buildfire

10

ACTIONABLE LESSONS

we learned from building
10.000 mobile apps

Introduction

Building apps requires skill, dedication, and experience. However, that experience comes at the price of making some very costly mistakes and then learning very important lessons.

We've built more than 10,000 mobile apps for every kind of company in every kind of industry, and after working with so many we've seen certain lessons keep coming up again and again.

Now we want to pass those lessons on to you. Every lesson that we provide here will save you time and money, but only if you learn them before you make the mistakes others made.

Let's start with the most important one.

Lesson 1:

An Idea is Not a Plan. You Need a Plan.

The first lesson we learned from the very beginning was the most important one: have a plan and stick to it. In fact, it's so important that it might as well be rule one and two.

What you should also learn is that AN IDEA IS NOT A PLAN. Lots of people have ideas for apps. Good ideas, bad ideas. Very few app ideas actually to turn into reality.

Why? Having ideas is easy. Making plans is hard.

Plans incorporate budgets and functional outlines and development milestones and promotional opportunities, tangible components that lead to complete a goal.

Ideas, well, are just ideas. They aren't useful until you put them into action.

Plus if you're working with a purpose and towards a goal, you won't be distracted by things that WILL come up during development, like meeting schedules and deadlines and conflict.

How you can learn from this:

Make a plan and stick to it. Don't be tempted to change things when they don't need to be changed.

Everything you want to build in an app needs to be mapped out during the planning phase, so when it's getting developed, there shouldn't be any unnecessary last minute changes after.

Then once you have a scope of work defined and outlined, do not change it.

Unless you absolutely have to.

Do not say "oh, but could we just add one more thing?" when your app is halfway built. Do that beforehand. Changing a prototype or a wireframe is a lot less expensive than changing code.

Otherwise, your developers, designers, project managers and everyone involved with the production of your app will hate you forever.

Aside from aggravating your technical team, there are financial reasons to consider. Every last minute minor change and tiny tweak takes time and costs money.

Also if you're racing competitors to market, and are trying to reduce prices for your consumers, constant changes will delay your app and cause you to miss your opportunity.

do not change it — otherwise your team will hate you forever

Lesson 2:

Apps Need to Do One Thing, and do it Well

Being a jack of all trades means that you're master of none. This is true in life, and is true for your app.

Whether you're building an app for internal communications or for promoting a product, or whether the app IS the product, you need to make sure that your app does what it does well.

Think of a Swiss Army Knife. It has a blade, a corkscrew, nail file, tweezers, and so on. But it trades usability for portability, and as such it doesn't do any one thing very well.

You wouldn't want to use one to cut an expensive steak, but that's ok, because it's not what it's designed to do. It CAN do it, but it doesn't do it well.

Now think of a steak knife. It is perfectly sharp, balanced, well proportioned and does its job of cutting and slicing with ease. However, it you probably wouldn't want to take it on a hiking trip.

Your app needs to take the same approach. For your app to be truly effective, focus it on what it is supposed to do, how users will use it, and then do that.

Plus, adding on functions and options to a simple app increases development time, which increases costs and decreases your profits or ROI.

How you can learn from this:

Determine the most important thing that your app is going to do, and focus your efforts on building that.

Here's a quick and easy way to determine if your app is overly complicated:

Ask yourself

> **Can I explain my app in the number of characters of a Tweet?**

(The 140 character classic Tweet, not the jumbo industrial-sized 280 ones.)

The point of this exercise is that if you can't find a way to accurately sum up your app in a short, concise, descriptive way, it's either too complicated or you need to work on your concept.

If you can communicate your app's value proposition to a stranger, while keeping it short and sweet, chances are you can keep yourself on track during development.

Lesson 3:

Speed is Everything (Execute Fast)

Building an app ALWAYS takes longer than you think it will. Always.

This is a universal truth that just about everyone knows, but it's still an important lesson we've learned after building thousands of apps, so it bears repeating.

What's worse is that the longer you take developing and building your app means less money in your pocket. That's for a couple of reasons:

- It decreases the return on your investment, whether that's because of an increased cost per user acquisition or failure to timely deliver proposed operational cost reductions.
- Opportunity cost is also a cost of your app and you have to factor that into building it. You can miss out on a chance to make money because you missed an opportunity.
- If you're paying developers or designers, the longer you employ them the more you will have to pay them.

How you can learn from this:

When you have a chance to speed up the development of your app, take it. Keep moving and keep focused.

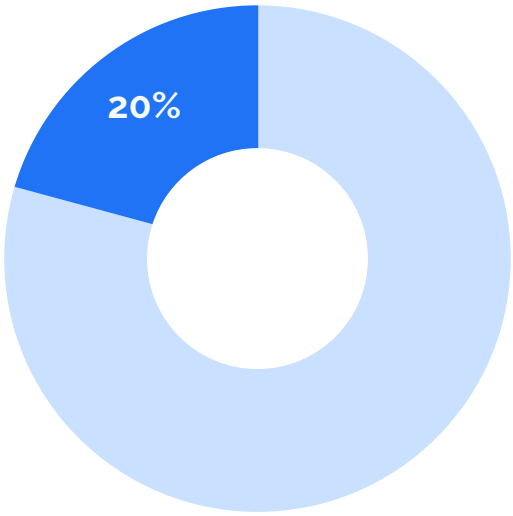
This helps you build quickly, and keep your momentum moving so you don't miss out on chances to capitalize on opportunities, and paying people you don't need to.

Learn the 80/20 principle. That means that 80% of the things you want your app to do is accomplished by only 20% of the technology.

The 20% is called the Minimum Viable Product (MVP), and it's the basis for everything that will come in the future. Define your MVP early as part of your plan.

Because there comes a time where budget will run out, time will run out, or some other resource will run out. Then you'll have to decide what to stop developing, and what to focus on.

Minimum Viable Product (MVP),



When that time comes, having an MVP will help you determine which feature or function can be sacrificed without compromising the entire project.

Without an MVP to clarify between **Nice to Have** and **Need to Have**, development will bottleneck as arguments and conflicts over what can go will halt production.

When you use this lesson, you will discover that it will speed up your production as you eliminate all the extra things in your app and remove bottlenecks that delay your launch.

BuildFire's core philosophy incorporates this 80/20 approach to rapidly accelerate app development.

After building 10k+ apps we learned almost all of them share 80% of functionality. We use our development platform to quickly and easily replicate those features at no further expense to you.

With the basics out of the way we can focus more of our energy (and your budget) on the features you really want.

Lesson 4:

KISS
(Keep It Stupid Simple)

This is a really simple lesson. About simplicity.

The best things in life are simple. Mobile apps are no different. Over-complicating and overthinking are so easy to do in app development.

Don't confuse "simple" with "easy", though. It takes time and skill to make something as complicated as the user experience on a mobile app look "simple".

Here's a super secret lesson that only the most successful app builders know:

The best mobile apps require the user to perform the fewest amount of taps in order to perform or reach the task or function they want to do.

Also, when building an app, remember your MVP, and then think of additional features that will **ENHANCE** the user experience, not complicate it.

Far, far too many people developing their app think that the more things that an app can do, the more it will appeal to users, and the more users using their app will make them more money.

However, too many features and the app becomes useless because it drifts too far from its original intent, and too complicated for someone to use effectively.

Simple procedures and operations will always create a better user experience, and happy users are engaged users. Engaged and happy users talk to their friends about your app. Make sense?

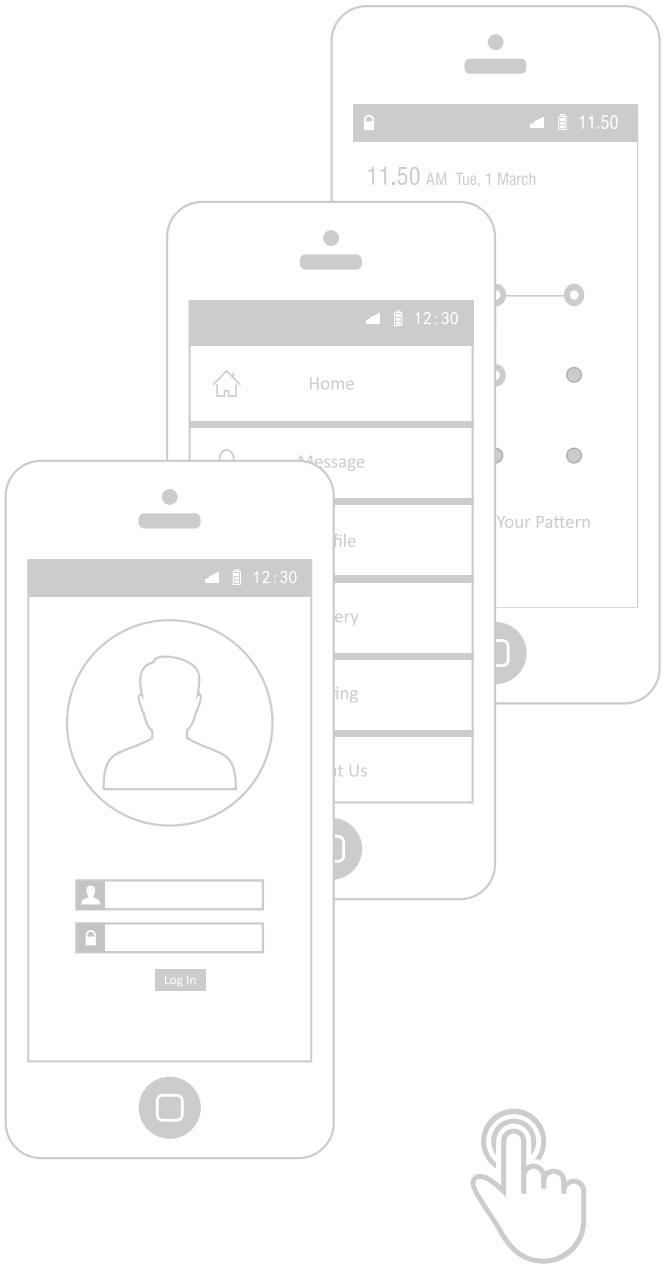
How you can learn from this:

The best way to test how easy your app is to use is to create a click-able wireframe prototype. Invision-app is a good program to use, and it's free.

Build a prototype, and let a friend use it. Don't tell them anything about how to operate it, or its features. Just give them a goal, turn it over to them, and take some notes.

If they can understand and operate it right away, you're on the right path. If not, you have some work to do.

Keep testing and keep cutting down. You'll have a great app in no time.



Lesson 5:

Know Exactly Who You Are Building Your Apps For

There are seven billion people on the planet. Half of them have smartphones. Half of those buy apps. That's almost 2 billion people.

Since people all have the same basic desires, if something appeals to everyone, then that is sure to be a hit, right? Go for the biggest amount of users, right off the bat.

Wrong, most of the time.

Far too many people want to start out with an app that does everything for everyone. Make sense, there are two billion people in the world who have smartphones, so that's a big market.

So, generally speaking, if you start out by trying to be everything for everyone, you end up being nothing for everyone.

Trying to tap into such a huge market right away is almost impossible because competition, economies of scale, and market research generally do not favor smaller developers.

Plus it's hard to build an app that works for everyone.

But there is a way you can put your size and scope to work for you in ways bigger companies and organizations cannot.

How you can learn from this:

Start small and then grow.

We learned from Kevin Kelly's 1,000 True Fans that the most effective strategy for building a mobile app is to structure it around a core group of true fans who share a desire or interest.

This number should be about 1,000 people. 1,000 people is a reasonable amount for a number of fans, and supporting them is a manageable figure for one person and app to handle.

Focusing on this relatively small number is precisely why you will succeed.

Because of their size, big companies and organizations cannot reach these groups, as the amount they'd have to spend to engage them is more than they would make from them.

That's where you come in. These groups are chronically underserved, and hungry for technology or tools that can unite them or give them the experience they crave.

These are the people that become the true fans of your app, the ones who will be hungry for your updates and the content and functions it can provide. Build an app for that niche of people.

Once that niche is satisfied, look at your app and do some analysis. Why did it work? What features were popular? Did it give a competitive advantage or appeal to a specific user base?

Careful analysis should reveal how to monetize their engagement, which you can turn into Version 2 of your app, and then expand your user base and increase your revenue, and so on.

There's an online company you might have heard of called Facebook that did exactly that.

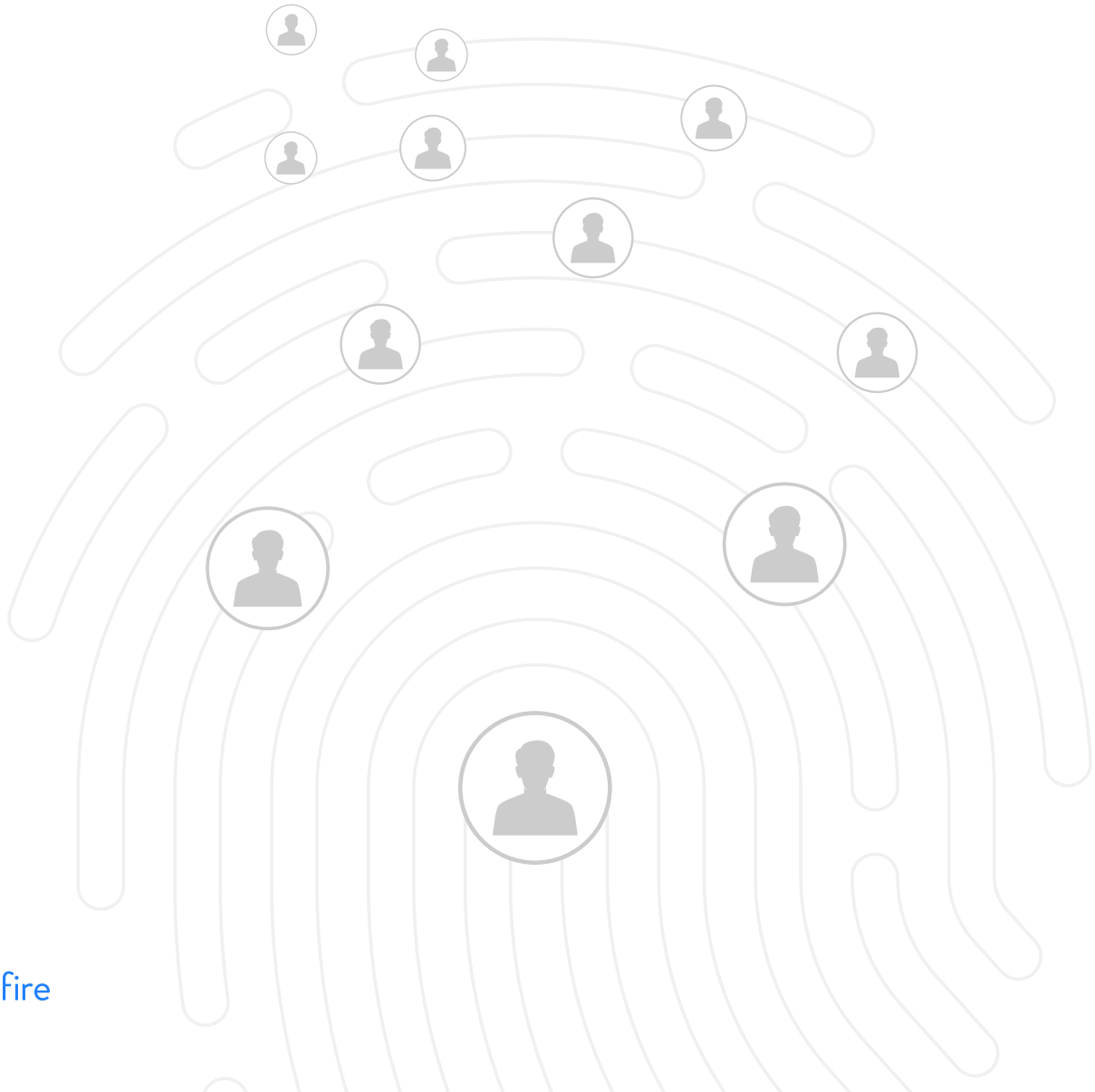
Facebook started as a social website for Harvard students. It was exclusive to students there, but these early adopters became True Fans, and saw the potential of the platform.

Once Facebook discovered its core value proposition (allowing students to easily communicate), it expanded to allow other students at other schools to access the network.

A few years later, after significant investment in infrastructure and user experience, Facebook opened its membership to everyone over 13 years old with a valid email address.

Now it's one of the biggest companies in the world with 2.07 billion active monthly users. Pretty good for a startup founded in a college dropout's dorm room.

Your app can work the same way, if you follow the 1,000 True Fans method like we did.



Lesson 6:

Eliminate the Middleman

When most people think of apps, they just think of the little program that lives on their phone, and does whatever it is it is supposed to do.

What few people consider, especially the people building them, is who is actually going to manage, operate, and update that app.

So they just build and design without a plan or focus.

This is the software equivalent of painting yourself into a corner: you're trapped, and if you need to do anything important quickly, you have to pay someone else a lot of money to do it.

Many people learn too late that you need to have a quick way to implement their app, because:

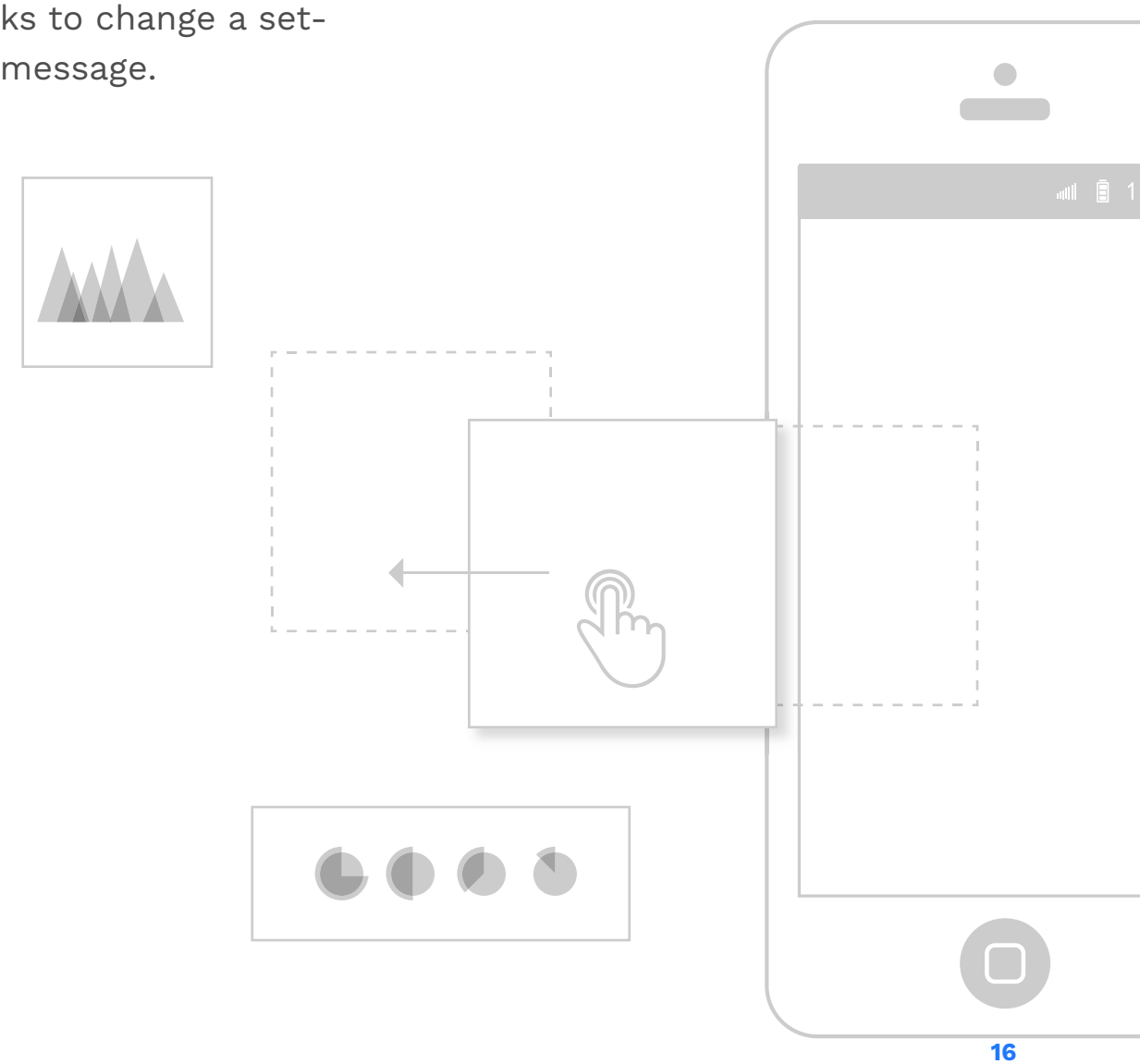
- If you have to rely on IT people or developers for quick notifications, updated content, or non-code based changes, your app will fail.
- If it takes too long to push notifications to users about a critical update or an offer it means your app is useless because it will delay timely user actions.
- If you have to go to the IT department for every minor update or thing you want to do, that reduces the efficiency of your app, and is honestly just a big hassle for everyone.
- If you have to call up a third party developer to change something, or send a push notification, or update a file, that gets really expensive really fast, and takes forever.

How you can learn from this:

The best way to eliminate the middleman is to build an app with a user-friendly and intuitive backend dashboard so non-technical people can easily make changes, updates, or messages.

Using BuildFire's Control Panel is a simple as point and click and drag and drop. Our dashboard was designed to require the fewest amount of clicks to change a setting or send a message.

This reduces the time needed to train employees to operate it, as well as the time it takes for them to implement critical updates or messages.



Lesson 7:

Timing is Everything

The mobile app world moves really, really fast. Consequently, lots of opportunities are time dependent.

There may be competing apps that could beat you to the market and capture your valuable user base and make your app obsolete.

Plus, your app might need to capitalize on opportunities tied to seasonal events, before a product launch, a new internal business initiative, or a big concert.

In addition to marketing and promotional concerns, smartphone hardware and software change so fast that smart developers structure updates around that timing and how it affects an app.

For example, in 2012 Instagram timed their release to correspond with iPhone and iOS (their primary platform) updates to ensure their app still worked with the newest camera and software.

If they had hesitated their app wouldn't have worked and competing apps might have capitalized on this to establish themselves as the dominant photo sharing app.

That's one of the reasons that Instagram is now the world's largest photo-sharing app and one of Facebook's biggest acquisitions.

How you can learn from this:

Do some research, find the right time to release your app and set a solid deadline, then work backwards from there to build your MVP based on available resources.

Also, when creating a timeline be sure to estimate the optimal return on investment.

The optimal ROI includes how much it is going to cost to meet the timing goals against what the costs of upgrading the app is going to be versus what the upgraded app is going to bring in.

For example, if your app relies on a critical piece of hardware such as GPS or camera, research what the most popular operating system your app is used on.

Once you know that, see if there are any scheduled updates to the firmware that operates those features, and test your app to see if it will still be compatible with those updates.

If not, determine if you the costs to add additional resources to ensure your app remains compatible with the new software are less than what it would take to ignore it.

But, in the long term, if you don't keep pace with the updates, chances are your competitors will, and they are probably already working to update their apps.

Plan accordingly.

Lesson 8:

Manage Expectations

Here's a hard life truth. You're not good at everything. You probably aren't great at slam dunking, or painting a picture, or rebuilding a car engine. Maybe you are, but probably not.

You might not even know you're not good at these things, so you try anyway. Then you break your arm dunking, or blow up your car driving it, or make a huge mess painting.

The same type of consequences can apply to building an app. You probably won't break your arm, but you might lose your shirt. That's if you're lucky. If you're not, you might lose your mind.

That's because a thing many people aren't good at is managing expectations and the project scope. Things like deadlines, budgets, and deliverables.

Most people seem to think these skills come naturally, rather than needing to be trained and practiced with experience. They are wrong.

What's even worse, most people aren't aware they they AREN'T good at these things, and so they set out, make a bunch of mistakes, and their project comes crashing down.

Ultimately it's nothing to be ashamed of, but if you are one of the lucky few who recognize your limitations, you are already ahead of the game when it comes to building a successful app.

How you can learn from this:

Whether you're at the start of your app building project, or halfway through, just ask yourself, and be honest:

> **"If I had to pay someone to do the quality of job that I'm doing, would I?"**

If the answer is "no", then maybe you shouldn't be the one doing it, be that design or project management or coding.

Sure you might have to because of budget or funding or operational reasons, but if you have the choice, perhaps it's best to look beyond yourself and your own capabilities.

Find someone who can help you set realistic deadlines, realistic performance expectations, a realistic budget, and your app development will be on the right track.

Even better than using a single person to plan out your timeline and resources is to consult with a team of professional app developers who understand how to build something with you.

Lesson 9:

Don't Forget About Maintenance and Infrastructure!

The biggest expenses, headaches, and time sinks in app development aren't the design, or what the buttons do, or any other obvious feature. It's the stuff behind the scenes.

It's how you set up the backend for everything and then keep it running. No one, NO ONE first starting out to build an app knows how much time and money it costs to maintain and grow one.

Directly tied to the failure of people to account for the additional costs of backend maintenance and infrastructure, is their failure to consider how their app will grow and expand.

In app development, we call this "scalability".

Scalability is critical for both the frontend and the backend. For example, the more people use your app, the more space will have to be devoted to storing and accessing and organizing them.

Those pathways have to be built to be scalable and expandable otherwise data bottlenecks will occur, which drastically detract from a user's experience and will drive them away.

Plus, we've had countless people come to us after they've built their apps themselves or with another platform and beg us desperately to untangle their technical messes. Why?

Lesson 9:
Don't Forget About Maintenance and Infrastructure!

Because they built their apps with no regard for how they will add more features, or improve their users' experience, or even how they will access it, and now need us to fix everything.

These people learn too late that the things you can't see are the things that hurt you the most.

They also learn a very expensive lesson. A little known fact of app development is simply take whatever you paid or budgeted for your app and **add 30% to it**.

That additional amount accounts for what the ongoing costs of time and money it's going to take to maintain, and operate, and improve your app.

Thirty percent is also a conservative estimate, based on a basic app. Your results may vary.

How you can learn from this:

Speak with a developer right from the beginning of your app building project. If you're not experienced in software development, you likely have no idea what you are in for.

The sooner you can settle these problems the sooner you can start developing your app, and as we've seen above, you have to develop your app quickly if you are to be successful.

BuildFire offers a cost calculator for non-technical people to estimate their app's scale and cost, and our BuildFire Geniuses are happy to consult on your custom app development needs.

Lesson 10:

Don't Recreate the Wheel

Closely tied to the mistakes that people make in setting their MVP is trying to cram too many features into an app that cannot support them.

This takes away from the focus of the app, and makes for a poor user experience. Reducing the number of things you want your app to do to improves the user experience.

On the other hand, most people fail to consider how their app's features can be reused. If it works well in one app, it might work well in others.

Think about Angry Birds.There are dozens of different versions of the same basic game concept, simply with different design elements or skins or themes.

The developers came up with a winning concept, and then replicated it a lot over instead of building something from scratch each time with more development and production costs.

Just don't overdo it. You can only go to the well so many times before it dries up.

Reusing or recycling too much too often can make your app feel stale, and also confuses users if its look and feel is too similar to previous versions.

Also, slapping a new coat of paint on an old app and trying to pass it off as new wastes your time, insults your users, and is not generally the best strategy.

How you can learn from this:

Chances are an app already exists that has some of the features you want in. If you can find a way to clone that functionality, do that so you won't have to recreate it from scratch.

Find that app and modify the user interface so that it's what you want. If there's a working model that makes it even easier, because its already been built and you can just adjust it.

It's just that easy, right?

Sadly, not in most cases.

There is a solution. Modern development platforms feature a control panel that allow you to clone a feature and plug and play it into another app easily, and do it as many times as you like.

Copying a feature is only the first thing you have to do though. Once you've found one that works, you then have to integrate it into your new app.

But, a platform that allows drag-and-drop plugins simplifies and streamlines the process lets you easily build your app and get it to market quickly.

Final Thoughts...

If you learn these lessons you will have a much smoother app building project. Some of them might seem obvious, but it helps to mention them to even the most experienced app builders.

Armed with this knowledge, you should be ready to build a great mobile app. Get started and see what you can do, and we can't wait to see what you'll build.

But if these lessons have helped you identify some of the shortcomings in your app development project, and you don't know if your tech team can fix them, BuildFire can help.

At BuildFire, we've learned these lessons and countless others in building over 10,000 apps for companies like PayPal, Wienerschnitzel, LA Philharmonic, and Ohio State University.

When people at those organizations needed to build quality apps quickly and cost effectively, they turned to our Pro-Services App Geniuses.
These dedicated consultants work with each customer on an individual level to truly understand their unique situations and goals, for a truly holistic approach to app building.

Then they apply their industry experience and technical knowledge to create a truly unique and stunning mobile app. All much quicker and better than using traditional development strategies.

Put our experience to work for you.