|  |  |
| --- | --- |
| **Want To Be a Better Developer? Work in UX**  I only worked as a PM/UX designer for two months during my first internship, but it taught me more about engineering than my time as a developer.  https://cdn-images-1.medium.com/max/1250/1*X8GPKCO8drb6nCdjEw9TnQ.jpeg  Photo by [rawpixel](https://unsplash.com/photos/SMJC_JJm7W4?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText) on [Unsplash](https://unsplash.com/@allychow/likes?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)  Technical skills are a must when it comes to tech. But for most people, algorithms and implementation are just a Google search away. That’s not to say that engineering is just copy-paste — the real engineering comes in figuring out the *how*and *why* we build products.  So while I gained a lot of experience interning as a software developer, I find my short time as a PM more valuable in terms of self-improvement and general learning as an engineer.  **1. Start with the MVP**  I have a bad habit of getting caught up in building super cool features that I’m excited to test my technical abilities on.  I spent a long time during my first internship coding up a feature I loved, but that no one would ever get to use. It worked, but I hadn’t built the necessary foundational aspects to the project that would enable them to get use out of my feature.  The users needed the basic functionality implemented first, otherwise why would they waste their time on a product that couldn’t do what was necessary?  *You can’t sell someone a house if there is no house, even if you have the nicest windows and tile choices. You still need the damn house.*  In other words, it doesn’t matter what you build if no one is going to use it.  **Build the MVP (minimum viable product) first**, so that your users will see that they can accomplish what they need on it, and *then* go add your fancy features.  **2. You need to talk to people.**  Why do you need to talk to users?  Sounds like a stupid question, I know, but as a developer I found myself asking that question because all I wanted to do was start writing code.  I thought that I could just put myself in the shoes of a user, and figure what they needed just by thinking real hard about it.  But even if you *think* you know what a user wants and needs, you don’t really know. Especially if you aren’t a user yourself.  https://cdn-images-1.medium.com/max/750/1*O_Qt_nL1VHUCsbbc5KNxag.jpeg  Photo by [Samuel Zeller](https://unsplash.com/photos/VK284NKoAVU?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText) on [Unsplash](https://unsplash.com/@allychow/likes?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)  Someone once described it to me as being like VR. You, the developer, put on the headset and experience what a user would through virtual reality.  You can simulate being a user, but how do you build the right environment? Someone has to build the environment and experience that you get when you put on the headset. If you’re also the one describing the scenario, then the scenario is limited by your imagination and understanding. It’s completely different to what someone who uses the product would describe the experience to be like.  *You can’t expect to learn anything new about the product you designed from yourself.*  You can’t know just by thinking about it, because your thoughts are shaped by your experiences and knowledge. You need someone else to teach you and frame the product to see and experience it like they would.  You aren’t them. And odds are, you aren’t designing a product for you, you’re designing it for them.  **3. Throw away your surveys, because you need interviews.**  We thought that we could reach a larger user group if we sent out surveys instead of going to meet with every single person.  And of course, surveys are cheaper, cost less time, and can go out to a huge audience.  **Unfortunately, most people don’t take the time to actually fill out the surveys.**  Especially if they don’t know you (no one knew us, a few random new interns), and especially if they don’t already desperately want what you’re selling (which they didn’t).  Even when we tried to sell them on our product first with an outline of our project preceding the survey, we got minimal responses. Most people are going to ignore things in their always filled mailboxes if it isn’t urgent, if they don’t recognize the sender, or if it gets filtered out.  So how do you get users to buy into something they don’t desperately need, from you, who they don’t know or care about, in a short period of time?  Well, we spent the better part of two weeks organizing face-to-face interviews with potential users who, if bought in, could spread word for us to other potential users. We went to meet all of them, pitched our project, got their feedback, and they became our first users. The conversion rate was pretty high after we got them to spend some time understanding what we were offering, and convincing them of the value they didn’t know they needed.  **Was it time consuming? Absolutely.**  **Was it more valuable than surveys? Exponentially.**  *Also some of our interviewees brought along some exciting tech and surprises to share with us, which made these interviews super fun.*  **4. Good user experience breeds successful engineering.**  I’m the most proud of my coding when I can leave it to run on its own when I’m done.  If it’s self-sustaining, then I won.  If I can leave it and it works great without me, then it can be used (so I built something valuable) and I don’t need to babysit it anymore. Which means I can go work on the next new exciting project.  However, if users are constantly confused and are making mistakes, then your email is going to be flooded with requests and questions. So even if you’re done coding, you’re not done correcting data sets or responding to angry people who don’t want to use your product anymore.  What I learned, is that a lot of this can be avoided by thoughtful design and description.   1. Hide anything and everything that users don’t need. If they can’t see it, then they can’t click on it, and no mistakes can be made. Don’t just make it greyed out, because then your users will want to know why they can’t use it, even if it has nothing to do with them. 2. Use colour to draw attention to main functionality and what users will find important. 3. Loading messages or animations are EVERYTHING. If users can’t see that something is happening, then they don’t know and they assume there isn’t. That means repetitive clicks and changes that your code might not know how to handle. Just give them a damn spinner! 4. Link to documentation with FAQs. 80% of the questions I got from users were about the same kinds of things. Redirecting them to a FAQ page meant my inbox was a lot less full. 5. Accessibility is key to allowing as many potential users as possible to get some use out of your site (and it’s not hard to do!).   Good design helps users learn how to use their product on their own, which is important because no one wants to read the instruction manual.  Save yourself some time and headaches while giving more users the tools they need to love your product through thoughtful design. | **Хотите стать классным разработчиком? Работайте с UX**  В свою первую стажировку я проработал продуктовым менеджером и UX-дизайнером всего лишь два месяца, однако узнал о проектировании куда больше, чем за время работы разработчиком.  https://cdn-images-1.medium.com/max/1250/1*X8GPKCO8drb6nCdjEw9TnQ.jpeg  Фото [rawpixel](https://unsplash.com/photos/SMJC_JJm7W4?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText) на [Unsplash](https://unsplash.com/@allychow/likes?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)  Технические навыки – это must-have, когда речь идет о технологиях. Но большинство людей считает, что про алгоритмы и реализации можно просто нагуглить. Нельзя сказать, что проектирование сводится к банальному копипасту – настоящее проектирование сопряжено с выявлением причин, *как* и *почему* мы создаем продукты.  Несмотря на обилие опыта, полученного при стажировке по разработке ПО, наиболее полезным с точки зрения саморазвития и общего накопления знаний по проектированию я считаю именно непродолжительный стаж пребывания продуктовым менеджером.  **1. Начните с MVP**  Есть у меня дурная привычка – с головой погружаться в создание навороченных функций, на которых можно опробовать свои технические навыки.  Очень много времени в своей первой стажировке я тратил на написание кода полюбившихся мне функций, которыми никто и никогда не смог бы воспользоваться. Все работало, но я не выстраивал нужной фундаментальной базы проекта, которая помогла бы людям воспользоваться написанной функцией.  В первую очередь пользователям важна реализация базового функционала. В противном случае, зачем они будут тратить свое время на продукт, который не делает того, что им нужно?  *Нельзя продать кому-то несуществующий дом, даже с самыми красивыми окнами и шедевральной плиткой. Все равно людям нужен именно дом.*  Другими словами, не важно, что именно вы создадите, если этим все равно никто не воспользуется.  **Сначала выпускайте MVP (минимально жизнеспособный продукт)** – так пользователи увидят, что получат именно то, что им нужно. И только *потом* добавляйте туда навороченный функционал.  **2. Говорите с людьми.**  Зачем вам нужно беседовать с пользователями?  Знаю, звучит глупо. Но как разработчик я не раз задавался таким вопросом, ведь единственное, чего мне действительно хотелось, – это поскорее приступить к написанию кода.  Мне казалось, что я и сам вполне подойду на роль пользователя, и усилием мысли смогу разобраться во всех пользовательских потребностях.  Но даже если вам *кажется,* будто вы знаете, чего именно пользователь хочет и ждет, вы гарантированно заблуждаетесь. Особенно, если сами вы не являетесь пользователем.  https://cdn-images-1.medium.com/max/750/1*O_Qt_nL1VHUCsbbc5KNxag.jpeg  Фото [Samuel Zeller](https://unsplash.com/photos/VK284NKoAVU?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText) на [Unsplash](https://unsplash.com/@allychow/likes?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)  Однажды кто-то сравнил этот процесс с виртуальной реальностью. Вы, разработчик, надеваете гарнитуру, и с помощью виртуальной реальности начинаете видеть все глазами пользователя.  Да, вы можете притвориться пользователем, но как подобрать нужную среду? Кто-то должен будет создать эту среду и опробовать то, что вы получите при подключении гарнитуры. А если сценарий тоже пишите вы, то он будет ограничен вашим воображением и пониманием. И все это будет кардинально отличаться от того, как другой пользователь продукта опишет собственный опыт.  *Вы не сможете научиться ничему новому о продукте, который сами же и создали.*  Одних мыслей и умозаключений здесь будет мало, ведь ваш процесс мышления выстраивается на основании собственного знания и опыта. Вам потребуется другой человек, который покажет и разъяснит структуру проекта – так вы сможете увидеть его глазами других и прочувствовать тоже, что и они.  Вы – не они. Причем, скорее всего, продукт вы создаете не для себя, а для других.  **3. Выбросите опросники. Вам нужны интервью.**  Нам казалось, что более широкую группу пользователей можно охватить рассылкой опросников, а не личными встречами.  Конечно же, опросники намного дешевле, занимают меньше времени и позволяют обратиться к большей аудитории.  **К сожалению, большинство людей не тратят время на вдумчивое заполнение опросников.**  Особенно, если вы лично не знакомы (а ведь нас не знает никто, кроме парочки новых стажеров), или они не испытывают ярко выраженной потребности в том, что вы продаете (а именно так всегда и бывает).  Даже когда мы пытались продавать продукт с наброском проекта до создания опросников, то получали минимальные отклики. Большинство людей никак не реагируют на письма, если в них отсутствует срочность, они не знают отправителя или письмо не проходит через фильтры почтового сервера.  Каким образом заставить потребителя как можно быстрее купить что-то, не особо ему нужное от вас – незнакомого человека, до которого пользователю нет никакого дела?  В итоге мы потратили почти две недели, проводя личные встречи с потенциальными пользователями, которые при положительном решении о покупке, могли бы рассказать о нас другим. Мы встретились с каждым персонально, представили наш проект, получили обратную связь, и обрели в их лицах наших первых пользователей. Коэффициент конверсии возрос после того, как мы заставили людей потратить время на вдумчивое изучение нашего предложения, и убедили их в ценности продукта, потребности в котором они не ощущали ранее.  **Было ли это затратно по времени? Конечно.**  **Оказались ли интервью более ценными, чем опросники? Еще как!**  *А еще некоторые кандидаты принесли с собой всякие интересные технические примочки и сюрпризы, что сделало само интервью куда более занимательным.*  **4. Хороший пользовательский опыт ведет к успешной разработке.**  Наибольшую гордость от написанного кода я испытываю тогда, когда после завершения работы он может выполняться самостоятельно.  Если код может поддерживать себя сам, то я победил.  Если все работает без моего участия, то продукт можно пускать в дальнейшую реализацию (то есть, я создал нечто ценное), и мне не придется лишний раз нянчиться с кодом. То есть, я смогу взяться за следующую интересную разработку.  А если пользователи постоянно путаются в программе и допускают ошибки, то вас ждет целый шквал вопросов и запросов по электронной почте. И даже если работа над самим кодом уже завершена, то корректировка наборов данных и общение с разъяренными и отказывающимися от вашего продукта пользователями идет в самом разгаре.  Я понял одну вещь: очень многого негатива можно избежать продуманным дизайном и описанием.   1. Спрячьте все ненужное пользователю. Если человек чего-то не видит, то он не станет там кликать и совершать разные ошибки. Не оставляйте серых полей, потому как пользователи захотят узнать, почему этим нельзя пользоваться, даже если им оно совершенно не нужно. 2. Выделяйте цветом главный функционал и все важное для пользователя. 3. Загрузка сообщений и анимации – это НАШЕ ВСЕ. Если пользователи не видят прогресса происходящего, то им кажется, будто ничего не происходит. Тогда ждите многочисленных кликов и неожиданных поворотов, к которым код может оказаться не готов. Просто покажите пользователям крутящийся спиннер! 4. Ссылка на документацию с FAQ. В 80% случаев пользователи спрашивают об одном и том же. Перенаправление их на страницу с FAQ значительно экономит место в моем почтовом ящике. 5. Доступность – это ключ, открывающий двери ко многим потенциальным пользователям, которые смогут опробовать ваш сайт (и для них он не покажется трудным!)   Хороший дизайн позволяет пользователям научиться работе с продуктом самостоятельно. Это крайне важно, т.к. никто не хочет возиться с документацией.  Сэкономьте собственное время и избавьтесь от головной боли, дав пользователям нужные инструменты, и они полюбят ваш продукт за продуманную структуру. |

https://medium.com/swlh/want-to-be-a-better-developer-work-in-ux-33a040970bcb