Load Balancing, AutoScaling and Serverless Computing

Load Balancing

- Cloud load balancing is the process of distributing workloads and computing resources in a cloud computing environment.
- Load Balancing allows enterprises to manage application or workload demands by allocating resources among multiple computers, networks or servers.
- Cloud load balancing involves hosting the distribution of workload traffic and demands that reside over the Internet.
- Load balancing helps enterprises achieve high performance levels for potentially lower costs than traditional on-premises load balancing technology.

Load Balancing

- Many cloud providers offer cloud load balancing technologies, including Amazon Web Services (AWS), Google, Microsoft Azure.
- AWS offers Elastic Load Balancing, which distributes workloads and traffic among EC2 instances.
- Google Cloud Platform offers load balancing for its infrastructure as a service, Google Compute Engine, which distributes network traffic between VM instances.
- Microsoft Azure's Traffic Manager distributes traffic for its cloud services across multiple data centers.

AutoScaling

- Auto scaling, is a method used in cloud computing, whereby the amount of computational resources in a server farm, typically measured in terms of the number of active servers, scales automatically based on the load on the farm.
- Autoscaling is useful whenever your site/application needs additional server resources to satisfy the number of page requests or processing jobs.
- You can now design a scalable architecture that will automatically scale-up or scale-down to meet your needs over the lifetime of your setup regardless of how fast/slow or big/small your site grows over that time.

Serverless Computing

- Serverless computing is a cloud computing execution model in which the cloud provider dynamically manages the allocation of machine resources.
- Serverless computing executes code that developers write using only the precise amount of compute resources needed to complete the task.
- When a pre-defined event occurs that triggers that code, the serverless platform executes the task. The end user doesn't need to tell the serverless provider how many times these events or functions will occur.

Serverless Computing – AWS Lambda

- AWS is regarded as the groundbreaking innovator that created the concept of serverless and has integrated it completely as part of its cloud portfolio.
- Lambda has native support for code written in JavaScript, Python, Java (Java 8 compatible), and C#, and allows the creation of a wrapper that can be added to Go, PHP, or Ruby projects, thus allowing execution of the code when triggered.
- Lambda is positioned at the heart of AWS's offering, acting in some ways as a gateway to almost every other cloud service it provides.
- Integration with S3 and Kinesis allows log analysis and on-the-fly filtering or image processing and backup—triggered by activities in these AWS services. The DynamoDB integration adds another layer of triggers for operations performed outside the real-time echo system.

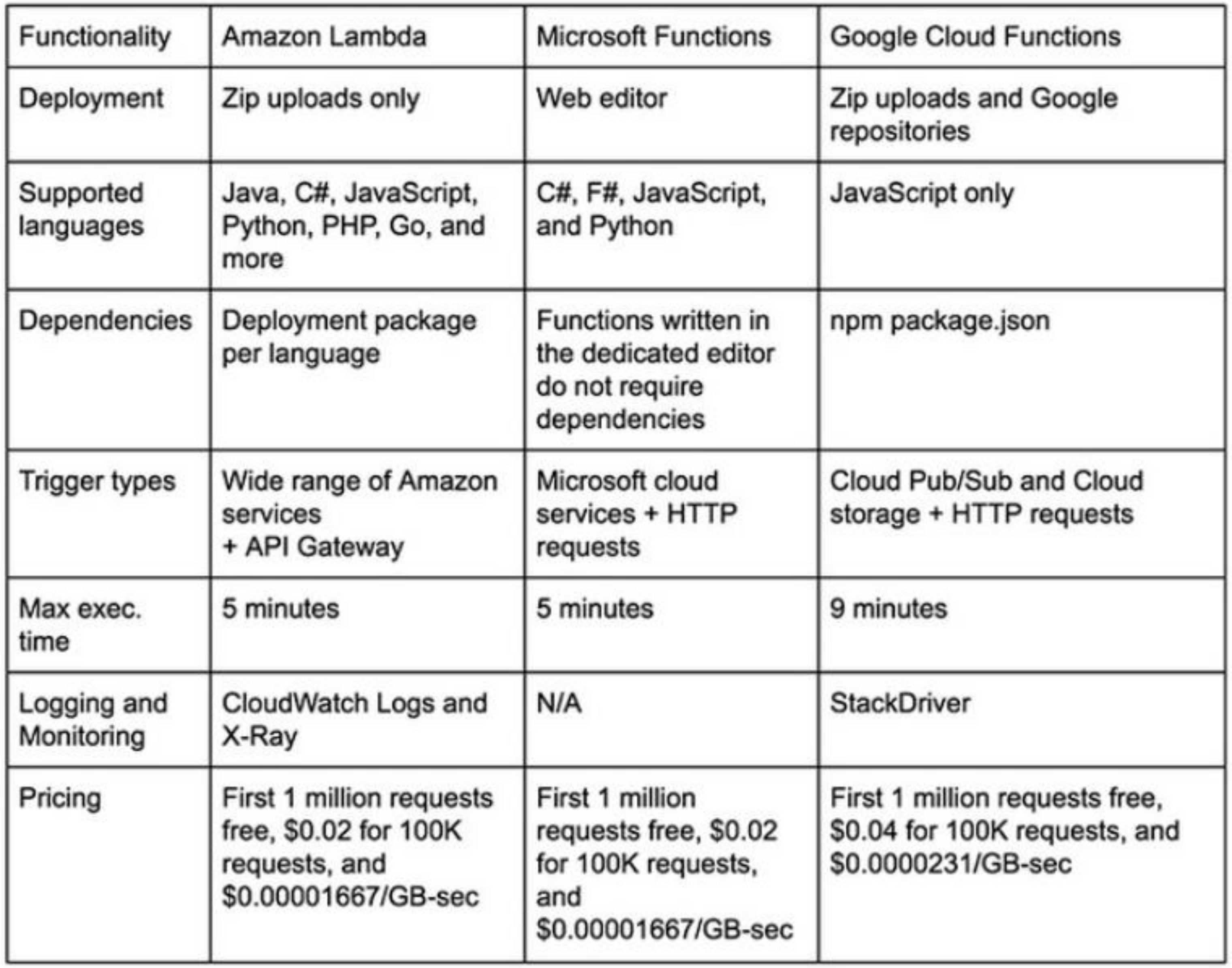
Serverless Computing – Azure Functions

- Azure Functions, Microsoft’s counterpart of Lambda, was created and introduced only at the end of March 2016.
- Microsoft allows functions to be coded in its native languages—C# and F#—inside the web functions editor, or they can be written and uploaded using common script-based options—Bash, Batch, and PowerShell. Developers are also able to write functions in JavaScript or Python.
- Azure has out-of-the-box integrations with VS Team Services, GitHub, and Bitbucket, allowing easy setup of a continuous integration process and the deploying of code in the cloud.
- Azure Functions supports several types of event triggers. Cron jobs enable timer-based events for scheduled tasks, while events on Microsoft’s SaaS services, for example, OneDrive or SharePoint, can be configured to trigger operations in Functions.

Serverless Computing – Google Cloud Functions

- Google was the last to join the serverless scene. Its current support is very limited, allowing functions to be written only in JavaScript, and triggering events solely on Google's internal event bus—Cloud Pub/Subtopics. HTTP triggers are supported as well, as mobile events from Firebase.
- Monitoring is enabled via the Stackdriver logging tool, which is very handy and easy to use, but does not supply all the information and metrics serverless users require.
- On top of all the missing functionality and limited capabilities, Google charges the highest price for function triggering. After using the one million free requests, its prices are double those of the other providers—$0.04 for 100,000 invocations, plus $0.04 for 100,000 milliseconds.

AWS Lambda vs. Azure Functions vs. Google Functions



Functionality	Amazon Lambda	Microsoft Functions	Google Cloud Functions
Deployment	Zip uploads only	Web editor	Zip uploads and Google repositories
Supported languages	Java, C#, JavaScript, Python, PHP, Go, and more	C#, F#, JavaScript, and Python	JavaScript only
Dependencies	Deployment package per language	Functions written in the dedicated editor do not require dependencies	npm package.json
Trigger types	Wide range of Amazon services + API Gateway	Microsoft cloud services + HTTP requests	Cloud Pub/Sub and Cloud storage + HTTP requests
Max exec. time	5 minutes	5 minutes	9 minutes
Logging and Monitoring	CloudWatch Logs and X-Ray	N/A	StackDriver
Pricing	First 1 million requests free, $0.02 for 100K requests, and $0.00001667/GB-sec	First 1 million requests free, $0.02 for 100K requests, and $0.00001667/GB-sec	First 1 million requests free, $0.04 for 100K requests, and $0.0000231/GB-sec