Mobile Backend as a Service

In this lesson, we will learn about mobile backend as a service and when to use it.

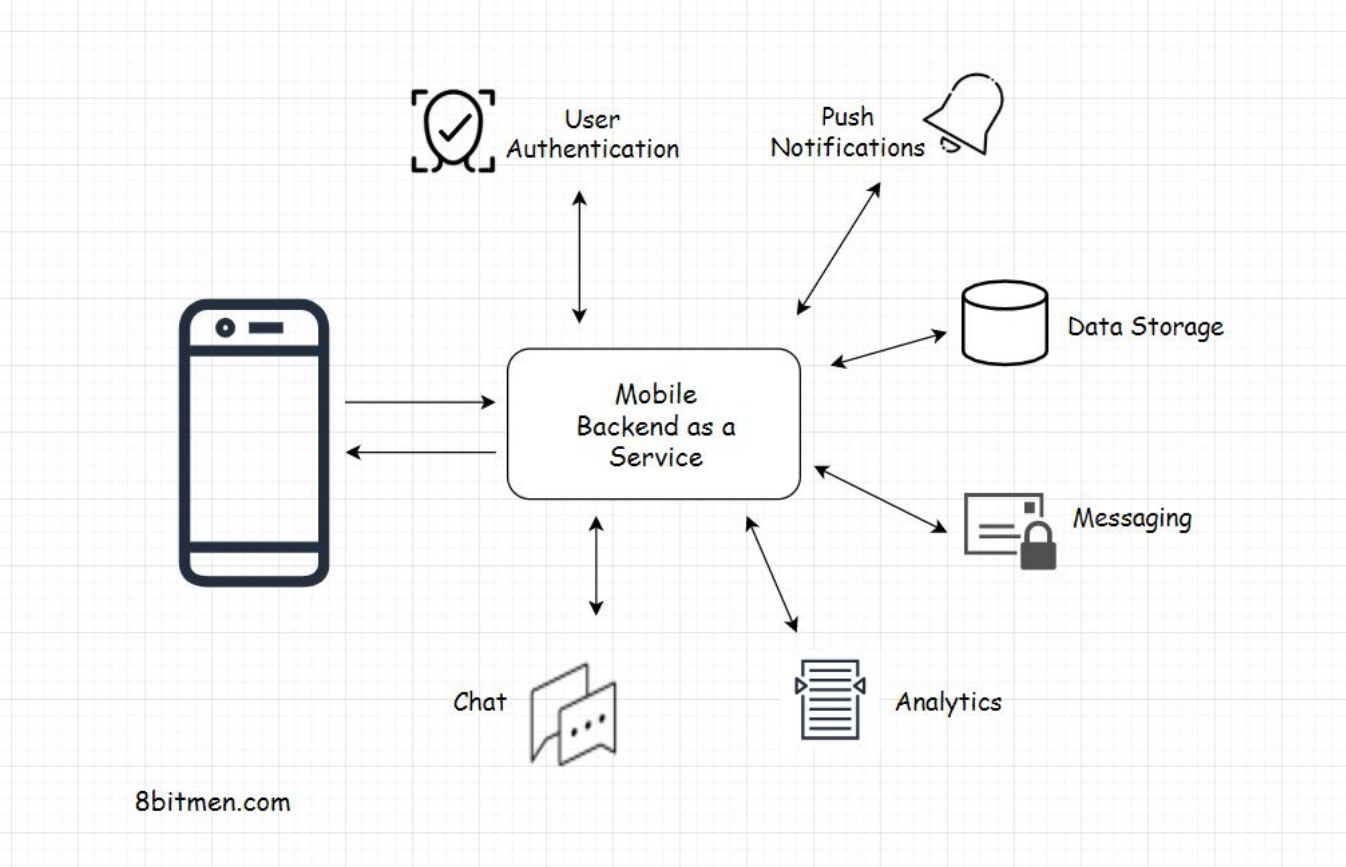
WE'LL COVER THE FOLLOWING

What Is Mobile Backend as a Service?

Mobile Backend as a Service or *MBaaS* is a cloud-based service model that takes care of the backend infrastructure of our mobile app and enables us to focus on the business logic and the user interface.

So, what are the things a MBaaS takes care of? What features does it bring along?

An online service besides the *business logic* and the *user interface* contains several other key features that collectively make the service functional, top-notch, a service worthy of getting the user attention. These features are user authentication, integration with social networks, push-notifications, real-time database, caching, data storage, messaging, chat integration, integration of third-party tools, analytics, crash reporting and so on.



A mobile backend as a service takes care of all these features making a developer's life a hell lot easier during the bootstrapping phase. Imagine writing and maintaining all these features yourself from the bare bones. I mean it's not even possible unless you have a team.

With these freemium cloud-based services, you don't have to worry much about the app hosting costs during the initial days as these services offer a generous free tier. So, if you are a solo developer, with these services, you can always bring your idea to reality & show it to the world.

Deploy your app to the cloud. Show it to the community. Have some initial customers. Get feedback. Pitch it to the potential investors without paying a dime for hosting & the infrastructure. *Well, what more can I say?*

This is the whole reason the cloud service model blew up. It provided a way for solo, indie developers to bootstrap their business, get a foothold in the market by just focussing on the idea implementation part and letting the cloud service take care of the rest.

In case you aren't much aware of the cloud. I have written a blog post about it why use the cloud? How a cloud is different than traditional computing?. This will give you an insight into it.

A MBaaS typically offers an API for every feature. There will be an API for user authentication, an API for real-time database, an API for messaging and so on. Our code can directly interact with the respective API and exchange information.

Also, since, we do not have to manage the infrastructure, a mobile backend as a service cuts down the time it takes to develop an app by notches. A few popular examples of MBaaS are Google Firebase, AWS Amplify, Parse.

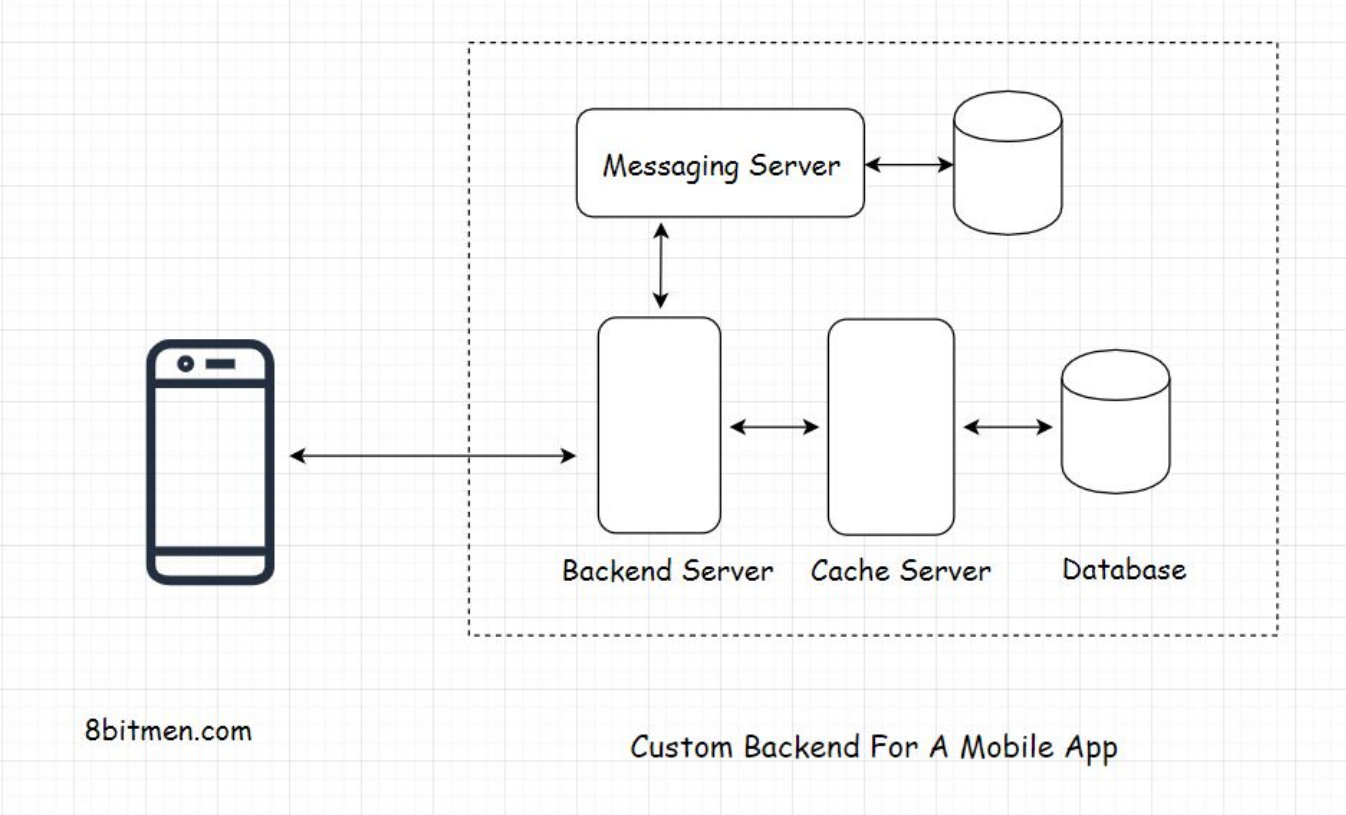
Parse was the early leader in this space but was shut down by Facebook.

When Should You Use A Mobile Backend as a Service?

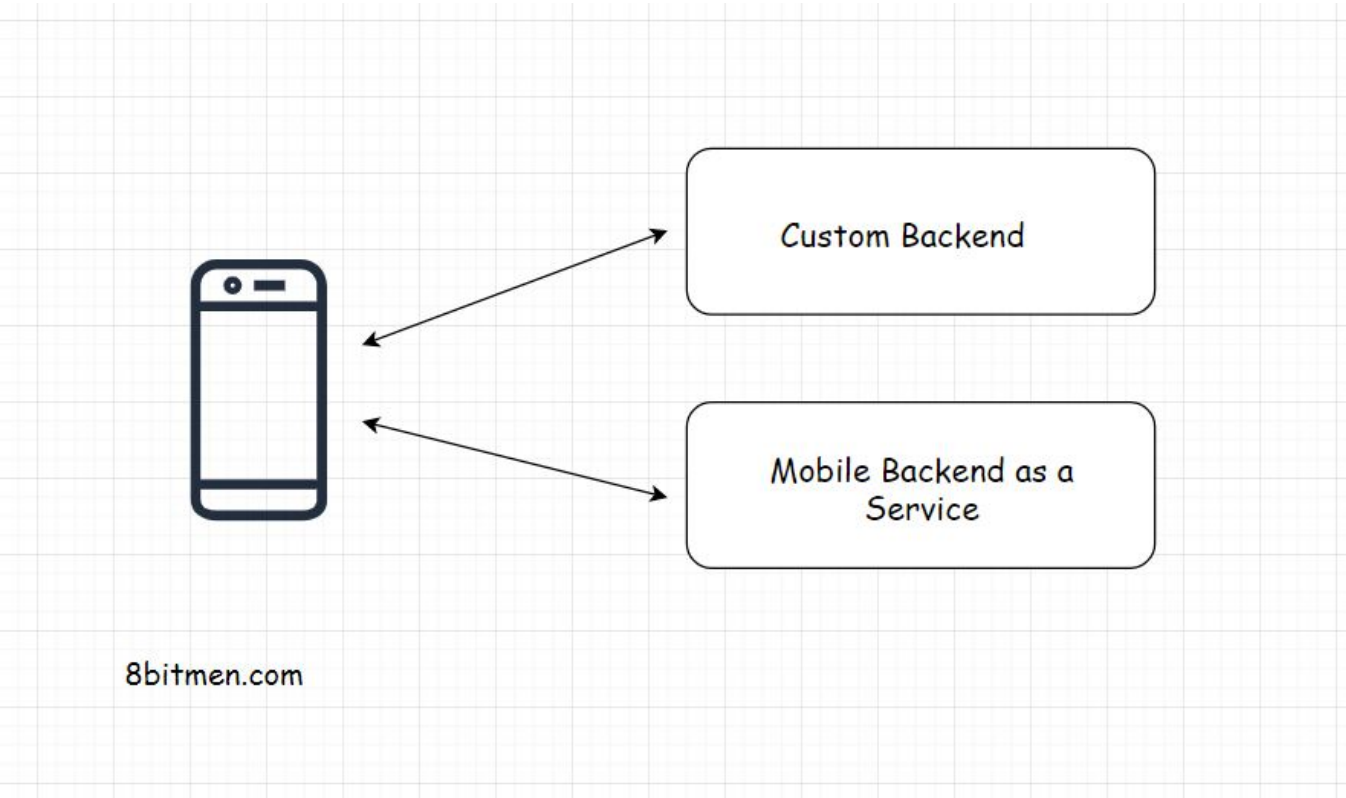
MBaaS is great for *mobile-only* services, great for use cases where you do not need or you don't already have a custom backend, up and running for your service. In case of a MBaaS, all the business logic resides on the client which is the mobile app. So, the app is a *Fat client*.

MBaaS is best for apps like mobile games, messaging apps, to-do list kind of apps. When using a MBaaS, there are a few things that I would want you to keep in mind. Since we don't have much control over the backend we always have to keep the business logic on the client. If we ever need to add a new feature that would require the business logic on the server. We will have to design a custom backend from the bare bones.

On the flipside, if we start with a custom backend and then write a mobile client which is the conventional way. You can always customize the design of your service, introduce new clients and stuff, just with an introduction of dedicated APIs for respective clients.



We can also use MBaaS & a custom backend setup in the same app in scenarios where we are required to integrate a legacy enterprise system with our mobile app or if we need to leverage some additional features that the custom backend server hosts. Think of a banking app built using a MBaaS that needs to interact with the legacy enterprise backend to cross verify the data entered by the user every time.



Also, not having much control over the backend, makes this kind of a *vendor*

lock-in situation. Just like parse.com what if the service provider decides to

close his shop. Or he stops upgrading his service, which may result in severe security flaws or he stops adding new features to his service or you in future disapprove of his updated billing rules. *What are you gonna do next? Keep that in mind.*