Side Projects Shipping & Growing

From Build Side Projects With 500k+ Users: Shipping & Growing

Course Objectives

1. Understand what makes a great **first version**
2. Learn **easy, low-effort** ways to improve a product
3. Set up the right mindset for **processing user feedback**
4. Know how to build a **smooth user interface**
5. Get tactics to **acquire users organically**

Done Is Better Than Perfect

"If you are not embarrassed by the first version of your product, you've launched too late." - Reid Hoffman [Co-Founder of LinkedIn]

Try to launch your v1 within **1 week**

Overall Point of This Lesson

Just move fast, be embarrassed by that first version, and launch it super super fast

Crafting the MVP

It's All About the Hero Flow

- Every product needs to have some "magic moment"
- The hero flow is the steps the user has to take in your UX to activate this moment
- The hero flow is all that matters for your MVP

Takeaway

Find your hero flow

Planning Your Initial Launch

1. Decompose the MVP
2. Do 1 piece per day (each one doable in a few hours)
3. Repeat #2 until all the pieces are done
4. Ship the MVP within 1 week

Make It 5x Better With 0 Effort

- Make your product free and add-free
- Make your product open source (will prove I'm not taking your user data, you can just read the code)
    - Don't leak your API keys though

Least Clicks Design

Click (Step) Sources

- Clicks with a mouse
- Taps on a mobile phone
- Keystrokes on a keyboard (search, user input)
- General passage of time

One Of The Easiest Ways to Reduce Clicks

- Remove user inputs entirely (e.g. import from a .csv file)

Takeaway

Produce value in as close to 0 clicks as possible

KISS

"Simple can be harder than complex: You have to work hard to get your thinking clean to make it simple. But it's worth it in the end because once you get there, you can move mountains." - Steve Jobs

Keep It Simple Stupid

- Don't assume that users will know certain gestures, patterns, or iconography
    - Don't even assume they'll try hard to use your product
- Follow WYSIWYG principles
    - Make everything intuitive and obvious

Easy Design Wins

1. Make things more obvious (instead of long press to trigger deletion, use a trash can)
2. Increase the hitbox
3. Hand-hold users (e.g. homebrew, type brew in the terminal, great reputation of helping users)
4. Automatically handle user tasks (e.g. how Airbnb merged login and signup together)
    a. Example of least clicks design

Leverage Existing Patterns

You Probably Aren’t a Designer

- … and you don’t have to be to build an amazing side project with a great user experience and interface
- Use designs of popular products and companies as a basis
- It’s better to copy existing patterns as that’s what users are used to

Dogfooding

- Dogfooding = Organically using the product yourself
- This is a staple of Meta engineering culture
- Critical in the beginning as you need self-power for feedback
    - It’s why choosing the right idea is important
        - Build a side project that you yourself would use, that you yourself can find a place in your life for this side project to add value organically
        - So that you yourself feel the pain and the weight of your crappy code
        - You can use that to become a better engineer that builds better products

Dogfooding Tips

- Be objective - Forget you’re the developer
- Be harsh - Your bar isn’t just “Does it work?”
    - Your bar should be: Does it work well?
- Do it when you haven’t touched the code in a while
- Test features you built a long time ago (because you likely don’t remember how it works, you’re more likely to interface with it as a innocent regular user)
- Just try everything (don’t restrict yourself to just the happy flow)

Be a Feedback Spong

Feedback Is Your Project’s Growth Fuel

- Alex personally responds to the first ~250 reviews on every project he launches
- Think about what you can uniquely offer that big companies can’t
    - One powerful angle is stellar user support

One Of The First Things Alex’s Does On A Side Project

- Add a button for users to send feedback on the settings page
    - Opens the email app directly and it prefills the draft with my side project developer email so they can give me feedback

Respond to feedback within 24 hours

1 Star Reviews Are The Best

Why?

- They give you clear, actionable product direction advice
- It's free product management

Users reward attentive product owners (e.g. best feature of the app is the developer)

Observer Your Users

Giving Feedback Takes A Lot Of Effort

- Most unhappy users won't give you feedback directly
    - Instead, they'll stop using the product
- Observing your users is more powerful than listening to your users
- Set up table-stakes analytics (also useful for impact measurement [how much user value is delivered])
    - Impact measurement: how many users you have, how many core actions that they've done, how many hero moments they've unlocked
- If you see users leaving, hopefully you'll be able to find out why and fix that

Protect Your Users By Caring About These

- Crashes
- Latency (ANR, TTI, FCP, ART)
    - ANR: App not responsive
    - TTI: Time to interaction
    - FCP: First contentful paint (how long it takes to render something when someone first goes to your web page
    - ART: Average response time (e.g. you should know how fast your APIs are, if you're running a model, you should know how fast your models are, how fast your LLMs are if you're doing that)
- Failed logins
- Errors in general (non-200, exceptions)
    - You should know how often this stuff is happening
    - After you observe these things, you want to bring them down

Posthog (free)

Posthog is a really great analytics tool and it comes with a user recording feature

- There's a lot of tools out there like Posthog with a very generous free tier

SEO

Search Engine Optimization

- Improving your SEO means Google showing your website more often in search results (the more search queries where you ran on the first page the better)
- This is a staple organic and sustainable way to get traffic and users
- Billion dollar companies are built off of this

Don't Waste Your Time With These

- DR (Domain Reputation)
- Backlinks
- Core Web Vitals
- Writing a billion blog posts

What Should You Focus On?

1. Have high quality pages
2. Have a lot of them (focus on this)

This is the only SEO tool you need: https://search.google.com/search-console/about

ASO

App Store Optimization

- Getting your app to show up when people search on the Google Play Store or Apple Play Store
- This is way easier than SEO (Instead of the entire Internet, closed ecosystems managed by Google and Apple)
- Source of 90%+ of Alex's app downloads
- For the most part, just making a good app makes you rank higher
- Main thing to think about is the app/store listing title
    - Do searches and gauge app traction
    - Your goal is to find something where the auto-complete shows up and then when you tap into it, there's many apps, 100's of thousands, millions of downloads that shows this is a frothy space that you can compete in

Do These To Look Good To Google/Apple

- Have a low crash and ANR rate (<1%)
- Target the latest SDK
- Update your app frequently
- Respond to reviews (this one I'm not so sure about)

- Polish your Play/App Store Listing
    - Make amazing mobile app pictures for free here: https://app-mockup.com/

Word Of Mouth

The Best Sign Of A Great Product

- This is the most sustainable form of product growth
- If you're following the advice from this course, you'll naturally get it
- There are product-driven ways to juice up **k-factor**
- K-factor: For every user of your app, how many other people do they recommend your product to
- Ways to juice up k-factor:
    - Share flow
        - Airbnb blog post about how they sharply refined their sharing flow (an oldie but a goodie): https://medium.com/airbnb-engineering/growth-at-scale-getting-to-product-sharing-fit-ccb4b501ecf
        - Simply having a nice share sheet puts you well ahead of most side projects
        - Think about how you can make sharing as easy as possible
        - Messaging is important
            - If you click on the share button, what message does it pre-compile to?
        - Least clicks design, how can I make it as easy for a happy user of my product to share it as possible?
            - If you reduce the amount of stuff they need to type when they share your product, then you're removing clicks and you're making that experience more delightful and more likely to be used
    - Referrals
        - This is a lot trickier as you need to create some sort of incentive
        - Ideally, you can create an in-ecosystem reward
        - However, if you're following the advice in this course, this is hard to do
        - Don't feel pressure to have a referral system
        - If you can do referrals great, if not, no worries

Social Media

Powerful But At A Cost

- Getting traction on social media takes real effort
    - Just like building a successful side project
    - It can easily devolve into a job/chore

- In Alex's opinion, it's the most inorganic of side project growth mechanisms
- Not to say that you shouldn't do it, it's just that chances are, you shouldn't do it and you should prioritize it lower than the other growth channels that I've already talked about

Let's Focus On These 2

- LinkedIn
  - Is a natural fit as it's a professional social network
  - Images tend to do well on the platform (videos are dicier)
  - There's a reason you likely found Taro on LinkedIn
- Reddit
  - The most communal of the social media platforms (strong community)

Paradigm For How People Create Software Products In General

Old Way

Build a product -> Foster a community around it

New Way

Foster a community around a topic -> Build product on top of that community

- A great example of this https://www.reddit.com/r/TheRaceTo10Million/

GitHub

GitHub Is A Huge Lost Opportunity

- 95%+ of GitHub READMEs are complete garbage
  - Putting in just a 1-2 hours of effort puts you in the Top 5%
- GitHub repos are getting more stars than ever before due to homepage feed
- Go fork good READMEs

Examples Of Good GitHub READMEs

- https://github.com/tsenart/vegeta
- https://github.com/afollestad/material-dialogs

Ingredients For A Good GitHub README

- Demos (the more visual the better)
- Setup instructions
- Fun backstory
- Proof of impact (all the top tier companies that use their library)
- Testimonials / case studies

Takeaway

Put actual effort into GitHub

Managing Your Project

Don't Overthink This

- Side projects are meant to be chill
    - It shouldn't feel like Work: Part 2
- If you're senior enough, you can internalize everything
    - If you're a junior engineer and you're still building up the skills of having a plan and decomposition, writing it out and formalizing it will be quite helpful
    - Alex did this, internalized everything, for the most part
- For the love of god, please don't use Jira (overkill for a side project)

The Best Tool: GitHub

- GitHub Issues
    - Make your issues, tag them, and you're good, you really don't need much else
- GitHub is enough but if you want to use something heavier, use something like Asana

When To Give Up

Where You Want To Be

Somewhere between "give up on every side project before it gets any good" and "spend years on a project that isn't getting any traction" (like 4,6, 9 months)

Commitment Is… Tricky

- Genuine good software isn't built in a day/week
    - Even with AI
- To get users, you need to deliver quality consistently over a long period of time
- However, some projects simply won't have legs
- Work backwards from your priorities

Signs You Should Abandon A Project

- You don't enjoy working on it anymore
- You realized it's a bad product
- No traction (after 1 month+)
- Tactical limitations (e.g. the technology is not there)
- Market change

Don’t Let It Get You Down

- You’re going to have to abandon some amount of side projects if you’re following the advice from these side project courses
- Most side projects fail, especially when you’re starting out
- Giving up isn’t a sign of weakness, it’s a moment of learning
- Every failed side project makes you stronger

Go Deeper: Follow Through

One Of The Core Takeaways From This Lesson And Overall

The Power Of Tiny Gains

- 1% better every day 1.01^{365} = 37.78
- 1% worse every day 0.99^{365} = 0.03

What This Means

- Turn improving side projects into a habit
- Spend 15-30 minutes a day making **something** better
- Especially when you launch, it’s so important to be hyper responsive, I would try to push minimum of 2-3 updates a week for the first couple weeks of launching the v1 of your side project
- Even a small bug fix is progress

Big Core Learnings

- Respond to feedback lightning fast (within 24 hours with the exception of weekends)
- Strive to deliver value in zero clicks
- Put actual effort into GitHub (low hanging fruit)
- Make it free and add free
- Find your hero flow

Professor Alex’s Homework For You

- ☐ Identify your hero flow
- ☐ Ship your v1 within 1 week
- ☐ Add a “Send Feedback” button somewhere (if you want to be really spicy, you can put it in the flow of the app, where after your user activates the hero flow, maybe then you ask them for feedback, you can be creative getting the feedback)
- ☐ Make your GitHub README not garbage
- ☐ Push 15 updates minimum across 2 months (if you work on a side project, you like it and use it yourself, you believe in it - this is what Alex does before he gives up on it)

Ways To Go Deeper

- Office hours
- Taro Forum (paused for the moment)
- Attend Taro Events