香港中文大學
The Chinese University of Hong Kong

MORE ON WEB APP ARCHITECTURE

CSCI2720 2022–23 Term 1

Building Web Applications

Dr. Chuck-jee Chau
chuckjee@cse.cuhk.edu.hk

OUTLINE

- Web API
- The RESTful API
- The three-tier architecture
- From Cloud Computing to Serverless Computing
- FaaS
- Why Serverless?
- Security Threats

THE EVOLVING WEB

- The traditional "web" was built for browsing by human beings
- Towards the ***Semantic Web***
  - Information on web understandable by machines
  - HTML5 semantic elements, e.g., `<footer>`, `<article>`
  - Metadata in web pages
    - `<meta>` for page description and keywords
  - Programmatic interfaces, e.g., Web API, Webhook

  - See: Tim Berners-Lee *et al.* "The Semantic Web"
    - https://www-sop.inria.fr/acacia/cours/essi2006/Scientific%20American_%20Feature%20Article_%20The%20Semantic%20Web_%20May%202001.pdf

WEB API

- Web API (Application Programmatic Interface) can provide
  - Data resources
    - e.g., bus arrival time, restaurant ratings, …
  - Services or ***microservices***
    - e.g., converting coordinates into place names, creating QR codes

- Developers can then easily incorporate these building blocks into other web applications

WEB API

- Public vs. private
  - Public APIs are available to everyone, but may subject to licenses
  - Private APIs are only available to internal developers
- Free vs. premium
  - Premium APIs may charge on-demand
  - "*API Economy*"
- API marketplace and directory sites
  - RapidAPI
  - ProgrammableWeb

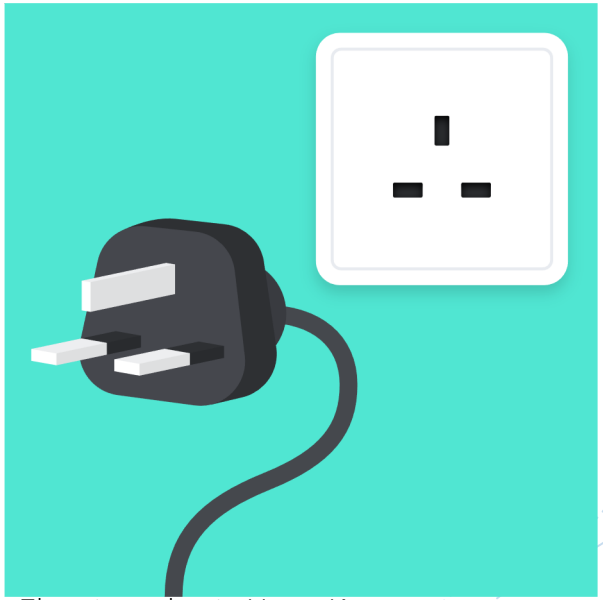
PUBLIC API

- Google
  - *https://developers.google.com/apis-explorer*
- Facebook
  - https://developers.facebook.com/docs/apis-and-sdks/

- Hong Kong Government
  - *https://data.gov.hk/en/help*
- Data portals of many cities
  - *https://dataportals.org/search*

API ENDPOINTS

- API usually provides endpoints as a URI to provide services
  - It needs to be static, without affecting applications built upon
- RESTful API *vs.* SOAP
- JSON *vs.* XML, *vs.* GraphQL
- Security and authentication
  - HTTPS
  - API Key
  - Rate limiting
  - OAuth – authentication + authorization



Electric socket in Hong Kong
https://www.electricalsafetyfirst.org.uk/guidance/advice-for-you/when-travelling/travel-adaptor-for-hong-kong/

BEING RESTFUL

- ***REpresentational State Transfer*** (REST) architectural style
  - Uniform interface
    - Unique and consistent methods of resource identification and manipulation, with clear messages to define actions and information
  - Client-server architecture
    - Separation of concerns: user interface vs. data storage
  - Statelessness
    - Each request must already contain all knowledge needed to complete the request
  - Cacheability
    - Allowing identification of cacheable contents, and to avoid stale or inappropriate data
  - Layered system
    - Hierarchical layers of components, without affecting communication between client and server (e.g., proxy and load balancers)
  - Code-on-demand (*optional*)
    - Executable code can be sent from server to provide extra temporary functionalities
  - *See: https://restfulapi.net*

RESTFUL API: COMMON PATTERN

- List container contents: **`GET /items`**
- Add an item to container: **`POST /items`**
  - with item in request
  - URI of item returned in HTTP response header, *e.g.*

    `Location: http://host/items/itemid`
- Retrieve an item: **`GET /items/itemid`**
- Update an item: **`PUT /items/itemid`**
  - with updated item in request
- Delete an item: **`DELETE /items/itemid`**

RESTFUL API

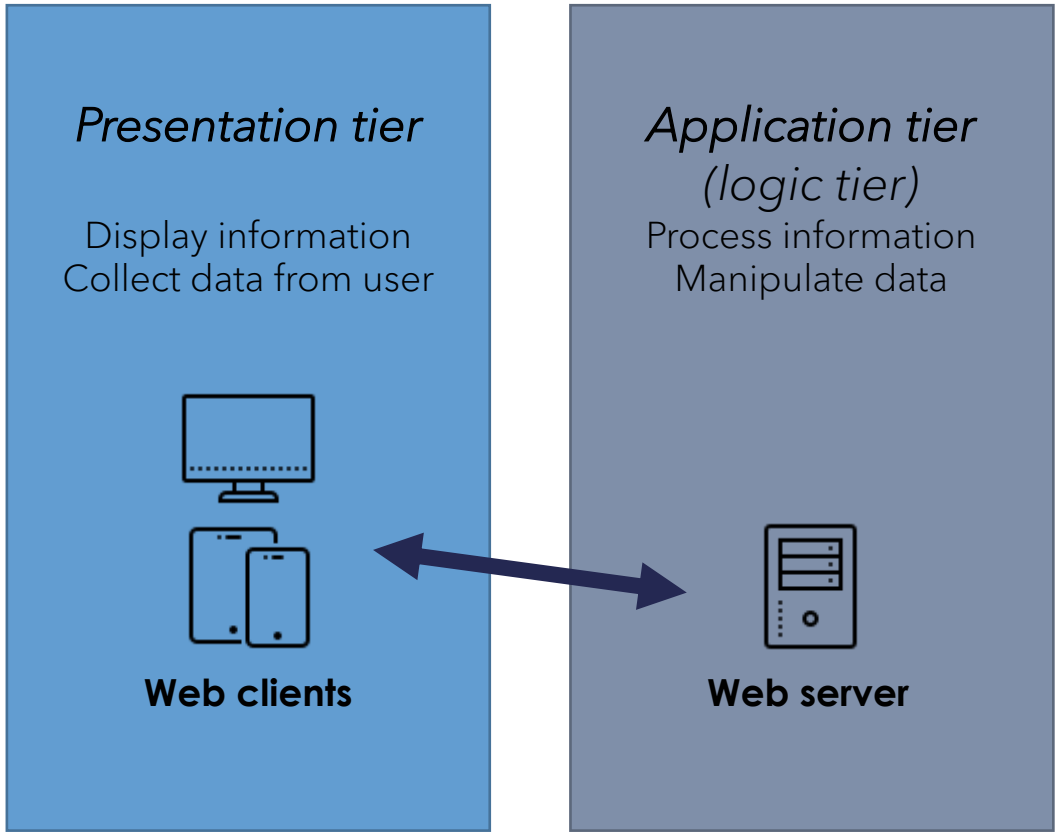
- RESTful services are easy to work with
  - Clients do not need to use specialized API
  - Clients just use standard HTTP
  - You can use browser for experimentation and testing
- Uniformity
  - URI is a uniform way to identify resources
  - HTTP uniform interface to manipulate resources
- Scripting language friendly
  - Easy to consume and develop

WEBHOOK

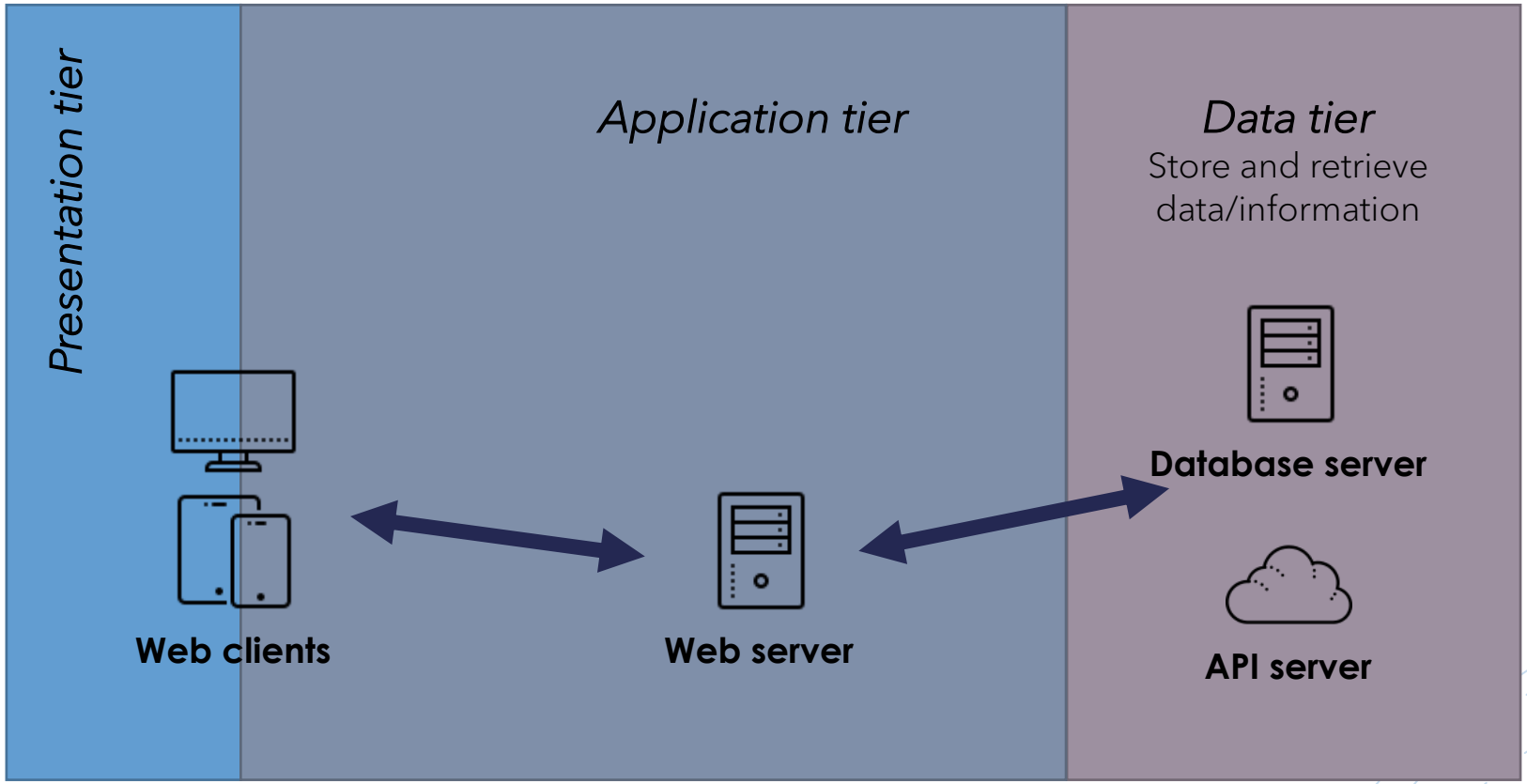
- Nicknamed “reverse API”
- Webhook is an event-triggered action
  - i.e., the workflow is initiated by the server, not the client
  - e.g., when a new user is created in company, send an email to everyone else of the announcement
- Requires developer to register a URL at the service provider which supports webhook events
- A POST payload will be delivered to the configured URL
- e.g., GitHub Webhooks: https://developer.github.com/webhooks/creating/

THE CLIENT-SERVER (TWO-TIER) ARCHITECTURE

- We have seen the usual client-server model
  - Good enough for smaller scale of services



THE THREE-TIER ARCHITECTURE



THE THREE-TIER ARCHITECTURE

- Logical and physical separation of functionality
- Each tier could run on different platforms
- Faster development by separate teams
- Improved scalability, reliability, and security

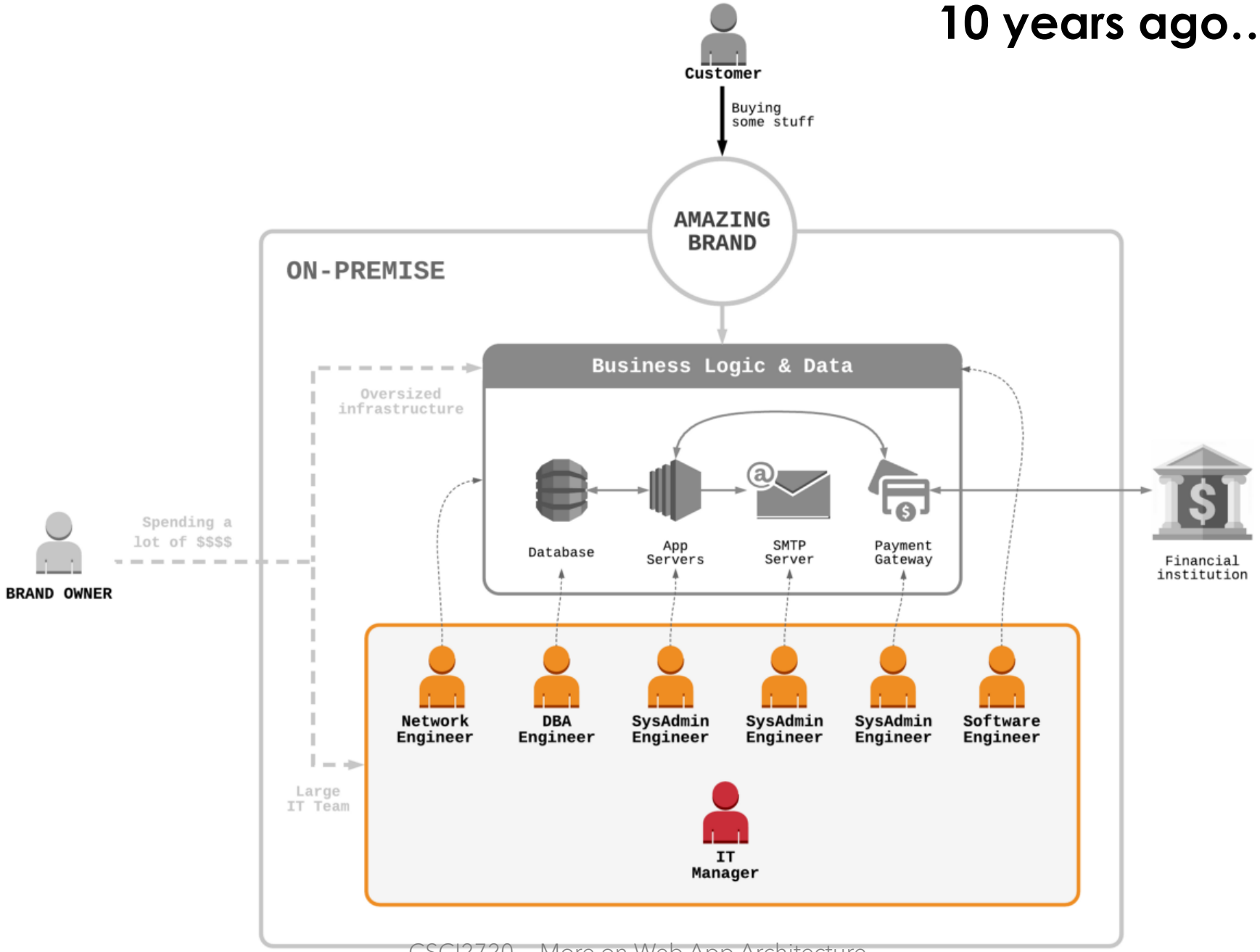
- *See: https://www.ibm.com/cloud/learn/three-tier-architecture*

SERVERLESS COMPUTING

- Web pages or web applications are served by ***web servers***
  - Ever since the birth of HTML, served on a NeXT computer at CERN
- In a ***serverless*** architecture, web server is still there
- Yet, web developers don't need to worry about it
  - "Focus on the ***application***, not the ***infrastructure***."

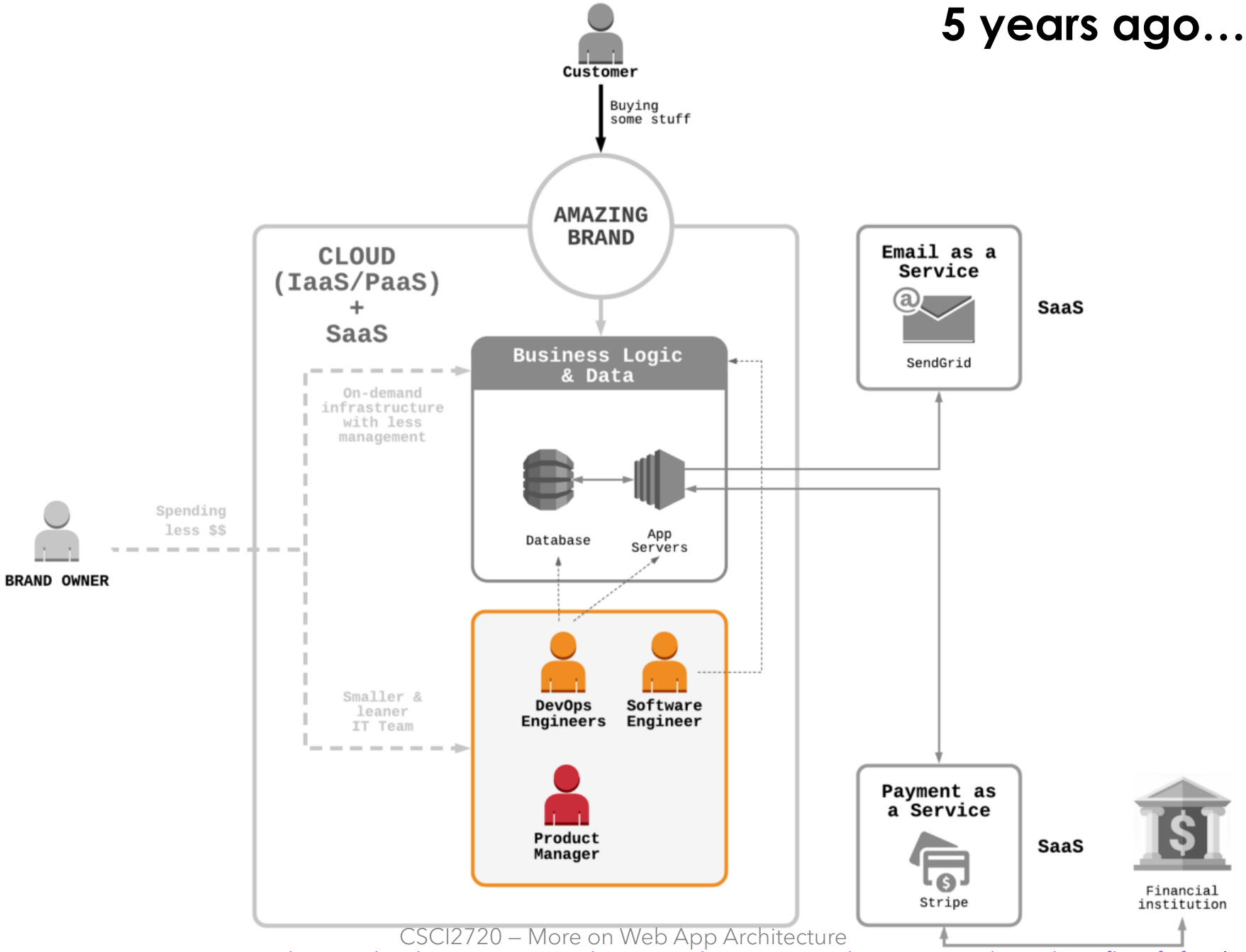
The first web server, in 1990, used by Tim Berners-Lee. *See:* https://cds.cern.ch/record/42413

10 years ago…



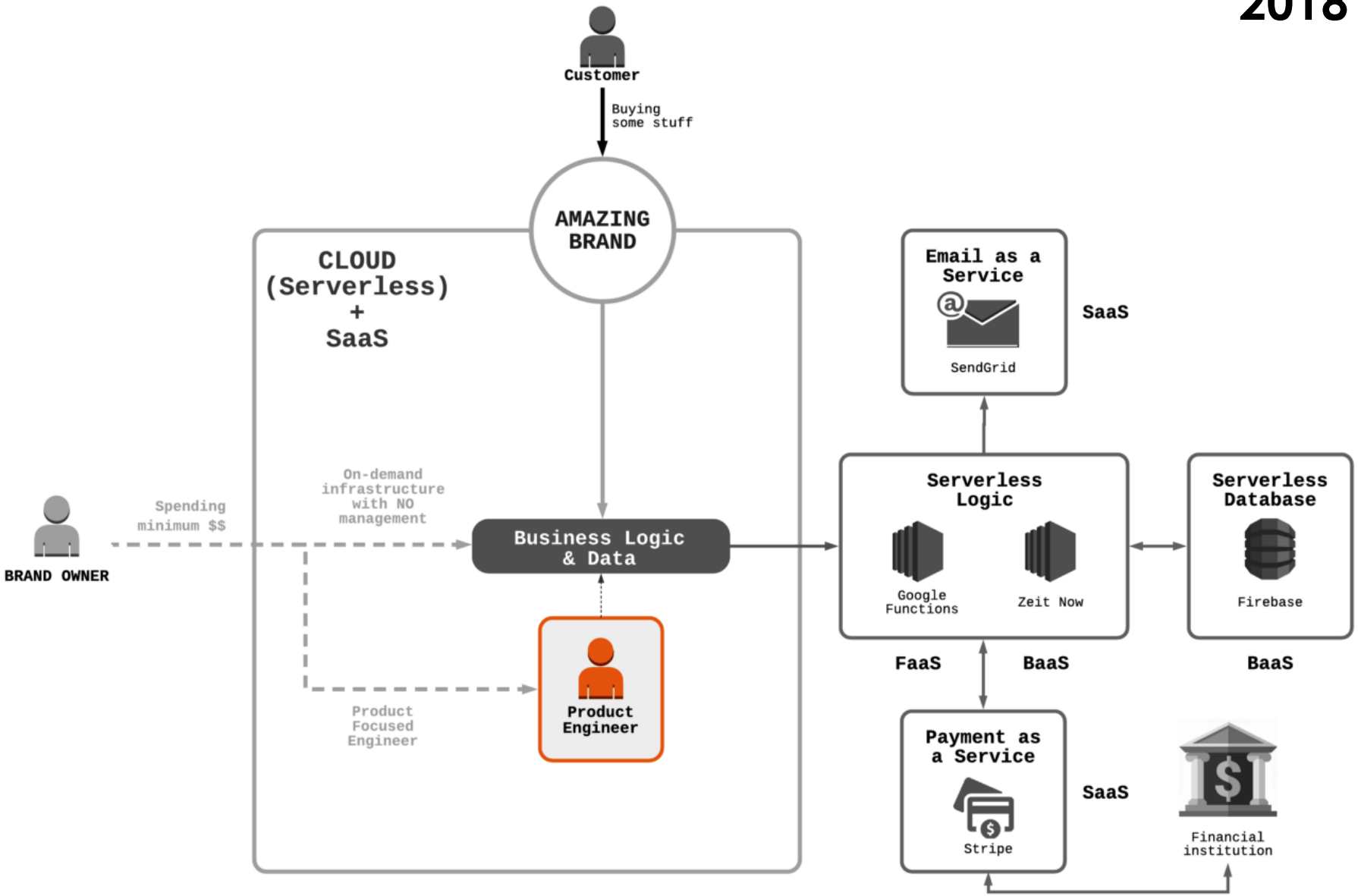
Source: https://hackernoon.com/the-serverless-series-what-is-serverless-d651fbacf3f4 *(link is no longer available)*

5 years ago…



Source: https://hackernoon.com/the-serverless-series-what-is-serverless-d651fbacf3f4 *(link is no longer available)*

2018



Source: https://hackernoon.com/the-serverless-series-what-is-serverless-d651fbacf3f4 *(link is no longer available)*

FROM CLOUD COMPUTING TO SERVERLESS COMPUTING

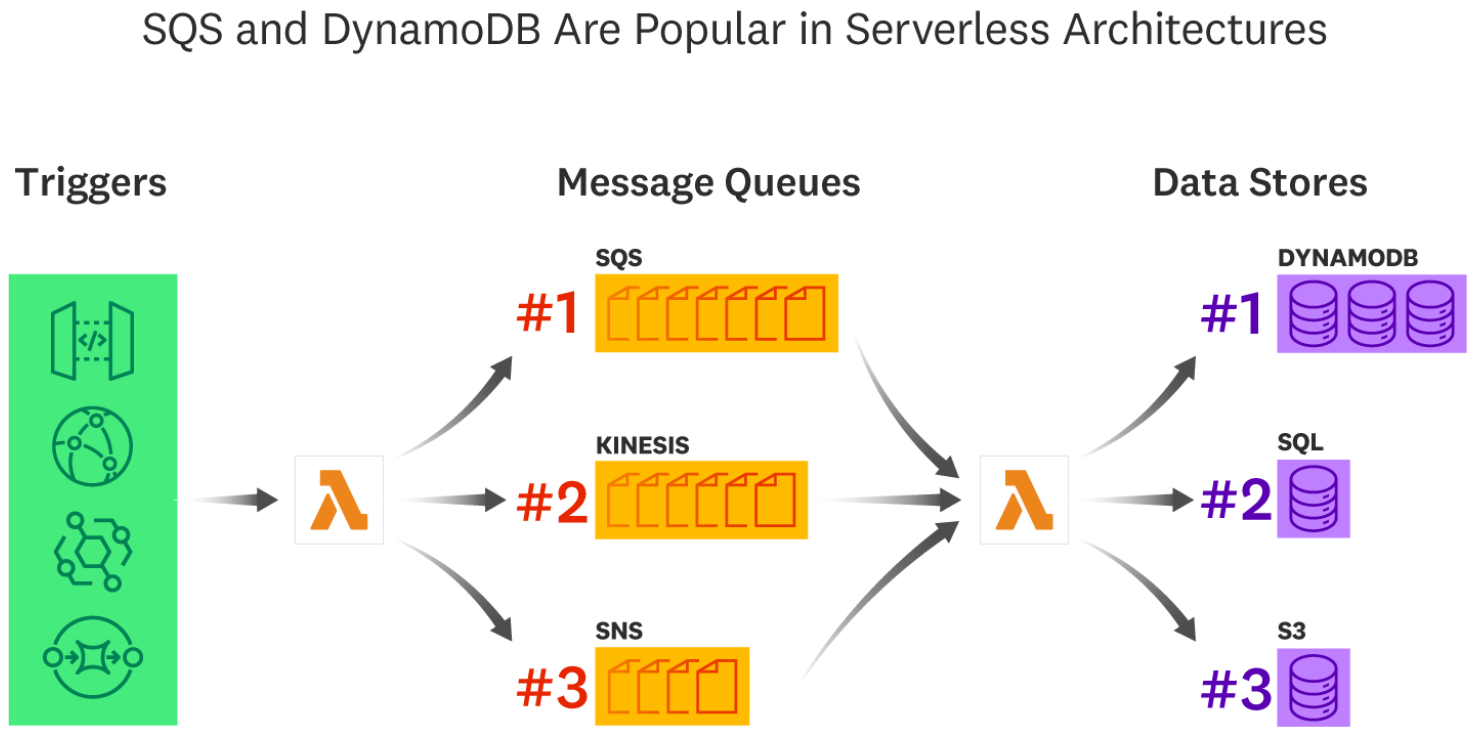
- Some *fancy* terms to know!
  - Software as a Service (***SaaS***)
    - Web 2.0 "***the cloud***" (~1999)
    - Providing apps/software
  - Infrastructure as a Service (***IaaS***)
    - Amazon Web Services (~2002)
    - Providing basic hardware and storage
  - Platform as a Service (***PaaS***)
    - Providing partial application stack
      - *e.g.*, OS, database
  - Backend as a Service (***BaaS***)
    - Providing SDK and API
  - Function as a Service (***FaaS***)
    - AWS Lambda (~2014)
    - Event-driven, on-demand, charge per use
- More ****aaS*** for you to name it! *See:* *https://en.wikipedia.org/wiki/As_a_service*

FAAS: WHAT CAN BE DONE?

- A Function-as-a-Service allows you to execute short operations in response to events (website click, image upload, etc.), using the cloud infrastructure, e.g.,
  - Payment processing
  - Data conversion
  - App notification

- Could be further linked up with container infrastructure in PaaS, *e.g.*, Docker

FAAS: WHAT CAN BE DONE?



See: https://www.datadoghq.com/state-of-serverless/

Source: Datadog

WHY SERVERLESS?

- FaaS gets popular for a number of reasons
  - Forget about server infrastructure (hardware/software)
  - Development is modular ➔ *agility*
  - Inherently scalable: scale the functions
  - Not charged for idling resources
- However…
  - Not everything under your control
    - Privacy? Debug? Transparency?
    - Tricky for development/testing locally
  - Auto-scaling of cost with usage ➔ *unpredictable*
  - Response latency

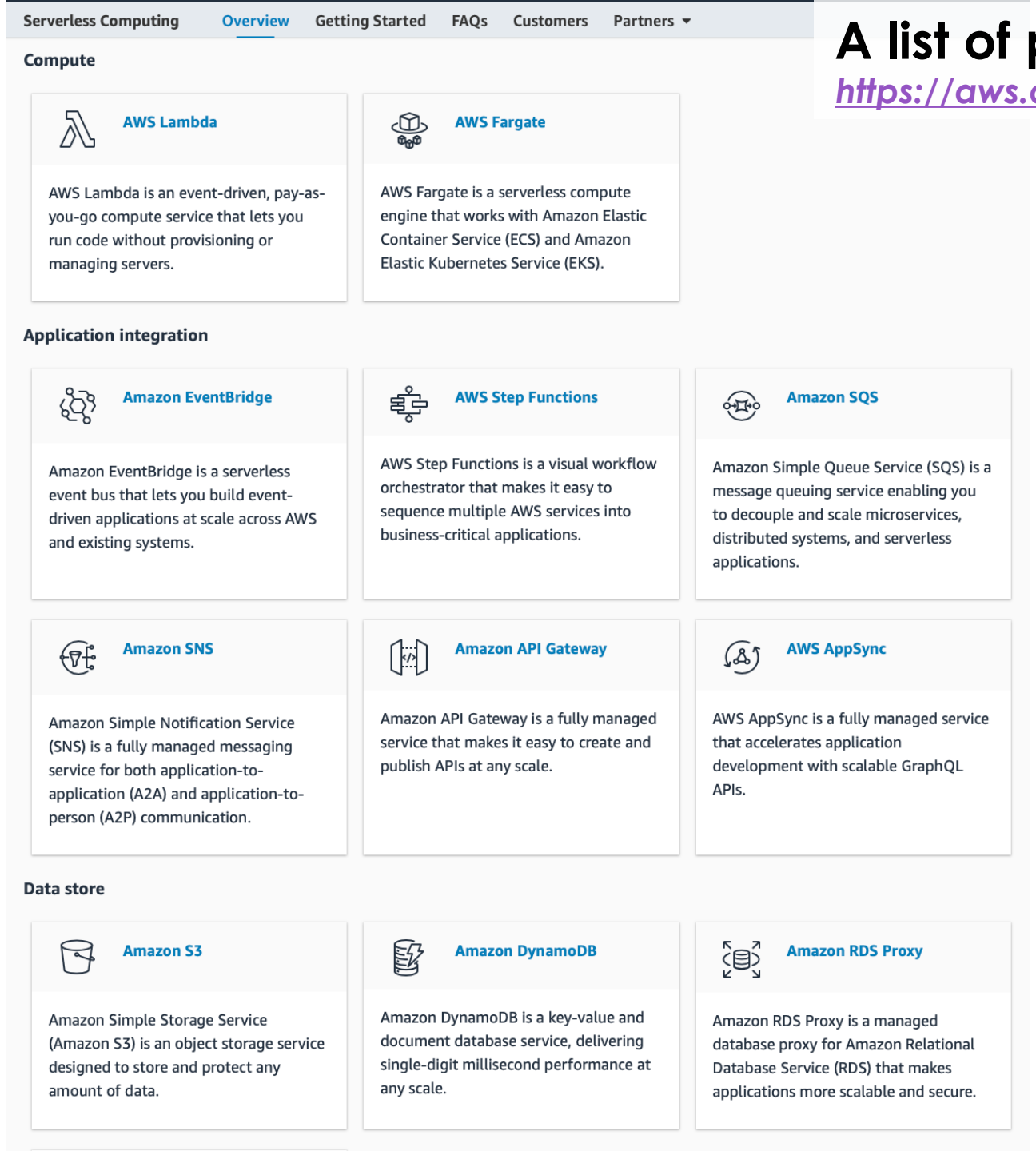
CLOUD PLATFORMS

- The big names
  - Amazon Web Services
    - AWS Lambda
  - Google Cloud
    - Google Cloud Functions
  - Microsoft Azure
    - Azure Functions
- Basic features
  - Load balancing
  - Performance monitoring
  - Usage analytics
  - Billing reports
  - Easy development tools
  - And… free trials!

- *See: https://hackernoon.com/serverless-101-an-introduction-to-faas-function-as-a-service*

A list of products in AWS

https://aws.amazon.com/serverless



A list of products in Google Cloud

https://cloud.google.com/products/#section-20

Google Cloud products

- Overview
- Featured products
- AI and Machine Learning
- API Management
- Compute
- Containers
- Data Analytics
- Databases
- Developer Tools
- Healthcare and Life Sciences
- Hybrid and Multi-cloud
- Internet of Things (IoT)
- Management Tools
- Media and Gaming
- Migration
- Networking
- Operations
- Security and Identity
- Serverless Computing
- Storage
- More Google Cloud products

Serverless Computing

Learn more

App Engine

Serverless application platform for apps and back ends.

Cloud Functions

Platform for creating functions that respond to cloud events.

Cloud Run

Fully managed environment for running containerized apps.

Workflows

Workflow orchestration for serverless products and API services.

Storage

Learn more

Archive Storage

Data archive that offers online access speed at ultra low cost.

Cloud Data Transfer

Tools and services for transferring your data to Google Cloud.

Cloud Storage

Object storage that's secure, durable, and scalable.

Cloud Storage for Firebase

Object storage for storing and serving user-generated content.

Filestore

File storage that is highly scalable and secure.

Google Workspace Essentials

Cloud-based file sharing, content collaboration, and storage.

A list of products in Microsoft Azure

Select a category:

- AI + Machine Learning
- Analytics
- Blockchain
- Compute
- Containers
- Databases
- Developer Tools
- DevOps
- Hybrid + Multicloud
- Identity
- Integration
- Internet of Things
- Management and Governance
- Media
- Migration
- Mixed Reality
- Mobile
- Networking
- Security
- Storage
- Web
- Windows Virtual Desktop

Compute https://azure.microsoft.com/en-us/services/#compute

Learn more >

Access cloud compute capacity and scale on demand—and only pay for the resources you use

API Apps
Easily build and consume Cloud APIs

App Service
Quickly create powerful cloud apps for web and mobile

Azure CycleCloud
Create, manage, operate, and optimize HPC and big compute clusters of any scale

Azure Functions
Process events with serverless code

Azure Kubernetes Service (AKS)
Simplify the deployment, management, and operations of Kubernetes

Azure Quantum PREVIEW
Experience quantum impact today on Azure

Azure Spring Cloud
A fully managed Spring Cloud service, jointly built and operated with VMware

Azure VMware Solution
Run your VMware workloads natively on Azure

Batch
Cloud-scale job scheduling and compute management

Cloud Services
Create highly-available, infinitely-scalable cloud applications and APIs

Container Instances
Easily run containers on Azure without managing servers

Linux Virtual Machines
Provision virtual machines for Ubuntu, Red Hat, and more

Mobile Apps
Build and host the backend for any mobile app

Service Fabric
Develop microservices and orchestrate containers on Windows or Linux

SQL Server on Virtual Machines
Host enterprise SQL Server apps in the cloud

Static Web Apps PREVIEW
A modern web app service that offers streamlined full-stack development from source code to global high availability

Virtual Machine Scale Sets
Manage and scale up to thousands of Linux and Windows virtual machines

Virtual Machines
Provision Windows and Linux virtual machines in seconds

SERVERLESS SECURITY THREATS

- Fragmented environment == New home for ***security threats***
  - Function event-data injection
  - Broken authentication
  - Insecure serverless deployment configuration
  - Overpriviledged function permissions and roles
  - Inadequate function monitoring and logging
- *See: https://newsletterest.com/message/48105/The-Ugly-Truth-About-Serverless-Data-Security*

READ FURTHER…

A RESTful Tutorial

https://www.restapitutorial.com/

What are Webhooks?

https://zapier.com/blog/what-are-webhooks/

Webhook vs API: What's the Difference?

https://hackernoon.com/webhook-vs-api-whats-the-difference-8d41e6661652

Serverless architectures

https://martinfowler.com/articles/serverless.html

What is Serverless? The "2020" edition

https://medium.com/swlh/what-is-serverless-the-2020-edition-5a2f21581fe5