

Università degli studi di Verona

Tesina su FaaS (Function-as-a-Service)

VR443470 - Valentini Andrea

giugno 2023

Indice

1	Introduzione	3
1.1	Che cos'è FaaS	3
1.2	FaaS e serverless	4
2	Aziende che offrono un servizio FaaS	5
2.1	IBM: cloud functions	5
2.1.1	Panoramica	5
2.1.2	Caso studio: GreenQ	6
2.2	Amazon: AWS Lambda	7
2.2.1	Panoramica	7
2.2.2	Caso studio: Coca-Cola	7
2.3	Google: cloud functions	9
2.3.1	Panoramica	9
2.3.2	Caso studio: Commerzbank	9
2.4	Microsoft: Azure functions	11
2.4.1	Panoramica	11
2.4.2	Caso studio: Fujitsu	12
2.5	Oracle: OCI	13
2.5.1	Panoramica	13
2.5.2	Caso studio	14
3	Esempio di applicazione: Azure Functions	15
3.1	Introduzione: costruire API serverless con Azure Functions	15
3.2	Creazione dell'environment	16
3.3	Creazione delle API	17

1 Introduzione

Con l'avanzare della tecnologia e del *cloud computing*, è aumentata sempre di più la richiesta di servizi online che consentissero di utilizzare calcolatori già pronti e con grandi disponibilità di calcolo.

La crescita del *cloud computing* è stata esponenziale nell'ultimo decennio, soprattutto anche grazie, purtroppo, alla pandemia del COVID-19. Tant'è che il CEO di Microsoft, Satya Nadella, disse:

“We’ve seen two years of digital transformation in two months.”

1.1 Che cos'è FaaS

Function-as-a-Service (FaaS) è una tipologia (*event-driven*) di servizio *cloud computing* che consente ai programmatori di sviluppare, eseguire e gestire pacchetti di applicazioni come se fossero funzioni, senza preoccuparsi della manutenzione di una propria infrastruttura.

Tipicamente, l'*hosting* di un'applicazione software su Internet richiede: la gestione di un server virtuale o fisico e la gestione di un sistema operativo. Con FaaS, viene tutto gestito in automatico dal *cloud service provider*.



I vantaggi di questa tecnologia sono molteplici e verranno spiegati più avanti. Per esempio, i programmatori possono concentrarsi solamente sul codice delle loro applicazioni.

1.2 FaaS e serverless

Serverless è un modello di sviluppo ed esecuzione di applicazioni di *cloud computing*, il quale consente ai programmatori di costruire ed eseguire il codice dell'applicazione senza preoccuparsi dei server o del *backend* dell'infrastruttura.

Nonostante spesso le persone scambino i modelli serverless e FaaS tra di loro, la verità è che sono due concetti diversi ed è più corretto dire che FaaS è un sottoinsieme dei modelli serverless.

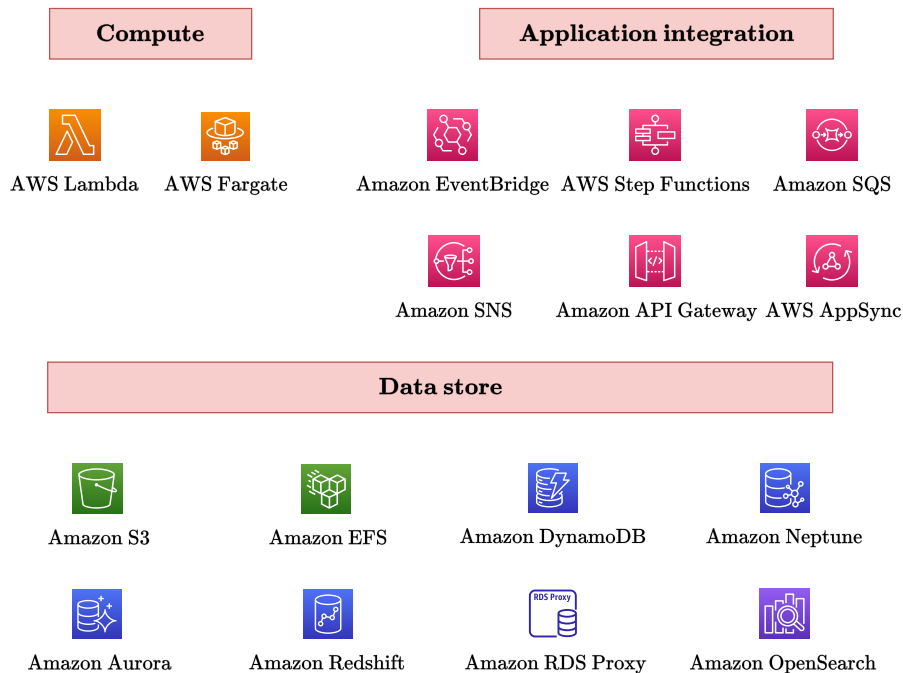


Figura 1: Esempio dei vari servizi *serverless* offerti da Amazon.

Il modello *serverless* è focalizzato su **qualsiasi** categoria di servizio, come la computazione, l'archiviazione, i database, la messaggistica, le *api gateways*¹, etc. In tutte queste categorie, la configurazione, la gestione e il costo effettivo dei server, è nascosto all'utente finale.

La figura 1 dovrebbe dimostrare come il servizio FaaS sia solo un sottoinsieme dell'enorme modello *serverless*. Amazon, come verrà anche approfondito più avanti, come servizio FaaS propone AWS Lambda.

Il modello Function-as-a-Service può essere considerato la tecnologia più centrale nell'architettura *serverless*. In altre parole è il cuore pulsante. FaaS è focalizzato sul paradigma *event-driven* (orientato agli eventi), dove il codice dell'applicazione, o del *container*, viene eseguito solo in risposta a determinati eventi o richieste.

¹Un *api gateway* è uno strumento di gestione delle API che si trova tra un client e una raccolta di servizi back-end.

2 Aziende che offrono un servizio FaaS

2.1 IBM: cloud functions

2.1.1 Panoramica

L'azienda IBM offre come servizio FaaS [IBM Cloud Functions](#) e si basa su Apache OpenWhisk².

Cloud Functions è diverso dalle tecnologie di calcolo tradizionali; si paga solo per il tempo per cui il codice sta soddisfacendo le richieste, arrotondato al 100ms più vicino. Questo significa che si potrebbero vedere dei notevoli risparmi rispetto ad altre tecnologie come le macchine virtuali e i contenitori, che probabilmente non sono utilizzati al 100% pur continuando a utilizzare memoria sul sistema del provider cloud.

Cloud Functions viene fatturato al secondo per gigabyte di memoria. Questo significa che è possibile ulteriormente ridurre i costi allocando solo la quantità di memoria necessaria perché le funzioni svolgano il loro lavoro.

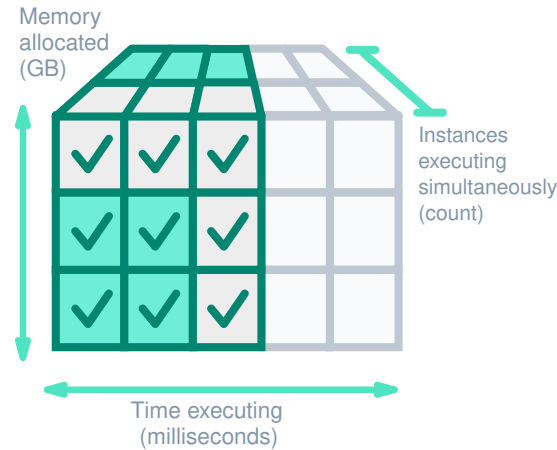


Figura 2: Funzionamento del calcolo del costo di IBM Cloud Functions.

Tempo exec. azione	Memoria dell'azione	Esecuzioni al mese	Costo al mese
500ms	128MB	5.000.000	Gratuito
500ms	256MB	5.000.000	\$4,00
500ms	512MB	5.000.000	\$15,10
1.000ms	128MB	10.000.000	\$15,10
1.000ms	256MB	10.000.000	\$37,31
1.000ms	512MB	10.000.000	\$81,72

Tabella 1: Stima di costi a seconda dell'utilizzo

²Apache OpenWhisk è una piattaforma open source che consente di eseguire funzioni in risposta ad eventi.

2.1.2 Caso studio: GreenQ

Tra tutti i casi studio che propone l'azienda IBM sul proprio sito, ne esistono 3 che riguardano singolarmente la tecnologia FaaS: GreenQ, Articoolo e SiteSpirit. Si approfondisce l'azienda GreenQ.³

La GreenQ Ltd. è un'azienda americana (Santa Monica, California) che ha l'obiettivo di portare la tecnologia IoT (Internet of Things) al servizio dei municipi in cui gestiscono i rifiuti delle città americane.



La sfida da superare. I municipi spendono cifre significative per raccogliere i rifiuti, ma spesso potrebbero essere ammortizzati eseguendo un'ottimizzazione dei processi.

L'azienda GreenQ sfrutta tale gap per introdurre una piattaforma IoT che consenta di salvare dati, come il tempo di ritiro, la posizione e il peso del cestino, grazie a schede hardware sui furgoni dell'immondizia. Questi dispositivi inviano i dati all'azienda GreenQ per monitorarli e analizzarli così da migliorare l'organizzazione della gestione del ritiro dei rifiuti. Tuttavia, dopo il lancio di un prototipo, l'azienda ha iniziato a ripensare alla sua architettura e infrastruttura.

Edy Candel, il CTO (*Chief Technology Officer*) e cofondatore di GreenQ, disse: “Dopo aver assunto i primi clienti, ci rendemmo conto che stavamo spendendo tanto tempo nel gestire le nostre virtual machines e aggiungere potenza computazionale. Vedemmo che la nostra infrastruttura aveva un numero di richieste che dipendevano dal numero di clienti e camion che stavano lavorando, quindi ci rendemmo conto che era difficile ottenere la scalabilità desiderata”.

La trasformazione. GreenQ entra a far parte del programma “IBM Alpha Zone Accelerator” creato appositamente per le *startup*. Così facendo, viene eseguita una migrazione dalla vecchia architettura alla FaaS di IBM, ovvero la IBM Cloud Functions.

Dato il paradigma event-driven, ad ogni pezzo dell'applicazione di GreenQ viene assegnato un trigger a seconda dell'azione scatenata o dell'insieme di azioni scatenate, eliminando in questo modo l'attivazione manuale nel *workflow*. Per esempio, quando il sensore del furgone dell'immondizia raccoglie dati da un cestino dei rifiuti durante il prelievo, l'evento *triggera* la soluzione adottata da GreenQ per controllare, analizzare e salvare automaticamente le informazioni.

Ultimamente, la piattaforma *cloud-based* fornita ai municipi e ad altre organizzazioni di raccolta dei rifiuti, è stata migliorata inserendo una *dashboard* in cui è possibile visualizzare in tempo reale: la posizione e il percorso dei furgoni della spazzatura, inviare e ricevere notifiche, e analizzare i dati per migliorare i servizi. Per continuare ad evolvere la sua offerta, GreenQ prevede di incorporare altre tecnologie di IBM, come IBM Watson Visual Recognition, per catturare e tradurre informazioni visive sui percorsi di ritiro dei clienti.

³Link completo dell'articolo: [link](#)

2.2 Amazon: AWS Lambda

2.2.1 Panoramica

L'azienda Amazon offre come servizio FaaS [AWS Lambda](#). Inoltre, Amazon offre più di 200 servizi AWS e *Software-as-a-Service* (SaaS) che consentono di *triggerare* AWS Lambda.

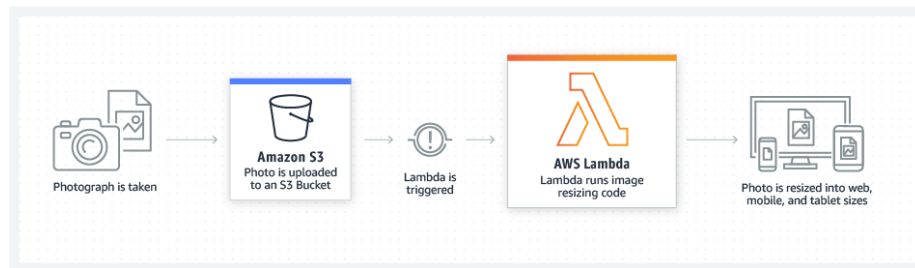


Figura 3: Esempio di utilizzo di AWS Lambda. Utilizzando il servizio di Data Store chiamato Amazon S3 è possibile salvare una foto carica da un utente, *triggerare* AWS Lambda e quest'ultimo eseguire l'azione di ridimensionamento dell'immagine.

Su AWS Lambda è possibile scegliere se eseguire le funzioni su processori x86 o architetture Arm. Il prezzo varia a seconda delle architetture e quindi si rimanda al sito ufficiale: <https://aws.amazon.com/it/lambda/pricing/>

2.2.2 Caso studio: Coca-Cola

Tra la marea di casi studio proposti dall'azienda Amazon, qua di seguito viene proposto quello di Coca-Cola Company.⁴

La Coca-Cola Company è un'azienda statunitense (Atlanta, Georgia) ed è una delle più grandi aziende produttrici e distributrici di bevande analcoliche e concentrati di sciroppi a livello mondiale.



Figura 4: Logo Coca-Cola Company.

⁴Link completo dell'articolo: [link](#)

La sfida da superare. Nel 2020, il mondo è stato colpito da una pandemia chiamata COVID-19. Questa tragedia ha provocato un grosso cambiamento nel comportamento degli esseri umani e l'azienda Coca-Cola stava cercando di migliorare i suoi distributori chiamati "Coca-Cola Freestyle".



Figura 5: Coca-Cola Freestyle.

Thomas Stubbs, vice presidente dell'ingegneria e dell'innovazione al *Coca-Cola Freestyle Equipment Innovation Center*, ha affermato: “*Tutti i dispenser Coca-Cola sono sicuri grazie ad una minuziosa cura e pulizia. Ma in questi periodi incerti, Coca-Cola vuole proporre ai clienti una nuova opzione: un’esperienza touchless*”. Per *touchless* si intende un distributore che non necessita di essere toccato, quindi limitando al massimo il contatto tra le persone.

La realizzazione. Per realizzare l'idea di Coca-Cola, la bassa latenza (*low latency*) era la chiave essenziale. Per questo motivo, Amazon ha pensato di adottare la sua *AWS serverless architecture* al nuovo distributore *contactless* così da far scegliere la bibita desiderata ai clienti tramite il proprio *smartphone* in pochi secondi **senza** il bisogno di scaricare un'applicazione o avere un account!

Il team Freestyle ha creato una web app *serverless* che si integrava con i distributori Coca-Cola Freestyle, in appena 4 mesi.

2.3 Google: cloud functions

2.3.1 Panoramica

L'azienda Google offre come servizio FaaS [Google Cloud Functions](#). Nel complesso sembra un servizio come gli altri, ma il sito fornisce una grande documentazione facilmente consultabile e una serie di casi d'uso.

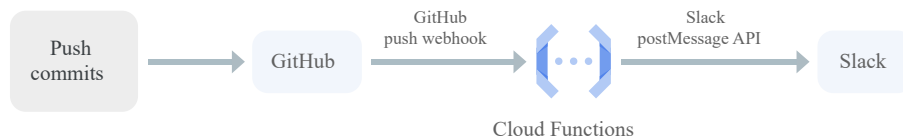


Figura 6: Esempio di caso d'uso: integrazione con servizi di terze parti e APIs.

Il prezzo varia a seconda dell'utilizzo che se ne fa: <https://cloud.google.com/functions/pricing>

2.3.2 Caso studio: Commerzbank

Tra i vari casi studio proposti da Google, qua di seguito viene approfondito Commerzbank.⁵

La Commerzbank è la quarta più grande banca della Germania, dopo Deutsche Bank, DZ Bank e KfW, con sede a Francoforte sul Meno, fondata nel 1870 ad Amburgo.



Figura 7: Foto della sede tedesca Commerzbank.

⁵Link completo dell'articolo: [link](#)

La sfida da superare. L'83% dei servizi finanziari, secondo Google Cloud survey, vengono sviluppati utilizzando servizi cloud. Per alcune banche, tra cui Commerzbank, la sicurezza è un aspetto fondamentale e questo cambiamento ha influito su tale aspetto. Le aziende che forniscono servizi finanziari devono maneggiare una grande quantità di dati sensibili che non possono essere compromessi.

Gli standard di sicurezza che sono cresciuti negli ultimi dieci anni, non possono essere spostati in massa all'interno del cloud, ma devono essere ridisegnati.

“Un fattore critico della nostra adozione della tecnologia cloud, è sempre stato le varie funzionalità che il cloud porta con sé quando deve processare una grande quantità di dati, per esempio, per ricavare informazioni dettagliate. Al momento del salvataggio in cloud, abbiamo bisogno di essere sicuri che i dati siano protetti e rispettino gli stretti standard di sicurezza. Questo è il motivo per cui abbiamo scelto Google Cloud.”

- Christian Gorke, Head of Cyber Center of Excellence, Commerzbank

La realizzazione. Il nuovo sistema di sicurezza implementato da Google è implementato all'interno (*built-in*). Questo si traduce in un'automatizzazione in larga scala, poiché i dipendenti non dovranno preoccuparsi di configurare software complessi come firewall, ma sarà tutto automatizzato da Google. Questo approccio viene chiamato *invisible security*.

Gorke spiega come funziona: “La sicurezza invisibile di Commerzbank è un approccio diviso in 4 passaggi. Primo, viene azionato Cloud Logging e Asset Inventory per ottenere una panoramica completa di tutte le nostre risorse nel cloud. Dopodiché, si implementa un filtro e uno strato d'azione basato sul modello Pub/Sub e BigQuery, i quali ci consentono di definire programmaticamente un ampio range di sicurezza per i vari casi d'uso. Successivamente, si valutano le misure corrette di sicurezza basate sugli eventi scatenati utilizzando Cloud Functions e Cloud Run, a seconda sempre dei casi d'uso. Dopo troviamo le misure di sicurezza corretti con BigQuery e Cloud Functions. Infine, eseguiamo un report dei risultati al Security Command Center.”

La soluzione di automatizzazione dei livelli di sicurezza di Google, consente ai dipendenti di Commerzbank di focalizzarsi sul proprio lavoro invece che sui possibili problemi di sicurezza.

2.4 Microsoft: Azure functions

2.4.1 Panoramica

L'azienda Microsoft offre come servizio FaaS [Microsoft Azure](https://azure.microsoft.com/en-us/pricing/). Sul proprio sito, Microsoft presenta le varie potenzialità del proprio servizio. Inoltre, Microsoft afferma che il proprio servizio è 5 volte più economico del concorrente Amazon: <https://azure.microsoft.com/en-us/pricing/>

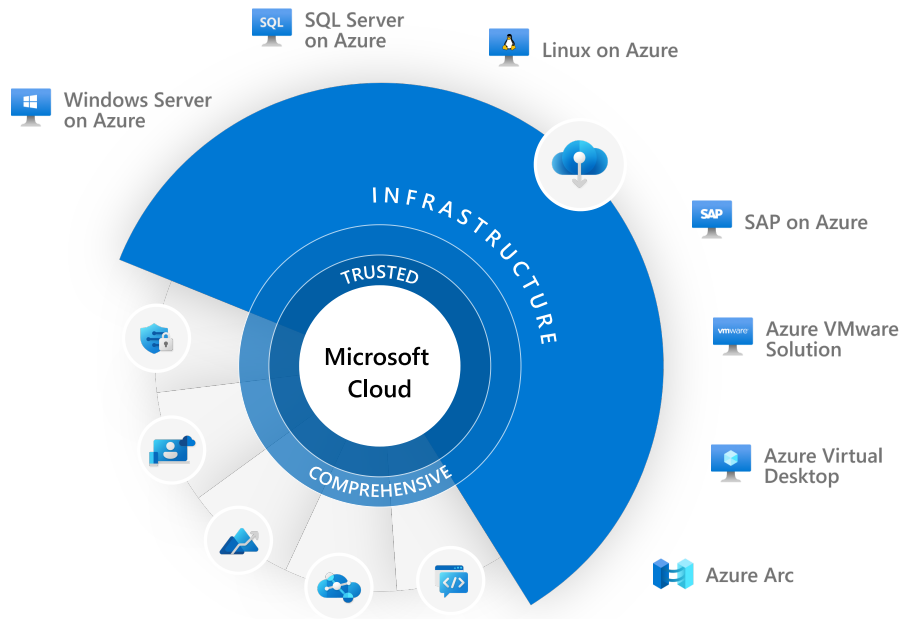


Figura 8: Infrastruttura di Azure.

2.4.2 Caso studio: Fujitsu

Tra le aziende più famose con cui ha lavorato Microsoft, si presenta qua di seguito Fujitsu.⁶

Fujitsu è un'azienda giapponese con sede a Tokyo e Kawasaki. È uno dei maggiori fornitori mondiali di prodotti e servizi per l'*information technology*, dai personal computer, midrange, grandi server e sistemi di storage, fino al software.



Figura 9: Logo di Fujitsu.

La sfida da superare. Recentemente, l'azienda Fujitsu ha adottato un nuovo tipo di *business brand* chiamato Fujitsu Uvance. Il nuovo *brand* si focalizza su come Fujitsu aiuta i suoi clienti a risolvere le sfide della società, utilizzando le sue funzionalità tecnologiche e l'esperienza del *problem-solving*.

Con il lancio di questa nuova sfida, Fujitsu sapeva che doveva rimodernare la sua presenza online per migliorare il supporto. Dato che la presenza online è un sistema di business critico, Fujitsu si rese conto che il suo sistema era da rinnovare a causa dei continui aggiornamenti giornalieri ai suoi sistemi. Fujitsu scelse di adottare la tecnologia cloud, così decise di effettuare una migrazione.

La realizzazione. Dopo il confronto di vari servizi cloud, Fujitsu scelse Azure di Microsoft. In particolare, scelse i prodotti: Azure High Priority e FastTrack. La migrazione, effettuata in pochi minuti, rese il sistema più scalabile, flessibile ed efficiente. Inoltre, Fujitsu poteva dare servizi correlati a Microsoft ai loro clienti. Caso completo: [case study](#)

⁶Link completo dell'articolo: [link](#)

2.5 Oracle: OCI

2.5.1 Panoramica

L'azienda Oracle offre come servizio FaaS [Oracle Cloud Infrastructure \(OCI\) Functions](#). Il prezzo risulta essere, forse, simile a quello di Google. Si lascia comunque il link per approfondire: <https://www.oracle.com/cloud/costestimator.html>

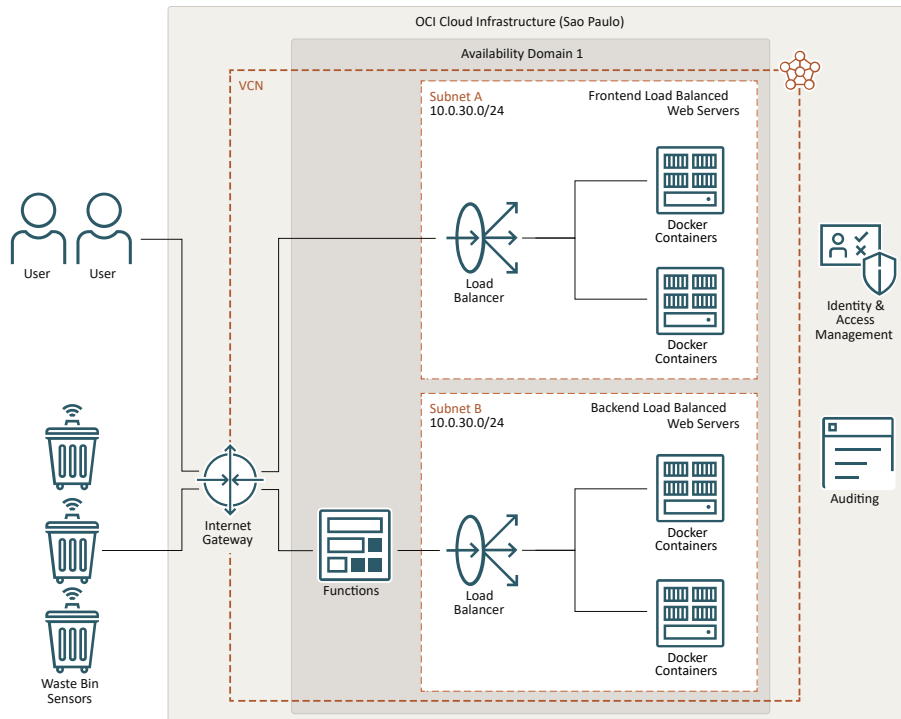


Figura 10: Caso studio: Waste2Go. Streaming dei dati IoT mediante funzioni su Oracle Cloud Infrastructure ([link articolo](#)).

2.5.2 Caso studio

Tra le aziende di spessore che hanno lavorato con Oracle, si presenta qua di seguito Generali.⁷

Generali è la più grande compagnia di assicurazione italiana e da terza per fatturato nel settore delle assicurazioni, dopo Allianz ed AXA.



Figura 11: Logo di Assicurazioni Generali.

La sfida da superare. Generali possedeva più di 70'000 dipendenti e operava in oltre 50 paesi. È chiaro che il numero di dati era piuttosto elevate.

Il dipartimento delle risorse umane compilava i *template* dei dati manualmente e li inviava al team capo, così da eseguire *reports* e analisi. È evidente che questo approccio richiedeva un consumo di tempo elevato ed era soggetto ad errori umani. L'obiettivo era quello di trovare una soluzione che incrementasse la mentalità *data-driven* e il reparto di risorse umane.

La realizzazione. Generali ha scelto i prodotti di Oracle come migliore opzione per migliorare le proprie performance grazie alla facile scalabilità e alle varie integrazioni. Il risultato è stato soddisfacente, poiché Generali ha automatizzato il processo di *reporting* e ha consentito allo staff delle risorse umane di essere più produttivo e focalizzato sulle attività pertinenti. Inoltre, è stato implementato un meccanismo di Machine learning così da aumentare la produzione di report.

⁷Link completo dell'articolo: [link](#)

3 Esempio di applicazione: Azure Functions

3.1 Introduzione: costruire API serverless con Azure Functions

Guida ufficiale: [link academy](#)

Con Azure Functions è possibile costruire rapidamente una API HTTP adatta alle applicazioni web. Il processo non prevede alcun web frameworks, quindi è molto semplice! Dato che Azure Functions è un servizio FaaS (Function-as-a-Service), il servizio risponderà ad un trigger HTTP solamente quando verrà invocato l'API. Quindi, finché l'API non viene utilizzata, eventuali costi del servizio non vengono addebitati. Questi vantaggi rendono Azure Functions un servizio molto richiesto dalle aziende.

Gli obiettivi in questo modulo sono:

- Costruire una API HTTP utilizzando l'estensione Azure Functions per Visual Studio Code.
- Imparare come modificare Azure Functions per renderlo *RESTful*.
- Specificare chi può accedere alla API utilizzando CORS (*Cross-Origin Resource Sharing*).

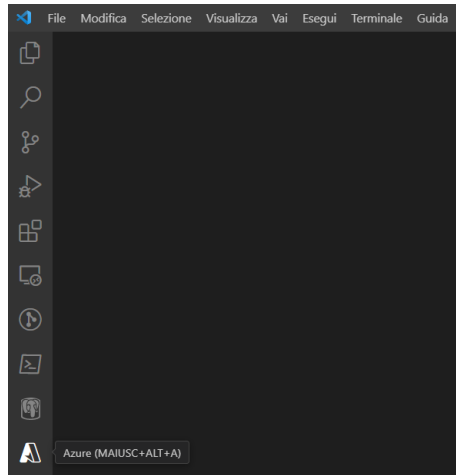
I prerequisiti sono semplicemente 5:

- Conoscenze base dei web services e concetti di API, incluso i metodi HTTP e REST;
- Conoscenza di Azure Functions, incluso gli HTTP Triggers;
- [Visual Studio Code](#) installato;
- [Azure Functions Core Tools](#) installato;
- [Azure Functions](#) installato.

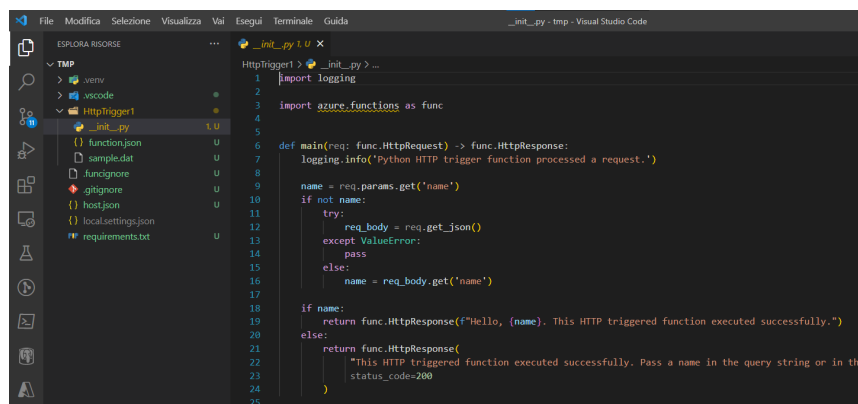
3.2 Creazione dell'environment

Una volta installati tutti i componenti necessari, si esegue quanto segue:

1. Apertura di Visual Studio Code;
2. Sulla colonna di sinistra, cliccare sull'icona di Microsoft Azure ed eseguire il login



3. Adesso si procede alla creazione di un nuovo progetto, quindi si apre la *palette* di VSCode (Windows CTRL + SHIFT + P) e si digita "Azure Functions: Create New Project...";
4. Si seleziona la cartella in cui verrà salvato il progetto;
5. Si seleziona il linguaggio di programmazione (nel nostro caso Python), le versione (3.10 nel nostro caso), il template che dovrà essere HTTP trigger per creare le API, il nome della funzione (potrà essere cambiato in futuro), poi si deve selezionare "Function" e scegliere se aprirlo nella finestra corrente;
6. Una volta creato l'ambiente, la schermata iniziale dovrebbe avere dei file con un codice d'esempio:



3.3 Creazione delle API

L'attuale *environment* non è caricato sul cloud! È solamente una versione di debug che viene utilizzata dagli sviluppatori per eseguire delle prove delle API sviluppate.

Una possibile implementazione delle API è fatta nel seguente modo. Si crea una cartella, contenente del codice python, che rappresenta la pagina web che deve essere ritornata nel caso in cui venga richiamata quella determinata API. Quindi:

1. Si creano due cartelle: Somma e Sottrazione;
2. Si copiano all'interno i file d'esempio che sono stati creati durante la creazione dell'*environment*, quindi:
 - `__init__.py`
 - `function.json`
 - `sample.dat`
3. Si modificano i file `function.json` e si aggiunge:
 - Nella cartella Somma, il file deve essere modificato aggiungendo sotto il parametro `methods`, il parametro `"route": "calcola-somma"`
 - Nella cartella Sottrazione, il file deve essere modificato aggiungendo sotto il parametro `methods`, il parametro `"route": "calcola-sottrazione"`

Così facendo, le API potranno essere raggiunte tramite il link: `localhost:7000/api/calcola-somma`

4. Adesso è necessario modificare i file python. L'implementazione della differenza si lascia al lettore, ricordando che la variabile `req_body` contiene un dizionario del file json che viene passato durante l'invocazione della API. Quindi, scrivendo `req_body["name_val_inside_json"]` è possibile acquisire il valore corrispondente.

```
1 import logging
2
3 import azure.functions as func
4
5
6 def main(req: func.HttpRequest) -> func.HttpResponse:
7     logging.info('Python HTTP trigger function processed a
8     request.')
9
10    name = req.params.get('name')
11    if not name:
12        try:
13            req_body = req.get_json()
14        except ValueError:
15            pass
16        else:
17            name = req_body.get('name')
18
19    if name:
20        return func.HttpResponse(f"Hello, {name}. This HTTP
21        triggered function executed successfully.")
```

```

20     else:
21         sum = req_body["val1"] + req_body["val2"]
22         return func.HttpResponse(
23             f"This HTTP triggered function executed
                successfully. Pass a name in the query string or in the
                request body for a personalized response.\nIl risultato e
                ': {sum}",
24             status_code=200
25         )

```