Lecture 7: How to Build Products Users Love, Part I

Kevin Hale

Alright, so when I talk about making products that users love, what it means specifically is "How do we make things that have a passionate user base, that our users are unconditionally wanting it to be successful, both on the products that we built and the companies behind them?" We're going to go over tons of information; try not to take too many notes - mostly just try to listen. I'll post a link to the slides on my Twitter account, and on that link, there will be a way for you to annotate the slides. So you can ask me questions, and if we don't get to them, I'll answer them after the talk.

So you guys have been listening to a lot about growth over the last several weeks, and to me, I feel like growth is fairly simple. It's the interaction between two concepts or variables: conversion rate and churn. The gap between those two things pretty much indicates how fast you're going to grow. Most people, especially business-type people, tend to look at this interaction in a very mathematical, calculated sort of way. Today I want to talk about these things at a more human scale because in a startup when you're interacting with your users, you have a fairly intimate interaction in the early stages, and so I think there's a different way of looking at this stuff in terms of how we build our products. We'll look at a lot of different examples of that and how it's executed well.

My philosophy behind a lot of things that I teach in startups is, the best way to get to $1 billion is to focus on the values that help you get that first dollar to acquire that first user. If you get that right, everything else will take care of itself. It's a sort of faith thing.

I came to be a partner at YC by a way of being an alumni. I went to the program of Winter 2006 (it was the second-ever program), and I built a product called Wufoo. Wufoo is an online form builder that helps you create contact forms, online surveys, and simple payment forms. it's basically a database app that looks like it's designed by Fisher-Price. What's interesting though is that because it was fairly easy to use, we had customers from every industry market and vertical you can think of including a majority of the Fortune 500 companies.

I ran the company for five years, and then we were acquired by Survey Monkey in 2011. At the time, we were a very interesting acquisition. We were only a team of 10 people at the time, and while we acquired funding here in Silicon Valley through Y Combinator, we actually ran the company from Florida. We had no office, everyone worked from home, and we were an interesting outlier. So each dot here represents a startup (PowerPoint slide) that exited through IPO or acquisition, we are the outlier to the left. The bottom represents the funding amount that they took, and vertical axis is the valuation of the company at the time. To sum it up the average start up raises about $25 million, and the return for their investors is about 676%. Wufoo, raised about $118,000 total, and our return to our investors was about 29,561%.

So a lot of people are interested in what makes Wufoo a little bit different, or how do we run the company differently. And a lot of it was focused on product. We weren't interested in building software that people just wanted to use, that reminded you that you worked in a cubicle because it was a database app at its core. We wanted a product that people wanted to love, that people wanted to have a relationship with, and we were actually very fanatical about how we approached this idea, to the point where it was almost sort of in a science-y way. So what we said was like, "What's interesting about startups in terms of us wanting to create things that people love, is that love and unconditional feelings, are difficult things for us to do in real life. In startups, we have to do it at scale." So we decided to start off by asking, “How do relationships work in the real world and how can we apply them to the way we run our business and build our product that way?”

We'll go over these two metaphors: acquiring new users as if we are trying to date them, and existing users as if they are a successful marriage.

When it comes to dating, a lot of the things that we uncovered, had to do with first impressions. All of you often talk about your relationships in the origin story. You guys will tell me about your first kiss, how you met, how you proposed. These are the things that we say over and over again, basically the word-of-mouth stories for relationships. There are similar things that we do with companies. Human beings are relationship-manufacturing creatures. We cannot help but create, and anthropomorphize, the things we interact with over and over again. Whether it's the cars we drive, or the clothes we wear, or the tools and softwares we use, we eventually prescribe characteristics to it, a personality, and we expected it to behave a certain way - that's how we sort of interact with it.

First impressions are important for the start of any relationship because it's the one we tell over and over again, right? There’s something special about how we regard that origin story. Let me give you an example. If you're on the first date with somebody, and you’re having a nice dinner, but you catch them picking their nose, you are probably not going to have another date with them. But if you're married to someone for about 20 to 30 years, and you catch them on the BarcaLounger digging for gold, you don't immediately call your lawyer, you know what I mean, and say, "We have a problem here, you have to start drawing up papers for divorce." You shrug your shoulders, and say, "At least he has a heart of gold."

So something about first-time interactions means that the threshold was so much lower in terms of pass fail. So in software and for most products in Internet software that we use, first impressions are pretty obvious and there are things you see a lot of companies pay attention to in terms of what they send their marketing people to work on. My argument for people who are very good at product is that they discover so many other moments and make them memorable: the first email you ever get, what happens when you got your first login, the links, the advertisements, the very first time you interacted with customer support. All of those are opportunities to seduce.

So how do we think about making first moments? We actually took this concept from the Japanese. They actually have two words for how to describe things when you're finished with them, in terms of saying, "is this a quality item?" The two words for quality are atarimae hinshitsu and miryokuteki hinshitsu. The first one means taken for granted quality, which basically means functionality. The last one means enchanting quality. Take for example a pen. Something has miryokuteki if the weight of the pen, the way the ink flows out of it, the way it's viewed by the people reading the hand writing from the pen, is pleasurable both to the user of the pen and the people who experience the byproducts, taking it to the next level. Let's start with some examples.

This is Wufoo's login link, and it has a dinosaur on it, which I think is awesome! But if you hover it, the spec has the added benefit of having a tool tip that doesn't tell you how to log in or what it does, but basically "RARRR!" What we noticed about this in early usability stages, is that this put a smile on people’s faces, like hands down, universally. I think a lot of times when we are assessing products we never think about, "Hey, what is the emotion on the person's face when they interact with this?"

This is Vimeo's launching page; this is actually a couple iterations ago, it's the one that I find to be the most beautiful. It lets you know that when you're starting out on this journey with them, it is going to be something different - they do this all over. If you search for the word "fart," as you scroll up and down, it makes fart noises. There's something different, like this site interacts with you, it's a little bit magical, and it’s a little bit different. It's something that you want to talk about.

You don't always have to do it with design. This is a sign-up form for Cork'd, which used to be a social network for people that love to drink wine. On it, it says, "Email address- it's also your sign in name, and has to be legit. First name – what mom calls you. Last name - what your army buddies call you. Password – something you'll remember but hard to guess. Password confirmation – type it again, think of it as a test." It's literally a poem as you fill it out the form. And this is the kind of thing when you're like, "Oh, I like the people behind this, I’m going to enjoy this experience." Now what does it say, when you fill out a form like this, about what their personality looks like it's going to be? And what's disappointing to me is Yahoo forces every product and service under them to use this exact same login form.

Flickr I thought had one of the best call-to-actions. It was, "Get in there!"

This is for Heroku's signup page. I think this is an older version. What's remarkable about it is that what you start getting a feel for, is like scaling up the backend services, it's as easy as dragging up-and-down different knobs and levers. It looks fairly easy to scale.

This is for a room full of computer science people; I think you'll appreciate that. This is Chocolat, a code editor. And they only have one call to action: when the time limit is up, everything in terms of all the features is all the same, except we changed the fonts to Comic Sans, and what they're basically saying is "Hey, we know who our users are, who our real customers are. They're going to be the people who care about this."

This is Hurl, a website for checking HTTP requests, and sometimes the places where you get errors are opportunities for first moments. If you hit a 404, this is what you get: (Unicorn throwing up a rainbow.)

Often times what we do is we create really beautiful marketing materials, but when you actually need documentation, we sort of skimp out on design features. This is something that happens over and over again. A company that gets this right is MailChimp. What they did was they redesigned all of their help guides so that they looked like magazines covers, and overnight basically readership goes up on all these features, and customer support for these things that help people optimize emails, goes down.

Speaking of documentation, Stripe - what's interesting about an API company, is that there is no UX. The UX is actually just documentation, and there are opportunities even in documentation to sort of enchant and amaze. One of the things that I love about them is that their examples are wonderful, but if you log into the app, one of the things that is a super pain for most people is when you’re doing most people's APIs, is grabbing your API credentials and keys. And what shocked us is that it says, "If you are logged into the app, we automatically put your API credentials into the examples, so you only have to copy and paste once, when trying to learn their API.”

When Wufoo wanted to launch the third version of our API, we realized, "Okay finally this is good enough that we want people to build on top of it." We were trying to figure out how we launch this out to the world that sort of has our personality behind it, because a lot of people usually do things like a programming API contest that give out iPads and iPhones; it makes you look like everyone else. So at our company, one weird value we have is that our cofounders are big medieval nuts, and we would take everyone out to Medieval Times every single year on the anniversary and founding of the company. So we said we have to do something in that flavor. We contacted the guys at armor.com, and said, "Can you forge us a custom battle axe?" We said, if you win our programming contest, you would win one. The result is, people wanted to talk about this. People wanted to say they were working on this because they wanted to say, "I am programming for a weapon." What's cool is we had over 25 different applications created for us, of quality and quantity that we could not have paid for on the budget and time that we had. We got things like an iPhone app, an Android app, a Wordpress Plugin, and all we did was change the way people talked about our origin story of how they interacted with one of our services.

I'm going to shorthand this by saying you should just subscribe to Little Big Details. It's basically tons of screenshots of software that just shows that they are doing it right and being conscientious of the user and the customers.

When it comes to long-term relationships, or marriages, the only research that we ended up having to read is the stuff done by John Gottman. He's been featured in "This American Life," Malcolm Gladwell's books, etc. He's a marriage researcher up in Seattle, and he has an interesting parlor trick that he can do. He can watch a video tape of a couple fight over some issue for 15 minutes, and predict within 85% accuracy rate, whether that couple will be together or not, or divorced, in four years. If he increases that video up to an hour and asks them to also talk about their hopes and dreams, that prediction rating goes up to 94%. They showed these same video tapes to marriage counselors, successfully married couples, sociologist, psychiatrist, priests, etc. They can't predict with random chance, whether people are going to be together or not.

So John Gottman understands something fundamental about how relationships work in the long term, and that basically how we fight even in the short term period can indicate the whole system and what it's going to look like. One of the surprising things he discovered is not that successfully married people don't fight at all; turns out, everybody fights and we all fight about the exact same things: money, kids, sex, time, and others ("Others" are things like jealousy and the in-laws.) To bring this around, you can actually attribute every single one of these to problems to things you see in customer support when you're building out your products, so Money - this costs too much, or I'm having trouble with credit cards. Kids - users' client. Sex - performance, how long you're up and how fast. Others - I said was jealousy or in-laws, so that's competition and partnerships, anything weird happening there, people are going to write to you about. And the reason I like to think about this in terms of customer support is that, in everyone's processing of a conversion funnel, customer support is a thing that happens in between every one of the steps; it's the reason why people don't make it further down there; it's the thing that prevents conversion from happening.

Now as we were thinking through all of these ideas, and as we were building up the company, we realized that there's a big problem with how everyone starts up their company or builds up their engineering teams. There's a broken feedback loop there. People are divorced from the consequences of their actions. This is a result from the natural evolution of how most companies get founded, especially by technical cofounders. Before launch, it is a time of bliss, Nirvana, and opportunity. Nothing that you do is wrong. By your hand, which you feel is like God, every line you write and every code you write feels perfect; it's genius to you. The thing that happens is after launch, reality sets in, and all these other tasks come in to play; things that we have to deal with. Now what technical cofounders want to do is get back to that initial state, so what we often see is the company starts siloing off these other things that makes a startup company real, and have other people do them. In our minds these other tasks are inferior, and we have other people in the company do them.

So for us, what we're trying to figure out is how we change software development so that we inject some values that we don't talk about enough, like responsibility, accountability, humility, and modesty. We call this SDD (Support Driven Development). It's a way of creating high-quality software, but it's super simple; you don't need a bunch of Post-it notes. All you have to do is make everyone do customer support. What you end up having is you fix the feedback. The people who built the software are the ones supporting it, and you get all these nice benefits as a result.

One of them is support responsible developers and designers. When people built the stuff, they give the very best support. Now we are not the first people to think of this. Paul English was a big supporter of this in Kayak. What he did was install a red customer support phone line in the middle of the engineering floor, and it would just ring with customer support calls. People would often ask him "Why would you pay engineers $120,000 or more to do something that you can pay other people a fraction of." He said, "Well, after the second or third time that the phone rings, and the engineer get the same problem, they stop what they're doing, they fix the bug, and they stop getting phone calls about it." It's a way of having QA in a sort of nice, elegant solution.

Now, John Gottman talks about the reason that we often break up with one another is due to four major causes. They are warning signs. He calls them the Four Horsemen: criticism, contempt, defensiveness, stonewalling. Criticism is basically people starting to focus, not just on the specific issue at hand, but on the over arching issues like "You never listen to users" or "You never think about us" all the time. Contempt is when somebody is purposely trying to insult another person. Defensiveness is not trying to take accountability, or trying to make excuses for their actions. Stonewalling is basically shutting down. Stonewalling according to John Gottman, is one of the worst things we can do in a relationship. Often times we don't worry about these things in customer support, criticism or contempt. Defensiveness, you see this all the time especially in companies as they get older. But stonewalling, this is something I see happening with startups all the time. You get a bunch of customer support calls coming in, and you just think, "I don't need to answer, I don't need to respond." That act of not even getting back to them is one of the worst things you can do, and it's probably some of the biggest causes of churn in the early stages of startups.

This is how support worked out with Wufoo. When we were acquired we had about 500,000 users on the system, 5 million people used Wufoo forms and reports whether they knew it or not, and all those people got support from the same 10 people, and usually there was one person dedicated to support a day, for any shift. Resulting in about 400 issues a week, that's about 800 emails. But our response time from 9 AM to 9 PM was between 7 to 12 minutes, from 9 PM to midnight was an hour, and then on the weekend it would be no longer than 24 hours. We carried this up all the way up to the scale.

What a lot of people talk about and often forget about Airbnb, is how they did this interesting thing where they went up to New York, and offered professional photography, and the founders would go up there and actually take pictures of the people's apartments to help them sell more, focusing on the stories about conversion. What most people don't realize is, a lot of times when I saw Joe in the early times of Airbnb, he had a phone headset stuck to his head all the time because he was doing phone support nonstop.

Churn is the story we don't like to talk about. Airbnb's growth really started picking up when they figured out how to match capacity to the demand or the phone calls they were getting into their support system.

At Wufoo we actually constantly did experiments around support because we were so obsessed with it. One experiment we did was, we heard someone here do a talk about how there's a disconnect between the emotions that we have when we need help, and the content and the reaction we get from people when we get help to people, especially online, because they just don't see those nonverbal cues. So she said, unless there's face recognition on the web, we are just always going to be disconnected from our users. Our feeling was like, "We're not face recognition experts, but we think there's another way of getting empathy." So, as form builders we added a drop-down, and what we said was, "what's your emotional state." Our hypothesis was nobody's going to fill this out; we thought this was going to be a lame experiment, but we'll see how it goes. It turned out that this field was filled out 75.8% of the time. The Browser Type drop-down field in comparison was filled out 78.1% of the time. So people were basically telling us, "For my technical support issue, how I feel about this problem is just as important as all the technical details you need to figure out in order to debug it."

We didn't prioritize things or triage things by emotion, so for the most part people didn't game the system. One of the interesting byproducts of it was that we noticed that people started being nicer to us. We went back and looked at the data, did some text analysis and realized when it comes to communicating with people over written words like email, there's only three ways in which you show strong emotions: exclamation marks, curse words, AND ALL CAPS. Sure enough, on all three of those metrics, they've gone down in the way people were talking to us in customer support. Once people had a simple outlet for their emotions, it made them a lot more rational, and a made our jobs much more pleasant as a result.

The other byproduct that is awesome is that you actually build better software when you do this – far better software. This is actually backed up by a whole bunch of research. Jared Spool, at User Interface Engineer (one of the biggest players in the space) says that there's a direct correlation to how much time we spend directly exposed to users and how good our designs get. He said it has to come in this specific way. It has to be a direct exposure. It can't be something where someone generates a report or through a graph. You have to be interacting with them in somewhat real time. It has to be a minimum of every six weeks, and it has to be for at least two hours; otherwise your software will get worse over time. Our developers, the people who are with Wufoo, are getting exposed to our users 4 to 8 hours every single week. What it does is that it changes the way you sort of build software.

Jared Spool has another way of talking about how we build products. Imagine that this represents all the knowledge needed to use your app on a spectrum (PowerPoint slide). This is like no knowledge (far-left) and this is all the knowledge needed (far-right). These two lines are pretty much your interaction with users. This is currently where their knowledge point is (PowerPoint slide), and this is the target point where you're trying to get them to. The gap between the two is called the knowledge gap as Spool calls it. And what's interesting about this is there's only two ways to fix this. That gap represents how intuitive your app is. You either get the user to increase their knowledge or decrease the amount of knowledge that's needed to use the application. And often times as engineers or people who build and work on these products we think let's add new features. New features only means let's increase the knowledge gap.

So for us we actually focused on the other direction. What that meant is that we spent 30% of our engineering time on internal tools to help with our customer support. But often times it was spent on helping people help themselves, like frequently asked questions, or tooltips; things like if you just click the help link, instead of taking you to the generic help documentation page, you go to the specific page that's going to be the most appropriate for what you're working on. We redesigned our documentation over and over again, A/B tested it constantly. One iteration of our documentation page reduced customer support by 30% overnight. It's one of those things where overnight, all the people that work on the product immediately had 30% less work to do.

What happens if you have everyone working on customer support constantly? I talked about in the very beginning that growth is a function of conversion and churn. This is Wufoo's growth curve for the first five years (PowerPoint slide). What's interesting is that we paid no money on advertising or marketing; all of it was done by word-of-mouth growth. The interaction between new users and downgrades are this (PowerPoint slide). It's so slight what it takes, that gap, making that work. What a lot of people keep forgetting is that there's almost no difference between an increase in conversion rate, 1% increase, and 1% decrease in churn; they do the exact same thing to your growth.; however, the latter is actually much easier to do, and much cheaper to do. And a lot of times we neglect this until way far along, and we usually have our B team work on these projects and services.

This is actually not one of the graphs we tracked most of the time at Wufoo, it's not even the one I'm proud of. This is the one I'm proud of because even though we had this nice, awesome curve of growth, this is what allowed us to scale, keep the company small, and have an awesome culture. And that required doing a lot of these things to help people do what they need.

So John Gottman noticed that there was a different type of behavior for relationships, and why people divorced. Basically there were these subsets of people who stayed together 10 to 15 years and all of a sudden divorced. None of the other indicators would show that this was going to happen. He was looking through the data and realized, "Oh there's no passion, there's no fire between these people." When it comes to relationships, they kind of follow the second law of thermodynamics: In an enclosed energy system, things tend to run down, so you constantly have to be putting energy and effort back into it. The way a lot of people think showing people that "I care about you" in products and companies is by doing things like creating a blog or making a newsletter. The thing is we look at these rates and basically it was such a small percentage of our active users, most of them had no idea about the awesome things that we were doing for them. So we built a new tool and called it the Wufoo Alert System. It allowed us to timestamp every new feature that we are building for users, and that every time they would login we would look at the difference between their log in time, or last login in time, and the new features that were implemented, and they would have this message show up, "Hey since you've been gone, here's all the awesome stuff that Wufoo did for you." Hands-down, this was the most talked about feature that I heard every time I went out to talk to users. They would say things like, "Dude I love that 'Since you've been gone' thing. Even though I pay the same amount every single month, you guys are doing something for me almost every week. It's totally awesome; it makes me feel like I'm getting maximum value."

The other thing that we did in addition to having everyone support the people that paid their paycheck, is have them say "thank you." And this was in large part due to us injecting humility and modesty into the equation. Every single Friday we would get together, we would write simple handwritten thank you cards to our users. And I know there are tons of people who would not be sort of excited about doing this; it was a ritual that made all the difference in terms of like having a team that was very tightly knit, and working on stuff that they really cared about. They constantly knew what the mission was for, and why we sort of did what we did. These aren't fancy thank you cards; they're just simple handwritten stuff on index cards, we threw in a sticker, and slapped on a dinosaur on the front of it.

What's interesting is we started this practice as a result of the early days of starting Wufoo. Chris, Ryan, and I were talking to try to figure out what we were going to do to show users that we appreciated them around Christmas, and Chris came up with this idea where he said, "Hey guys, a couple years ago my mom made me write thank you letters to all my relatives for my Christmas presents, and I really didn't like to do it, but the following year all my presents were super good... so I think we should try this for our business and see how it goes."

So that first year we wrote handwritten Christmas cards to all of our users that first year. Second year rolls around, and we have too many customers with just the three founders. We were thinking, "We're kind of screwed; we don't know what we're going to do." Well, we read a book called The Ultimate Question and in it, he talks about focusing on your most profitable users; if you just take care of them, things will work out. So we thought, "That will work out, that's scalable." Basically we only wrote to our highest-paying customers. So January rolls around that second year and one of our longtime loyal users writes to us. He basically says, "Hey guys I really loved the Christmas card you guys sent me the first year, and I just wanted you to know, I haven't received my second card yet and I'm just looking forward to it; I know you didn't forget about me. Thanks a lot." So we were like, fuck. The best way to exceed expectations is not to set any in the beginning; we were sort of in this conundrum. What we decided after thinking about it for a while was that we had to stop doing it just one time a year; it needs to be something that happens every sort of week. And even though we'll never catch up to all of our customers, just a practice of doing it will make all the difference.

I talked a lot about lovey-dovey, touchy-feely stuff that I think a lot of engineers don't like to think about too often, so I'll end on hard business data or research. There's an article that was put out by the Harvard Business Review several years ago by Michael Treacy and Fred Wiersema and in it they talk about the discipline of market leaders. They say there's only three ways that you achieve market dominance, and depending on how you want to achieve that market dominance, you have to organize your company in a very specific way: best price, best product, and best overall solution. For best price, you focus on logistics, so Wal-Mart and Amazon. If you want to be the best product out there, you focus on R&D, Apple is usually a quintessential example of that. Best overall solution is about being customer intimate. This is the path that you see all luxury brands follow, as well as the hospitality industry. What I love about this path towards market dominance is that the third one is the only one that everyone can do at any stage of their company. It requires almost no money to get started with it. It usually just requires a little bit of humility and some manners. And as a result, you can achieve the success of any other people in of your market. That's all I got, thank you very much.

Q: So what do you do when you have a product with many different types of users? How do you build one product that all these users love?

A: There is an interesting fine line for that. What I usually tell people is focus on those who are the most passionate, especially in the early stages. Whatever niche it's going to be in, that's where I'm going to focus on completely. I think Ben Silverman from Pinterest started off with design bloggers. Tailor your thing for them, and eventually you'll figure out universal values that will appeal to a lot of other people. So just start one at a time. And a lot of examples that you see up there, a lot of companies make the mistake of just thinking "Oh I'll just make my app funny." Humor is like really difficult to do. When you want to shoot for something witty, you have to get functionality right. So like the Japanese quality. If you don't have atarimae, don't try to do anything witty, because it'll backfire. So hands down our number one focus in Wufoo was to make everything as easy to use as possible; everything else was just polish.

Q: How do we balance being obsessed with working on product with all the other skills that are needed by a company, such as marketing, branding, etc?

A: If you're working on product, you should also always have this flip-side where you're talking to users. For us inside of Wufoo, the way we got people to talk to users was through customer support. They got to see firsthand whether the features worked or not, and it also impacted everyone else in the company because everyone had a customer support shift, so they had a social incentive to make everything work. There should be no point where you are only focused on product. You should always have time where you work on product, and then you see what users say to you - like ongoing virtual feedback. So be careful when you don't have that.

My feeling on marketing and sales, my feeling is marketing and sales is a tax you pay because you haven't made your product remarkable. Word-of-mouth is the easiest kind of growth, and it’s how a lot of the great companies grow. Figure out how to have a story that people want to tell about your product where they are the most interesting one at the dinner table. And then that person is your sales person. That person is your sales force for you.

Q: How do you make a decision on product and communicate that with your engineering team when there are lots of different directions to go?

A: We just looked at customer support, which is really easy because you see what people are having the most amount of trouble with. You cannot help but get feature requests from people. No matter whatever openings you have in your product or app, people will jam feature requests in there, so you're going to know what they want. Your job as a product person and an engineer is to not just do what they say, because that way you'll just be a slave. You have to figure out and solve what they really want, that deep underlying reason. The thing is if everyone wants to have a different way to go, then ultimately someone's going to figure something out. But also, make the smallest version of each little idea, no longer than 1 to 2 weeks to build it, so you can try it out to see what works and what doesn't. It's dangerous to have multiple product directions that require a lot of time to figure out.

Q: Can you relay the story about how the King for a Day thing was good at Wufoo?

A: Yeah, okay so I don't like hackathons. I think they sort of suck in terms of those done inside of companies because you spend like 48 hours working really hard on something that you're really passionate about, and 99% of them never make it out to production. It's super sad. So we came up with an idea called "King for a Day." It worked over the weekend. How it worked is someone randomly in the company got drawn and they got to be the king. The king got to tell everyone else what to do on product. So everything that was bothering them about Wufoo or any other feature that they wanted to have built, they got engineering, marketing, and advertising resources of everyone in the company to make it happen. And of course we worked with them to figure out what we could do in 48 hours. We would do this one to two times a year. It was a huge hit and a boost to morale because what people most loved, was working on things that they felt made a difference, like, I made a difference to the app. So for us, that's one way that we would divide time for product direction. Sometimes, the people that work for you are the people who have the strongest opinions about where the product should go. And that's a good way to democratize it a little bit, by rotating it around.

Q: You said you guys all work from home, which usually seems like a nightmare. How did you make that work?

A: We all work from home, and we all work around the Tampa Bay area. We would allow anyone to work from anywhere but usually as we tried to recruit them and meet our team, they usually decided to come and move here anyway. Remote working is especially tricky. A lot of people like to romanticize it, especially people who are employees, but the thing is an office gives you a lot of benefits and efficiencies that you now have to compensate for when you have remote working. But remote working also has these sort of efficiencies. For example, I don't have to worry about my employees losing two hours of their day to commuting. So the biggest thing we had to do for remote working is to respect people's time. The way we had it set up is we actually had a 4 1/2 day workweek at Wufoo; half-day on Friday was for all the meetings and stuff. We said, no biz dev meetings, no talking with other outside parties. They'd have to be done on Friday, on that half-day; they couldn't be done in the middle of the week. And then also one day of everyone was already dedicated to customer support. So everyone in our company effectively only had three days each week to actually build and work on whatever they were doing. But I actually firmly believe that if you have three solid days, 8 to 10 hours, when you're only working on what you need to build, you can get a ton of shit done. So, what we said was, you have to respect everyone's time during that three day period.

What we came up with was a 15 minute rule. You could have a chat or a phone call with someone, but it could last no longer than 15 minutes. So if you had some complicated issue that you couldn't figure out, at 15 minutes you are to immediately table that item, and have us discuss it on Friday. You'd move on to the next item on your list. I would say 90% of the time, the item never got brought up on Friday, because usually what would happen is people would sleep on it, and then you would magically say "Hey I found a solution!" Or "Hey that's not a big problem whatsoever." Most problems inside a company don't need to be solved in real time or right away. The only things are like when the site is down or when payments aren't working. Everything outside of that is kind of luxury. So focus on your priorities as much as possible, and as a result our 10 person team did far more than many many other companies.

But it takes extra work to make remote working happen. We are an extremely disciplined team, and I would have to say, there are not many YC companies that have been able to replicate what we do. I think there are only two companies in YC that have been able to replicate our discipline style. It takes more work in a very different fashion. And often allows you to be a little bit lazier, in terms of all these things around productivity.

Q: How do we set up accountability for our employees as a manager?

A: We were profitable nine months after launch, so we had profit-sharing, which makes incentives pretty simple and clear. It would be a multiple of whatever sort of bonus pool that we had, and performance measures would be based on how they did in customer support, on their duties there, and what they said they wanted to accomplish. I don't like process and I don't like a lot of tools to help people to be productive, so the only thing that we had to help people manage their projects is a To-do list. It was a simple text file that we shared in a Dropbox account. Each person had their name on it, and you got to see every time someone updated things on the To-do list. What we said was every single night, you wrote everything you did that day, and on Friday we would just go over "This is what you said last week you were going to do; this is what you actually got done. What are the problems at hand?" And it's super simple. It creates this nice written trail for how to handle stuff, and I don't have to worry about managing them. They set the tone for how they want to be assessed. And for people who are excellent at what they do, it works very very well. And when you actually have problems, it's very easy to fire people. I was fortunate enough not to have to fire anyone at Wufoo, but we were able to correct everyone's behavior very very quickly because we just looked at this and evaluated the problem: "Look this is a pattern of behavior. You've been doing your work at last minute, etc. This is evidence that you've provided to us; all we have to do is describe it back to you." And because everyone in the company sees it, there's social pressure that's put into place to help make it all happen.

Q: How do you hire people that can work remotely and work in this fashion?

A: Pretty easily, you have them work on a side project for you. So you contract them out, and have them work remotely as such. Usually the projects I like to have them work on are about one month long so you can get a good sense of how people manage themselves and work on things. That was always the first assessment; we never did things just by interviews.

The other thing we had to screen them for was their ability to do customer support because not every engineer has those empathy skills to handle that stress. So sometimes I would have people write "break-up" letters to me in an interview, giving them 15 minutes to write it. That way you get a good sense of their writing skills because 90% of what you're doing in customer support is telling customers bad news like "we don't support that feature, sorry," or "no that's not going to work," or "that's not going to be available."

Q: Are there any tricks or experiments that didn't work out in your company?

A: Okay I'll talk about one. So one of the things that we did early on to try to motivate ourselves was - like, we understood the idea of crunch mode, and that it's really bad for people. Like if you're doing the subscription business, you need people to last for the long-term, and in video games, a lot of the time they crunch people for a specific time and they have multiple sprints. Most of the time the deadlines can get super exhausting. You might get an increase in productivity, but the recovery that you need for people is always greater than the productivity you gain. And in a company where you need everyone doing customer support, being on their game, and constantly pushing out features, you don't have time for recovery.

So we were thinking that we wanted to build a company vacation into how Wufoo works to reward our users every single year. So we thought, if the vacation is built in for the recovery, we can have one crunch period before the vacation set up and only do customer support that will sort of scale with people. So the way we did the very first crunch mode was that, it was just between the three founders, and we had each of us draw a 10 item To-do list that would be fairly aggressive. The first person to get through seven of their items would win, and the last person to get through seven of their items, would become what we called "Trip Bitch." Trip Bitch meant that you carried the other person's luggage and got people drinks when you're on the company vacation. So we did that, and during that period, everyone was pretty excited about it. The winner also got to choose the next company vacation. But all of a sudden, Ryan had basically poorly estimated the items on his list and realized very quickly, "I'm going to fucking lose," and he just sort of gave up. So crunch mode, turned out to be blah mode for him because he knew he was going to lose and became pretty demoralized. So as a result of that we decided not to do it in that similar fashion anymore. Good idea that we like to talk about, but it was one that we never did again.

Alright guys, thanks a lot! You can email me at kevin@ycombinator.com.