Back To Module Home

Architecture of Scalable Applications

0% completed

Different Tiers in Software Architecture

Monolith and Microservices

Monolith and Microservices Quiz

Conclusion

Mark Module as Completed

When should you pick a Monolithic Architecture?

In this lesson, you will learn about the pros and cons of monolithic architecture and when to choose it for your project.

We'll cover the following

Pros of monolithic architecture#

Simplicity#

Monolithic applications are simple to develop, test, deploy, monitor and manage since everything resides in one repository.

Things are relatively simple when dealing with one repository averting the complexity associated when handling and monitoring multiple components deployed separately.

Cons of monolithic architecture#

Continuous deployment#

Continuous deployment is a pain in monolithic applications as even a minor code change in a certain application layer or a feature necessitates a re-deployment of the entire application.

Regression testing#

The downside of re-deployment of the entire application is that we need to perform a thorough *regression testing* of the whole application after the deployment is done. A minor code change in one feature can potentially impact the functionality of other features significantly since all the features are tightly coupled with each other.

Single points of failure#

Monolithic applications have a single point of failure. A bug in any of the application features can bring down the entire application.

Scalability issues#

Maintenance and *scalability* are a challenge in monolith apps as all the components are so tightly coupled with each other. As the code size increases, things get trickier to

manage.

Cannot leverage heterogeneous technologies#

Building complex applications with a monolithic architecture is tricky since multiple technologies and programming languages need to be leveraged to implement the respective features of an application.

Using multiple programming languages in a single codebase becomes a mess also often times not possible. Heterogeneous technologies have compatibility issues and microservices architecture suits best to leverage them.

It is tricky to use *Java* and *NodeJS* together in a single codebase, and when I say tricky, I am being optimistic. I am not sure if it is even possible.

Not cloud-ready, hold state#

Generally, monolithic applications are not cloud-ready as they may hold state in the static variables. Though not all monolith apps are designed that way, it's a general observation that legacy apps use static variables to quite an extent. An application to be cloud-native, to have a consistent behavior on the cloud, has to be stateless.

When should you pick a monolithic architecture?#

Monolithic applications fit best for use cases where the app requirements are pretty simple; when the application is not that complex. Examples of this are a to-do list app, a sports news app, an organization's internal tax calculation app, a similar open public tool, etc.

These are the use cases where the business is certain that the application will have limited features and complexity and will not require any serious expansion in the near

future.

In the next lesson, we will learn about microservice architecture.

What is Monolithic Architecture?

Next

What is Microservice Architecture?

Mark as Completed

Report an Issue