service, nicknamed the Uber-for-Weed. They also focus heavily on cannabis-as-medicine education. The U.S. medical marijuana industry is growing fast as more and more states legalize it. Don't know about Meadow? You'd be a lot cooler if you did.

INTRO

GROWTH

MATURITY

DECLINE



you're in

FULLSCREEN is the first demographically-targeted video on demand, mobile streaming service. Their plan is to utilize YouTube celebrities to act in their series' while bringing back re-runs of shows like

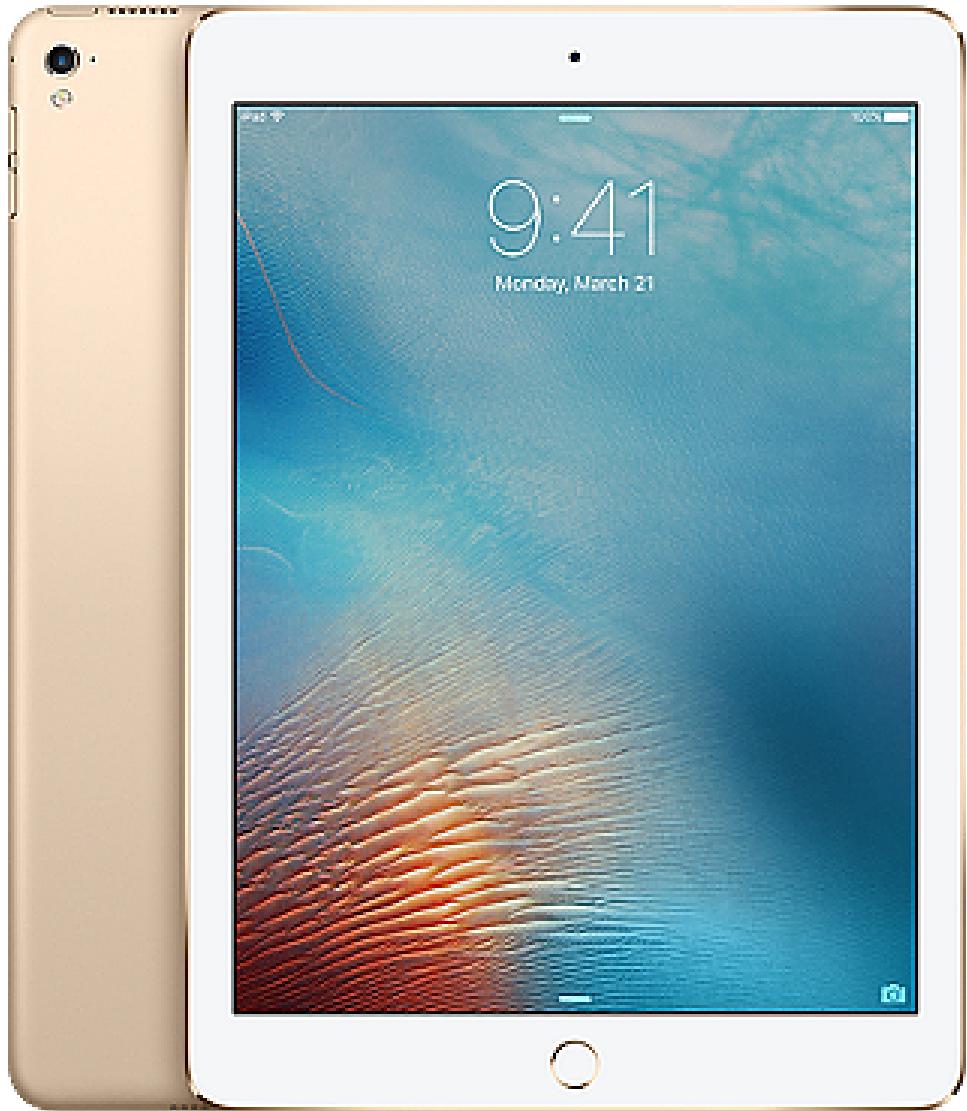
Saved by the Bell and Daria. This combined with the rise of mobile usage is in the hopes of attracting a certain age group and a whole lot of money. Fullscreen's launch date is set for late-April 2016.

INTRO

GROWTH

MATURITY

DECLINE



The 9.7-INCH IPAD PRO is the latest edition to the iPad family. The iPad has been responsible for a 75% increase in insomnia and the popularity of Battlestar Galactica. Just kidding. The IPAD is like a computer, but it's not, ya know? This version hopes to attract those with iPad V.1 or those who think the jumbo 12.9-inch iPad is just too much. It's too much.

INTRO

GROWTH

MATURITY

DECLINE



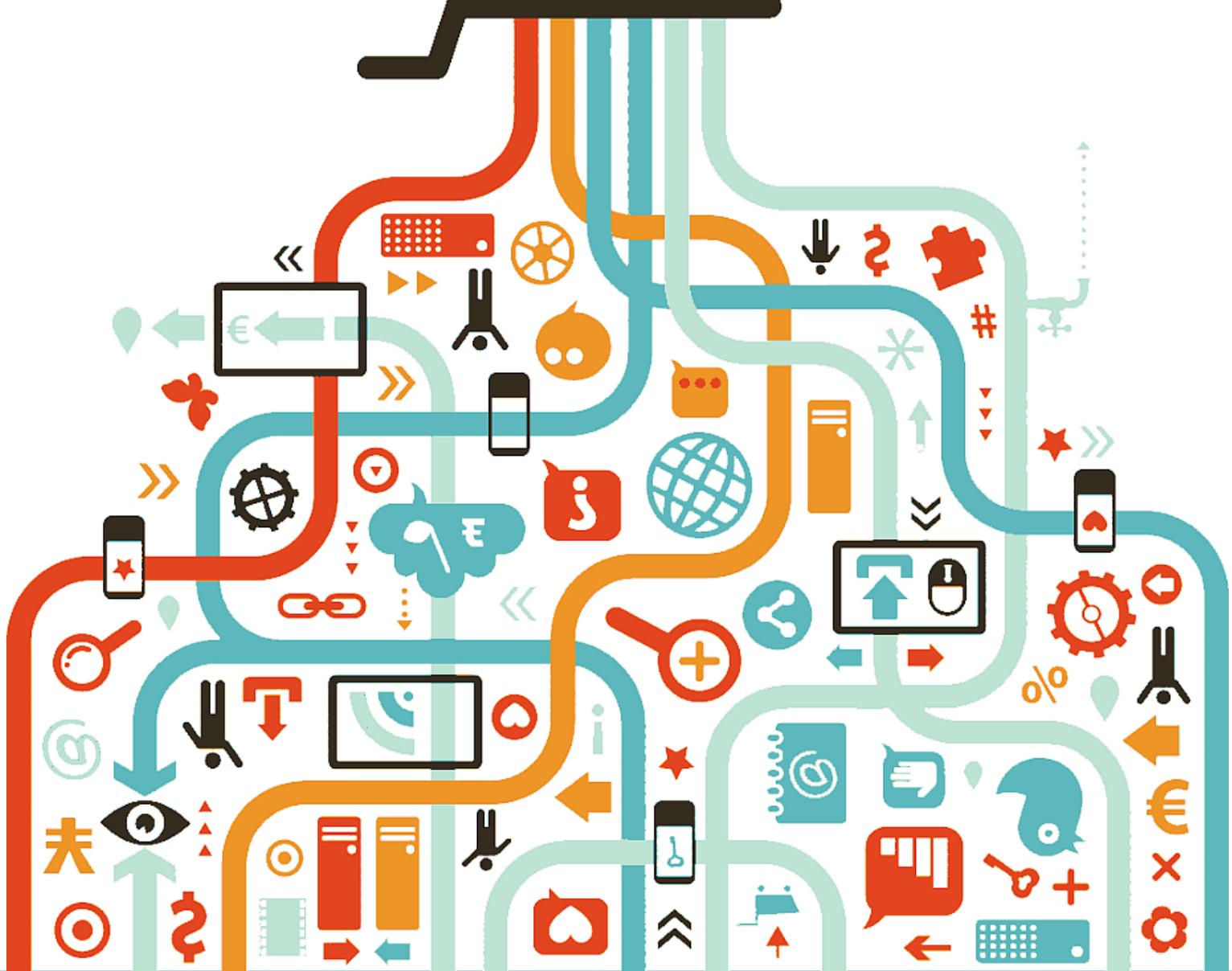
Although streaming sites like Spotify and - ahem - Soundcloud, make listening to music easy and convenient, VINYL RECORD sales grew for the tenth consecutive year in 2015. In fact, vinyl and music streaming sites account for the same share (roughly 9%) of music revenue in Millennials ages 18-34 (Nielsen, 2015).

INTRO

GROWTH

MATURITY

DECLINE



So...what phase are these products in? Where do you think these products will be in a year?

INTRO

GROWTH

MATURITY

DECLINE

Post your answers in the discussion.

Product Lifecycle phases

- Real examples



Covered in this lecture:

Real life examples of products
that are in each of the
four product lifecycle phases

Taught by:



- ▶ **Introduction** - rythm.co
 - ▶ **Growth** - Snapchat
 - ▶ **Maturity** - Twitter
 - ▶ **Decline** - Yahoo
-
- You can check out Google Trends to see how a company is evolving and in which stage it might be at the moment

See you next lecture!

The product development process



Covered in this lecture:

The steps you need to follow before launching your product

Taught by:



- ▶ There are 7 phases that products go through while they're being developed:
 - > **Conceive**

- > Plan
- > Develop
- > Iterate
- > Launch
- > Steady state
- > Maintain or kill

● #1 Conceive

- we collect user problems and brainstorm solutions
- the biggest source of ideas is usually inside of the company

● #2 Plan

- market research on the ideas we got in phase #1
- customer interviews
- roadmapping

● #3 Develop

- make timelines and write our features
- user stories, specs, estimations on very specific details
- setting the requirements

● #4 Iterate

- we finish the MVP or the early prototype
- we test the assumptions that we've made
- we use tools for testing different versions of the product



See you next lecture!

Getting deeper into the product development process



LECTURE
SUMMARY

Covered in this lecture:

How to make decisions
about your product
that benefit the company

Taught by:



- ▶ There are 7 phases that products go through while they're being developed:
 - > Conceive
 - > Plan
 - > Develop
 - > Iterate
 - > Launch
 - > Steady state
 - > Maintain or kill
- #5 Launch
 - working with the marketing team, the legal team, the PR team, the sales team, and position the product for public launch
 - we launch it and see what reaction we get

- **#6 Steady state**

- collecting metrics and optimizing them
- sales continue

- **#7 Maintain or kill**

- analyzing the data: how competitive the product is, what is the return on investment
- we decide if we kill it or maintain it
- even if it produces revenue, it could just not fit the company's vision anymore
- if we decide to kill it, we do "sun-setting", which is a slow transition to the end of life of the product

► This is a natural process and every product goes through these phases



See you next lecture!

What is Lean?



LECTURE
SUMMARY

Covered in this lecture:

Explaining the lean framework
and the lean mindset

Taught by:



- ▶ The lean framework is a product development philosophy that revolves around cutting all of the unnecessary work or effort until you're absolutely sure you need to do it
- Example: Creating a food delivery app

Normally, you would hire drivers, buy a cell phone number, and build the actual app

The lean way of doing this is doing everything yourself in the beginning, trying to get as far as you can by using the least amount of resources

- Being lean means you're not building anything until you know for sure that there is an interest in it

See you next lecture!

What is Agile?



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Explaining the Agile framework

- ▶ Agile is a way of applying the lean mindset to software development
- In the Agile framework, we group things into small batches and do them one by one, in order to not waste resources

Example: Researching the most important 2-5 features of a product instead of developing all of them

See you next lecture!

What is Scrum and how does it work?



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Explaining the Scrum methodology

- ▶ There are two Agile development methodologies:
Scrum and Kanban

Scrum is the most common and it works like this:

- **1. The sprint planning meeting**

- you take the most important features from the top of your product backlog and you move it to the sprint backlog
 - you talk about what needs to be done in order to implement it
 - you put the work into a project management software, and into tickets

- **2. The start of the developing process**

- a sprint usually takes 2 weeks
 - your team works on the tickets by taking them off the top of the sprint backlog and moving them to "In progress" and then to "Done"
 - at the end of the 2 weeks, you should have completed everything in the sprint backlog; if not, they go into the next sprint

● 3. Standup meetings

- daily meetings held in the mornings
- people remain standing during the meeting, in order for it to remain brief and concise
- every team member makes a summary of their work

● 4. Retrospective meetings

- you meet with your team at the end of each sprint
- you talk about 3 main things: the last sprint, what went well and what didn't, any questions people have

See you next lecture!

What is Kanban and how does it work?



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Explaining the Kanban methodology

- ▶ Kanban is not as strict as Scrum in terms of meetings and times
- A Kanban board has columns with cards that you can move from one column to another, to reflect the state of the item: "To do", "In progress", or "Done"
- How many items can be in each particular state is up to you or your team to decide
- Kanban doesn't use sprints
 - > There is no sprint backlog, only the product backlog itself
 - > The team works on their ticket, they move it to done, and they take the next task off the top of the product backlog
- Kanban doesn't prescribe any particular meetings types
- Kanban is more relaxed, but it makes it more difficult to evaluate how much time it's going to take to develop items

See you next lecture!

Waterfall



Covered in this lecture:

Explaining the Waterfall framework

Taught by:



- ▶ The Waterfall framework is the opposite of Agile
- ▶ In the Waterfall framework, we take all the features of a product and develop them all at the same time
- Doing things in the Waterfall way is riskier
- It's much harder to adapt to the market feedback after you've already built everything

See you next lecture!

Real life examples of Agile and Waterfall



LECTURE
SUMMARY

Covered in this lecture:

Examples of where you can use the Agile and Waterfall frameworks

Taught by:



► Agile

Example: Creating a music playlist in a music app

- everyone on the team gets together and decides on core features
- developers create the database, product managers create wireframes
- implementing basic functionality
- everyone is in close communication and collaboration

► Waterfall

Examples:

- Operating systems that can't operate only with a few features
- Engine control systems in cars
- Building skyscrapers

See you next lecture!

Resources - Section 3

<https://www.technologyreview.com/s/601425/in-global-shift-poorer-countries-are-increasingly-the-early-tech-adopters/>

<http://allthingsd.com/20130723/sunrise-sunset-why-companies-kill-products-we-love/>

<https://www.fastcompany.com/3028213/how-to-manage-your-startups-fast-growth>

https://www.chicagobooth.edu/entrepreneurship/docs/aec_2006_11-30_overcoming_growth_plateau_final_study.pdf

<http://www.timeslive.co.za/sundaytimes/stnews/2016/05/06/Millennials-spinning-the-%E2%80%98vinyl-revival%E2%80%99>

<http://www.billboard.com/articles/columns/chart-beat/7348566/record-store-day-gain-vinyl-album-sales>

<https://www.bloomberg.com/news/articles/2016-04-21/groove-masters-of-the-vinyl-revival>

https://www.washingtonpost.com/entertainment/music/vinyl-revival-czech-record-maker-aims-for-global-dominance/2016/05/06/e87c44ec-1373-11e6-a9b5-bf703a5a7191_story.html?utm_term=.c814ecfaff9b

<https://www.recode.net/2016/4/27/11586476/twitter-is-going-to-have-a-hard-time-fixing-its-ad-problem>

<https://rythm.co/>

<https://www.quantcast.com/>

<https://trends.google.com/trends/>

<https://hbr.org/2014/09/how-to-market-test-a-new-idea>

<https://beta.apple.com/sp/betaprogram/>

<https://www.forbes.com/sites/neilpatel/2015/03/16/8-elements-of-a-robust-product-launch-strategy/#2f9a9fa0220a>

<https://www.fastcodesign.com/3058625/how-products-can-die-gracefully>

<https://hbr.org/2015/07/know-when-to-kill-your-brand>

<https://hbr.org/2016/03/lean-strategy>

<http://theleanstartup.com/>

<https://hbr.org/2016/05/embracing-agile>

<https://www.wrike.com/library/ebooks/the-state-of-agile-marketing-2016/>

<https://www.infoq.com/news/2016/05/agile-marketing-survey>

<https://hbr.org/2016/04/the-secret-history-of-agile-innovation>

<http://agilemanifesto.org/>

<https://en.wikipedia.org/wiki/Kanban>

http://www.toyota-global.com/company/vision_philosophy/toyota_production_system/just-in-time.html

https://www.gamasutra.com/view/feature/181992/waterfall_game_development_done_.php

<https://www.ibm.com/developerworks/rational/library/4243.html>

<https://www.scrumalliance.org/agile-resources/mayden-s-transformation-from-waterfall-to-scrum>

Intro to Ideas and User Needs



LECTURE
SUMMARY

Covered in this lecture:

Introduction to Section 4

Taught by:



- ▶ Any product was first an idea or a need in someone's mind
- ▶ The product manager is **NOT** the idea person
- ▶ Your job is to gather ideas from everyone and decide what to build and when to built it, given all the things that are happening inside the company and in the market

See you next lecture!

Where ideas come from as a PM



Covered in this lecture:

The main sources for gathering ideas as a product manager

Taught by:



- ▶ Any company you will work for as a PM will probably have a long list of ideas that need to be developed. But where do you get new ideas from?

Ideas come from 4 main places: E.M.U.C.

- **Employees** - coworkers, management, and yourself
- **Metrics** - problems and inefficiencies you find when you're looking into how users use your product
- **Users** - user feedback from forums, emails, social media
- **Clients** - this applies primarily to business to business product managers

- ▶ Depending on your PM role, ideas might come from different sources:
 - **Internal PM** - stakeholders
 - **Business to consumer PM** - users, metrics, coworkers
 - **Business to business PM** - employees, clients



See you next lecture!

Getting to the real user needs



Covered in this lecture:

Figuring out if you are actually solving a real problem

Taught by:



- ▶ Once you have a list of ideas, you don't just go out and build them
- ▶ A big component of the product manager mindset is being able to understand the real problems behind what people ask you to build
- ▶ If you give people what they say they want, you're not always solving the actual core problem
- Being a product manager is all about finding a solution to a problem, instead of trying to fit the problem to a solution

- ▶ When someone asks you to build something, ask yourself:
 - Is this solving an actual problem?
 - Can this have any unintended side effects?
- The easiest way to get to the core issues of a request is to ask "Why?" for three times in a row, until people get to the real pain point



See you next lecture!

Namaste

You have officially launched a meditation app. Welcome to the world of helping others find their centers, align their chakras, and reclaim their True Norths.

The user feedback is rolling in...take a look at a few below. Think about what core issues they allude to or what their comments might suggest. Think past the problem. What should you take away from these comments? Post your thoughts in the Q&A Section.



“

All of these meditation apps say the same thing. You can find this stuff for free on YouTube. Not worth it.



“

I enjoyed the speakers relaxing voice. However her technique did not help me relax. I feel that she encouraged me to keep focusing on the negative thoughts and pressure, rather than just observing them and letting them go. Ideally, she would instruct us to return our focus to the breathing and let go of the “pressure” (or just ignore it and let it be). Otherwise, if one continues to FEEL the pressure, it will bring more energy to the pressure (instead of drawing energy and attention AWAY from it). I am not suggesting one should suppress the pressure feelings, but we should not give it that much attention either. Returning to the breath is ideal.



“

The one thing that I don't like is that you have to pay for the extra packaging I would really love to try the emergency packages since I have such a bad anxiety and just random attacks all the time but \$12 a month has me a little hesitant on my budget. I wish you didn't have to pay so much.... As to any money at all to get help. Because I feel like 10 days for me just isn't enough.



“

I have gotten hours of use from this app. There are two different voices that lead meditations, one is calming and pleasant, the other sounds like a deflating circus balloon. Why cant i choose which voice I get? The paid subscription for this app is really high as far as apps go so I should be able to have options. Also, why are all these meditations on tracks? Why cant I customize my meditation program?



“

Bit of a sham, isn't it? Not much content outside of the paid section, which wouldn't be as unfortunate had the app's adverts not insinuated there was more access to understand what it is this has to offer. The price levels are quite steep! The free app requires you to register, so you can be advertised to, and then pushed into a subscription.



“

The app won't allow me to stay signed in. When you're using the app almost daily it's extremely annoying. Either save my password or allow me to use TouchID but stop making me sign in every dang time. Some of us take security seriously and use long, hard to guess passwords it's annoying to have to go into 1Password to copy and paste my password every time I want to open this app. This almost never happened before you released this new “better” app.



“

I seldom pay for apps, but I didn't flinch at the \$10/year charge. They're jumping to \$40, and I'll miss it, but it's not worth it as they keep adding content I don't use.



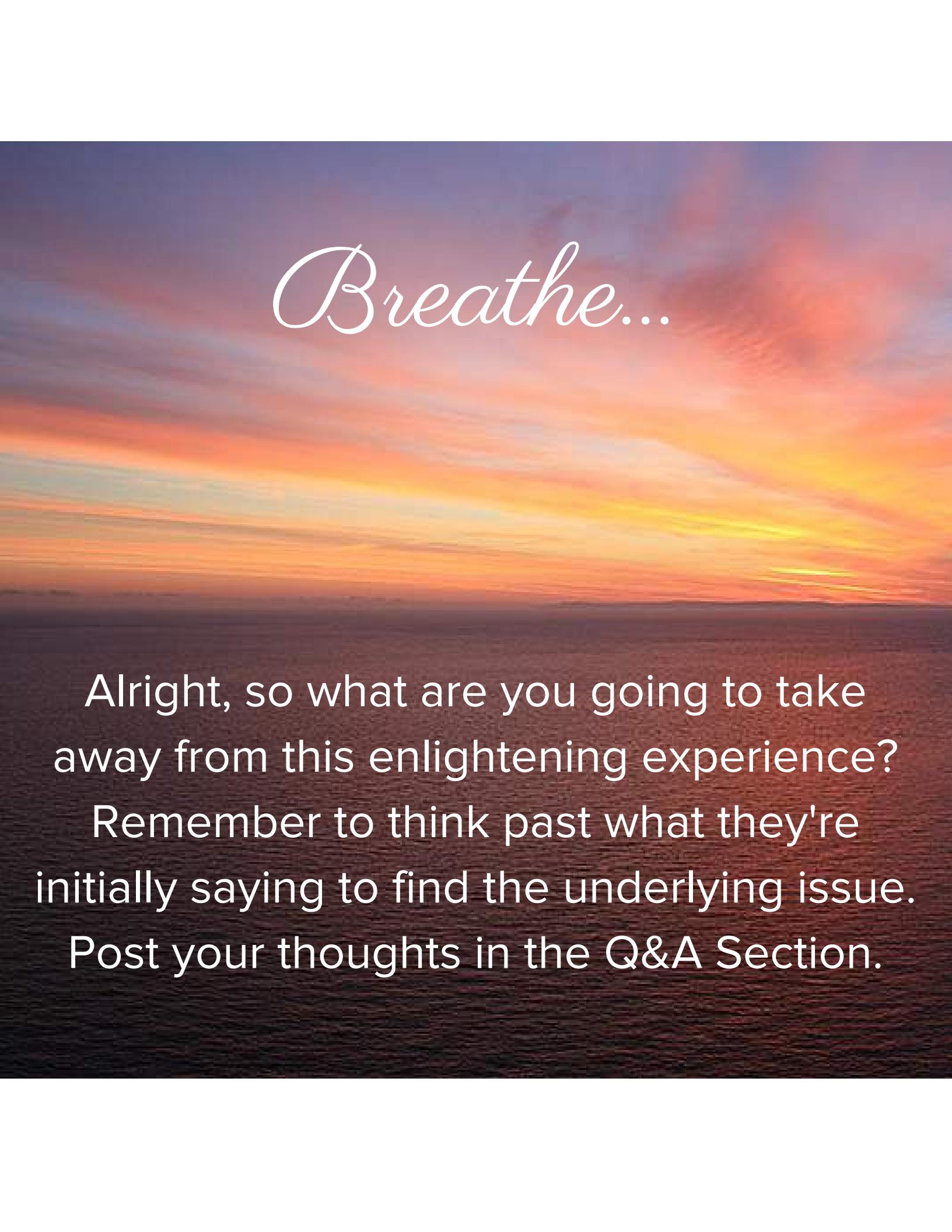
“

The queuing feature is unintuitive. I had the foundation courses in my queue and was in the middle of foundation 2 when I decided to add pro 1 to my queue, thinking it would tack on at the end of my queue, after foundation 3. Instead, it swapped out with foundation 3, and when I tried to add foundation 3 back it was locked because I haven't finished foundation 2. I asked customer support about this and they added foundation 3 back to my queue but only by telling the system I had completed foundation 2, which I haven't....so now my timeline and progress stats are incorrect. Queue would be better if you could queue up an infinite number of courses and move them around in your timeline (like Netflix queue :(



“

Can we gamify this? Like there's another app out there that helps you stay positive by playing games...so what if we create a SIMs-like world where we can all go and meditate together? And we can choose different backgrounds like being on a sunshine hill or under the ocean. And then we can each create our own meditation music playlists and share with each other and take turns DJing during our meditations. And we can be friends and see who meditates more and win things like free yoga classes nearby or shirts or a meditation bell. And also we can make a meditation game game where you like see who meditates the longest at one time.

The background of the slide features a photograph of a sunset or sunrise over a calm sea. The sky is filled with horizontal clouds, transitioning from deep blue at the top to vibrant shades of orange, yellow, and red near the horizon. The water in the foreground is dark and reflects the warm colors of the sky.

Breathe...

Alright, so what are you going to take away from this enlightening experience?

Remember to think past what they're initially saying to find the underlying issue. Post your thoughts in the Q&A Section.

Users vs. customers



Covered in this lecture:

The difference between users
and customers

Taught by:



- Sometimes, the people who pay for your product (customers) are not the same people that use it (users)

There is a different kind of feedback that you will get from each category:

- The company that buys the product will give you feedback about general features that they need it to have
- The actual users will give you feedback about technical issues they encounter while actually using the product

See you next lecture!

Resources - Section 4

http://www.huffingtonpost.com/quora/what-are-the-top-10-philo_b_2828845.html

<https://www.uxmatters.com/mt/archives/2009/01/the-ux-customer-experience-communicating-effectively-with-stakeholders-and-clients.php>

<https://medium.com/on-products/a-practical-guide-to-user-needs-89a1e0c03f95>

https://thenextweb.com/dd/2014/04/28/context-design-anticipate-users-needs-theyre-needed/#.tnw_xQKflRD7

<http://jacks.tumblr.com/post/33785796042/lets-reconsider-our-users>

<https://www.forbes.com/sites/katelee/2012/10/18/user-vs-customer-does-it-matter/#6c9407e65dcd>

Market research: Sizing the market



Covered in this lecture:

How to figure out the size of the market that you're getting into

Taught by:



- ▶ There are two common approaches to thinking about the market size:
 - **Top down**
 - based on finding the total market and then estimating what your share of that market is
 - it's a more optimistic approach
 - **Bottom up**
 - based on thinking about the current sales of similar products and estimating how much of those sales you can capture
 - it's a more conservative approach
 - this is the best way to do it but it takes more effort and time

- Tools and techniques you can use to look at market data:

- search Google for "industry reports for _____"
- compete.com - look at the amount of traffic the competitors' websites get
- use Google AdWords Keyword Planner - it shows you the volumes and related terms of searches on Google
- search Twitter
- search Reddit



See you next lecture!

Introduction to finding competitors



LECTURE
SUMMARY

Covered in this lecture:

What you need to know before starting to search for competitors

Taught by:



- ▶ You need to know how to collect information on your competitor and make a judgement on what that means for your product
- ▶ If you go into a market with a lot of competitors, you need to analyze them before tackling that opportunity
- ▶ If you go into a market that seems to have no competitors, you have to know what is the reason for that - there might be no customer demand, or maybe you actually have a brilliant idea
- As a product manager, you are in charge of feature triage

- The features you choose to build have to:
 - get you more users
 - make your users happy
 - enhance your brand
- ▶ You can't make decisions until you know what your competitors look like, what they are doing, and what they currently offer



See you next lecture!

Finding competitors as a PM



LECTURE
SUMMARY

Covered in this lecture:

The main methods you can use to search for competitors

Taught by:



- ▶ The most important thing is capture. You have to record every competitor that you find
- ▶ List your competitors under two categories:
 - Known
 - if you work in a company, you should already know who your biggest competitors are
 - if you don't know, search Google for "[your company name] versus"
 - Unknown
 - these are competitors that might not be that obvious
 - in order to find them, first you have to figure out what problem your product solves and for whom
 - use the following 3 techniques to find them on Google:

► **1. Channel the type of user**

- how exactly would the user complain about his problem?

- Google these exact complaints

>> You will find:

- Companies that are using that verbage

- Individual complaints online

- Google connects the dots, so you might find competitors

► **2. Search for "site: [popular website]" and your search term**

- popular websites: Reddit, Quora, Yahoo Answers

- describe what your product does and search for that

>> You will find:

- Companies using literal descriptions

- Ads at the top of the search list - those could be paid by competitors

- Google's input

► **3. Search for exact phrases using quotation marks**

>> Tip: How would you pitch your product?

Search for that exact phrase

See you next lecture!

Direct / indirect / potential competitors and their impact



LECTURE
SUMMARY

Covered in this lecture:

The differences between the four main types of competitors

Taught by:



- ▶ There are four types of competitors:
 - Direct
 - they're going after the same customer group as you
 - they're solving the same problem as you
 - customers have to make a decision between you and them
 - Indirect
 - they solve the same problem as you, but in a different way
 - they have a different target customer group, but which might overlap with yours
 - Potential
 - they offer something to the same target customer group
 - they don't address the same problem as you
 - they're also called "peripheral competitors"

● Substitute

- they solve the same core problem
- they're not angled and delivered the same way at all
- they don't generally target the same people
- customers could substitute your product with theirs

► You should be the most concerned with direct competitors

- Be competitive with direct competitors
- Make sure you don't lose too many customers to indirect competitors
- Make sure the potential customers can't do the same thing easily
- Be at least better than substitute competitors



See you next lecture!

The 5 criteria for understanding competitors



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Explaining the five dimensions you should compare yourself to your competitors on

● #1 The product core

- > Who is in charge of making the product?
- > How good is your competitor's product team?

● #2 The size of their user base

- > Having a large user base has certain advantages:
 - any time they launch a new feature, they can dominate the market
 - they have an easier time getting press coverage
 - it allows them to strike deals with other companies more easily
 - the larger the company, the better the product

● #3 Design

- > How good is the competitor's ability to make products that have great design?

● #4 Brand

- > How highly regarded is the brand of your competitor?
- > Having a strong brand is a huge competitive advantage
- > Strong brands demand a higher level of customer loyalty, can charge higher prices, and they get the benefit of the doubt

● #5 Speed

- > How quickly can they make changes?
- > As a company gets larger, they get slowed down



See you next lecture!

What's a feature table?



LECTURE
SUMMARY

Covered in this lecture:

What a feature table is and
what it's useful for

Taught by:



- The feature table is a comparison chart we use to compare our product to our competitor's product on a list of dimensions
- You need to know how competitive you are before you launch your product
- You don't necessarily need to make a feature table, but it's the best way to compare yourself to others

See you next lecture!

Putting together a feature table



LECTURE
SUMMARY

Covered in this lecture:

How a feature table looks and
what you should write in it

Taught by:



- ▶ Draw a table with multiple columns and rows
- On the X axis, put the direct competitors
- On the Y axis, put the features and factors you want to compare

Why should people choose your product?

- Think about what is important to your customer group
- Do some research

- Examples of features and factors:
 - price
 - reliability
 - it has to have feature x
 - ability to do y

- ▶ Make sure that your solution is obviously better than the others you're comparing yourself with

See you next lecture!

OCULUS RIFT



LET THE RACE BEGIN

MEET THE OCULUS VR HEADSET...



Virtual reality, or VR, is one of the latest emerging tech crazes and companies like Google, Facebook, Sony, and Microsoft are all in the midst of developing and launching hardware (headsets, controllers) and apps for VR.

People are saying that VR is going to change the way people do business, travel, socialize and more. Right now VR has launched mainly in the gaming markets, but the possibilities are coming.

Google released Google Cardboard a few months ago at around \$25 per in a smart move to open the market for VR (and the future of VR content), but the Oculus Rift (Microsoft) and Gear VR (for Samsung phones) are first-to-market in terms of headsets.



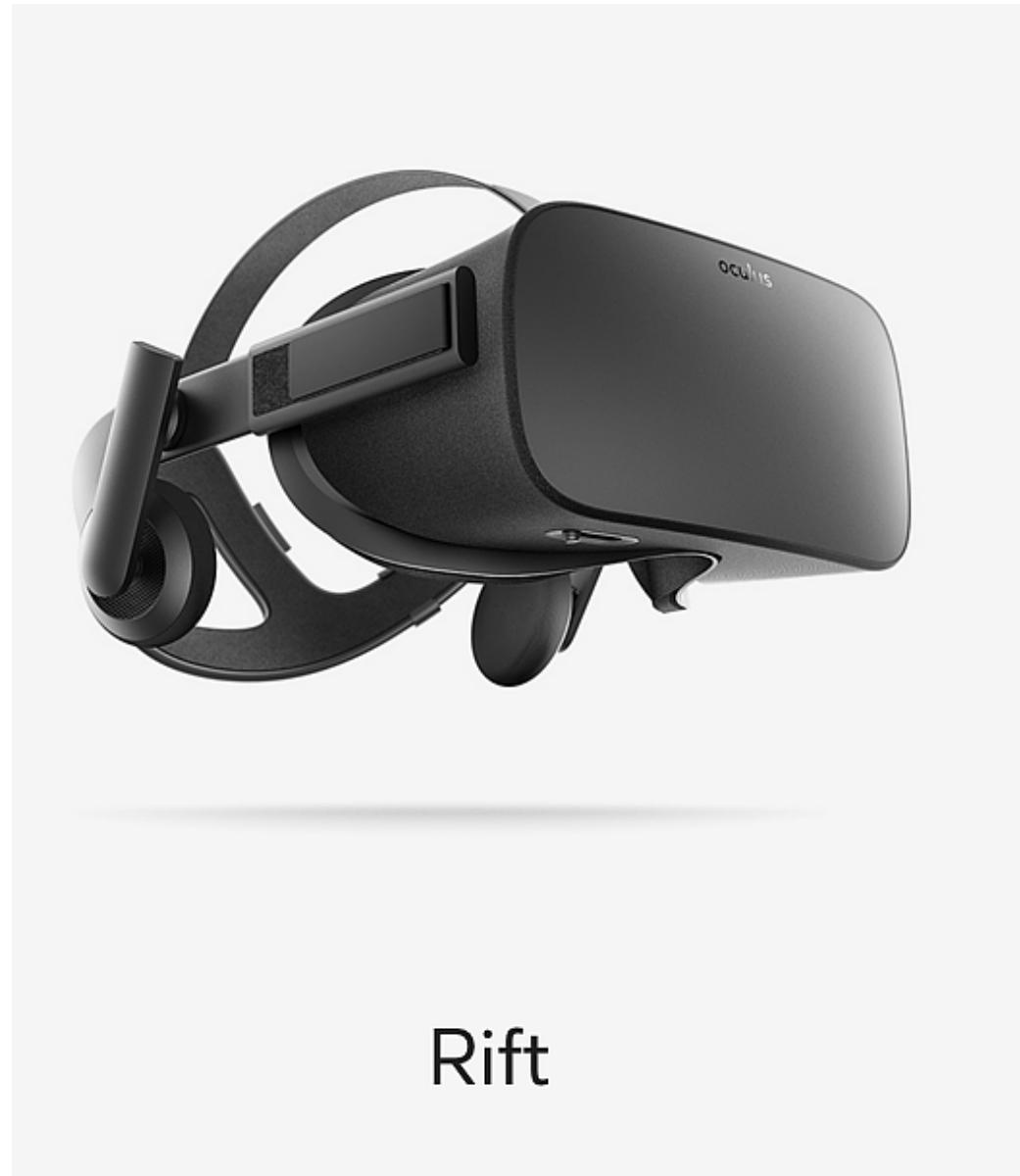
REACTIONS TO IT HAVE BEEN VARIED...



AND SOME WONDER IF THIS IS THE NEXT GOOGLE GLASS...

FEATURES

- Immersive, wide field of view
- High refresh rate
- Custom optics
- "Magic presence" (not real magic)
- Game(s) included with purchase
- Lightweight
- Customizable, comfortable design
- Integrated VR audio (headphones)
- Accessories: Oculus remote, VR sensor, Oculus Touch controllers
- Xbox One wireless controller with purchase
- Oculus-Ready PC (no Macs allowed)
- Guaranteed to polarize friends



This is an emerging market with emerging products.
What features do you think are going to be most important?
Who's going to win the VR race?

[Click here to finish the activity](#)



Analyze specific features



LECTURE
SUMMARY

Covered in this lecture:

Using a feature table for
comparing specific features

Taught by:



- ▶ After you build your product, you can compare it to what other competitors are doing
- Example: Comparing the discussion board across a number of platforms for online courses
- This type of feature table can be used when you propose a new feature
- See what others are doing wrong or right
How does your product differentiate?

See you next lecture!



Wunderlist

...and their not so wunderful dilemma.



What is Wunderlist?

Wunderlist is a multi-platform app that helps you manage and share all of your to-dos. Whether you're a freelancer managing multiple clients, planning a trip to Saigon, or just need a reminder to restock on mouthwash - Wunderlist makes it effortless.

Wunderlist was founded in Berlin by six friends who wrote out a business plan, posted an ad on a social media site requesting funding, and within three days two angel investors flew down and showered them in seed money.

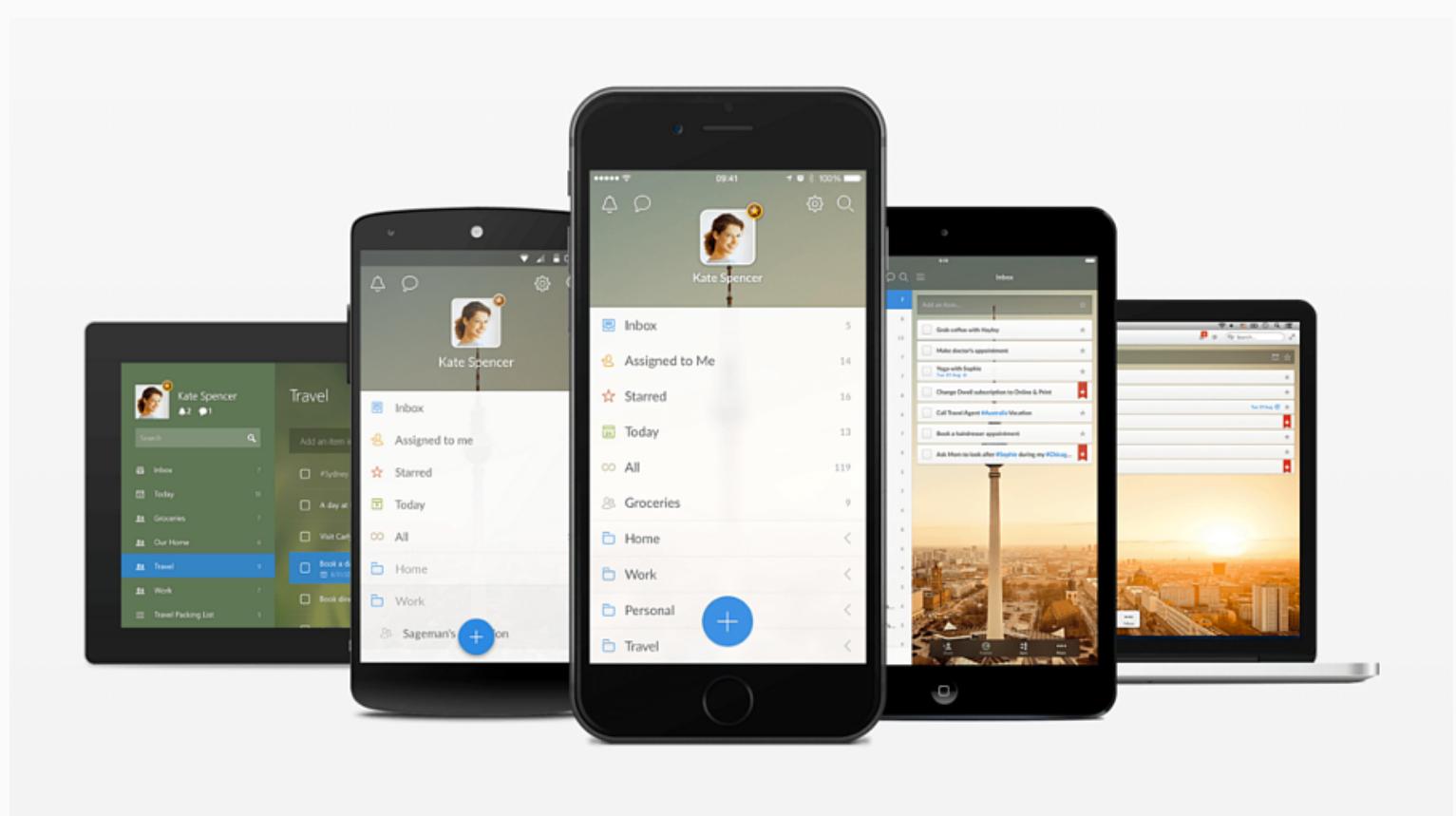
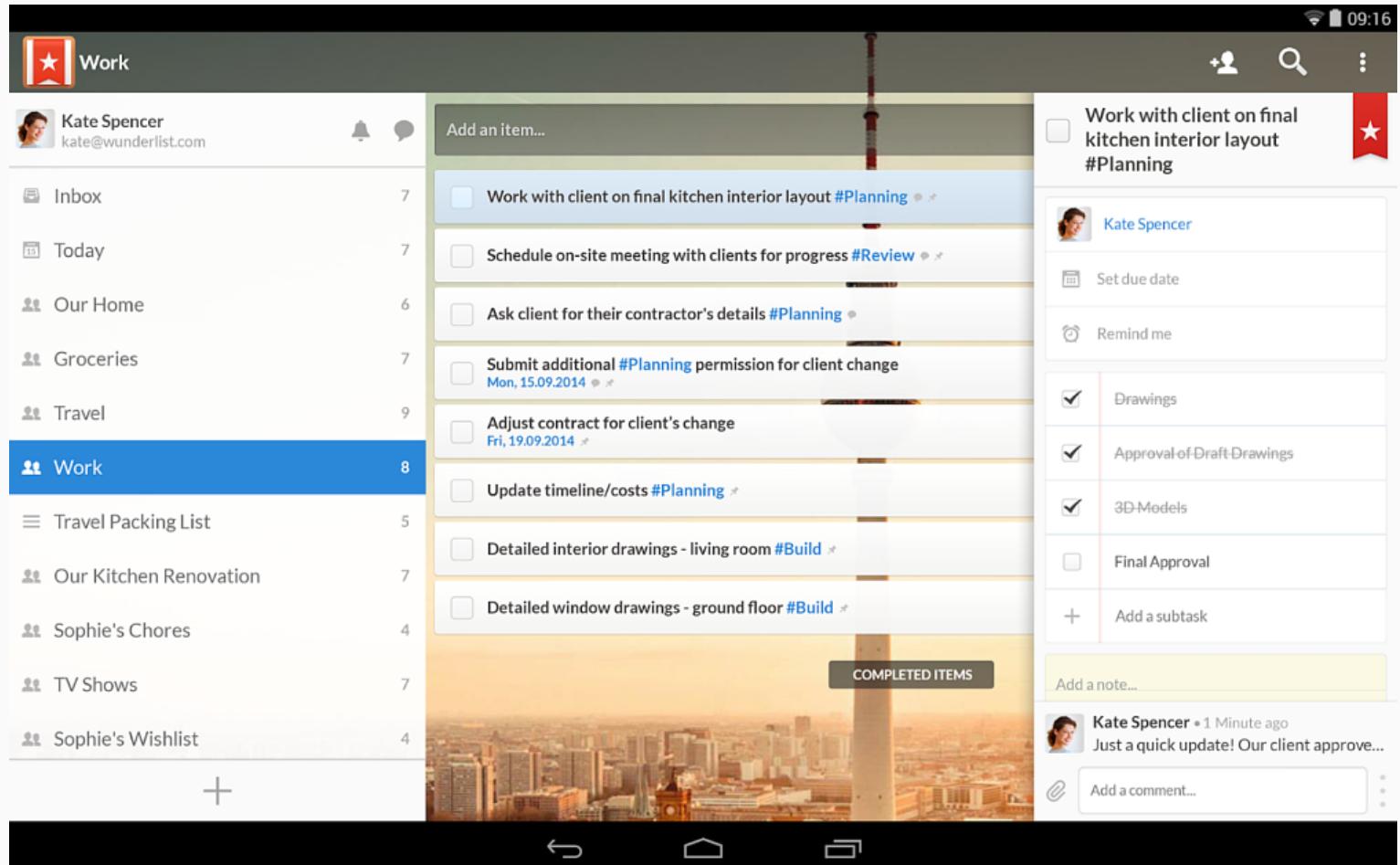
That was in 2010 - now Wunderlist is one of the most popular productivity applications in the world.

But it's not always raining money in wunderland...

Wunderlist has a classic Product Management dilemma: they offer a great feature set for their free version. The upside of this is that a great free version will attract a lot of new users. BUT, on the other hand, it leaves very little incentive to upgrade to Pro, which reduces total revenue and profitability

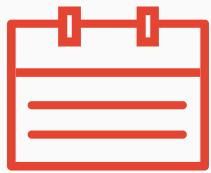
So if they reduce the free features, they'll make more money (good) but potentially lose competitive edge and see a slower growth in new users (bad).

Wunderlist started as a desktop app but has expanded to have companion iPhone, iPad, Android, Windows, Kindle Fire, Blackberry, and Web applications



Wunderlist by the numbers

A "wunderkinder" is "a person who is exceptionally successful in his field while still young."



5+

Years in business



13M

Total users



2.6M

Active monthly users

Monday is the most productive day of the week: together each Monday around 1,310,170 to-dos are created and 979,895 completed.



\$23M

Funding to date.



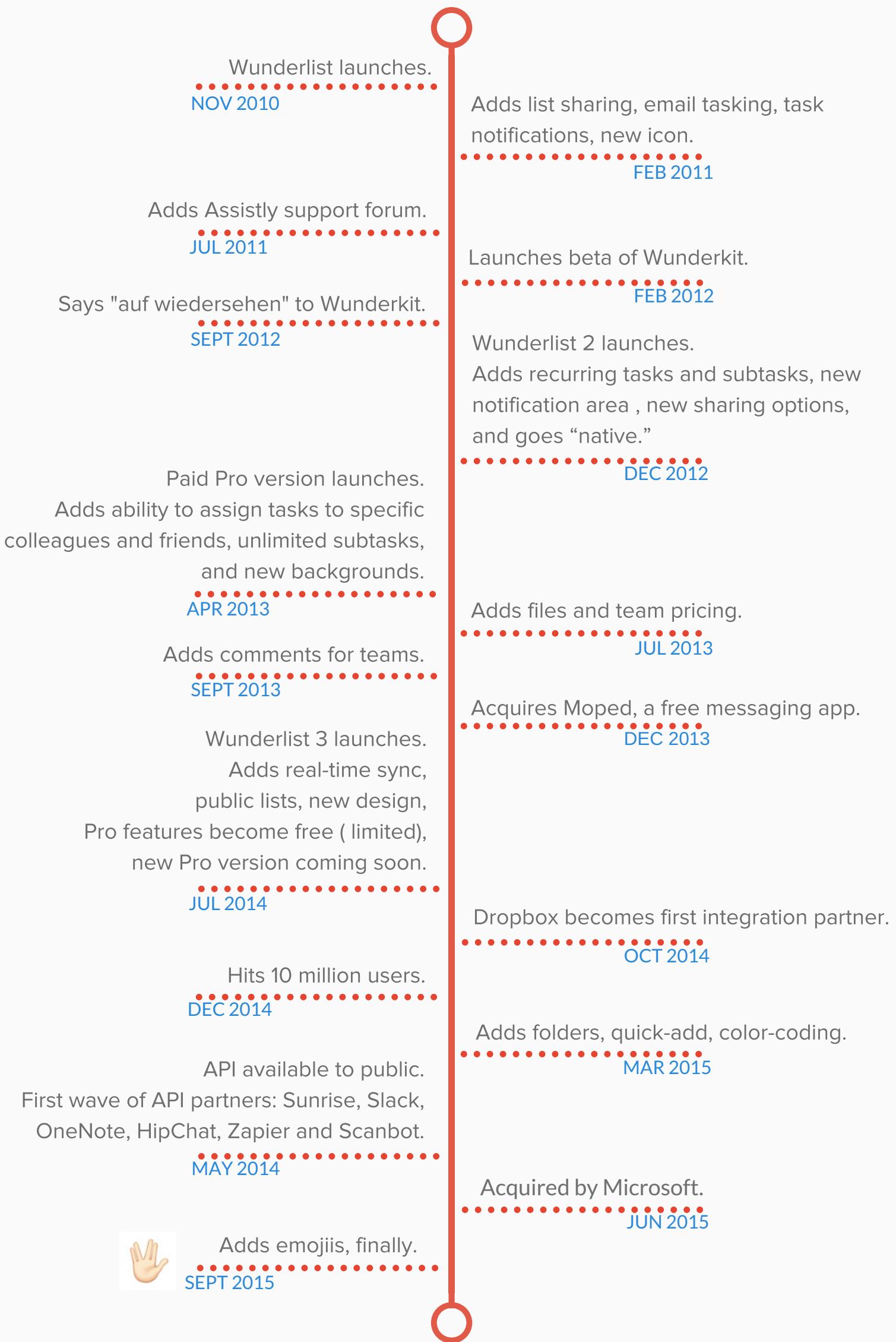
991M

To-dos created
to date.



742M

To-dos completed
to date.



Feature Table

Where do you think Wunderlist is most vulnerable?

What's their biggest opportunity?



Let's pretend that you're the latest and greatest Product Manager for Wunderlist...

The market is saturated with productivity-everything.

What you need to do is decide which is more important: increase users or increase revenue. Would you change the free/paid system? How?

- Drop features from free
- Add features to free
- Add features to paid
- Charge more
- Don't do anything
- Other

What are you going to do? Post your answers in the Group Discussion and make sure to justify your case.

What Wunderlist should do

Ask Question

Monitoring your competitors



Covered in this lecture:

How to keep track of the changes that happen with your competitors

Taught by:



- ▶ You need to keep tabs on your competitors in order to continuously adapt to the market
- ▶ Keep an eye on the 5 criteria for understanding competitors

There are 3 events that you should be looking for:

● #1 Funding

- many companies receive venture capital
- more money means more people, more ads, more contractors, more press, increased speed, more to work with, better product or design team, and it could also affect the user base size

>> Where to look:

- Crunchbase.com - keep track of how much money people have raised

● #2 Acquisitions

- acquisitions happen when a company buys another company, leading to its growth
- acquisitions can help improve a company's product team, but they get slower

>> Where to look:

- Crunchbase.com - keep track of the latest acquisitions

● #3 New features or new product launches

- you have to know what's new and whether it's competing with you

>> Where to look:

- Mention.com - it tracks your competitors' social media accounts and any online mention of the company; you can get notified when something new happens
- Google Alerts - you get notified when any new search results show up for your competitor, you can see how their SEO changes

See you next lecture!

What do we ultimately care about as a PM?



Covered in this lecture:

The most important things
to keep in mind

Taught by:



- ▶ If you work at a company, they will already know exactly who their competitors are
- ▶ Tip: Stay on top of industry news and pay close attention to the new features that your competitors build
- ▶ See what you can learn from your competitors

See you next lecture!

Resources - Section 5

- <http://www.planprojections.com/projections/market-size-estimation/>
- <https://productmanagementtips.com/2012/04/21/market-sizing-quick-and-dirty-techniques/>
- <https://www.appannie.com/en/>
- <https://www.inc.com/jeff-haden/bottom-up-or-top-down-market-analysis-which-should-you-use.html>
- <https://www.inc.com/guides/201105/10-tips-on-how-to-research-your-competition.html>
- <https://www.blueoceanstrategy.com/what-is-blue-ocean-strategy/>
- <http://motto.time.com/4116259/google-search/>
- <https://techcrunch.com/2012/04/24/one-sentence-pitch-winners/>
- <http://www.foodnetwork.com/recipes/giada-de-laurentiis/cheesy-baked-tortellini-recipe-1916784>
- <https://hbr.org/2011/06/why-a-great-individual-is-better>
- <https://hbr.org/2011/06/why-a-great-individual-is-better>
- <https://consequenceofsound.net/2016/05/spotify-trolls-apple-music-thanks-it-for-its-unprecedented-growth/>
- <https://blog.hubspot.com/marketing/branding-differentiate-competition-examples#sm.0000wd7126uxme1wzee2maui06me>
- <https://hbr.org/2015/06/a-better-way-to-map-brand-strategy>
- <https://hbr.org/2015/10/how-tesla-under-armour-and-sonos-do-branding>
- <https://techcrunch.com/2013/01/21/dave-morin-ceo-of-path-says-if-facebook-is-a-chevy-then-were-a-bmw-as-family-friendly-network-pushes-6m-users-looks-to-launch-more-virtual-goods/>
- <https://brainmates.com.au/brainrants/product-differentiation-what-does-your-product-do-better/>
- <http://gizmodo.com/oculus-rift-review-this-shit-is-legit-1767483554>
- <https://techcrunch.com/2016/03/28/SMACK-TO-THE-FUTURE/>
- <https://www.forbes.com/sites/insertcoin/2016/03/29/when-is-a-launch-not-a-launch-when-its-the-oculus-rift/#21d914d92087>
- https://www.tomtom.com/en_gb/action-camera/action-camera/
- <http://www.veho-muvi.com/>
- <https://www.sony.com/electronics/actioncam/fdr-x1000v-body-kit>
- <https://www.crunchbase.com/#/home/index>
- <https://mention.com/en/>
- <https://www.google.com/alerts>

What is customer development?



LECTURE
SUMMARY

Covered in this lecture:

Explaining the customer development cycle

Taught by:



- ▶ Customer development is one of the most important tools that a PM uses to find out if they're building the right thing
- ▶ Customer development is the practice of establishing a continuous and iterative communication line with your customer, so that you can come up with ideas, hypotheses, try them out, get feedback, and adapt your product accordingly
- Customer development is a framework developed by Steve Blank

In this framework, there are 4 steps in the product lifecycle: Discovery, Validation, Creation, and Building

- We're focusing on the customer interviews
Customer interviews help you understand the reasons why customers buy or don't buy your product

- ▶ The customer development cycle:
- 1. Validation
 - in this phase, you use customer interviews to figure out if your product is needed and if it solves a real problem
- 2. Development of the first version
 - in this phase, you use customer interviews to figure out what features you should build
- 3. Iteration
 - in this phase, you are improving the product
 - you use customer interviews to figure out if they are enjoying the product, who is getting the most out of it, what you are missing, what new features it might need
- ▶ Product managers use customer development as a tool for
 - risk mitigation
 - opportunity recognition



See you next lecture!

4 Types of Interviews



Covered in this lecture:

Explaining the types of interviews
and their purpose

Taught by:



► #1 Exploratory interview

- the most free form
- you are trying to establish whether or not they have certain pain point and are open to certain solutions
- you are exploring, looking for insight
- you can use it to come up with ideas
- talk about their day, the context in which they would use your product, see how badly they want it and if they would pay for it
- ask open ended questions

► #2 Validation interview

- you have a theory and you want to test it out
- these interviews are run in a scientific way
- they are hyper-sensitive to bias
- you don't introduce your theory or idea until the very end
- you try to be as objective as you can when describing your idea
- see if they talk about that problem on their own

► #3 Satisfaction oriented interview

- find out what parts of your product are good and what are not good
- understand why they are satisfied or unsatisfied
- example questions:
 - >> What should we stop doing?
 - >> What we can do to do this better for you?

► #4 Efficiency interview

- find out how can you improve your product to better serve its purpose
- find out when they use it and where it is most helpful
- are they using x-y-z feature and what for?
- example questions:
 - >> How easy is it for your to use feature x?
 - >> If you wanted to do __, how would you do it in our product?

- All interviews are usually a mix between these types



See you next lecture!

Key differences in customer development



Covered in this lecture:

How to approach customer development depending on the stage you are in

Taught by:



- ▶ Customer development evolves throughout the life cycle of the product
- ▶ We use the same guiding principles for customer development, but the difference is in the type of information you need to gather
- Pre-product
 - potential customers
 - they don't know you
 - harder to reach
 - focus on pain points and validation
- Post-product
 - existing customers
 - they know you
 - easier to reach
 - focus on satisfaction, usability, pain points

See you next lecture!

Who you should talk to



LECTURE
SUMMARY

Covered in this lecture:

How to establish your target customer group

Taught by:



- ▶ First, you have to figure out who you are targeting

Your target group will depend on whether you are pre-product or post-product

- ▶ Pre-product

You are still in the stage of validation

What should go in it? Do people want it? Is there a market?

- Situation #1 - you already have in mind a group of potential customers you will target

- Situation #2 - you have no idea

Exercise: 3 Question Stud

- Write down at least 3 types of potential customer groups
- All products solve a problem. What problem does your product solve and who has this problem?

- Draw a table with 3 columns and 3 rows
- On top, write the customer groups
- On the left side, write the 3 different criteria you are going to use to judge the groups

The 3 criteria:

- **Size** - Market size
- **Pain:Payment** - How much pain do they have associated with the problem? How likely are they to pay for your solution?
- **Accessibility** - How easily can you get in contact with these people?

Rate each criteria 1-10, add up the 3 ratings, and see which group has the highest number; that's the group with the highest potential.

► Post-product

You already have an established customer base, people paying and engaging with the product, and a lot of leads

You can choose from your user base which customers are the best to interview for the specific feedback you want to get

See you next lecture!

Finding interviewees externally



Covered in this lecture:

Taught by:



Where to find people to interview when you are in the pre-product phase

► 1. LinkedIn

- this works if your target customers are defined by a certain job, location, or education
- search for the specific job you target
- the results include your connection and your connections' connections
- contact them:
 - >> inmail (paid option)
 - >> connect with them and message them (free)

► 2. Forums

- these are places where people talk about their interests or their problems
 - >> Reddit.com - subreddits for everything
 - search the website
 - search through Google - site:reddit search term
- >> Quora.com
 - look for specific topics or groups of questions and look at the people that are constantly replying in them
 - if they fit your target demographic, contact them

► 3. Twitter

- people are self-identifying with things they like and they complain about what they don't like
- search for phrases people might be saying
- contact the ones who match your target group

► 4. Your competitors

- go to their social media pages and see who is engaging in conversations there
- see who is commenting on their blog
- contact the most active people

See you next lecture!

Finding interviewees internally



Covered in this lecture:

Where to find people to talk to when you are in the post-product phase

Taught by:



- ▶ You already have a list of customers or people that are interested in your product
- ▶ You can contact all your users, but it comes off very spammy if you do it this way, and you're going to have a hard time getting them to talk because it will be a cold email
- ▶ The best customers to talk to are people that are enthusiastic or have a problem with something in your product
- #1 Use your company's live chat system
 - examples: olark, tawk.to, intercom.io
 - passive approach - ask to see the chat logs from your customers, to see what they're saying, find individual people that meet your criteria and message them referring to what they said

- active approach - you can talk to people randomly or wait for specific users to come online
- ask the people who are in charge of solving users' problems to ask these people if they would be willing to talk to a product manager for 10 minutes

● #2 Your blog

- look at the people who are commenting
- they are probably users and they want to be heard

● #3 Power users

- these are people who use your product frequently, or buying things often, sending messages often
- they are more informed about your product and they're more invested in it, so there's a good chance that they'll be interested to talk

● #4 Twitter

- look at who is tweeting at you, replying, or sharing your posts
- it will be easier to get these people to talk to you



See you next lecture!

How to get them to talk



Covered in this lecture:

Strategies you can use to convince
people to talk to you

Taught by:



- ▶ Cold emails are not very efficient because there is no personal connection
- ▶ The response ratio is usually 3:1 - you message 3 people, and 1 will respond
- ▶ Follow these 3 rules:
 - **1. Be short**
 - no one will read a long email from a stranger
 - the ideal length is 4-7 sentences
 - **2. Be personal**
 - people don't like talking to robots
 - mention how you found them and then ask them to talk to you and help you with feedback
 - you can make a semi-template, but at least one sentence has to be personalized

● 3. Be valuable

- show them that this conversation is valuable to them
- people want to feel like they're helping
- tell them that you value their input
- you can offer to incentivize them

► Bonus Tips

1. Mention that you're not from sales

2. Make them feel special

- assure them that you want to fix their problem and that they have valuable information that can help you
- give them the VIP treatment

See you next lecture!

Practice writing emails



Covered in this lecture:

How to structure your emails
and what to write

Taught by:



- ▶ **1. Introduction: 1-2 sentences (personal)**
 - say where you found them and mention the problem you noticed they have
- ▶ **2. Two sentences about why you want to talk**
 - say you want to solve their problem and you need their help in order to do that

>> Techniques you can use to get them to talk to you:

 - appeal to their pride ("You appear to be an expert")
 - appeal to money
 - imply association - mention that you know people in a space that the person cares about
- ▶ **3. One sentence scheduling a time**
 - "Are you available to talk Wednesday at 11:00 AM?"
 - it's easier if you propose a specific time
 - you can use online scheduling softwares to let them pick a time

See you next lecture!

How to run a customer interview correctly



LECTURE
SUMMARY

Covered in this lecture:

Best practices for getting real
and helpful feedback

Taught by:



- ▶ You have to make sure you get the most accurate and helpful information
- ▶ Your topic is NOT going to be your product
- #1 Don't talk about your solution
 - Talk about their problems
 - "Customers might not know what they want, but they can't hide what they need"
- #2 Don't talk about your own opinions
 - The customer should talk more than you
 - The point is to get off point
 - Sometimes you might get new ideas from their answers
- #3 Create a comfortable environment
 - Don't ask questions that might make them nervous or uncomfortable
 - Don't react negatively to their feedback
 - Respond in a neutral, non-judgemental way

- #4 Don't force the conversation, guide it

- Let them talk about what they care about
 - You can get more information this way

- ▶ Tip: Whenever you get stuck, say:

- "That's interesting, tell me more."

- ▶ Try to get an answer to these questions:

- >> Who are your customer?
 - >> What are their habits?
 - >> When do they need your product?
 - >> Where do they need it?
 - >> Why do they need it?

- ▶ The more conversations you have, the more you can understand

See you next lecture!

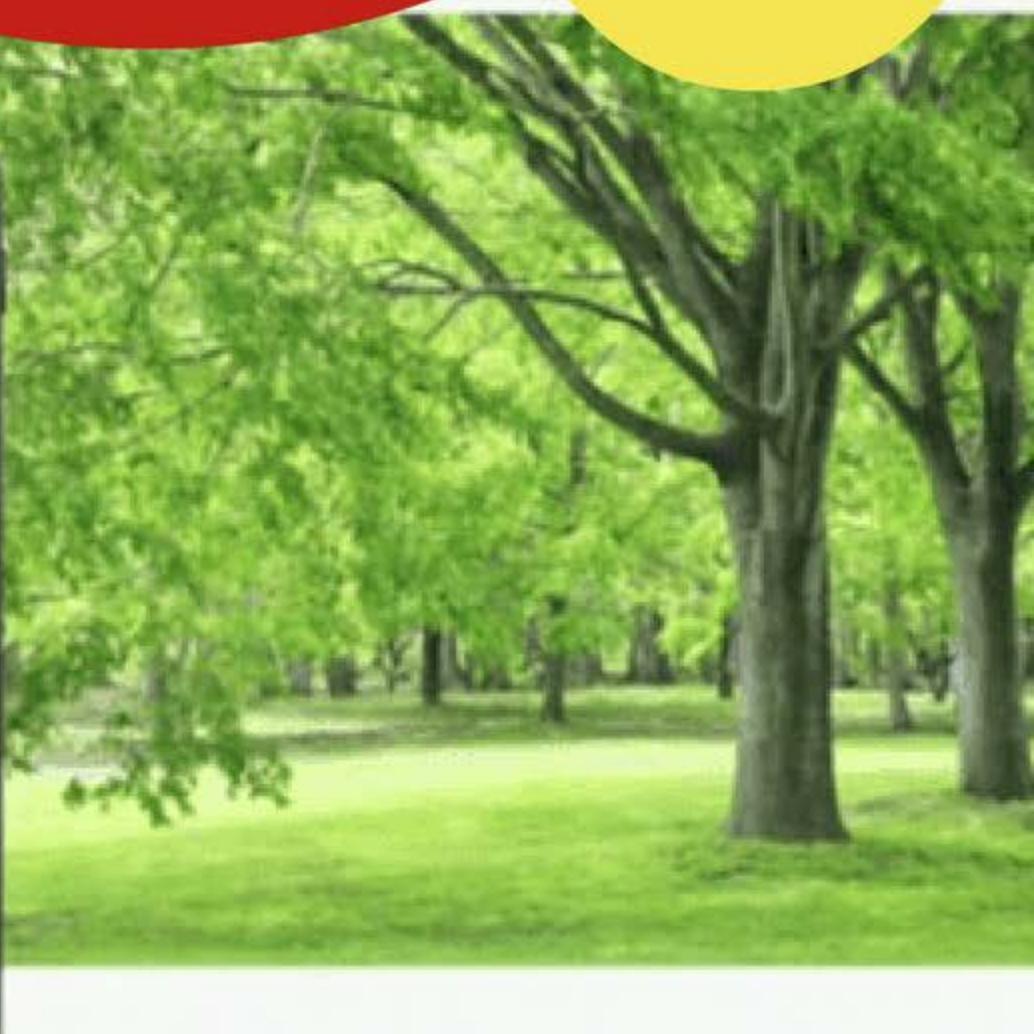


Let's Play...



THE MAIN FEATURE OF POKEMON GO IS THAT IT BLENDS THE REAL WORLD WITH THE MOBILE GAME WORLD. IN ORDER TO FIND AND CATCH POKEMONS, YOU HAVE TO WALK AROUND IRL (IN REAL LIFE). FOR EXAMPLE, AN EGG COULD BE AT THE ART MUSEUM. ONCE YOU GRAB IT, YOU'LL HAVE TO WALK THREE MILES TO HATCH IT, AND ONCE IT'S HATCHED YOU'LL HAVE TO TRAIN IT BY GOING TO A GYM IRL. THINK OF IT AS FITBIT MEETS AN MMORPG (MASSIVELY MULTIPLAYER ONLINE ROLE-PLAYING GAME).

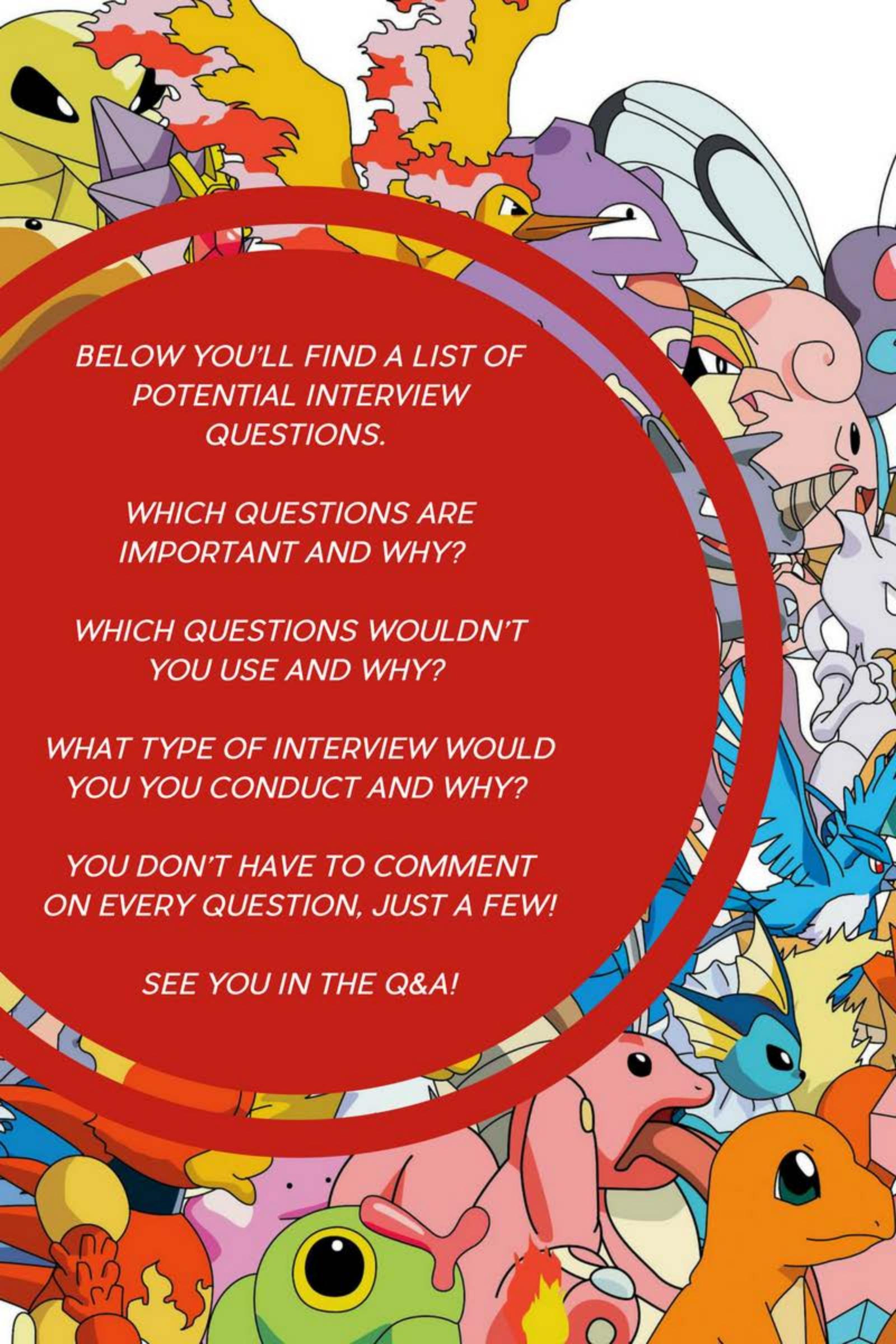
SEE THE ARTICLES IN RESOURCES FOR MORE INFORMATION!



*YOU'RE GETTING READY
TO INTERVIEW BETA
TESTERS WHO HAVE BEEN
TESTING POKEMON GO.*

*YOU WANT TO KNOW IF THEY
ARE ACTIVELY PLAYING THE
GAME OR JUST PLAYING
WHILE ON THE GO.*





*BELOW YOU'LL FIND A LIST OF
POTENTIAL INTERVIEW
QUESTIONS.*

*WHICH QUESTIONS ARE
IMPORTANT AND WHY?*

*WHICH QUESTIONS WOULDN'T
YOU USE AND WHY?*

*WHAT TYPE OF INTERVIEW WOULD
YOU CONDUCT AND WHY?*

*YOU DON'T HAVE TO COMMENT
ON EVERY QUESTION, JUST A FEW!*

SEE YOU IN THE Q&A!



1. Did you find yourself going out more into the real world?
2. Did you discover new places by trying to find eggs?
3. How much time did you spend training at the gym?
4. When do you use this game?
5. When notified there was a Pokemon nearby did that cause you to go out
6. Have you noticed more physical activity in your daily life?
7. How motivated were you to play the game?
8. What motivated you to play the game?
9. Do you play the game on your morning commute?
10. What was your general impression of the game?
11. When you're at a gym using the game do you feel judged by other people?
12. What are the positives and negatives of the game?
13. What's your opinion on the interface of the game?
14. Let's all go around the room, introduce ourselves, and say how many Pokemons we have.
15. Do you play the game in your spare time?
16. Do you feel like telling your friends to play it?
17. Would you use this game to exercise with your friends?





18. You said that you didn't like having to walk a certain distance to hatch eggs, why? The whole point of the game is to be physically active so what you're saying defeats the entire game.
19. How did the game help you to think about your physical activity?
20. Where do you play this game?
21. Has the game motivated you to pay more attention to the physical activity levels of others?
22. Are there any other subjects you would like to discuss?
23. If we held Pokemon GO events at gyms would you attend?
24. I feel that this is incredibly beneficial for my health, do you feel the same way? Why or why not?
25. That's interesting, tell me more.
26. Why do you think you need this game in your life?
27. When you're playing the game and you want to see if there are any Pokemon nearby what do you do?
28. Would it help if we gave you control over how much physical activity was needed to unlock features and rewards?
29. What do you use this for? Fun or exercise?
30. How easy is it for you to catch Pokemon?





PUT TOGETHER YOUR INTERVIEW QUESTION LIST:

Question #1

Question #2

Question #3

Question #4

Question #5

Question #6

Question #7

How would you conduct the interview?



Good questions, bad questions



LECTURE
SUMMARY

Covered in this lecture:

Best practices regarding
the questions you ask

Taught by:



- ▶ Bad questions can bias the customer's answers
- ▶ Good questions help you get real feedback
- Rule 1: Always ask open-ended questions
 - it gives them the room to give any information they see fit
 - they talk more
- Rule 2: Don't ask binary questions
 - these are questions with only 2 possible answers
 - you don't get any useful information this way
- Rule 3: Don't ask hypothetical questions
 - people don't really know what they would do in hypothetical situations
 - you won't get a helpful answer

- Rule 4: Don't ask leading questions

- these are questions that somehow include the answer
- you will influence their answer and bias it

- Rule 5: Don't ask questions that might make them lie

- don't put them in an awkward position
- you will never know if their answer is real



See you next lecture!

Building user personas off your interviews



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Centralizing the information you gathered and making a general customer profile

- ▶ The companies usually already have user personas, but sometimes you might need to make them yourself as a product manager
- ▶ User personas are fictional people that represent groups of similar users that behave in certain ways
- User personas help increase empathy towards your users
- User personas are profiles that include:
 - name, age, photo/sketch
 - what is most important to them about your product
 - what they want to be able to do easily

See you next lecture!

Real life example of a user persona



LECTURE
SUMMARY

Covered in this lecture:

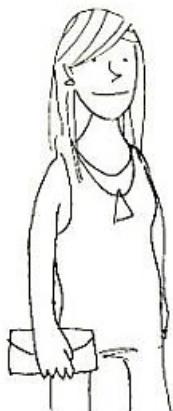
Taught by:



What a user persona looks like

Becky

*The Social Listener
(Leisurely Mainstream)*



- 26, Female
- Single. Works in Chicago as a nurse
- Drives 30 minutes each day from the suburbs
- Has an iPhone 4S
- Listens to what's new on the radio and what her friends like

I'm tech savvy	X	no - yes
Music defines who I am	X	no - yes
I want control	X	lean back - lean in
I want to support artists	X	no - yes
I accept ads	X	no - yes
I want to influence other people	X	no - yes

The product manager & the data diet



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



How to make sure you gather enough information

- ▶ You can't make product decisions based on only one single source of information, because you risk building the wrong product
- ▶ Sources of information you will use:
 - internal feedback
 - user test data
 - online feedback
 - analytics stats
 - news
 - market trends
 - watching competitors
- Customer interviews are just a dish in your data diet

Customer interviews have vulnerabilities:

1. They don't scale
 - you can't talk to too many people
2. The data is qualitative
 - you also need quantitative data because it's more precise

See you next lecture!

Resources - Section 6

<https://usabilityhub.com/>

<https://www.amazon.com/Four-Steps-Epiphany-Steve-Blank/dp/0989200507>

<https://steveblank.com/category/customer-development/>

http://www.huffingtonpost.com/michael-b-fishbein/7-common-customer-develop_b_4311877.html

<http://customerdiscovery.com/mistake-avoid-customer-discovery-interviews/>

<https://hbr.org/2015/03/putting-yourself-in-the-customers-shoes-doesnt-work>

<https://www.fastcompany.com/56447/customer-experience>

<https://jasonevanish.com/2013/08/11/95-ways-to-find-your-first-customers-for-customer-development-or-your-first-sale/>

<https://www.fastcompany.com/3059849/these-millennials-have-become-the-top-decision-makers-at-ibm>

<https://www.intercom.com/>

<https://www.tawk.to/>

<https://www.olark.com/>

<https://blog.hubspot.com/marketing/write-cold-email-get-response#sm.0000wd7126uxme1wzee2maui06me>

<https://www.fastcompany.com/3036672/what-we-learned-from-sending-1000-cold-emails>

<https://mailchimp.com/resources/guides/common-rookie-mistakes-email-marketers/>

<https://www.fastcodesign.com/1671033/why-focus-groups-kill-innovation-from-the-designer-behind-swiffer>

<https://www.fastcodesign.com/1671600/focus-groups-are-dangerous-know-when-to-use-them>

<https://www.mindtheproduct.com/2016/07/explosion-pokemon-go-product-designers-perspective/>

<http://www.thrv.com/6-steps-for-product-managers-to-handle-the-pokemon-go-augmented-reality-craze/>

<http://www.jasonshen.com/2016/product-insights-pokemon-go/>

<https://www.productschool.com/blog/get-job/product-perspective-pokemon-go/>

<https://www.qualtrics.com/blog/writing-survey-questions/>

<https://www.surveymonkey.com/blog/2015/02/11/5-common-survey-mistakes-ruin-your-data/>

<https://blog.kissmetrics.com/best-ways-to-get-feedback/>

<https://zapier.com/learn/customer-support/collect-customer-feedback/>

<https://www.forbes.com/sites/eilenezimmerman/2015/04/20/steve-blank-on-why-most-startups-fail-and-its-got-nothing-to-do-with-technology/#38c3d7f420cc>

What is an MVP?

Covered in this lecture:

The definition of an MVP and
what we use it for

Taught by:



- ▶ MVP = Minimum Viable Product
- ▶ The term "MVP" was first introduced in "The lean startup framework" by Eric Ries
- ▶ "A Minimum Viable Product is that version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort"
- In an MVP experiment, you build the smallest version of a product, with the least amount of resources, in order to get real feedback and find out if your idea will work
- Validated learning is learning from a scientific experiment where you are not biasing your customers
- Fail fast: run as many experiments as you can in order to collect more data and find the best version for your product

See you next lecture!

How do product managers think about MVPs?



LECTURE
SUMMARY

Covered in this lecture:

How to approach MVPs as a product manager

Taught by:



- ▶ As a PM, you will usually work in larger organizations that have a higher tolerance for risk than startups
- ▶ If your MVP experiment fails, the company will probably have enough resources to survive
- ▶ In larger organizations, product managers are not as concerned with resources, as they are with how a failure may affect the company's brand
- ▶ On the other hand, failed products might benefit the company in other ways

See you next lecture!

The 7 steps of the MVP process



LECTURE
SUMMARY

Covered in this lecture:

Listing out the seven steps of an MVP experiment

Taught by:



- ▶ The process that a product manager will go through in order to run the MVP experiment has seven steps:
 1. Figuring out what your problem/solution set is
 2. Identifying your assumptions and find the riskiest
 3. Building testable hypotheses around your assumptions
 4. Establishing your minimum criteria for success
 5. Picking what type of MVP you are going to run and your strategy
 6. Executing the MVP experiment
 7. Evaluating and learning from the experiment

See you next lecture!

Identifying your assumptions



Covered in this lecture:

The process of figuring out and writing down your assumptions

Taught by:



- ▶ People are always assuming that they know things, although there is always a possibility that they are not true
- ▶ We don't actually know anything until we test it

How to identify your assumptions:

- "In order for my idea to be successful, the following must be true..."
- Examples of frequent assumptions:
 1. My customer has x, y, z problem
 2. _____ matters to my customers
 3. _____ will pay for this product
 4. There are no satisfactory substitutes

See you next lecture!

Identifying your riskiest assumptions



LECTURE
SUMMARY

Covered in this lecture:

How to figure out which one of your assumptions is the riskiest

Taught by:



- ▶ If you have a lot of assumptions that have to be true in order for your product to be successful, you need to focus on testing the riskiest one first, so that you can save resources
- Riskiest assumption: That specific assumption that if it's not true, it means your product will definitely fail
- Usually, the riskiest thing to assume is the fact that your customers have that specific problem that you're trying to solve
- If they don't have that problem, they won't care about your product

See you next lecture!

The Risk / Difficulty square



LECTURE
SUMMARY

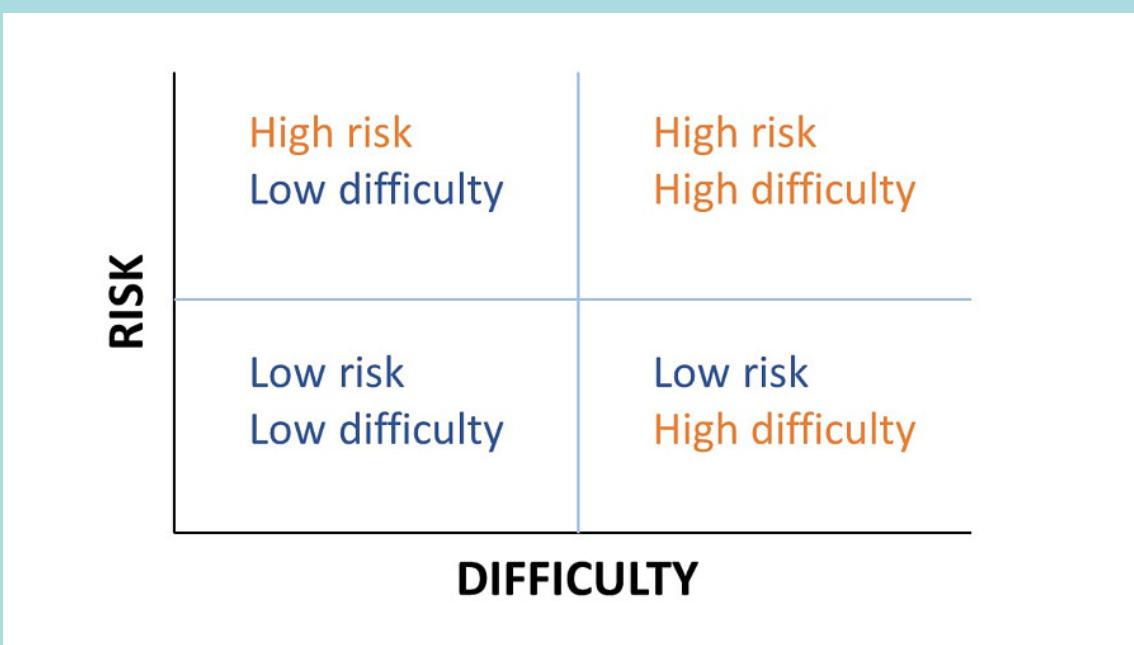
Covered in this lecture:

Taught by:



Prioritizing what you test according to the degree of risk and difficulty

- ▶ Large companies have to prioritize what assumptions to test, depending on their degree of risk and difficulty
 - ▶ Risk: How risky the assumptions are for the company or product
 - ▶ Difficulty: How much effort you need to make to test the assumptions
- Usually, these characteristics can be drawn in a table that looks like this:



- ▶ The first category to target should be "High risk / Low difficulty"
- ▶ Then, test the "High risk / High difficulty" category
- ▶ In the end, you can test the Low risk / Low difficulty" category and just ignore the "Low risk / High difficulty" assumptions



See you next lecture!

What is a hypothesis?

Covered in this lecture:

Taught by:



The definition of a hypothesis

- ▶ Once you have a list of assumptions, you need to create a specific hypothesis that you can test in your MVP experiment
- A hypothesis is a single, written, testable statement of what you believe to be true, with regards to the assumptions that you've identified
- Having a hypothesis that you are trying to prove true is part of running any scientific experiment

See you next lecture!

What is the minimum criteria for success?



Covered in this lecture:

What is a minimum criteria for success and how to set yours

Taught by:



- ▶ There are three different outcomes to an MVP test:
 1. You find out that your hypothesis is false
 2. You find out that your hypothesis is true
 3. You're somewhere in the middle
- ▶ 90% of the MVP experiments you are going to run are going to end up somewhere in the middle
- You need to establish a concept called Minimum Criteria for Success (MCS) in order to decide whether your product is worth building or not
- ▶ The MCS gives your experiments clarity and meaning
- ▶ In order to set your MCS, you need to take into consideration the cost and the reward of building the product
- ▶ Popular ideas that are monetary unsustainable are not good ideas

- ▶ Identify the metrics that you are targeting that will signal interest from your potential customers
 - ▶ Calculate the overall cost of building what you want, by including developer's time, your time, labor wages, advertising cost, brand effect, or opportunity cost
-
- In the end, the cost must be lower than the reward, in order for the product to be worth building
 - To figure out your MCS, you have to identify at what point the benefits outweigh the costs



See you next lecture!

Minimum criteria for success for startups



Covered in this lecture:

What startups look at when it comes to setting their MCS

Taught by:



- ▶ Startups look more at validation metrics, which demonstrate real interest from potential customers
- Examples: number of shares, percentage of people that sign up
- ▶ For startups, the economic viability of the project as a whole is a big concern, because the risk is always higher for new products
- ▶ MVP experiments will usually be shaped as coming soon pages, or as products presented as already real

See you next lecture!

The types of MVPs (1)



LECTURE
SUMMARY

Covered in this lecture:

**Listing out and explaining
the different types of MVPs**

Taught by:



- ▶ When you run MVP experiments, you're always going to present in some way something that implies that your new feature or product is already real or coming soon, because you need validated learning from real feedback
 - ▶ The extent to which you need to fake it will determine which type of MVP you need to use
-
- **The email MVP**
 - All you need is an email client, an email list, and some writing skills
 - By pitching them on a new product, you can see how they react in a simulated scenario
 - If you don't have an email list, you can gather emails from potential customers
 - Take a more personal tone to avoid getting into Spam

● Shadow button MVP

- Instead of building a new feature, you can show a button that supposedly links people to that specific feature
- The link might work and get them to another page, saying the feature is coming soon, or it might just look broken
- The number of clicks you get will signal the amount of interest

● 404 and coming soon page

- You act like you're adding a new feature, and when the user navigates to the page, it either displays a 404 error message, or a page that says the product is coming soon and that will ask you to sign up

>> Amazon.com is using this type of MVP very frequently



See you next lecture!

The types of MVPs (2)



LECTURE
SUMMARY

Covered in this lecture:

**Listing out and explaining
the different types of MVPs**

Taught by:



● Explainer videos

- You use videos that explain what the new features will do
- There are 2 types of explainer videos: tutorial style and sales style
- In tutorials, someone explains how he uses that specific feature, even if it's not real yet, by adding video effects that make it look real
- In sales related videos, you are making a fake promo that pitches the product by explaining its benefits

● Fake landing page pitch experiment MVP

- You create one singular page that pitches the benefits of the product, and that has a call to action somewhere on the page, and then you drive traffic to the page
- Landing pages are web pages that consist of just one page

- A pitch experiment is when you have an online experiment where you pitch for a new product/feature to an online audience, and you see what they do
- Landing pages are a natural fit for running pitch experiments



See you next lecture!

The types of MVPs (3)



Covered in this lecture:

Listing out and explaining
the different types of MVPs

Taught by:



● Concierge service MVP

- Concierge service is a service where you get one on one support from someone who manually goes through a task
- Instead of making a feature/product, you can launch an informal offering to a small subset of users as a beta version
- You can manually help them accomplish the task and see if the customers like it

● Piecemeal MVP

- Instead of building your product, you take what's available in terms of out of the box software, you piece them together, and you can get the functionality you need to test your basic version
- There are tons of services out there for anything you want to do

● Wizard of Oz MVP

- From the front, this MVP looks completely made, but all the tasks that computers and automated systems should be doing are being done manually

>> Zappos is a classic example of a company that used this type of MVP

See you next lecture!

In depth: Email based MVPs



Covered in this lecture:

Taught by:



Pros and cons of using an email MVP

► Pros:

- You can do it quickly
- You can limit your testing group to a small but statistically significant group of people
- You can segment your users
- If you already have a user base, it's easier to select your power users in order to test a new feature

► Cons:

- It may come off as sloppy
- You might dent your brand

● Tips:

1. Be aware of what your audience expects from you
2. Try to match the production value of the emails your company uses, in terms of design and images
3. Try to pair this with a landing page or a concierge service

See you next lecture!

In depth: Shadow buttons



Covered in this lecture:

Taught by:



Pros and cons of using shadow buttons

► Pros:

- Shadow buttons are easy to do and they give you great data to show you if people are interested

► Cons:

- They don't look good and your website looks broken
- They might create a negative response from your users, if they care about the polish of your product

● Tips:

1. Put a message that says "Thank you for clicking" and acknowledge what that thing was
2. Limit the amount of users that see this as much as possible

See you next lecture!

In depth: Coming soon and 404 MVPs



Covered in this lecture:

Taught by:



Using coming soon and 404 pages

- ▶ When choosing between these two, you need to think about which one is the least evil for you: getting an error or seeing a coming soon page
- ▶ Clothing companies might use this type of MVP by adding clothes that are out of stock
- Tips:
 1. Think about your users expectations and which of the two MVP types will result in the least negative outcome
 2. Put some effort into your landing pages, or make your 404 pages look funny in order to minimize the negative effects
 3. Websites you can check out for creating landing pages: Launchrock, Kickoff Labs

See you next lecture!

In depth: Explainer videos



Covered in this lecture:

Taught by:



Pros and cons of using explainer videos

► Pros:

- Video converts much better than just text pages
- Explainer videos can do a great job in getting across a product that needs in depth explaining
- You cover way more ground in a shorter time

► Cons:

- Explainer videos need a lot of work to create
- If your target customers are older, videos might not work well for them

● Tips:

1. Consider your users expectations and try to be as ordinary as you can in relation to the other videos you use
2. If you are in a company that relies heavily on video, they already have the infrastructure to create them, so it will be easier to use this type of MVP

See you next lecture!

In depth: Piecemeal MVPs



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Pros and cons of using piecemeal MVPs

► Pros:

- There are a lot of online softwares that can help you accomplish specific actions you might need

► Cons:

- If you need multiple functionalities, it's harder to get them chained together
- Sometimes they don't look that great

● Tips:

1. You can use online systems that help you facilitate interactions between all the softwares and connect the dots

>> Examples: Zapier, IFTTT

2. Look for softwares that allow white labeling, that allow you to put in your own images, colors, and fonts, so the new feature looks like it's part of your product

See you next lecture!

In depth: Concierge service MVPs



Covered in this lecture:

Pros and cons of using
concierge service MVPs

Taught by:



► Pros:

- It's cheap to run
- You don't actually build out anything in this MVP
- It's a hybrid between customer development and MVP testing
- You get to talk to your customers and see what their needs are

► Cons:

- It can be time absorbing and you will have to do all the work yourself
- You are going to spend a lot of time on each person

● Tips:

1. Think about the value of your time
2. If you don't have enough time for this, you need to pick another type of MVP

See you next lecture!

How do big companies think about MVPs?



LECTURE
SUMMARY

Covered in this lecture:

What large companies care about when it comes to MVPs

Taught by:



- ▶ Large organizations have the resources to weather failed products
- ▶ The financial risk is lower for them, but they care a lot about whether their brand is affected
- ▶ On the other hand, startups can use all their resources for making an MVP, which increases the risk if the product fails
- As a product manager in a big company, you will be expected to build an MVP that is a very basic version of your product, without spending unnecessary resources
- One feature MVP = creating a product with just one feature (the most important) and get feedback for that

See you next lecture!

Evaluate and learn



LECTURE
SUMMARY

Covered in this lecture:

How to evaluate your experiment
and learn from the data you gathered

Taught by:



- ▶ After you gather your data from your MVP experiment, you need to compare it with your MCS to see if it was successful
- ▶ Figure out why it worked or why it didn't work
- MVP experiments will primarily return quantitative data, in the form of numbers that indicate behaviors
- You also need qualitative data from customer interviews, in order to make the right decision
- Qualitative data helps you understand WHY the customers did what they did
- ▶ Putting together all the data will help you figure out whether your MVP experiment was successful or if you need to make some changes and run it again

See you next lecture!

Resources - Section 7

- <http://theleanstartup.com/principles>
- <https://www.quora.com/What-is-a-minimum-viable-product>
- <https://www.smashingmagazine.com/2014/04/a-guide-to-validating-product-ideas-with-quick-and-simple-experiments/>
- <https://blackboxofpm.com/mvpm-minimum-viable-product-manager-e1aeb8dd421>
- <https://www.mindtheproduct.com/2015/08/avoid-product-managers-worst-nightmare-part-1/>
- https://thenextweb.com/dd/2014/11/12/15-ways-test-minimum-viable-product/#.tnw_J4CTRpb2
- <https://www.forbes.com/sites/groupthink/2014/04/28/five-pitfalls-of-running-lean-startup-experiments/#4047c7146296>
- <https://www.smashingmagazine.com/2014/04/a-guide-to-validating-product-ideas-with-quick-and-simple-experiments/>
- <https://www.uxmatters.com/mt/archives/2010/04/dealing-with-risky-and-safe-assumptions.php>
- <https://www.uxmatters.com/mt/archives/2015/10/identifying-and-validating-assumptions-and-mitigating-biases-in-user-research.php>
- <https://dkander.wordpress.com/2013/05/07/how-to-diagnose-your-riskiest-assumptions/>
- <https://blog.intercom.com/first-rule-prioritization-no-snacking/>
- <https://www.mindtheproduct.com/2012/08/experiments-101/>
- <https://appsumo.com/adzoola/>
- <https://techcrunch.com/2011/10/19/dropbox-minimal-viable-product/>
- <https://blog.bufferapp.com/idea-to-paying-customers-in-7-weeks-how-we-did-it>
- <https://www.bullethq.com/lean-startup-zappos-how-zappos-validated-their-business-model-with-lean/>
- <http://ryanhoover.me/post/69599262875/product-hunt-began-as-an-email-list>
- <http://www.creativebloq.com/web-design/best-404-pages-812505>
- <http://www.hongkiat.com/blog/60-really-cool-and-creative-error-404-pages/>
- <https://medium.com/@joelgascoigne/how-to-successfully-validate-your-idea-with-a-landing-page-mvp-ef3c2d02dc51>
- <https://www.powtoon.com/index/>
- <https://goanimate.com/>
- <https://www.typeform.com/>
- <https://zapier.com/>
- <https://ifttt.com/>
- <https://grasshopperherder.com/concierge-vs-wizard-of-oz-test/>
- <https://medium.com/@jevy/building-i-open-sourced-all-my-business-ideas-b3e30b65abd5>
- <https://pando.com/2014/04/18/dancing-giants-why-large-companies-should-embrace-the-minimum-viable-product/>

Intro to wireframing



LECTURE
SUMMARY

Covered in this lecture:

What wireframes are and
why we use them

Taught by:



- ▶ In order to build a great product, you first have to conceptualize it
- ▶ One of the crucial skills for conceptualizing your idea is wireframing
- ▶ Wireframes are visual guides for websites or apps that lay out the rough structure for where the content is going to go
- Wireframing is the first step to take in order to materialize your idea
- Wireframes have a low fidelity/accuracy
- As you gather feedback, you add more details and more fidelity

- If you're on a smaller team, you will probably have to work on wireframes yourself
 - In a larger company, you might not be asked to do wireframes, but you need to be familiar with them or contribute to their creation
 - These skills are also helpful if you want to sketch out an idea you want to propose
- ▶ Wireframes make it easier to communicate feature ideas



See you next lecture!

Wireframe, mockup, prototype



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



The differences between wireframes,
mockups, and prototypes

► Wireframes

- you make these first
 - you draw the general structure, the layout
 - low fidelity
- >> Tools: Balsamiq, Axure, Omnigraffle, Hotgloo, POP

► Mockups

- static displays of what the final product looks like
 - they have more details and colors
 - are done by the designers
- >> Tools: Photoshop, Sketch, Illustrator, Axure, UXPin

► Prototypes

- they handle usability
 - they have basic interactions and high fidelity
 - you can see potential problems in the user flow
- >> Tools: Keynote, POP, Axure, Proto.io, InVision

See you next lecture!

Let's start sketching



Covered in this lecture:

Taught by:



How to draw your first sketch

- ▶ The best tools to use are pen and paper
- ▶ 1. Draw a browser
- ▶ 2. Draw the logo and the navigation bar
- ▶ 3. Ask yourself: What is the point of this page?
What should the users be able to do?
- Write out all the functions that the page should have
 - >> Example: if you're making a page that shows videos, draw how the thumbnails appear on the page, how the like button looks like, draw a search bar, etc.
- Look at your favorite websites to get inspired
- Remember:
The important thing is the function, not the form

See you next lecture!

Using POP



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



What is POP and how to use it

- ▶ You can get the app at www.popapp.in
- ▶ This app helps you use your phone to add basic user flow and interaction to your wireframe
- - Create a project
 - Select your device
 - Take photos of all your screen sketches
 - Add relationships between the screens by creating tappable buttons that link them
- POP helps you communicate your ideas easier

See you next lecture!

Intro to Balsamiq



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



What is Balsamiq and how to use it

- ▶ Balsamiq is a popular tool for creating low fidelity wireframes
- ▶ They have a web and a desktop version, and a free trial for both
 - - You can add elements to the canvas
 - There are categories for mostly any element you want to add
 - You can add comments for other people who are looking at your sketch
 - You have the option to export your wireframe or share it with others
 - Play with the settings to get closer to what you want to create

See you next lecture!

Resources - Section 8

<https://www.smashingmagazine.com/2009/09/35-excellent-wireframing-resources/>

<https://blog.balsamiq.com/wireframe-presentation-tips/>

<https://designmodo.com/wireframing-prototyping/>

<https://designshack.net/articles/layouts/vector-vs-raster-what-do-i-use/>

<http://www.ucreative.com/articles/how-to-explain-raster-vs-vector-to-your-clients/>

<https://www.axure.com/>

<https://www.hotgloo.com/>

<https://www.invisionapp.com/>

<https://dribbble.com/search?q=wireframing>

<http://keynotopia.com/wireframe-templates/>

<https://www.invisionapp.com/blog/the-big-list-40-rock-solid-design-prototyping-resources/>

<http://mashable.com/2013/04/02/wireframing-tools-mobile/#AA0truAUaPqm>

<https://dribbble.com/search?q=mobile+wireframes>

<https://marvelapp.com/pop/?popref=1>

<https://marvelapp.com/sketchpad?popref=1>

<https://balsamiq.com/>

<https://blog.balsamiq.com/category/products/>

Introduction to metrics



Covered in this lecture:

Taught by:



What metrics are and why we use them

- ▶ As a PM, your entire role revolves around metrics in one way or another. You use this science to help your product be successful.

"What gets measured gets managed" - Peter Drucker

- ▶ Feedback loops - the more frequently you're getting accurate feedback, the more effectively you're going to manage that

Example: Fitness tracker industry

- ▶ Metrics are numbers or measurements that describe what's going on with your product
- Metrics are also called KPI - Key Performance Indicators

- Examples of metrics:
 - monthly active users
 - returning users
 - churn users
 - app store reviews
 - Facebook and Twitter posts
- Metrics are different depending on the type of PM you are and the company you are working for
- Product managers define success by achieving their goals regarding metrics



See you next lecture!

Real life examples of metrics



Covered in this lecture:

Taught by:



Examples of metrics of popular companies

- ▶ Companies have a lot of similar metrics, but when it comes to engagement, they have some metrics that are specific to their product

- ▶ Twitter
 - Growth metrics: total new users per month, monthly/daily active users, activated users
 - Engagement metrics: multiple logins per day, time spent on the website, number of tweets sent per user, average number of likes, re-tweets and follows, number of private messages sent

- ▶ Youtube
 - Growth metrics: monthly/daily active users, total new users, activated users
 - Engagement metrics: video views per user, average viewing time per user

► Facebook

- Growth metrics: new users, monthly active users
- Engagement metrics: newsfeed position clicks, number of messages sent, time spent on the website and on their partner websites (Instagram, What's App), average number of likes users give and get



See you next lecture!

Metrics of all kinds



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Explaining the types of metrics and
what they track, with examples

► #1 Growth & Activation metrics

- track and measure how your product is growing
- examples: total new users, new users by source, activated users
- there's a big difference between someone who only installed your app and an activated user
 - >> An activated user is someone who actually signed up and performed an action inside the app

► #2 Retention metrics

- track how many people are coming back to use your app repeatedly
- examples: retained users, resurrected users
 - >> Retained users are users who are using your app all the time
 - >> Resurrected users are users who haven't been using your app for a while, but they're coming back after you notified them about, let's say, some attractive new feature

► #3 Engagement metrics

- track how many times users are engaging with the app
- these metrics are tailored to each company and product

► #4 User happiness metrics

- track how happy your users are
- examples: net promoter scores, number of customers that have written complaints, app store rating
- these are some of the most difficult metrics to measure, but are very important

► #5 Revenue metrics

- track how much revenue you are making
- examples:
 - lifetime value (how much revenue, on average, on a certain period of time, does each customer generate?)
 - cost per acquisition (how much money do you have to spend in marketing/ads/salaries to acquire a customer?)
 - monthly recurring revenue
 - annual recurring revenue



See you next lecture!

How to pick good metrics



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



What characteristics your metrics should have in order to get real feedback

- ▶ Even if you're not tracking the exact metric that the company is targeting, you might notice that improving other metrics might lead to achieving the company's goal
- ▶ Exploratory metrics - you're not always tracking them, but they're there and you can analyze them to see what's happening
- ▶ Reporting metrics - you're tracking them on a long period of time to make sure your product is headed in the right direction

We're going to talk about reporting metrics

- What makes a good metric? Your metrics should be:
 1. Understandable
 2. Rate or Ratio
 3. Correlated
 4. Changeable

See you next lecture!

Using the HEART metrics framework



Covered in this lecture:

Explaining the HEART metrics framework and how to use it

Taught by:



- ▶ This metrics framework was originally popularized by Kerry Rodden and it helps you think about the customer journey

The meaning of H.E.A.R.T.:

- ▶ **#1 Happiness**
 - how happy your user is
- ▶ **#2 Engagement**
 - how engaged your user is in the short term
- ▶ **#3 Adoption**
 - how many users tried your product
- ▶ **#4 Retention**
 - are your users returning every single month?
- ▶ **#5 Task Success**
 - what's the most important thing that your users should do with your product, and are they doing it?

For every metric, you will track 3 things:

- ▶ **Goals** - what do you want to happen?
 - ▶ **Signals** - what is the actual thing that you need to measure in order to know that you're getting closer to your goal?
 - ▶ **Metrics** - how do you take the goal and signal and express it as an actual metric over time?
-
- H.E.A.R.T. is only an acronym that sounds good. The actual order in which your customer uses your product would be A.T.E.R.H.
 - **Tips:**

You can use the HEART framework for anything, it's very flexible

This metrics framework is used for reporting metrics

You don't have to use all the metrics, pick the ones that are the most important for your company

The signals column can be used as a guide to what you need to do to process and track the metrics

See you next lecture!

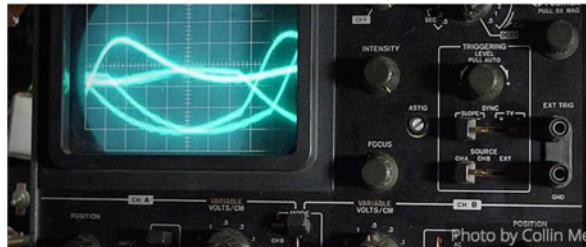
Why the HEART framework?

- IIAAYKIG - If it's an acronym, you know it's good
- Metrics are tougher than you think!
- It's a simple way to ensure you're thinking about every aspect of the user journey / way a user sees your product

How to choose the right UX metrics for your product

by Kerry Rodden

[Tweet](#) [Share](#) [Pocket](#)



When designing for the web, you can analyze usage data for your product and compare different interfaces in A/B tests. This is sometimes called “data-driven design”, but I prefer to think of it as data-*informed* design — the designer is still driving, not the data.

To make this work in practice it’s important to use the right metrics. Basic traffic metrics (like overall page views or number of unique users) are easy to track and give a good baseline on how your site is doing, but they are often not very useful for evaluating the impact of UX changes. This is because they are very general, and usually don’t relate directly to either the quality of the user experience or the goals of your project — it’s hard to make them actionable.

I’m part of a group of quantitative UX researchers at Google, and we like to think of large-scale data analysis as just another UX research method. We’ve developed a couple of useful methods to help choose and define appropriate metrics that reflect:

- The quality of user experience (the HEART framework)
- The goals of your product or project (the Goals-Signals-Metrics process)

The HEART framework

While helping Google product teams define UX metrics, we noticed that our suggestions tended to fall into five categories:

- **Happiness:** measures of user attitudes, often collected via survey. For example: satisfaction, perceived ease of use, and net-promoter score.
- **Engagement:** level of user involvement, typically measured via behavioral proxies such as frequency, intensity, or depth of interaction over some time period. Examples might include the number of visits per user per week or the number of photos uploaded per user per day.
- **Adoption:** new users of a product or feature. For example: the number of accounts created in the last seven days or [the percentage of Gmail users who use labels](#).
- **Retention:** the rate at which existing users are returning. For example: how many of the active users from a given time period are still present in some



for X

U B E R

Uber + Alcohol + Lemonade

UBER FOR SPIKED LEMONADE



Happiness

Engagement

Adoption

Retention

Task Success

Happiness

- How happy is your user?

Engagement

- How engaged is your user in the short term?

Adoption

- How many interested users have actually tried your product?

Retention

- How many users do you retain long term?

Task Success

- How successful are you at allowing users to perform the most valuable task?

Goals

What do you want to happen?

Signals

What is the thing we need to measure?

Metrics

A signal expressed over time.
This is the thing you watch.

Goals

Signals

Metrics

Happiness

Engagement

Adoption

Retention

Task Success

- 1. Happiness**
- 2. Engagement**
- 3. Adoption**
- 4. Retention**
- 5. Task Success**

- 1. Adoption**
- 2. Task Success**
- 3. Engagement**
- 4. Retention**
- 5. Happiness**

	Goals
Happiness	<ul style="list-style-type: none"> • Maximize drinker satisfaction with our spiked lemonade and the delivery service
Engagement	<ul style="list-style-type: none"> • Maximize # of orders and total value of the orders (through add-ons like additional shots in the lemonade or extra fast delivery)
Adoption	<ul style="list-style-type: none"> • Maximize the # of people who order at least 1 glass of spiked lemonade
Retention	<ul style="list-style-type: none"> • Maximize the % of users that return and place a repeat order on a monthly basis
Task Success	<ul style="list-style-type: none"> • Minimize # of "abandoned carts" (people that don't complete purchase) • Maximize the # of users that successfully complete an order within 5 minutes

Signals

Happiness

- App store rating
 - Scores from the NPS survey sent to customers
-

Engagement

- # of orders per customer
 - Order value in dollars per customer
-

Adoption

- # of users who order at least 1 glass of spiked lemonade
 - # of users that have downloaded and opened the app
-

Retention

- People that order spiked lemonade
-

Task Success

- Incomplete orders
- Time taken to place each order, from app start to "success" screen

Metrics

Happiness

- App store rating change, month over month
 - % of perfect 10 NPS scores
-

Engagement

- Average order value in dollars per day
 - Average # of orders per day **per user**
-

Adoption

- % of new users -> customers
(# of new users that order **divided by** the # of people that have opened the app to get a percentage)
-

Retention

- Retained users
(# of people ordering this month **divided by** the # of those same people that ordered last month to get a percentage)
-

Task Success

- Average time to checkout
- % of orders incomplete per month

	Goals	Signals	Metrics
Happiness	<ul style="list-style-type: none"> Maximize drinker satisfaction with our spiked lemonade and the delivery service 	<ul style="list-style-type: none"> App store rating Scores from the NPS survey sent to customers 	<ul style="list-style-type: none"> App store rating change, month over month % of perfect 10 NPS scores
Engagement	<ul style="list-style-type: none"> Maximize # of orders and total value of the orders (through add-ons like additional shots in the lemonade or extra fast delivery) 	<ul style="list-style-type: none"> # of orders per customer Order value in dollars per customer 	<ul style="list-style-type: none"> Average order value in dollars per day Average # of orders per day per user
Adoption	<ul style="list-style-type: none"> Maximize the # of people who order at least 1 glass of spiked lemonade 	<ul style="list-style-type: none"> # of users who order at least 1 glass of spiked lemonade # of users that have downloaded and opened the app 	<ul style="list-style-type: none"> % of new users -> customers (# of new users that order divided by the # of people that have opened the app to get a percentage)
Retention	<ul style="list-style-type: none"> Maximize the % of users that return and place a repeat order on a monthly basis 	<ul style="list-style-type: none"> People that order spiked lemonade 	<ul style="list-style-type: none"> Retained users (# of people ordering this month divided by the # of those same people that ordered last month to get a percentage)
Task Success	<ul style="list-style-type: none"> Minimize # of "abandoned carts" (people that don't complete purchase) Maximize the # of users that successfully complete an order within 5 minutes 	<ul style="list-style-type: none"> Incomplete orders Time taken to place each order, from app start to "success" screen 	<ul style="list-style-type: none"> Average time to checkout % of orders incomplete per month

Tips on using HEART

- Flexible – use it for a product area or a whole company
- Use it for reporting, not exploring
- It's not strict and you don't have to use them all – try to **pick one that matters most**
- You can use the signals column to inform engineers of tracking requirements

Using the AARRR (Pirate) metrics framework



Covered in this lecture:

Explaining the AARRR (Pirate) metrics framework and how to use it

Taught by:



- ▶ This framework was created by Dave McClure and it's commonly used with software and service businesses

The meaning of A.A.R.R.R.:

- ▶ **#1 Acquisition**
 - metrics for acquiring your user
- ▶ **#2 Activation**
 - metrics that show that the customer is now activated, they signed up and engaged with the app
- ▶ **#3 Retention**
 - customers come back to use your app
- ▶ **#4 Referral**
 - are users happy enough with your app that they refer it to others?

► #5 Revenue

- metrics that show you are making revenue from your users
- Think of metrics that you can measure in order to know how your product is doing in each one of these steps



See you next lecture!

Tracking your metrics in practice



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Tools you can use to track your metrics

- ▶ If you are at a company, you already have a list of metrics to track and the tools to do that

- ▶ If you just decided what metrics you want to track, but you don't know how to do it, you can use these tools:
 1. Google Analytics
 2. CrazyEgg
 3. KISSmetrics
 4. Mixpanel
 5. Optimizely
 6. Segment - this is a metrics hub, not a tool
 - it helps you keep your metric history from all the tools you used

See you next lecture!

Resources - Section 9

<https://www.entrepreneur.com/article/237484>

https://www.wired.com/2011/06/ff_feedbackloop/

<https://sproutsocial.com/insights/social-media-metrics-that-matter/>

<https://blog.kissmetrics.com/analytics-can-strengthen-engagement/>

<https://500.co/distribution-metrics-for-dummies-the-science-of-profitability/>

<https://hbr.org/2013/03/know-the-difference-between-yo>

<https://www.dtelepathy.com/ux-metrics/>

<http://500hats.typepad.com/500blogs/2007/09/startup-metrics.html>

<https://www.slideshare.net/dmc500hats/startup-metrics-for-pirates-long-version>

Dave McClure - Startup Metrics for Pirates: AARRR!

<https://www.youtube.com/watch?v=irjgfW0BIrw>

<https://www.optimizely.com/>

<https://www.crazyegg.com/>

<https://www.kissmetrics.com/>

<https://mixpanel.com/>

<https://segment.com/>

Introduction to epics



LECTURE
SUMMARY

Taught by:



What epics are and why we use them

- ▶ Everything that a company builds comes from the CEO's vision
- ▶ In order to achieve that vision, the company has goals that it has to reach, that are defined by metrics
- ▶ Teams come up with initiatives in order to reach the goals
After that, the team releases the new updates
- An epic is a grouping of one or more features or functionalities that we want to build

Examples of epics:

- "Translate the app to Spanish"
- "Implement photo sharing in direct messages"

- ▶ Not all the things that we do as a product team result in a new feature for the outside user, that's why we call them epics, not features
- ▶ Epics are also defined as pieces of work that take longer than one sprint to build
- ▶ If they are shorter than one sprint, they will be called user stories



See you next lecture!

Let's get into epic specs



LECTURE
SUMMARY

Taught by:



Covered in this lecture:

Explaining epic spec sheets
and what goes into them

- ▶ Epic spec sheets are documents that contain all the requirements for building a product
- ▶ Their purpose is to allow everyone in the company to read them and understand what they have to build

Epic spec sheets have four main areas:

● #1 Introduction

- summary of what the features you're building are for and why you're building them
- what metrics you are trying to improve
- links to specific documentation
- marketing plans, legal requirements
- early wireframes

● #2 Product requirements

- what is required for the specific features you want to build

- #3 Design requirements

- you and the designer fill this section together
- sketches, prototypes

- #4 Engineer requirements

- this section is mostly filled out by the engineers after you discuss with them
 - it contains what must be done on the technology side
-
- ▶ You are in charge of creating and maintaining the entire spec sheet, but mainly the first two areas
 - ▶ Each company has a different way of doing spec sheets

See you next lecture!

User stories and acceptance criteria



LECTURE
SUMMARY

Covered in this lecture:

Taught by:



Explaining user stories and acceptance criteria, with examples

- ▶ User stories are a way to describe a thing we're going to build that delivers some type of functionality to the end user
- ▶ User stories follow this format:
"As an X, I want to do Y, so that I can Z"
- This is a way to explain to the engineers what the feature needs to do without saying how to do it

Example: "As a user, I want to send pictures in a direct message to my friends, so that I can share my favorite photos with them"

- User stories belong inside your project management tool, written as tickets that can be moved in the "To do", "In progress", or "Done" categories

- ▶ Acceptance criteria are a set of conditions that software must satisfy in order to be considered complete
- ▶ The purpose of acceptance criteria is to be very specific on how a feature should function

Example: "Given I am a user and I click the "Add picture" button in the direct message, I am presented with a popup window to choose the file I can upload, submit it with the upload button and see a preview of the uploaded image."

- As a PM, you are responsible for testing the completed tickets and stories before approving them to be released to the public



See you next lecture!

Estimations and velocity



LECTURE
SUMMARY

Covered in this lecture:

Explaining velocity and how to
make more accurate estimations

Taught by:



- ▶ From company to company, the engineers are building things in different ways, in different languages and styles, and things are changing all the time
- ▶ That makes software estimation very hard
- ▶ In order to estimate accurately, you need to figure out something called velocity
- Story points: measuring the difficulty of a task by using a rating system that everybody in the company understands
- Velocity: the number of story points we were able to accomplish in a 2 weeks sprint

- Example:

- We have 5 items in total
- 3 were done but were very hard
- We rate them 5 on a scale of 1-5
- The velocity is $5 + 5 + 5 = 15$

- ▶ Doing these calculations increases the accuracy of the estimations in the long term



See you next lecture!

Roadmapping



LECTURE
SUMMARY

Covered in this lecture:

Reasons why we do roadmaps
and alternatives

Taught by:



- ▶ Every company does roadmapping differently
- ▶ Roadmaps are usually inaccurate, but they are good to have as a general guide
- ▶ Why do companies do roadmaps?
 1. Executives and investors like to see quarter-based maps
 2. You could be against an actual deadline
- The alternative is to sort things depending on priorities
 - >> Near-term, Mid-term, Long term
- This method keeps everyone in line but doesn't impose strict deadlines

See you next lecture!

Prioritization



Covered in this lecture:

Prioritization methods that
you can use as a PM

Taught by:



- ▶ As a product manager, prioritizing things is a major part of your role

Prioritization methods:

- ▶ **1. Assumption testing**
 - prioritizing by the assumptions
 - try the riskiest assumptions first
- >> Take your assumptions and assign them a value between 1-10 (10 = riskiest)
- >> Rate the importance of doing that thing (10 = very important)
- >> Add these two values and sort the assumptions based on that

► 2. The BUC method

- Business benefits
- User benefits
- Cost

>> Score these dimensions 1-10

>> Add the score of the benefits and subtract the cost to get the final score

>> Prioritize your tasks based on these scores, with the highest at the top

► 3. The MOSCOW method

>> Organize your tasks into these categories:

Must - things you absolutely have to build

Could

Should

Would

>> First do the things that MUST be done

See you next lecture!

Resources - Section 10

<https://www.atlassian.com/agile/delivery-vehicles>

<https://www.fastcompany.com/3015920/unblocked-a-guide-to-making-things-people-love-part-1>

<https://www.fastcompany.com/3015931/design-flow-achieving-breakthrough-creativity-and-high-yield-production-part-1>

<https://www.fastcompany.com/3015932/data-flow-using-data-to-constantly-improve-part-6>

<https://www.productmanagerhq.com/2014/10/what-is-a-user-story/>

<https://www.atlassian.com/software/jira>

<https://www.fastcompany.com/3015928/engineering-flow-planning-for-high-velocity-sprints-part-3>

<https://www.fastcompany.com/3015930/facilitating-high-velocity-engineering-sprints-part-4>

<http://www.romanpichler.com/blog/10-tips-creating-agile-product-roadmap/>

<https://www.atlassian.com/agile/roadmaps>

<https://www.fastcompany.com/3015926/value-driven-product-development-using-value-propositions-to-build-a-rigorous-p>

<https://www.fastcompany.com/3015933/concluding-thoughts-your-job-is-to-make-wonderful-things-part-7>

General communication skills



Covered in this lecture:

General advice on communicating
with people in your company

Taught by:



- ▶ Being a communications hub is one of the biggest parts of the product manager job
- ▶ #1 Meetings
 - learn how to take advantage of meetings
 - you have to make sure that activities that come from meetings are followed up on
 - send a recap email after the meeting
- ▶ #2 Conference calls
 - make sure that you have the right intonation and that you deliver a clear message
 - make sure there are links provided for all that you are discussing

► #3 Emails

- ensure that you are writing clear, concise messages, accompanied by the right resources and data that you refer to

► #4 Informal meetings

- these are great for connecting, getting feedback, ideas, and working to build support

- Learn the art of communicating differently with different types of people

See you next lecture!

Working with engineers



LECTURE
SUMMARY

Covered in this lecture:

Best practices for
communicating with engineers

Taught by:



- ▶ With engineers, you have to describe everything clearly and in details, and include technical information

Tips for communicating with engineers:

- ▶ 1. If something goes wrong, it's your fault because you didn't give them the right specs
- ▶ 2. When you're pitching a feature, make sure you have a good idea of where the feature is going to go in the future
- ▶ 3. When possible, you should do the work upfront, on things like checking logs, or looking up data
- ▶ 4. Watch out for tech debt (something which has to be dealt with later because it wasn't done right the first time)
 - engineers hate tech debt

- ▶ 5. Don't treat engineers like an agency
 - don't design or come up with all the concepts yourself and just hand them the requirements
 - let them provide feedback and come up with their own ideas



See you next lecture!

Working with designers



LECTURE
SUMMARY

Covered in this lecture:

Best practices for
communicating with designers

Taught by:



- ▶ Tips for talking to designers:
 - 1. Give designers their creative freedom
 - 2. Don't treat designers as an agency or as someone who just makes things pretty
 - ask them for feedback
 - 3. You and your designer should function as a team
 - you need to show them all the data involved in the project
 - 4. Don't ever tell the designer what to do
 - 5. Always talk about user problems first and solutions second
 - let them come up with the best solutions

See you next lecture!

Working with executives and others



Covered in this lecture:

Best practices for communicating with executives and others

Taught by:



► Tips for communicating with executives:

1. The secret to communicating with executives is being brief, because they are busy people
 - write two bullet points instead of two paragraphs
2. Always speak in terms of business effect
 - explain what things mean in terms of revenue and metrics
3. Communicate in their style

► Tips for communicating with other people:

1. Talk to people on each team as much as possible
2. Ensure that other teams understand that you know the user base, the technology, and the business very well
3. Update them on the latest developments as frequently as possible
4. Make sure you tell the stakeholders the reason you ended up with that specific solution

See you next lecture!

Resources - Section 11

<https://pmblog.quora.com/Tips-for-PM-Introverts>

http://www.huffingtonpost.com/brian-de-haaff/the-product-manager-vs-th_1_b_7733156.html

<https://www.mindtheproduct.com/2014/03/product-managers-5-ways-you-can-make-an-engineers-job-easier/>

<https://www.quora.com/Can-a-product-manager-be-effective-without-product-design-skills>

<https://www.mindtheproduct.com/2015/08/4-design-skills-every-product-manager-should-have/>

<https://blog.hubspot.com/marketing/communicating-effectively-with-your-boss>

Why learn technology?



Covered in this lecture:

Reasons you should learn
technology as a PM

Taught by:



- ▶ As a PM, you don't need to know how to code, but you need to understand how the most common technologies work

Reasons to learn technology:

- 1. You will know what can be realistically built
- 2. It helps you build a great relationship with engineers
- 3. It helps you understand the impact of the decisions that you're making

See you next lecture!

"The Cloud", servers, clients, and the inner workings of the internet



LECTURE
SUMMARY

Taught by:

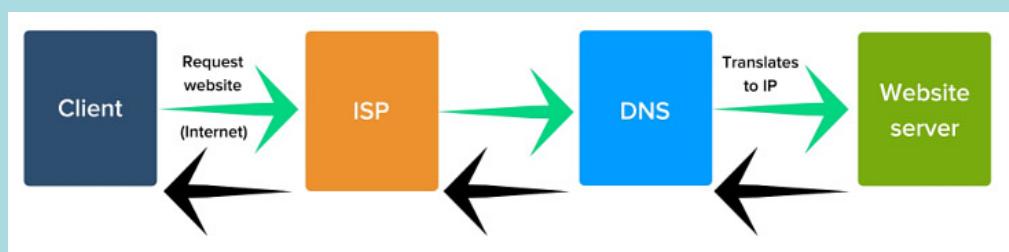


Explaining how the internet works

- ▶ "The cloud" is just a buzz word for the internet

How the internet works:

- Everything is connected to a large network that contains two types of computers:
 - >> Server - stores data in a database and sends it to the client
 - >> Client - something that people use to access the servers
 - examples: laptop, phone, printers, TVs, etc
- Each server has an IP address
- DNS: Domain Name System - it translates website names into IP addresses and back again



See you next lecture!

Understanding the Front End, Back End, and Tech Stacks



LECTURE
SUMMARY

Covered in this lecture:

Explaining what goes into the front end,
the back end, and what tech stacks are

Taught by:



- ▶ Front end is represented by what the users see and interact with
 - color, text, interactions on the web page
 - programming languages that are used for the front end:
 - >> HTML (structure and content)
 - >> CSS (style)
 - >> Javascript (interactivity)
- ▶ Back end is represented by information, servers, and databases that are located somewhere else
 - technology that is used in the back end:
 - >> MySQL (databases)
 - >> Amazon S3 (servers for rent)
 - you can use SQL queries to find information in the MySQL database

- Application layer: code or programming that does calculations to the data from the front end or the back end and it sends it to the right place
- ▶ Programming languages: Ruby, Python, PHP
- A tech stack is the set of technologies that were used to make an app, all the way from the back end to the front end



See you next lecture!

Understanding APIs - Application Programming Interfaces



LECTURE SUMMARY

Covered in this lecture:

Taught by:



What APIs are and types of APIs

- ▶ API: Application Programming Interface
- ▶ API is the messenger that takes requests and tells the system what you want to do, then it returns what you requested back to you
- Public APIs are provided by companies to allow products or services to get data from their services
 - >> Example: Searching flights on kayak.com
 - they contact various companies' databases and display their data in the results
- Private APIs are available only to developers inside a company to get information from different internal systems
 - >> Example: Facebook
 - there are probably databases holding all the user information, where developers can get the data they need

See you next lecture!

Additional technology



Covered in this lecture:

Taught by:



More advice on learning technology

- ▶ #1 Learn about frameworks
 - Frameworks are like speed dial for programmers
 - Frameworks are templates that are structured around programming languages that allow developers to get things done faster
 - >> Example: Ruby on Rails is a framework for Ruby
- ▶ #2 As a PM, it would be useful for you to learn SQL, because most companies are going to have MySQL databases
 - It makes it easy to get data by yourself, without needing help
- ▶ #3 Check out some data analysis languages, like Python
- ▶ #4 Learn about how databases work and the difference between relational and non-relational databases

See you next lecture!

Resources - Section 12

Larry Ellison & Ed Zander at the Churchill Club:

<https://www.youtube.com/watch?v=rmrxN3GWHpM>

<https://www.theguardian.com/technology/2015/may/06/what-is-the-internet-of-things-google>

<http://mashable.com/2014/01/21/learn-programming-languages/#NN61TONQaaqq>

<https://learn.onemonth.com/front-end-vs-back-end-developers-2527ee4f4c7d>

<https://sproutsocial.com/insights/what-is-an-api/>

<https://www.codecademy.com/learn/learn-sql>

<https://stackoverflow.com/questions/4811744/what-is-the-difference-between-a-relational-and-non-relational-database>

Getting relevant experience



Covered in this lecture:

How to get relevant experience
before applying for a PM job

Taught by:



- ▶ It's hard to get the job at first, because it requires previous experience. Once you get your experience, you have great chances to get the job
- ▶ There is no way to major in product management in college. In order to get the job, you have to demonstrate that you understand the concept and role of a product manager thoroughly
- There are 2 ways of getting experience:
 - at your current job
 - through a side project
- ▶ Something you do at your current job can often involve the same skills you need to be a product manager
- 1. Think about any time you've had multiple things to do, either by yourself, or with a team, and you had to prioritize what you do first

- 2. Think about all the times you've talked to any users of a product or a tool, internally or externally
 - 3. Think about any times you've given feedback or reported bugs on the product of your company or even another product
 - 4. If the company you're with now has a product team, then recall all the interactions you've had with them and how you may have helped them acquire data to make decisions or gave them feedback on the product
 - 5. Think about any times you have worked on something at your job with multiple other teams or groups of people
 - 6. Think about any work you've done to improve the efficiency of a process
- ▶ These experiences can be put on your resume and you can talk about them in the product management interviews



See you next lecture!

Building a portfolio with a side project



LECTURE
SUMMARY

Covered in this lecture:

How to get relevant experience
by doing a side project

Taught by:



- ▶ If you can't think of anything related to product management at your current job, doing a side project is the best option
 - ▶ Document your experience of coming up with ideas to solve users' problems through a new product or feature
 - ▶ Make prototypes of your ideas
 - ▶ Describe how you would change an actual product to serve its users better
- Tips:
1. Check out the other courses that Evan teaches
 2. Check out Ellen Chisa's course, "The Fundamentals of Product Management," at www.ellenchisa.com

See you next lecture!

Branding yourself



LECTURE
SUMMARY

Covered in this lecture:

How to create a strong online personal brand

Taught by:



- ▶ Another thing that will help you a lot in getting a job in product management is having a good personal brand online
- ▶ The best way to demonstrate credibility is to have a website or blog where you talk about PM and where you post your side projects
- Topics that you can write about:
 1. Why a product is poor and how it can be improved
 2. What you've learned from your top 5 mistakes
 3. Write product management book reviews
 4. Write a post on your thoughts on the effects of new technology and industry trends on the market
- ▶ It's very important that you have a public social media account where you can interact with your readers
- ▶ You can go to Quora.com and ask or answer questions related to products and product strategy

See you next lecture!

Resources - Section 13

<https://www.fastcompany.com/1769746/how-brand-yourself-celebrity-even-if-you-think-youre-not-special>

<https://medium.com/tag/pm/latest>

<https://www.udemy.com/user/evankimbrell/>

Where to look and what to look for



LECTURE
SUMMARY

Covered in this lecture:

Where to look when searching for open product manager positions

Taught by:



► #1 Looking inside your own company

- the best place to look for a job
- if you already have a product management team at your company, try to learn more about what they are doing
- show interest in their work, offer to do tasks and projects that can be added to your resume

► #2 Networking

- you need to meet people in the product management industry in order to get references
- go to Meetup.com and look for meetings in your area

► #3 Looking online

- Angellist.com - look for product management jobs
- Hacker News - look for "Ask HN: Who is hiring?" which is their monthly thread, full of open PM positions

- Facebook - look for groups around your city that involve product management jobs

- mailing lists

>> example: kennorton.com - he has job postings at the end of each newsletter



See you next lecture!

Inside advice on your PM job hunt



LECTURE
SUMMARY

Covered in this lecture:

Some tips to take into account
when looking for a job

Taught by:



- ▶ Tip #1: Make sure that the company has a good understanding of the product manager role
 - some companies don't see the difference between product and project management, and they might end up hiring you to do a different job
- ▶ Tip #2: Do a lot of research on the current product team or product management leader at the company
 - find them on Twitter, look at their blog
 - working for somebody who has worked in the industry in the past is a great way of learning

See you next lecture!

Resources - Section 14

<https://angel.co/>

<https://news.ycombinator.com/jobs>

<https://www.kennorton.com/>

https://www.meetup.com/?_cookie-check=p0xn0C7WfvoW7szd

<https://www.themuse.com/advice/how-5-product-managers-got-their-start>

<https://www.kennorton.com/essays/productmanager.html>

Resumes



Covered in this lecture:

Optimizing your resume before applying to the product management job

Taught by:



- ▶ If you don't have any prior product management experience, it's important to frame the experience you do have in a way that relates to the product manager job

Ways of optimizing your resume:

- **Step 1: Think about your relevant experience**
 - edit the subtext of each position you held on the resume and make it reflect your product management experience
- **Step 2: Make everything quantitative**
 - instead of describing what your job was, you have to describe how did you do at your job
 - product management is all about improving metrics
 - describe how you improved things at your job
 - >> Example: Don't say "I managed marketing campaigns"
 - Say "I increased the number of simultaneous campaigns managed at once by 20% over the course of the year"

See you next lecture!

Interviewing for product management



Covered in this lecture:

Taught by:



Best practices when being interviewed

► Tip #1: Use every interview to make the next one even better

- pay close attention to the questions they ask you
- take mental or physical notes for your answers
- go look the answers up if you didn't get them right
- be optimistic

► Tip #2: Ask good questions

- the more research you do before the interview, the better
- research the industry trends, the competitors, and ask them thoughtful questions

>> Examples:

- How do you try to differentiate yourselves in the market?
- What are the biggest obstacles for this company in the next year?
- What are your thought on that latest feature your competitor launched?

See you next lecture!

How to answer interview questions the right way



Covered in this lecture:

Tips on answering questions at your product management interview

Taught by:



- ▶ Product management positions are based on making good decisions based on data
- ▶ In all your answers, you have to mention how you came to that conclusion, why, and what the alternatives are
- ▶ Before giving an answer, you have to ask more details about the question, until you have enough data to base your conclusion on
- As a PM, there is never a single correct answer to a question
- You have to synthesize the information and come up with a good approach and be clear about why this is the best option

See you next lecture!

Insider tips for getting the job



Covered in this lecture:

Increase your chances of getting the job by creating a project

Taught by:



- ▶ TIP: Do a demonstration project for the company that you're applying to and send it to the hiring manager and the recruiter
- 1. Ask them some questions during your first interview with them
 - why they are hiring
 - which features this product position would work on
 - information on what they are building, the markets they target, and the clients they seek
- 2. Research the company's current product, the industry, the technology, and the competition
- 3. Develop a comprehensive overview on how you would approach the problem if you were to be hired

The final project will include:

- descriptions of your research
- your approach
- technical limitations
- wireframes of potential solutions
- A/B tests
- competitive overviews
- customer feedback

● **4. Email the completed document to them, thanking them for the interview**

- ▶ Doing this project will increase your chances to get the job, and if not, to get the next job
- ▶ You can put this project on your blog or website and talk about it in your other interviews
- ▶ Don't just wait for them to call you. Everybody else does this. Show them that you really want this position and you have the right skills to do it



See you next lecture!

Resources - Section 15

<http://todds-job-tips.blogspot.ro/2012/08/turn-your-resume-fluff-into.html>

<https://blog.ellenchisa.com/an-intro-to-pm-interviewing-a7b2e730cfbc>

<http://thepminterview.com/>

<https://www.producttalk.org/2012/09/the-most-common-mistakes-people-make-in-product-manager-interviews/>

<https://www.quora.com/What-are-frequently-asked-questions-in-product-manager-interviews>

<http://product.hubspot.com/blog/4-ways-to-get-into-product-management>

The first things to do



Covered in this lecture:

What to do after you've got the job

Taught by:



- ▶ It usually takes some time to be completely comfortable with the job and do the best work
- Tip 1: Schedule one-on-one meetings with team members
 - ask them what their goals are and what challenges they have
 - take notes so you can refer to them later
- Tip 2: Arrange a meeting with the lead engineer
 - have them explain to you what technologies they use and what the biggest technical challenges are
- Tip 3: Start talking to users
 - get an idea about what's important to your users
 - help the customer support team answer customers' questions

- Tip 4: Read as many internal documents as you can

- epics and mobile releases
- sales decks
- internal presentations

- Tip 5: Look at the data

- look at the existing metrics of the product
- find out what they plan to track

- Tip 6: Meet with the boss

- ask about the expectations for your role
- understand their goals and their biggest problems



See you next lecture!