Unsplash © Cade

This is why your customers hate your Minimum Viable Product

There are two distinct schools of thought around the concept of Minimum Viable Product (MVP). Some people think MVPs are the scourge of tech startups. They’ll point out that there’s a growing swamp of hastily-thrown-together apps, which constantly lower the quality curve of the ecosystem by purposely offering substandard builds. Harsh, but not wrong.

Others believe that MVPs are the single best way for an entrepreneur to prove product-market fit without sinking a ton of time and money into overbuilding a complete product that may never sell to a single customer.

They’re actually both right, but all that tension between the two camps springs from a number of misunderstandings about what makes a good MVP.

Bad MVPs put a lot of friction on the

customers

The MVP concept didn’t originate in the mobile app world. In fact, the concept has been around far longer than mobile devices and the cloud. That said, the emergence of the cloud as a primary development environment has made it much easier for startups to develop a new software product iteratively, and release that product to market in smaller and smaller chunks.

Meanwhile, the Internet, and more specifically, the mobile Internet, has made it possible to deliver perpetual versions of a software product at a frequency that we couldn’t even dream about back when software was shipped on physical media.

This means that at any given time, one small change can be made to a product, and within minutes, that updated product can be shipped, installed, and quickly expose new functionality to its customers.

So think of an MVP as the first change — from a product that does *nothing* to a product that does *something.* That brand new product is then released to the market to find and prove fit.

That’s where you get the Minimum. That’s where you get the Viable.

But that’s also where you get the confrontation. Because the bill that comes due for all that minimum viability is a ton of manual work, functionality that breaks, difficulty in usage, and poor performance at scale.

If you want your customers to love your MVP, that bill needs to be paid by your startup, not your customers.

Bad MVPs get launched too early

An MVP is not a product test, it’s a product market test. It’s not a beta test of the technology. It’s a pilot to determine product-market fit.

This means that before launch, these three basic assumptions about your MVP should already be in place:

1. It will work. Don’t treat your customers like a public QA team. They’re paying you. And if they’re not paying you, this isn’t an MVP, because unless the product is generating revenue, there is no way to determine viability. There’s nothing wrong with holding a public beta test of your technology, but at the end of that test, you’ll only know whether or not your technology will work, not that your product will sell.
2. It will sell. There are dozens of ways to market test your concept before

launch. You should have predetermined sales expectations when you launch an MVP, and if you don't meet those expectations, you can go back to your market testing and figure out why it undersold.
3. It will be used more than once. This is the assumption that trips up a lot of MVPs, because they're all splash and no value. Before you launch, you should know where your product creates value for the customer and where that value proposition can be repeated over multiple uses.

These tenets are rigid and inflexible. If you're not confident about any single one of them, your MVP is not ready for launch.

Read also

How to scale a startup to 50 employees without losing its culture

Bad MVPs overreach on functionality and use cases

An MVP is also not meant to test whether or not your customers will find value in every single bit of functionality you want to throw at them.

An MVP should be launched with what I call "core value." In other words, there should be one functional area that the MVP does very well, and when it does, data about the core value will imply the potential success of whatever derivative functionality you want to sell on top of it.

Whatever your product does, find the core value and launch that and only that as your MVP. That product should complete one use case, support very few outliers, and maybe only target a certain customer niche. Of course you can add a feature here and there, but the simpler you start, the more you learn, and the more successful each iterative step will be.

Also, don't get overly concerned with pricing, price model, upsell, or scale. Do one thing really well for the customer who will find the most value from that thing. The data that comes back from your MVP launch will be much more accurate at any darts you might have thrown at pricing and costs and scale before launch.

Bad MVPs treat the customer like a user

Here’s the actionable part of how to build a better MVP.

Your MVP should be able to quickly and completely handle all aspects of customer flow. If you do these things very well, you’ll keep most of the friction away from the customer.

I can find the MVP. Yes. You would be surprised at how many times a startup will make it difficult for their customers to find their product. Your first contact with your customers should never be about your startup or your mission or your bio or your history or your team or anything but the product.

I can try the MVP. Whether it’s a free trial, a fully interactive demo, a video, or just words and pictures, the customer should get a full sense of what your product looks like and how it works.

I can decide if I need the MVP. Quickly present all the information the customer needs to make a purchase decision, including price, terms, delivery method, and any prerequisites. And please notice I said NEED and not WANT. The difference between need and want is the difference between a successful product and a failed product.

I can buy the MVP. A callback to an earlier rule, an MVP needs to be *purchased* to prove viability of the product on the market. But beyond that, make sure that all your eCommerce and payments infrastructure is working, as is your method of delivery. Confirmation of the status of their payment or any impending payment is also a must.

I can start using the MVP. More than just an instruction sheet, your MVP should onboard the customer, get them started, and bring them face-to-face with the core value as quickly as possible

I can get help. Support options must be easy to find, easily accessible, and you should be quick to respond with an answer. If you don’t have a human staffed 24/7, make sure your customer knows the time frame for resolution. And it doesn’t hurt to set up alerts internally to make sure that someone does respond quickly.

I can give feedback. One of the most important aspects of an MVP is to get customer feedback on the product. Beyond just tracking their usage or popping up a rating solicitation, proactively solicit feedback from customers at regular intervals. Also make it easy for them to give you feedback unsolicited.

I can quit the MVP and get a refund. One of the main stopping points in a customer’s buying decision is if they aren’t sure how they can get their money back. Even if you don't offer refunds, let them know that so you don't waste their time during the buy decision, because a maybe always winds up a no.

When you nail all of these customer-facing elements, then it's time to launch your MVP. Remember, don't launch a broken product that does a lot. Launch a product that, from the customer's perspective anyway, does one thing perfectly.

This article was originally published on Medium by Joe Procopio

Joe Procopio is a multi-exit, multi-failure entrepreneur. He is the founder of startup advice project TeachingStartup.com and is the Chief Product Officer of mobile vehicle care and maintenance startup Get Spiffy. You can read all his posts at joeprocopio.com

If you want more direct advice and answers, look into Teaching Startup.

Article by JOE PROCOPIO