Decoding Serverless

An essential guide to determine software project vs serverless fit

Catalyst
by Zoho

Serverless technologies are eating the stack. They're changing application development and deployment, making it easy for developers to take their ideas to market fast and at a lower cost. This combination of agility and low costs leads to experimentation, leading to rapid innovation. With serverless technologies offering robust cloud-based IDE, SDK, and frameworks, developers have robust tools and frameworks at their disposal to address specific needs in development, testing, debugging, and deployment. Developers can gain immensely from using serverless technologies for app development, greatly enhancing their productivity.

So, what is serverless computing?

Serverless computing is a disruptive cloud-native paradigm that eliminates the need for developers to manage infrastructure by providing back-end or full-stack services on-demand for developers to build and deploy cutting-edge serverless applications at a fraction of the cost, time required to provision and deploy on applications on servers. In the process, the provisioning, deployment and management of servers is abstracted from the developers and is handled by the cloud provider. Serverless is eating the stack with benefits like faster time to market, ease of implementation, massive scalability and reduced engineering lead time and operational cost.

LEARN MORE ABOUT SERVERLESS

Are there servers in serverless computing?

The term "Serverless Computing" is a misnomer. Servers are still used by cloud service providers to execute code for developers. However, going serverless allows developers to build and run applications and services without thinking about servers or managing them, or interacting with them in any way.

Developer Advantage with Serverless

From enhanced scalability to greater flexibility, reduced liability, innovation, and much more, serverless architectures offer the developer a wide range of benefits, all at a reduced cost and a quicker time to market. In this section, let us consider some of the benefits:

Fosters innovation

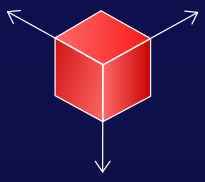
Applications hosted in a serverless environments are usually broken up into smaller functions and the abstraction adds many constraints. As developers are forced to face the challenges imposed by such constraints, they are expected to change their development process. With time, all these challenges foster innovation as developers adopt a wide range of strategies to deliver functional apps. When the serverless abstractions are used along with modern technologies like machine learning services for text analysis or image recognition, innovationation is accelerated. The abstraction provided by serverless along with the constraints leads to rapid innovation, meeting the business needs of today's globalized economy

Advanced coding

Higher-level programming abstractions offered to developers by some serverless frameworks can advance their coding skills. From function packaging to advanced sequencing and composition, you take your coding skills to the next level by exploiting these programming abstractions to construct highly complex serverless apps easily.

Seamless Upgrades

Apps developed for the serverless environment are collections of functions instead of single monolithic stacks. This means that developers have the option of uploading all their code at once or a single function at a time without having to manage servers and the stacks on top of the operating system. With this advantage, developers can update, patch, or add new features by changing one function at a time instead of reworking the entire app. This accelerates applications updates and upgrades and makes it easy to upgrade without any downtime.



Enhanced scalability

Serverless architectures are highly scalable, which means that developers can code without worrying about growth or the number of concurrent requests. During coding, developers can exploit seamless scaling features of such services by taking advantage of stateless application architectures. This allows for seamless scaling without having to wait for servers to be provisioned.

Greater flexibility

With tools like browser-based IDEs, SDK, and API access, developers get more flexibility in their application development and deployment. With Microservices architecture becoming the norm, serverless technologies are helping developers build applications using modular architectures.

Event driven apps

Since Functions as a Service model uses Action and Trigger as the two major primitives, newer programming paradigms that invoke Actions asynchronously, directly via REST API, is available for developers. Developers can benefit from executing event requests using a variety of sources to trigger parallel invocations or sequential invocations. This opens up new vistas for developers, helping them innovate much rapidly than in the past.

Zero system administration

With zero system administration, developers using Serverless technologies are free to adopt creative and novel coding strategies to expand their apps without worrying about server capacity constraints or the operational burden that comes with managing the underlying infrastructure stack.

With the developer advantage offered by serverless technologies, it is easy to see why serverless computing is growing rapidly in popularity.

Key considerations

Even though serverless technologies offer many benefits to developers, they also put some limits on the type of applications that can be deployed using these technologies. Developers should be mindful of this limitation while planning their deployment stack,

As more and more developers embrace serverless technologies, there's some confusion about picking the right platform for their needs. Ever since AWS Lambda became a success among developers, there are many other services that are made available in the market, including Azure Functions, Google Cloud Functions, Zoho Catalyst and Each service offers a different set of features and it is critical for the developers to understand what to look in a serverless computing service. In this post, we will talk about the top considerations for developers as they pick their deployment platform. Keep in mind that everyone's needs are unique, and there is no straightforward template which developers can use to select their platform

Key Considerations for Developers

Programming Language Support

Of course, the first consideration is the support for the programming language the developer wants to use. Not all Functions as a Service (FaaS) providers support all programming languages out of box. The programming language version is another factor that needs consideration. While the support for programming languages and different versions may be limited, some FaaS providers offer custom options where developers can upload the programming language of their choice using containers. Check if the FaaS provider support the programming language and version needed for your application, either out of the box or through containers

Payloads

When developers deploy their code for execution by FaaS, different cloud providers offer different ways to package the code. Some support deploying the code directly from Github or using a Zip file, few cloud providers allow deployment of binary code. Some also offer support for uploading Docker containers. Developers use different ways to package their applications for deployment and, in some cases, the applications demand specific kinds of payload support (eg: support for binary or containers). It's important to evaluate the payload support before picking a FaaS provider

Event Sources

FaaS, by its very definition, is invoked either by a trigger or at specific times, like a cron job. It is important to ensure that the FaaS provider supports the event source your application needs. This is as important as picking a provider based on programming language support

Data Stores

Depending on your application needs, you need to pick a cloud provider that offers support for the data store of your choice. It may be a relational database or object storage, so make sure it is available with the FaaS or integrable with the FaaS provider and meets your needs. Some FaaS providers support databases out of the box and, in others, there is an operational overhead. Depending on the application needs and the expertise of the developers, it is critical to pick a FaaS offering with right developer experience. Any friction in the developer experience will slow down application deployment, impacting the speed

Supplementary Cloud Services

For a functioning application, there are other dependencies like Identity and Authentication Management, Message Bus, Push and email notifications,etc.. Developers should evaluate if the FaaS platform offer integrations with various cloud services needed to meet these dependency needs

Documentation

Any good cloud service should provide comprehensive documentation about their platform and the APIs along with How-tos and other examples. Without a good documentation, developers waste their time finding the relevant information considerably slowing down the deployment process. Even though most people consider good documentation to be an afterthought, it should be part of any evaluation process

Browser based IDE and Workflow

Even though this is not as critical as some of the above mentioned considerations, support for browser based IDE and a visual workflow editor is a good to have feature. Even though developers have their own choice of IDE, a good visual editor removes the friction in their development process and accelerates deployment.

Ease of Adoption

While setting up an entire cloud environment for an end-to-end application, it's a good idea to check for the availability of polyglot stacks with more than one programming language and support for third-party libraries. This way you can harness the benefits of severless with simple serverless platforms that uncomplicate serverless for you.

CI/CD Support

With technology companies increasingly gearing towards CI/CD, it's critical to ensure that the serverless platform inherently supports CI/CD by providing decoupled environments for local debugging, staging and production.

These are some of the key aspects which every developer should consider as they evaluate different FaaS providers. The needs of every developer and the application are different, and this article only highlights some of the key considerations for developer

Tutorials

Expand your serverless computing skills with these tutorials:

- **Get started with Catalyst:** Start development right away from your local machine with **Catalyst SDKs** and **CLI toolkits**. See for yourself how easy it is to get the Catalyst CLI and environment up and running in under 5 minutes.

- **Build your first serverless web app:** Try this 10-minute tutorial to build your first serverless application using node.js functions, databases, and other Catalyst services

- **Build a lead milestone tracking microservice on Catalyst:** Try this simple tutorial to build a simple microservice using Catalyst Basic I/O Functions and integrate it with Zoho CRM to send automated counter-based email alerts when more than a specific number of leads are created.

Find more tutorials here.

Next step

Explore a free, full-featured sandbox and build your first serverless solution on Catalyst.

Get started for free

- Pay nothing till you deploy the project to production
- Get up to 125 million free invocations*
- Get a free, full-featured sandbox

*Free for the first year upon deploying your first project to production and considering single credit operations like Select Query in Data Store and heavy usage pricing

Resources


Getting Started


Component Help


CLI Documentation


SDK Documentation


API Documentation


Deployment and Billing


Tutorials


Release Notes


FAQ

Get free consultation

To discuss your ideas or explore how going serverless can boost your development process, consult with us for free.

To request a free consultation,
write to us at: **solutions@zohocatalyst.com**

Catalyst
by Zoho

The simplest serverless platform

Catalyst is a full-stack serverless developer platform by Zoho. At Zoho we've launched 50+ apps that serve more than 70 million users. To accomplish this, we've built the entire stack in house, from our own data centers and servers to high level services like authentication, machine learning, and notifications. With Catalyst, you get access to the expertise we've honed over the years in handling cloud infrastructure.