Cloud Architecture of an E-Commerce Platform

Patterns and Artificial Intelligence for Cloud and Cloud-Edge Continuum

Companion website:
https://dub.sh/11913446-cloud-arch

Application Overview

Péter Ferenc Gyarmati, 2022. 12. 15

Rationale

Online selling is becoming more popular.

Small and medium businesses face high barriers to entry in the market

E-commerce marketplaces offer a solution by providing a platform for vendors to list and sell their products

The marketplace takes a commission on each sale and handles platform development and maintenance

The marketplace also provides advertising and review capabilities for vendors and buyers

Functional Requirements

Customers can register and login to the system	Vendors can register and list their products	Vendors can promote their products for a period of time	Promoted products appear for users automatically based on their preferences and real-time interactions with the system	Customers can search for products and observe them in detail, with support for image recognition
Customers can buy products	Customers can choose from payment methods such as credit card, PayPal, and bank transfer	Customers can leave ratings and reviews for products	Customers can connect with vendors in built-in chat rooms	The system detects fraudulent activities and blocks the corresponding accounts

Non-Functional Requirements

The system should be highly available across the globe

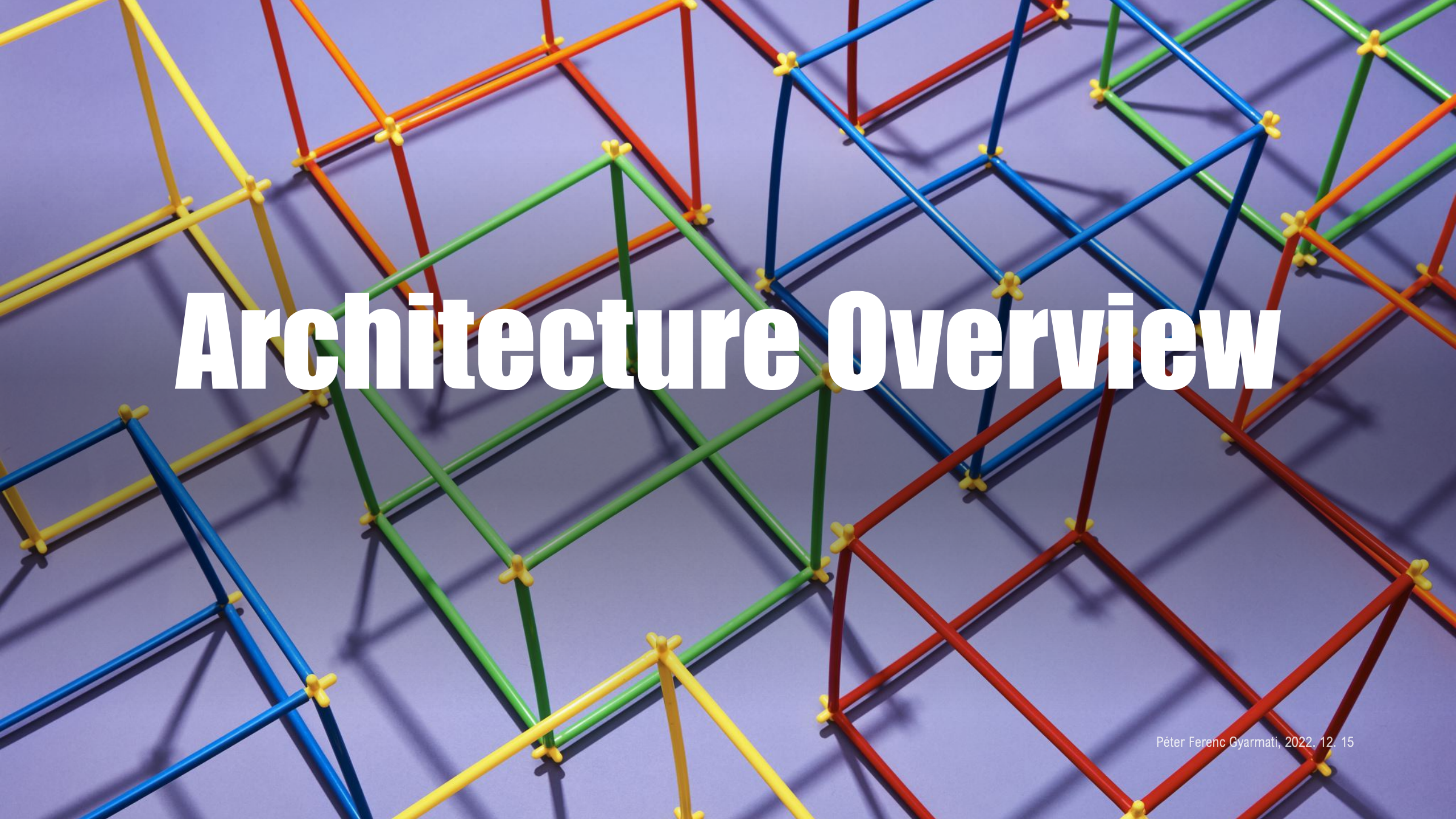
The system should have extremely low latency to ensure a good user experience and the maximization of conversions

The system should be scalable to handle a large number of concurrent users

The system should tolerate spikes in traffic (e.g. before Holidays, Black Friday, etc.)

The system should be secure and protect customer and vendor data, compliant with local and international laws

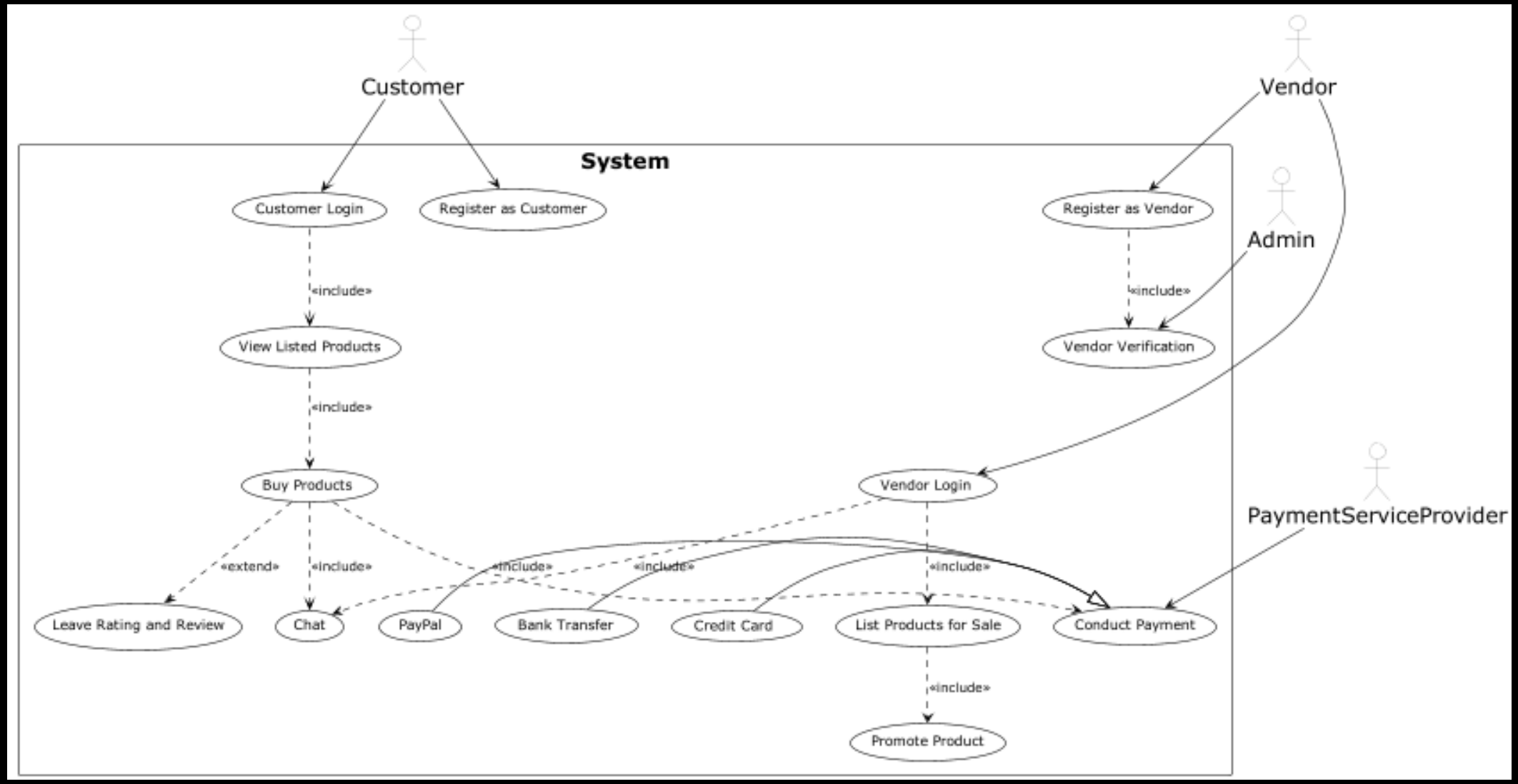
Data-loss should be prevented to maximize the satisfaction of customers and vendors



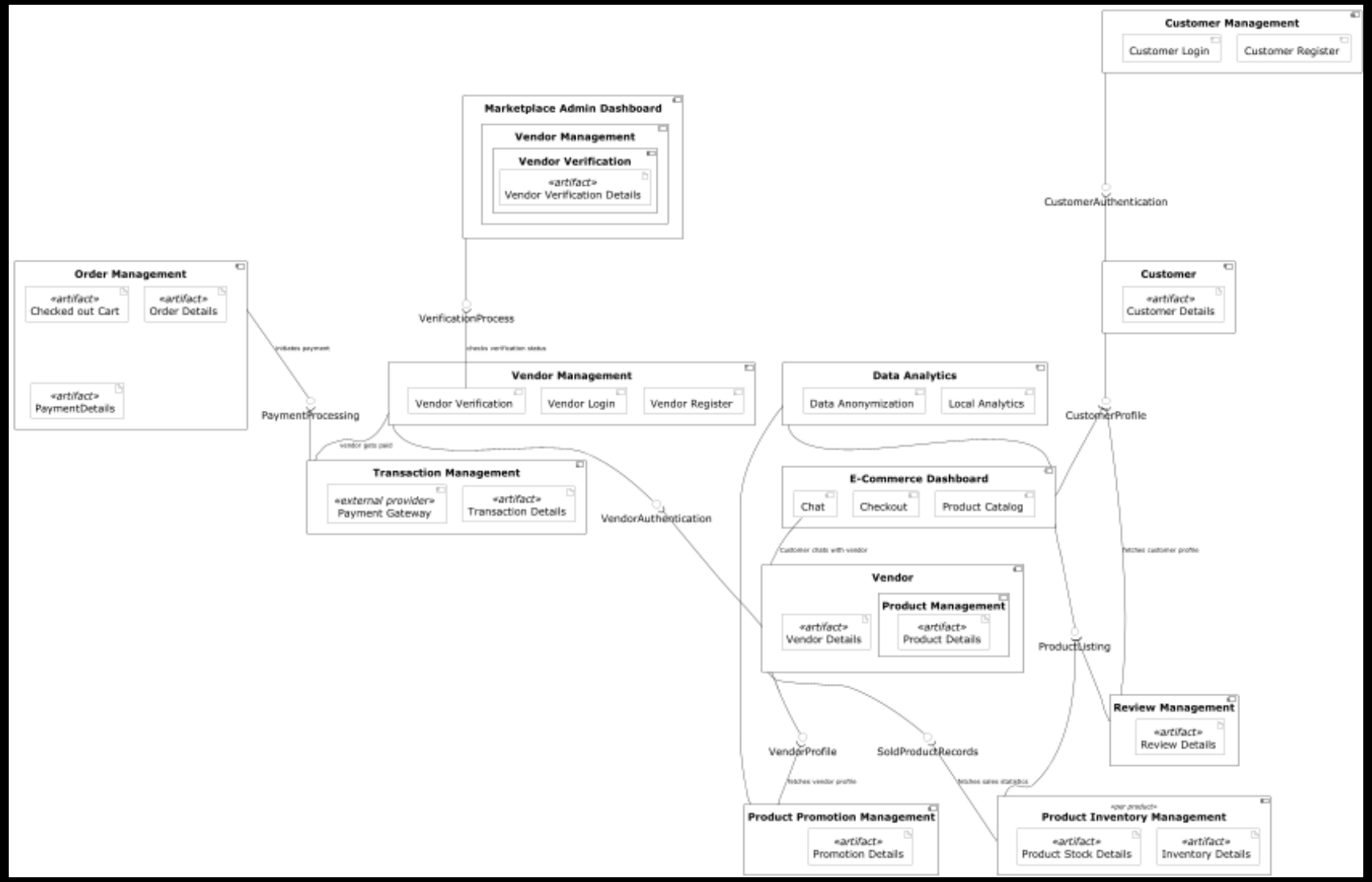
Architecture Overview

Péter Ferenc Gyarmati, 2022. 12. 15

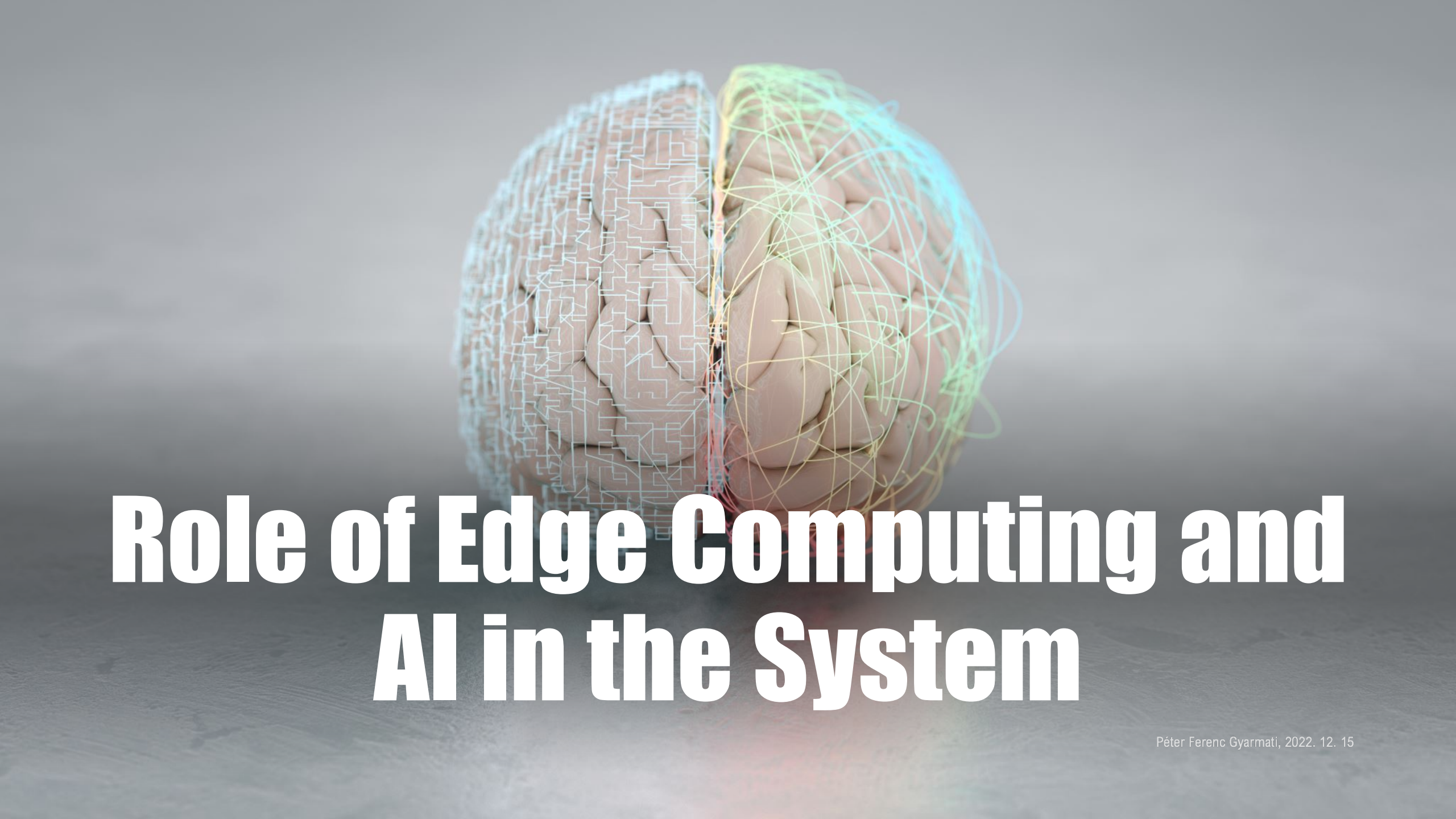
Use Cases


Customer
Vendor
Admin
PaymentServiceProvider
System
Customer Login
Register as Customer
Register as Vendor
«include»
«include»
View Listed Products
Vendor Verification
«include»
Buy Products
Vendor Login
«extend»
«include»
«include»
«include»
«include»
Leave Rating and Review
Chat
PayPal
Bank Transfer
Credit Card
List Products for Sale
Conduct Payment
«include»
Promote Product

Components



Péter Ferenc Gyarmati, 2022. 12. 15



Role of Edge Computing and AI in the System

Péter Ferenc Gyarmati, 2022. 12. 15

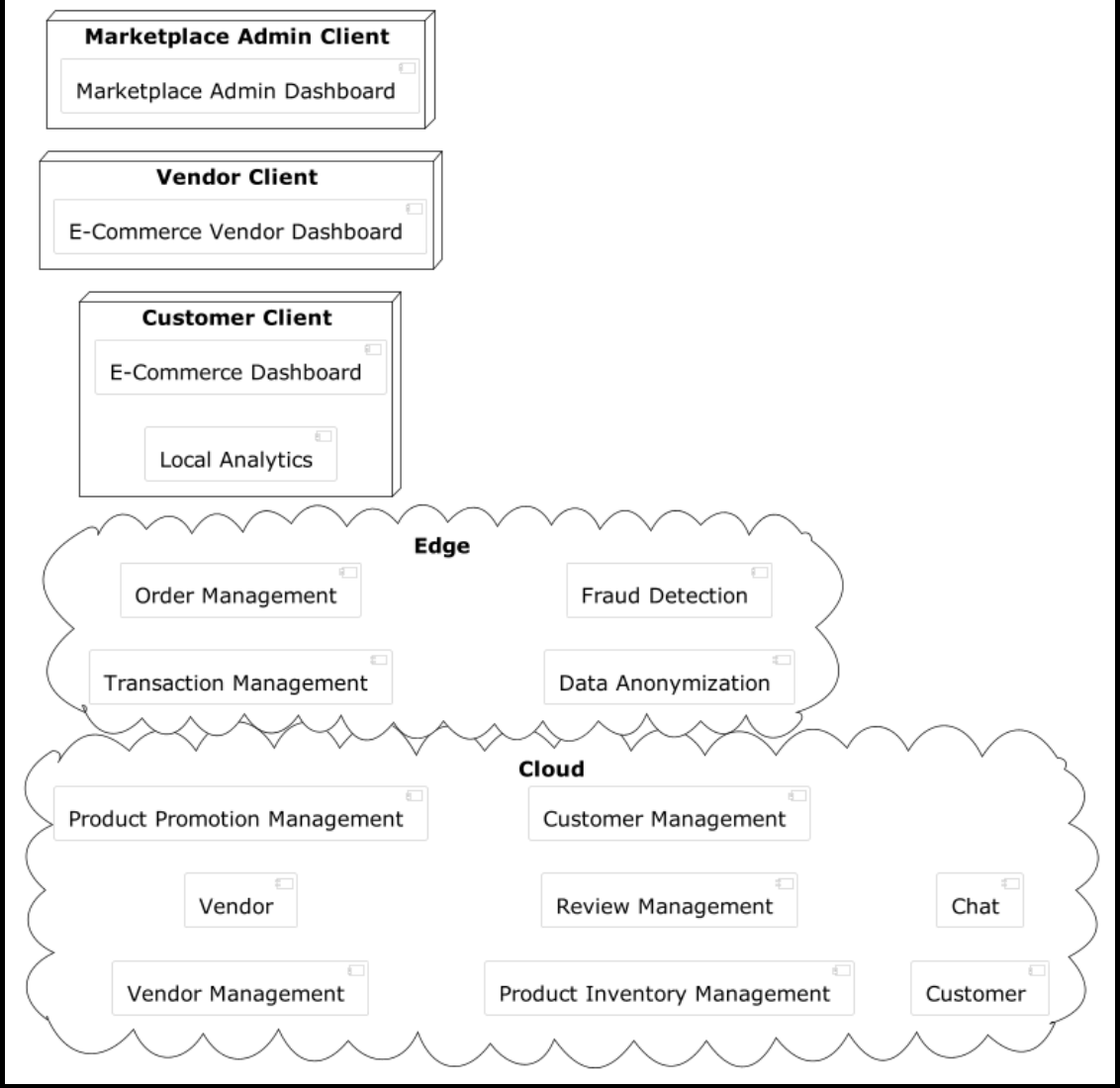
Making Use of Edge Computing

- Every millisecond counts when it comes to conversions
  - Joint research of Google and Deloitte: *“milliseconds make millions”*
- Targeted ads and personalized recommendations
  - Collection and processing of user data in real time
- Complying wuth local data privacy regulations
  - Data is processed as close to the user as possible

Making Use of AI Components

- Product search with image recognition and object detection
  - Users can browse listings based on images and descriptions
- Detecting fraudulent activities
  - Fake accounts, reviews, transactions can be detected in real-time
- AI-based advertisements
  - Users are prompted with ads and recommendations relevant for them

Deployments



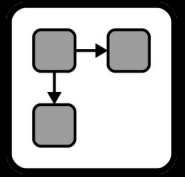
Péter Ferenc Gyarmati, 2022. 12. 15

Overview of the Identified Patterns

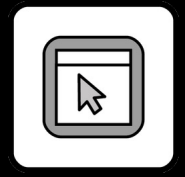
Public Cloud

CDN

Distributed Application



UI Component

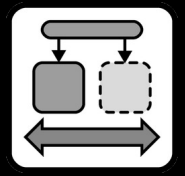


Data Access Component

Stateful Component

Exactly-once Delivery

Unpredictable Workload



Elastic Load Balancer

Key-value Storage

Relational Database

Péter Ferenc Gyarmati, 2022. 12. 15

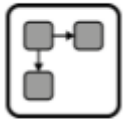
Cloud Service Models & Cloud Deployment Types

Component(s)	Pattern	Description
System-level		**Public Cloud:** *IT resources are provided as a service to a very large customer group in order to enable elastic use of a static resource pool.* Given that this E-Commerce platform should be available and scalable globally, the use of a public cloud is reasonable.

Native Cloud Applications

Component(s)	Pattern	Description
E-Commerce Dashboard		**Content Distribution Network:** *IT resources with a peaking utilization at reoccurring time intervals experience periodic workload.* The storefront UI needs to serve several media assets such as product previews, review images, etc. Using a CDN can help provide a better experience for users, by loading these assets quicker.

Fundamental Architecture Styles

Component(s)	Pattern	Description
System-level		**Distributed Application:** *A cloud application divides provided functionality among multiple application components that can be scaled out independently*. This pattern allows to break down the system into smaller, independent components that can be developed, tested, and deployed separately. This can make it easier to manage the development and maintenance of the E-Commerce platform, and allow to make changes and updates to individual components without affecting the entire system. This pattern can also improve the scalability of the platform and allow it to handle increased traffic and demand without performance degradation.

Application Components

Component(s)	Pattern	Description
E-Commerce Dashboard Marketplace Admin Dashboard		**User Interface Component:** *Customizable user interfaces are accessed by humans. Application internal interaction is realized asynchronously to ensure loose coupling.* The platform is made accessible to the end-users via easy-to-use, graphical user interfaces.
Order* Vendor* Transaction* Product Promotion* Product Inventory* Review* Customer* *Management		**Data Access Component:** *Access to data is handled by components that isolate complexity, enable additional consistency, and ensure adjustability of data elements.* Several components in the system need to store state in order to provide persistency and a consistent user experience. A data access component can be made responsible for communicating with the cloud-based data store, which contains the data for the products, customers, and orders in the ecommerce platform. It can expose a set of methods that can be used by the rest of the application to perform operations on the data, such as reading a product, updating a customer's address, or deleting an order. It abstracts the details of the cloud-based data store and the data access and manipulation logic, allowing the application to scale and evolve as the business grows and changes.
As above		**Stateful Component:** *Multiple instances of a scaled-out application component synchronize their internal state to provide a unified behavior.* Several components in the system need to store state in order to provide persistency and a consistent user experience. The stateful component is responsible for managing the state of the application, which includes data such as the products, customers, and orders in the ecommerce platform.

Communication Offerings

Component(s)	Pattern	Description
Transaction* Order* Product Inventory* Review* *Management		**Exactly-once Delivery:** *The messaging system ensures that each message is delivered exactly once by filtering possible message duplicates automatically.* After a customer buys a product, a workflow is triggered, notifying both the customer and the vendor about the event. Events in the order-, product-inventory-, and review management components also trigger such workflows.

Application Workloads

Component(s)	Pattern	Description
System-level		**Unpredictable Workload:** *IT resources with a random and unforeseeable utilization over time experience unpredictable workload.* While most of the time the workload of such a platform can be predicted, there are edge cases which could lead to unusual usage peaks: if a vendor introduces a new product which is extremely hot, or reaches unusually high traffic through own marketing campaings, the platform's load can vary highly.

Application Management

Component(s)	Pattern	Description
System-level		**Elastic Load Balancer:** *The number of synchronous accesses is used to adjust the number of required application component instances.* To serve users seamlessly, even when there is a peak in the number of visitors, new application instances might need to be provisioned automatically. A load balancer can assist in fulfilling this requirement.

Storage Offerings

Component(s)	Pattern	Description
Order Management Product Inventory Management Review Management Customer Management		**Key-Value Storage:** *Semi-structured or unstructured data is stored with limited querying support but high-performance, availability, and flexibility.* A key-value storage system is used in the platform because it provides fast, efficient access to data. Since data is stored as key-value pairs, it can be retrieved quickly by using the key as an index. This is useful as we need to access large amounts of data quickly and efficiently in order to provide a smooth and responsive user experience, especially in the order, product inventory, review and customer management modules so that related requests can be served quickly.
System-level		**Relational Database:** *Data is structured according to a schema that is enforced during data manipulation and enables expressive queries of handled data.* A relational database system allows us to organize data in a structured, logical way, making it easier to manage and maintain the data. This is important for the proposed E-Commerce platform, as it is expected to have large amounts of data, such as products, customers, and orders, that need to be organized and managed efficiently. It also enforces data integrity and consistency, which can help ensure that the data in the system is accurate and reliable.

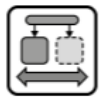
Péter Ferenc Gyarmati, 2022. 12. 15

Emphasis on Cloud Portability

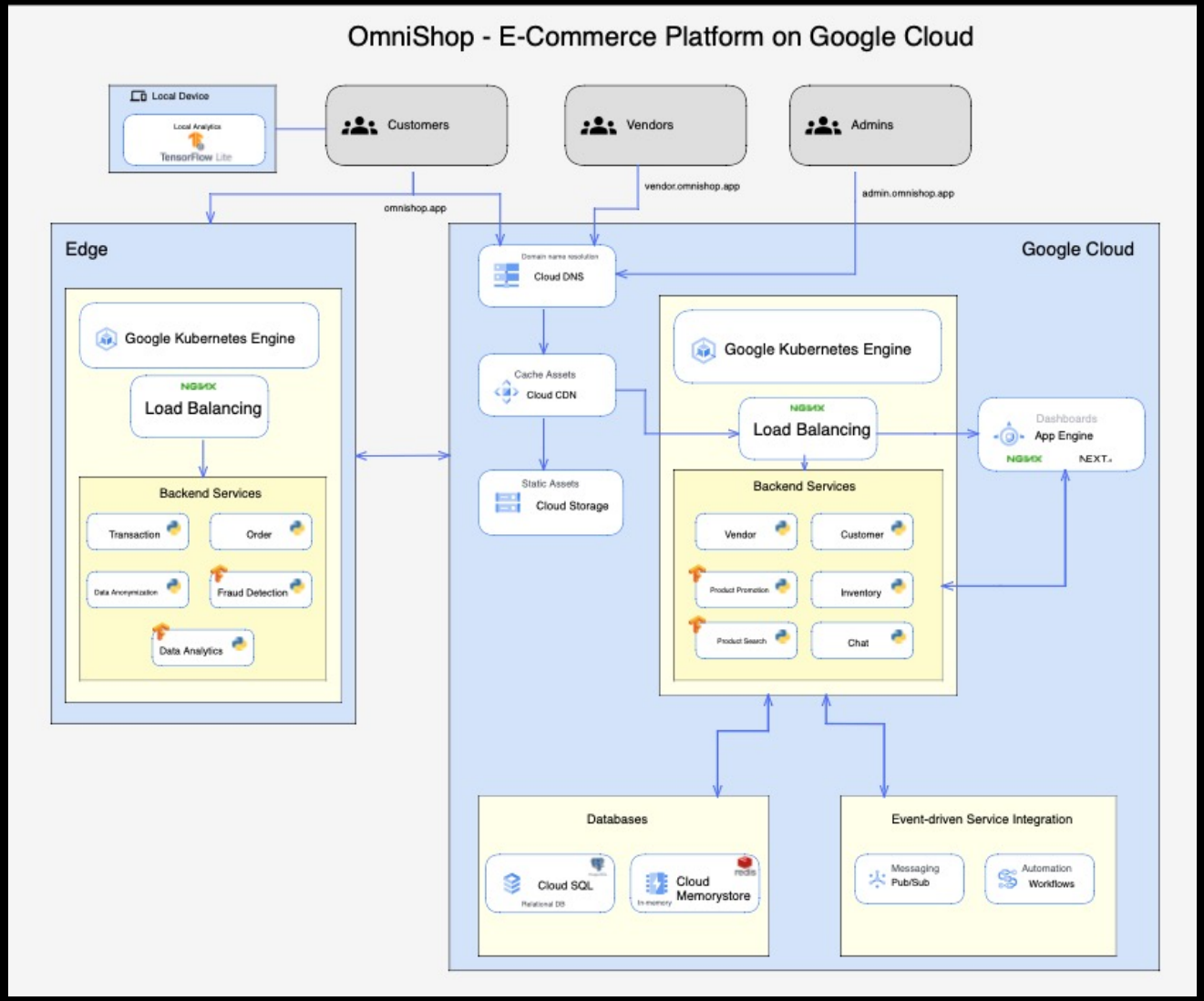
Designed system components based on Open Source packages

GCP provides only the infrastructure, the components could be hosted anywhere

Attempts to avoid vendor lock-in

Vendor Agnostic Pattern	Vendor Specific Pattern	Open Source Artifact(s)	Description
			**Public Cloud** ➡ **Google Cloud**
			**Cloud Distribution Network** ➡ **Cloud CDN**
			**Exactly-once Delivery** ➡ **Pub/Sub**
			**Key-value Storage** ➡ **Memorystore** Redis can be deployed to GCP's Memorystore offering as an Open Source artifact.
			**Relational Database** ➡ **Cloud SQL** PostgreSQL can be deployed to GCP's Cloud SQL offering as an Open Source artifact.
		 	**User Interface Component** ➡ **App Engine** nginx and Next.js can be deployed to GCP's App Engine offering as Open Source artifacts.
		 	**Elastic Load Balancer** ➡ **Google Kubernetes Engine - Ingress** Kubernetes and an nginx Ingress Controller can be deployed to GCP's Kubernetes Engine offering as Open Source artifacts. Please note that the Cloud Load Balancing offering of GCP would also be a viable choice here, however, since it would couple the E-Commerce platform more tightly to the cloud provider, the k8s-and nginx-based solution is preferred to enhance cloud portability.

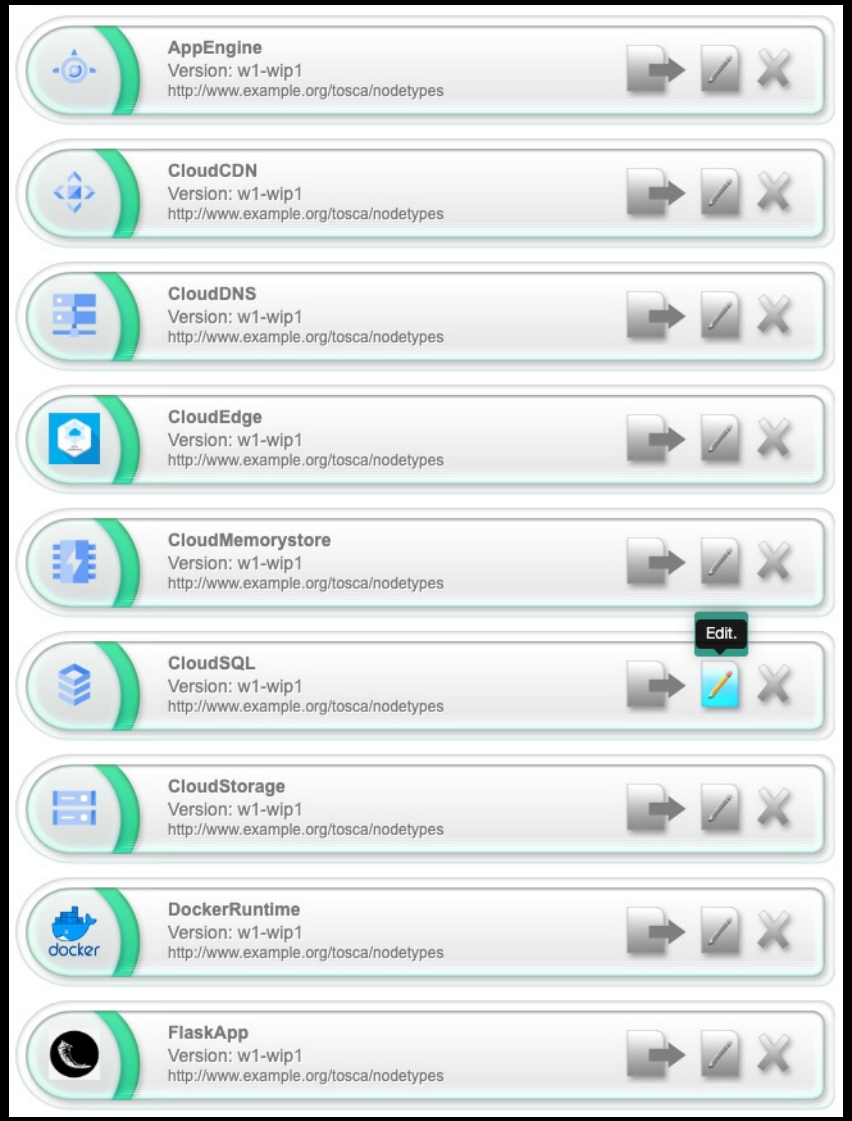
Vendor-specific Architecture

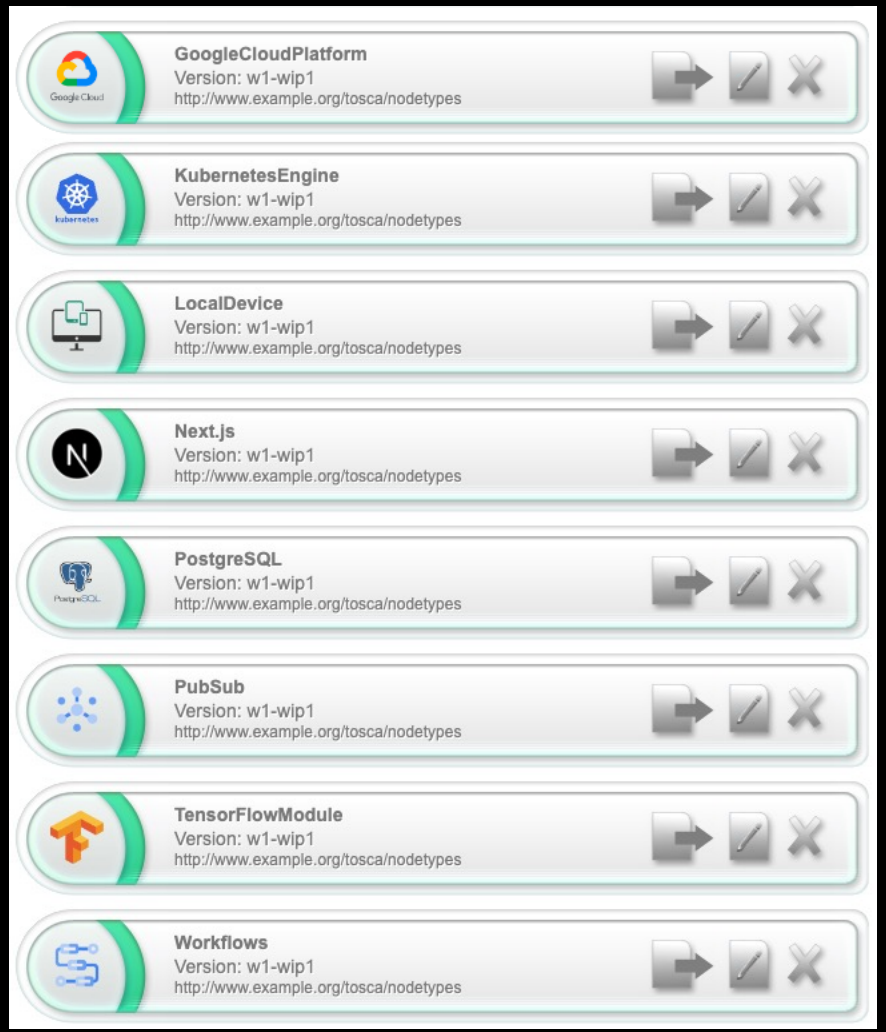


Modeling with OpenTOSCA

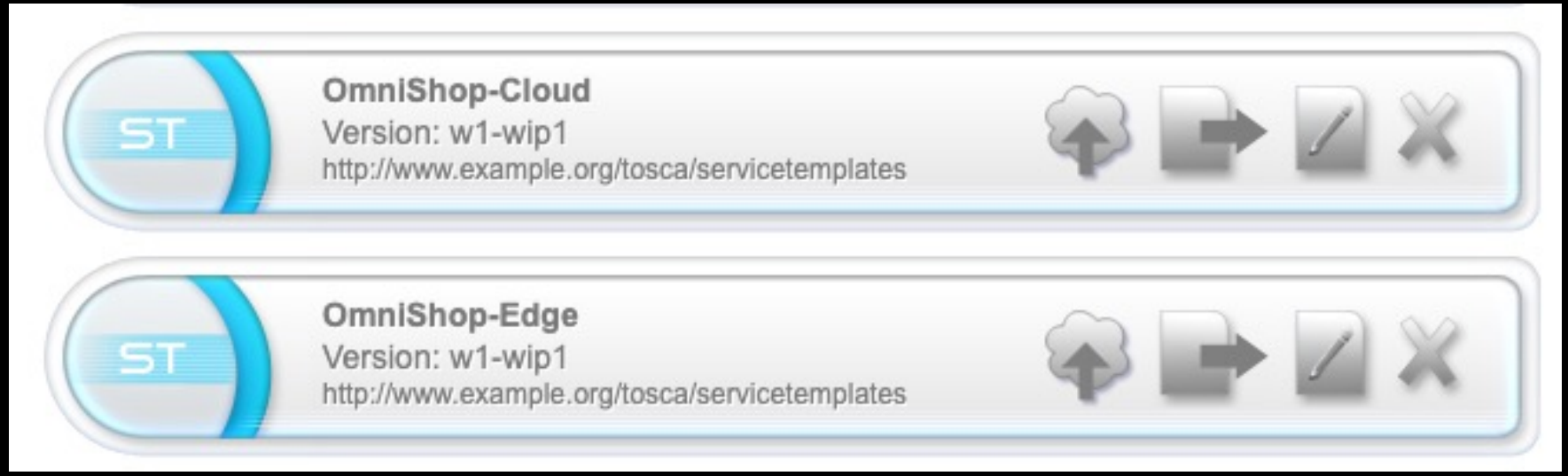
Péter Ferenc Gyarmati, 2022. 12. 15

Created OpenTOSCA Resources



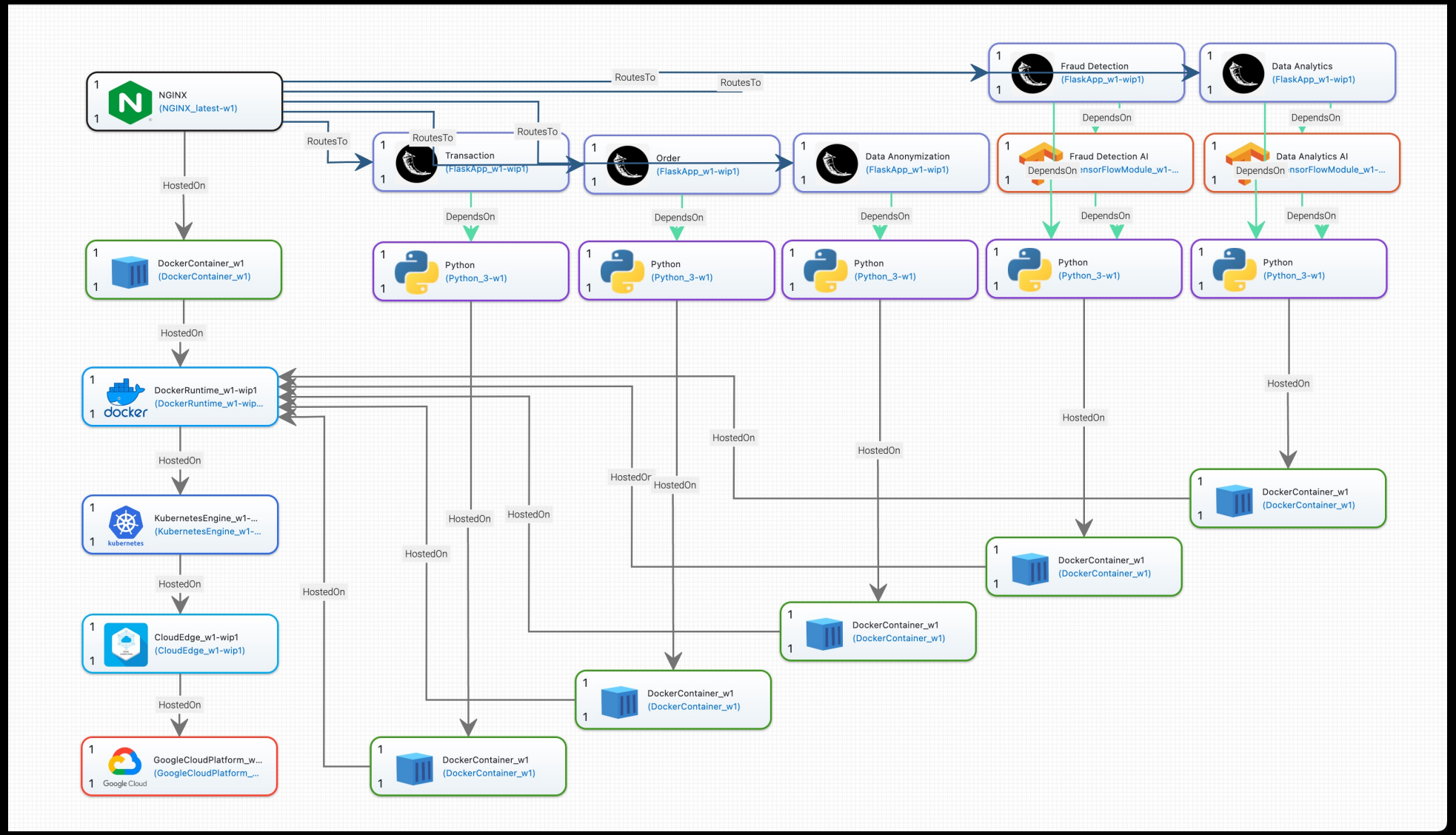


Created OpenTOSCA Resources



OpenTOSCA Topology

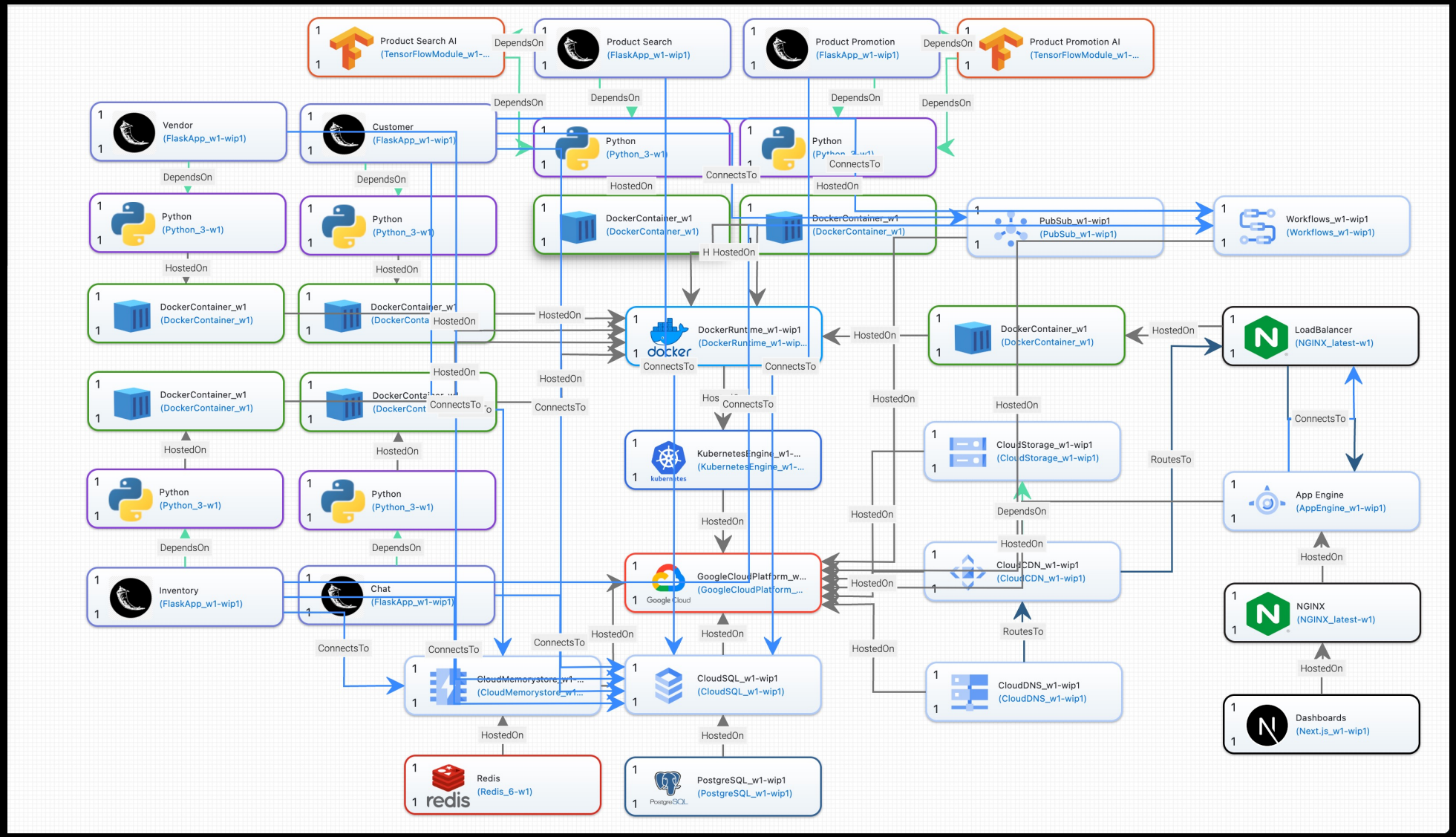
Edge



Péter Ferenc Gyarmati, 2022. 12. 15

OpenTOSCA Topology

Cloud



Péter Ferenc Gyarmati, 2022. 12. 15

Discussion

Thank you for your attention!

References

- *https://dub.sh/11913446-cloud-arch*
- https://miro.medium.com/max/7746/1*Iq5850WkFXq1BoDEKE2RDw.png