Log In Join

Back To Module Home

Architecture of Scalable Applications

0% completed

Search Module

Mark Module as Completed

What is Monolithic Architecture?

In this lesson, we will discuss monolithic architecture.

We'll cover the following

- What is monolithic architecture?

What is monolithic architecture?#

> An application has a *monolithic architecture* if it contains the entire application code in a single codebase.

A *monolithic application* is a self-contained, tightly coupled software application. This is unlike the *microservices* architecture, where every distinct feature of an application may have one or more dedicated microservices powering it.

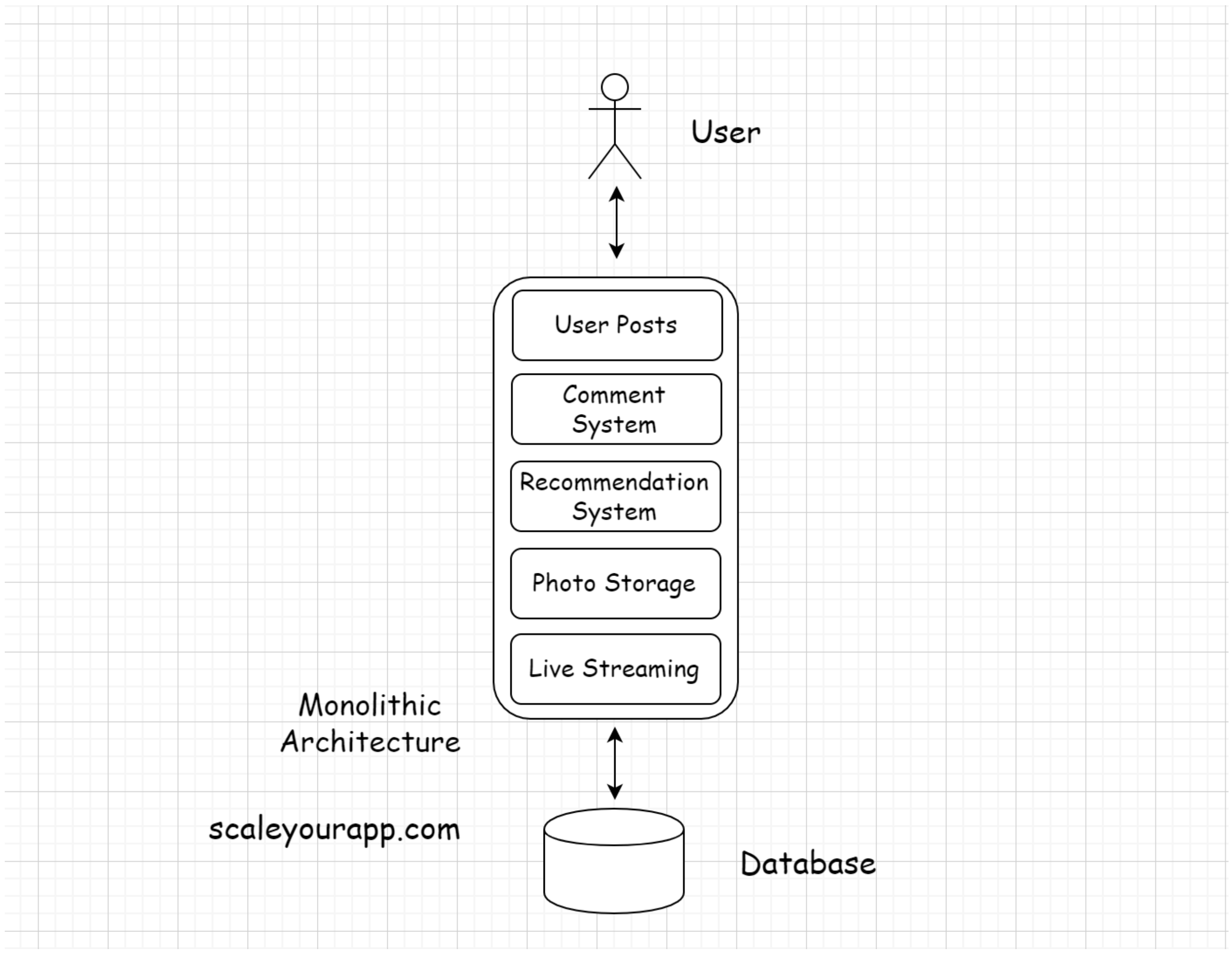
Taking example of a social networking site like *Facebook*. The application contains various features such as:

- User posts
- Comment system

- Groups
- Marketplace
- Portal Ads
- Photo storage
- Live streaming
- Recommendation system for recommending the latest and contextual content on the platform to the users and so on.

In a *monolithic architecture*, all the modules will be coded in a single codebase tightly coupled with each other as opposed to having one or more dedicated *microservice* for running respective features.

The diagram below represents a monolithic architecture.



Monolithic apps are simple to build, test, and deploy in comparison to a microservices architecture.

Often during the initial stages of a business, teams choose to move forward with a monolithic architecture, intending to branch out into a distributed microservices architecture later.

Well, this decision is a trade-off. We need to bear in mind refactoring and re-writing code has significant costs associated. Dismantling features from a tightly coupled architecture and re-implementing them into separate microservices demands a lot of time and resources.

There have been instances in the past where the dev teams decided to start with a monolithic architecture and later scaled out to a distributed microservices architecture.

This is what *LinkedIn* did. Though I would like to state here that *LinkedIn* started at a time when *cloud computing* and *microservices architecture* weren't the norm.

In the present computing landscape, applications are built and deployed on the cloud. Also, businesses have to move fast. A wise decision is to pick the loosely coupled stateless microservices architecture from the start if we have multiple distinct features in our application and expect things to grow at a rapid pace in the future.

On the flip side, if our requirements are simple, monolithic architecture would suit best. Implementing a microservices architecture in this use case would be overkill. After all, managing numerous modules running in conjunction in a distributed environment isn't a walk in the park.

In the next lesson, let's go through some of the pros and cons of monolithic architecture.

Different Tiers in Software Architectur…

Next

When should you pick a Monolithic Ar…
