Cloud-Native Applications

Tessema Mengistu(Ph.D.)

mengistu@cs.vt.edu

Outline

- Cloud Native Applications: An Overview
- Application Architectures
    - Monolith
    - Microservices
    - Serverless
    - Event Driven Architecture

Cloud-Native Application

- Cloud native
  - Software approach of building, deploying, and managing modern applications in cloud computing environments.

  - Cloud Native Computing Foundation (CNCF)

    "Cloud-native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach."

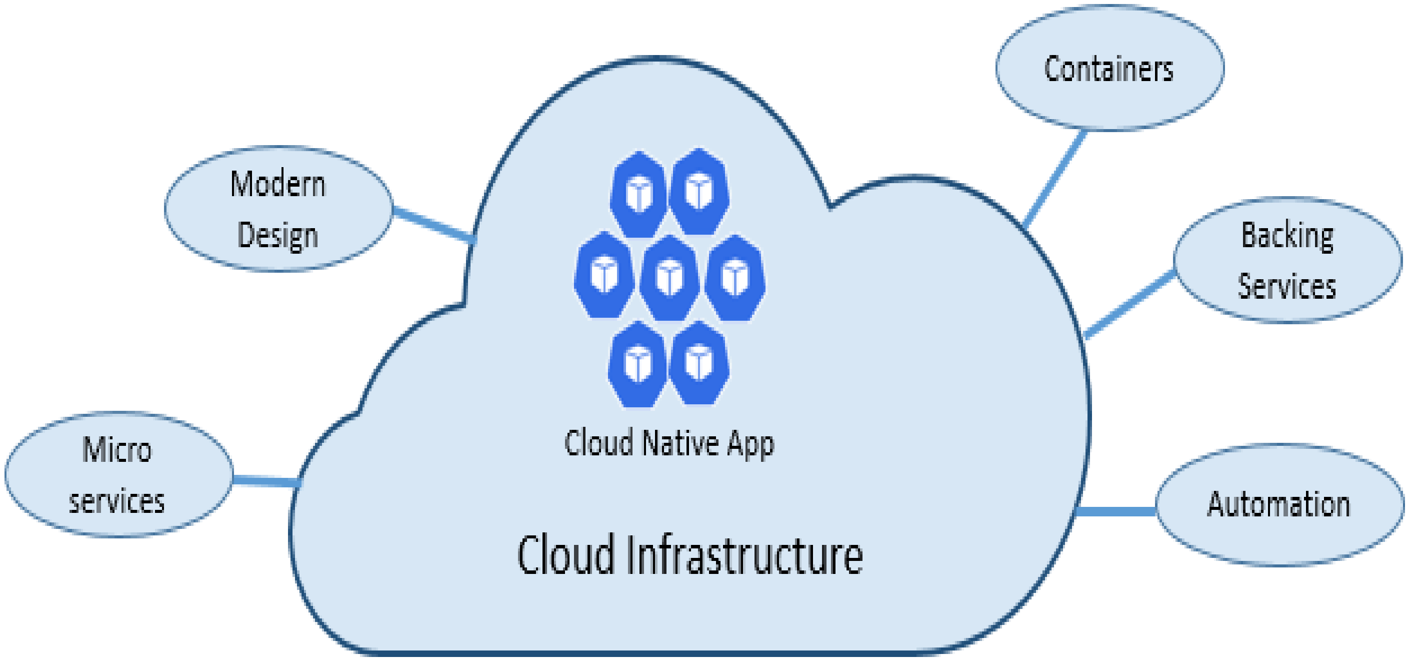
Cloud-Native Application

- Incorporates the concepts of:
  - DevOps
  - Continuous Integration and Continuous Delivery (CI/CD)
- Embraces modern architectures including **microservices** and **serverless**

Cloud-Native Application

- Advantages of cloud-native includes
  - Decrease time to market
  - Greater agility
  - Resilience
  - Portability
    - Suitable of hybrid and multi-cloud deployment

Cloud-Native Application



Cloud-native foundational pillars

Source: https://learn.microsoft.com/en-us/dotnet/architecture/cloud-native/definition

12 Factors for Cloud-Native Applications

- **Code base**
  - There is a single code base for each service
  - Tracked in version control systems
- **Dependencies**
  - Each service isolates and packages its own dependencies
- **Configurations**
  - Configuration separately stored through a configuration management tool
- **Backing Services**
  - Associated resources such as data stores, caches and message brokers should be exposed via an addressable URL
  - Decouples applications from resources

12 Factors for Cloud-Native Applications

- **Build, release, and run**
  - There has to be a strict separation between build, release, and run aspects
    - Facilitates easy rollback for an activity
- **Processes**
  - Each contributing services has to run on its own process
- **Port binding**
  - Every service functionality is exposed on its own port
- **Concurrency**
  - Multiple instances can be made to run via isolated containers to assure HA

12 Factors for Cloud-Native Applications

- **Disposability**
  - Maximize robustness with fast startup and graceful shutdown
- **Dev/prod parity**
  - It is recommended to minimize the difference between different environments such as development, testing, staging and production
- **Logging**
  - Treat logs as event streams
- **Admin processes**
  - Run administrative/management tasks as one-off process
    - Data clean-up and pulling analytics for a report

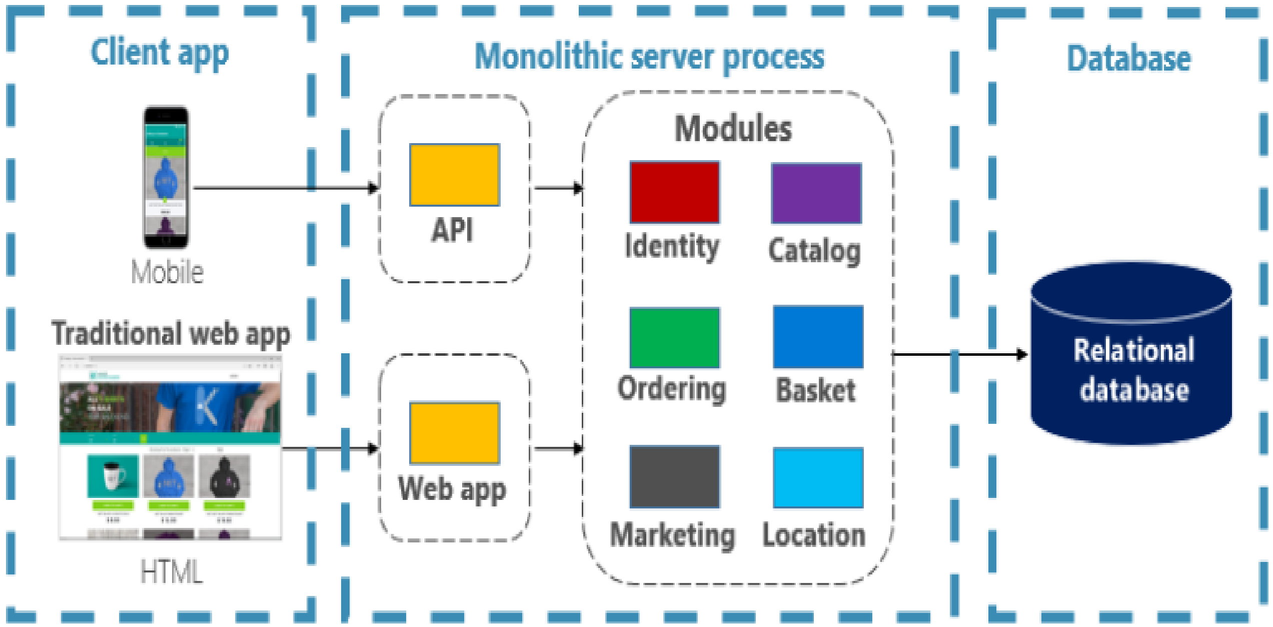
12 Factors for Cloud-Native Applications

- Additional factors
  - **API First**
    - Make everything a service
  - **Telemetry**
    - Your design includes the collection of monitoring, domain-specific, and health/system data
  - **Authentication/ Authorization**
    - Implement identity from the start

Application Architecture

- Monolith
  - Built as a single unit that runs all or most of the functions
  - Processes are tightly coupled
  - Advantages
    - Faster to design, develop, test and deploy
  - Disadvantages
    - Scalability
    - Reliability
    - Agility
    - Modifiability

Application Architectures



Source: https://learn.microsoft.com/en-us/dotnet/architecture/cloud-native/definition

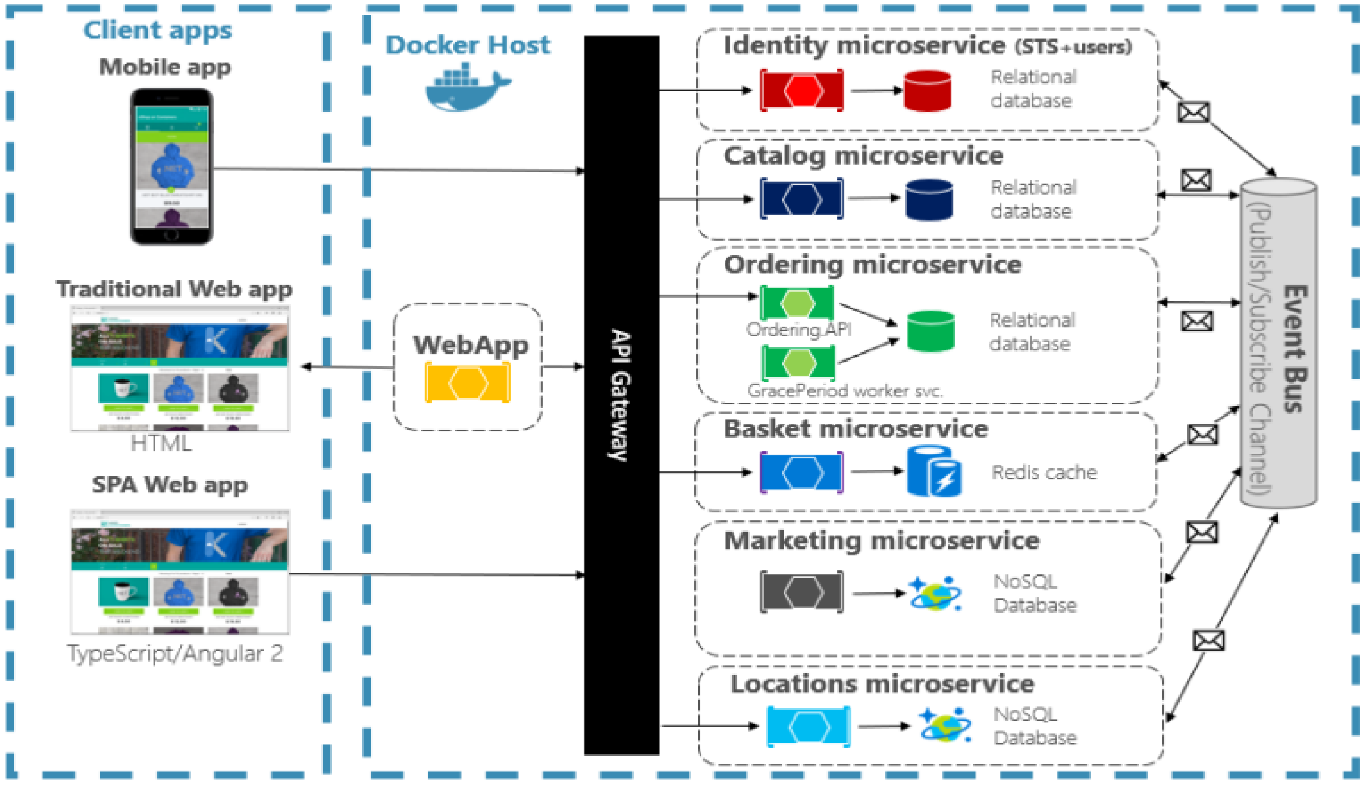
Application Architectures

- Microservices
  - Based on the principles of decentralization and modularity
  - Each microservice is built around a single business function and deployed independently
  - The most popular widely-applicable approaches to build highly-scalable modern applications
  - Common Characteristics
    - Independent and decentralized
    - Specialized services
    - Black box

Application Architectures

- Advantages
  - High reliability
  - Scalability
  - Faster time-to-market
  - Modifiable & Agile
  - Easier CI/CD
- Disadvantage
  - Complex
    - Development
    - Testing
  - Latency
    - Network communication
- Challenges
  - Data consistency
  - Security

Application Architectures



Source: https://learn.microsoft.com/en-us/dotnet/architecture/cloud-native/definition

Application Architectures

- **Serverless**
  - Applications are developed as a set of functions
  - No need to maintain server infrastructure, operating systems patching, etc.
    - A cloud-computing execution model
  - Handled by Cloud service provides (AWS, Google Cloud,…)
    - Automatic scaling
    - Built-in availability and fault tolerance
  - Advantage:
    - Reduced Total Cost of Ownership (TCO)
    - Faster time-to-market
  - Can provide:
    - Function as a Service
    - Back-end as a Service

Application Architectures

- Function as a Service (FaaS)
  - What drives serverless computing
  - Run a function with a set of resource constraints within a time limit
  - Could be triggered by an event or a schedule or even manually launched
  - FaaS offerings from the top cloud providers:
    - AWS Lambda
    - Google Cloud Functions
    - Azure Functions
    - IBM Cloud Functions

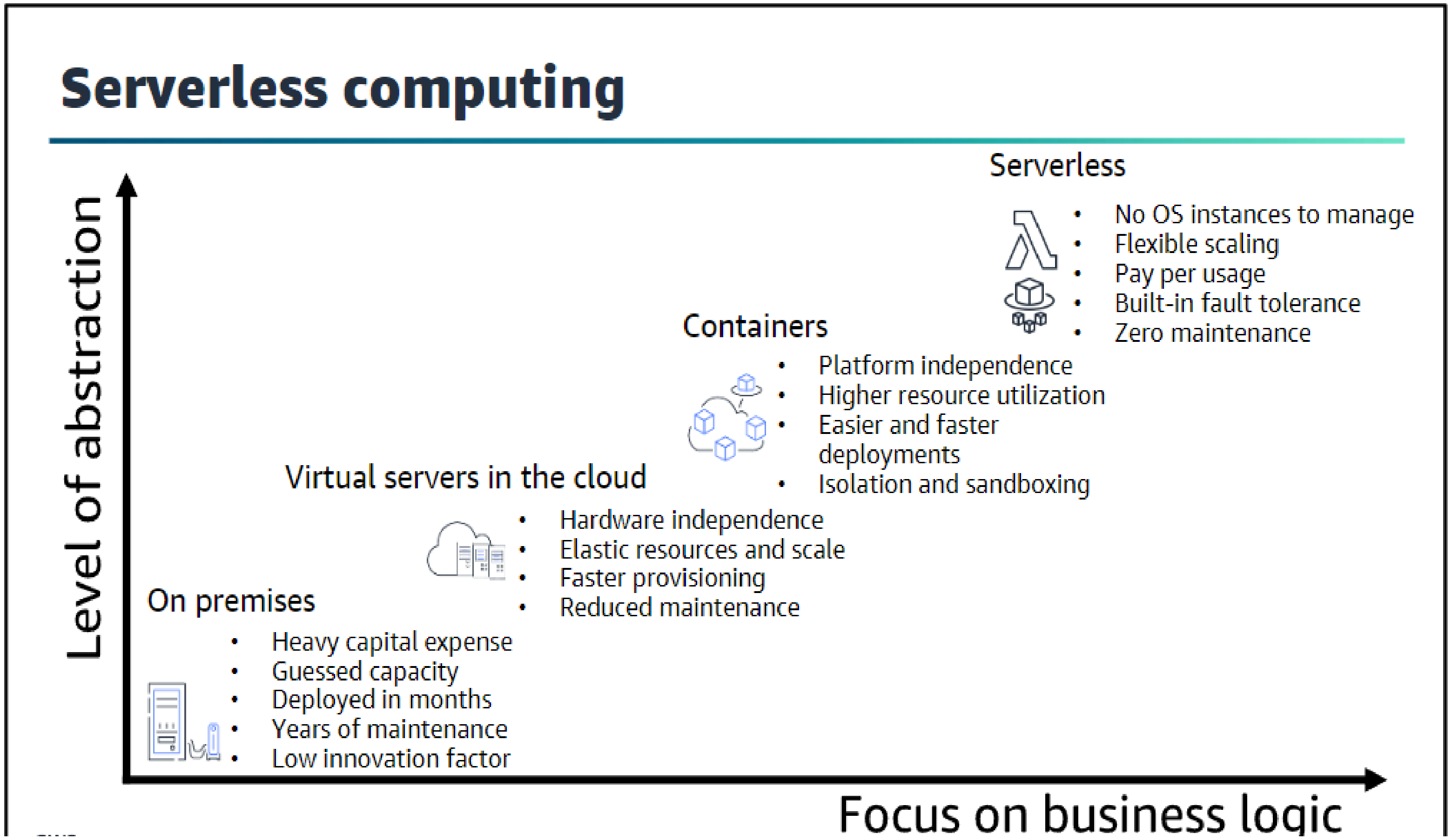
Application Architectures

- Back-end as a service(BaaS)
  - Provide pre-written software for activities that take place on servers
    - User authentication
    - Database management
    - Remote updating, and push notifications for mobile applications,
    - Cloud storage and hosting
  - Fully managed

Application Architectures

- Different serverless platforms available
  - AWS Lambda
  - Azure Functions
  - Google Cloud Function

Application Architectures



Serverless computing

Serverless
- No OS instances to manage
- Flexible scaling
- Pay per usage
- Built-in fault tolerance
- Zero maintenance

Containers
- Platform independence
- Higher resource utilization
- Easier and faster deployments
- Isolation and sandboxing

Virtual servers in the cloud
- Hardware independence
- Elastic resources and scale
- Faster provisioning
- Reduced maintenance

On premises
- Heavy capital expense
- Guessed capacity
- Deployed in months
- Years of maintenance
- Low innovation factor

Application Architectures

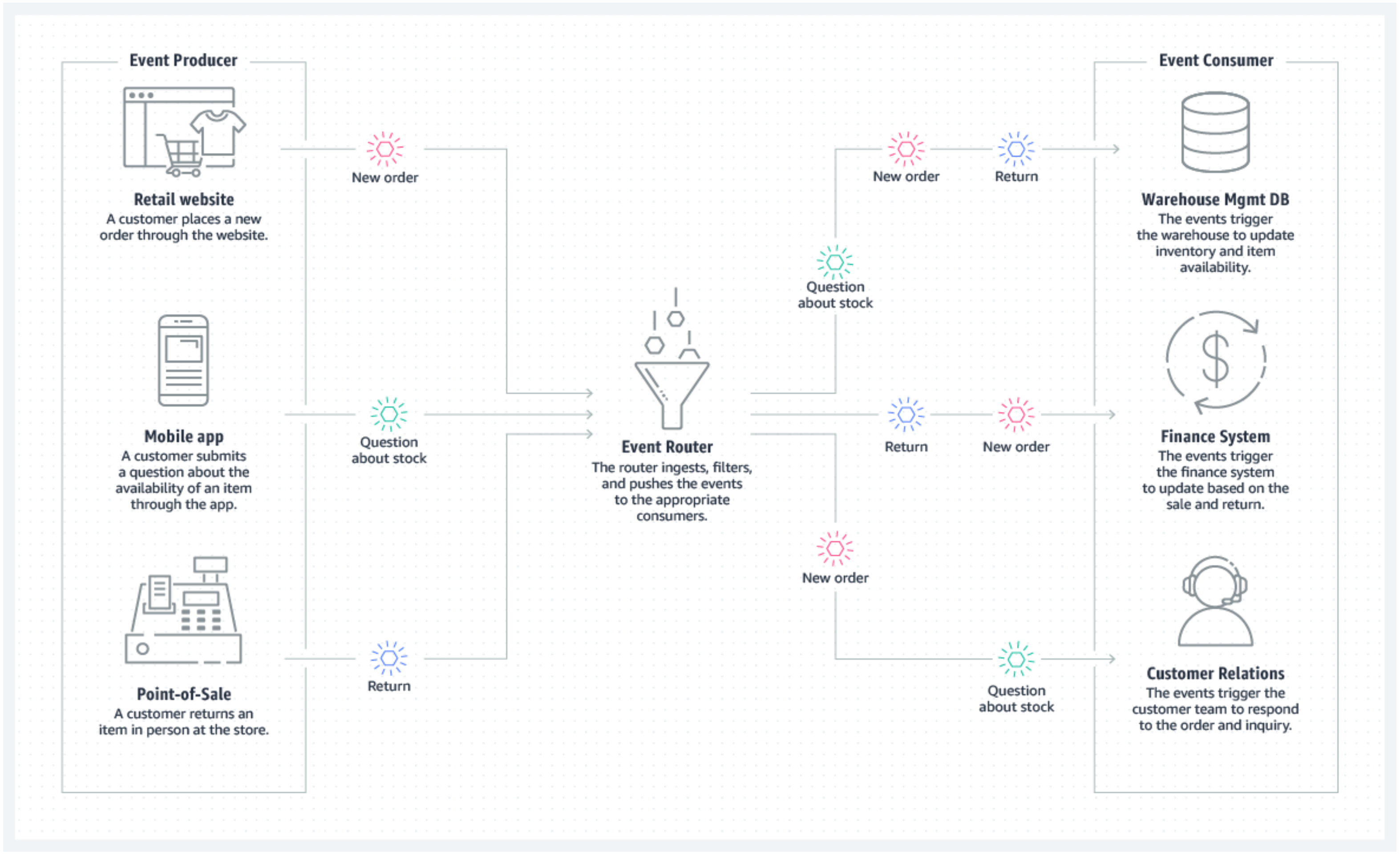
- Infrastructure as Code (IaC)
  - The ability to provision and support computing infrastructure using code instead of manual processes and settings
  - Describes a system architecture and how it works
    - Servers, networking, operating systems, and storage
  - Two approaches:
    - Declarative
    - Imperative
  - Advantages
    - Easy to replicate
    - Minimize configuration error
  - Example:
    - AWS CloudFormation, Azure Resource Manager, Google Cloud Deployment Manager, Ansible, Chef, …

Event-Driven Architecture

- Event-Driven Architecture (EDA)
  - An architectural pattern where capturing, processing, and storing events is the central theme
  - Event
    - The record of a significant occurrence or change that's been made to the state of a system
    - Immutable records and can be read and processed
  - Event-driven systems can respond to events as they are generated and take appropriate action

Event-driven architecture

- Components of an event-driven model:
  - Event producer
  - Event router (broker)
  - Event consumer
- Advantages
  - Fault tolerance
  - Real-time processing



Event Producer
Retail website
A customer places a new order through the website.
Mobile app
A customer submits a question about the availability of an item through the app.
Point-of-Sale
A customer returns an item in person at the store.
New order
Question about stock
Return
Event Router
The router ingests, filters, and pushes the events to the appropriate consumers.
New order
Return
Question about stock
Return
New order
New order
Question about stock
Event Consumer
Warehouse Mgmt DB
The events trigger the warehouse to update inventory and item availability.
Finance System
The events trigger the finance system to update based on the sale and return.
Customer Relations
The events trigger the customer team to respond to the order and inquiry.

Source: https://aws.amazon.com/event-driven-architecture/

References

- Architecting Cloud Native .NET Applications for Azure.
  - https://learn.microsoft.com/en-us/dotnet/architecture/cloud-native/
- https://aws.amazon.com/what-is/iac/
- https://aws.amazon.com/serverless/
- https://www.redhat.com/en/topics/cloud-native-apps/what-is-serverless