Ensar

AZURE

Service Bus

Overview of Azure Service Bus

The Document would explore relevant Azure service bus documentation to provide essential insights about a competent solution for Messaging as a Service on Azure.

Ensar Solutions.

USA | INDIA

www.ensarsolutions.com

AZURE SERVICE BUS OVERVIEW

Azure Service Bus serves as a fully managedenterprise integration message broker, offering ahighly reliable cloud service through Microsoft Azure.This powerful tool ensures efficient message deliveryat scale, making it an essential component for bothcloud-native and hybrid applications. With its queuesand topics, Azure Service Bus enables asynchronous,decoupled communication to facilitate a smoothflow of information between applications.

Key Features of Azure Service Bus

The Azure Service Bus's primary feature is its Message Queues, providingFIFO (First In, First Out) capability coupled with message duplicationdetection. This feature ensures an orderly processing environmentwhere each message is processed only once and strictly in thesequence it was received. Consequently, it aids in managing tasks likeorder processing with utmost efficiency.

Another essential characteristic of Azure Service Bus is its Publish-Subscribe model, encapsulated in its Topics and Subscriptions.Implementing the publish-subscribe pattern, this feature allows one-to-many communication, all in a unidirectional flow, ensuring broadmessage dissemination.

The Dead-Lettering feature of Azure Service Bus offers robust errorhandling. When a message cannot be processed, the serviceautomatically relocates it to a separate queue, known as the dead-letter queue, facilitating further analysis and investigation.

Azure Service Bus's Scheduled Delivery capability ensures messagesbecome available for processing at a predetermined time. This featureallows for the systematic management of tasks and operations.

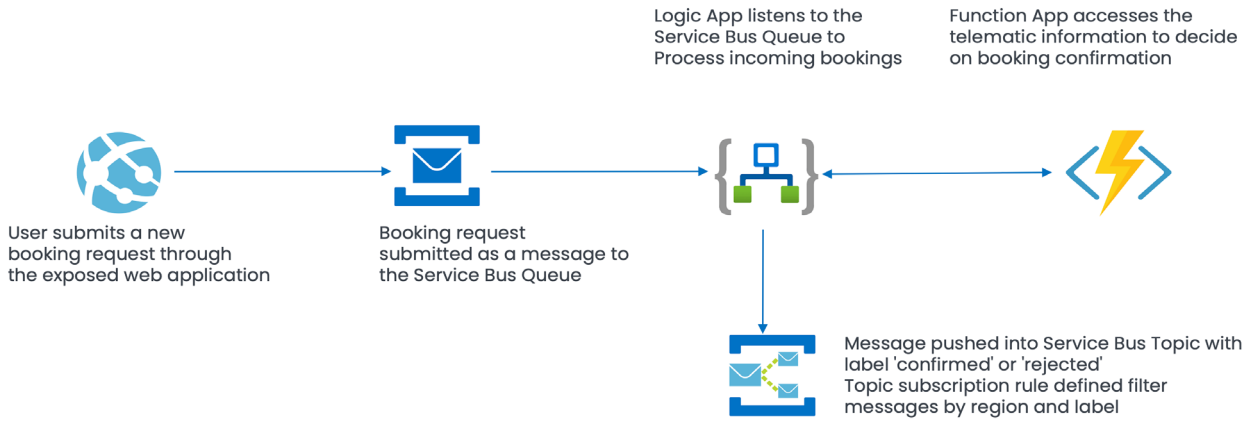
With its Batching feature, Azure Service Bus optimizes resource utilizationby minimizing the calls to Service Bus. It achieves this by batchingmultiple messages to be sent or received in one operation. federated security models and provides robust identity and access management features.

Transactions support in Azure Service Bus allows grouping of two ormore operations into a transaction that executes atomically, ensuringconsistency and reliability.

The Sessions feature provides an effective way to manage sequencesof related messages, thereby supporting workflows and interrelatedprocesses.

When it comes to Security and Compliance, Azure Service Bus complieswith industry-standard protocols and regulations. It employs afederated security model and provides comprehensive identity andaccess management features, ensuring that your data is protected.

Lastly, the Auto-forwarding feature offers advanced routing capabilities.This allows chaining of queues and topics to form intricate routingpaths, facilitating more complex communication structures.



COMPARE AZURE MESSAGING SERVICES

Microsoft Azure offers several messaging services, each of them designed to cater to different use-cases. The main three messaging services are Azure Service Bus, Azure Event Hubs, and Azure Event Grid.

Azure Service Bus:
It's primarily used for message-oriented middleware applications for high-value enterprise messaging. It supports a set of advanced features like ordered delivery, transactions, duplicate detection, sessions for handling message sequences, etc. It's designed for scenarios where high reliability is critical.

Azure Event Hubs:
This is a big data streaming platform and event ingestion service. It can receive and process millions of events per second, which makes it suitable for real-time and telemetry data scenarios like IoT device telemetry, user experience or operational analytics, log aggregation, etc.

Azure Event Grid:
It's a fully managed event routing service. Event Grid uses a publish-subscribe model and is designed for high availability, consistent performance, and dynamic scalability. It's best for reactive programming, microservices architectures, serverless applications, and automation of operational tasks like file cleanup and batch jobs.

Here's a comparison based on various factors:

Key Factors	Azure Service Bus	Event Grid
Message Size	Azure Service Bus supports a maximum message size of256 KB (Standard tier) to 100 MB (Premium tier).	EventHubs supports a maximum message size of 256 KB, andEvent Grid supports event data of 64 KB per event.
Delivery Guarantees	Azure Service Bus guarantees first-in-first-out (FIFO), At-least-once, or At-most-once delivery.	Event Hubs ensuresAt-least-once delivery. Event Grid guarantees At-least-once delivery.

Here's a comparison based on various factors:

Key Factors	Azure Service Bus	Event Grid
Retention Period	Service Bus retains messages until they get processed.	Event Hubs can be configured to retain events for aduration from 1 to 7 days. Event Grid doesn't store eventsand retries delivery for up to 24 hours.
Ordering	Service Bus maintains the order of messages.	Event Hubsmaintain the order of events within a single partition. EventGrid does not guarantee ordering.
Pull vs Push	Service Bus is primarily a pull-based system (with anoptional push-based system)	Event Hubs is a pull-basedsystem, and Event Grid is a push-based system.
Volume	Service Bus is designed for lower volume of messages thanEvent Hubs but with more features.	Event Hubs is designedfor high volume like telemetry data. Event Grid is designedfor event-based scenarios and can handle high volume ofevents but with smaller size.

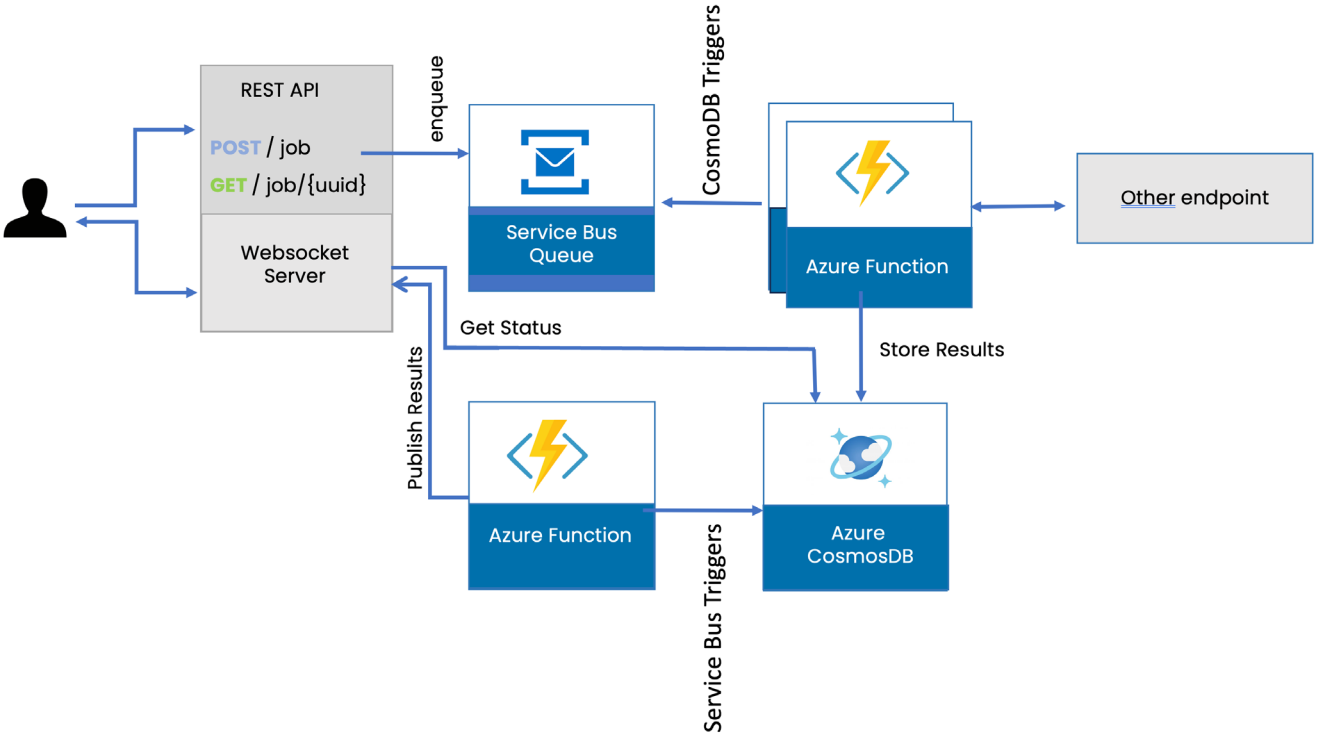
ARCHITECTURE

Azure Service Bus is a message broker system providing communicationcapabilities in the form of messages between various applications,services, and servers. The architecture of Azure Service Bus is centeredaround several key components:

At the core of the Azure Service Bus architecture is the Namespace, a container for all messagingcomponents. Each Namespace contains a set of queues, topics, relays, and event hubs, andserves as an application container and isolation boundary within the Service Bus. EachNamespace has a unique Fully Qualified Domain Name (FQDN), providing a distinct identificationwithin the Service Bus ecosystem.

Queues in Azure Service Bus facilitate a First In, First Out (FIFO) message delivery, enabling one ormore competing consumers to process messages in the order they were added to the queue. Thisarchitecture guarantees that each message is received and processed by only one receiver,thereby ensuring the integrity and sequence of the messages. Queues form the backbone ofpoint-to-point communication in the Service Bus.

Topics, on the other hand, cater to the publish/subscribe communication mechanism. A senderapplication sends a message to a topic, and any number of receiver applications withsubscriptions can receive that message. This many-to-one communication method is particularlyuseful when a single message needs to be received by multiple receivers, each with their uniquesettings.



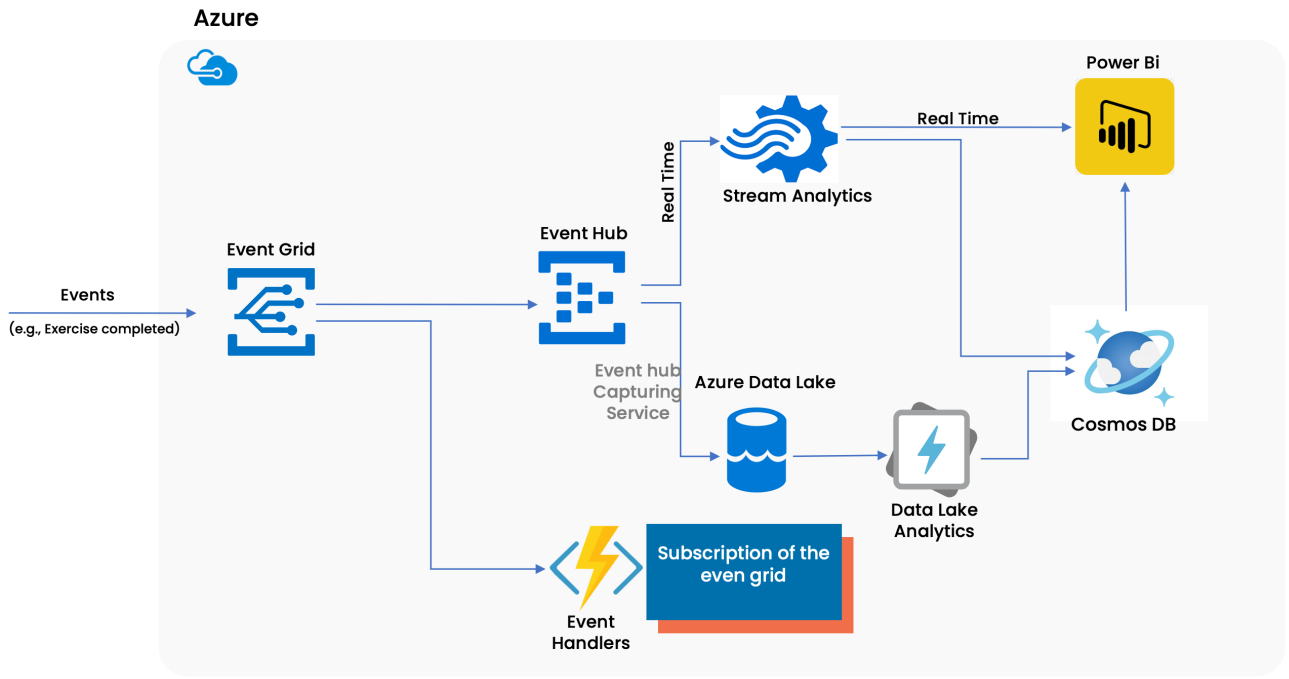
A subscription is associated with a specific topic, allowing for a more fine-tuned messagereception. Each subscription can use filters to receive only particular types of messages publishedto the topic, thereby allowing multiple subscribers to consume a message from a topic tailored totheir requirements.

Messages are the fundamental units of data exchanged between applications in Azure ServiceBus. A message consists of a header, containing standard and custom properties, and a bodythat contains the application-specific data. This structure allows for efficient message processingand data integrity.

Azure Service Bus Relays offer a unique feature allowing on-premises services to project publicendpoints, facilitating bi-directional communication without intrusive changes to the corporatenetwork infrastructure. They effectively bridge the gap between on-premises systems and thecloud.

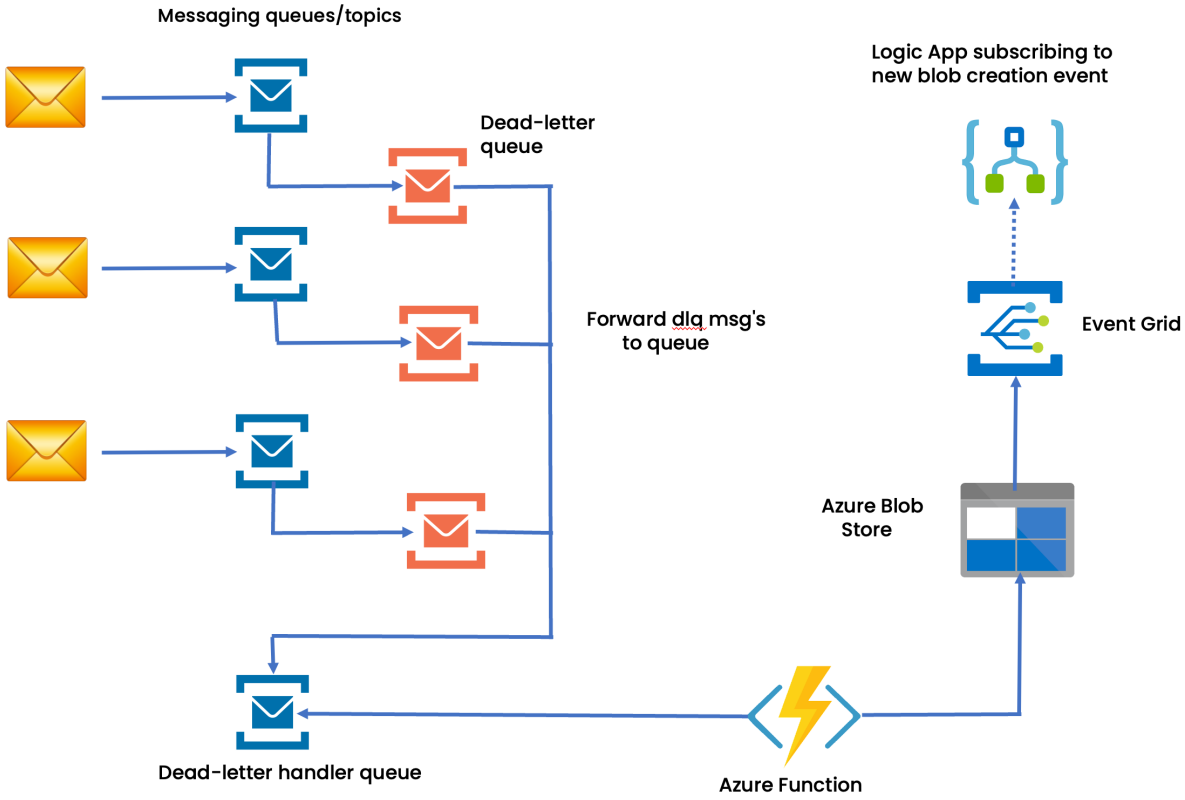
Event Hubs are part of the Azure messaging ecosystem, designed to handle event and telemetryingress at a massive scale. This is ideal for telemetry and event data scenarios where highthroughput is required.

he Azure Service Bus incorporates a robust security model. Access to queues, topics, andsubscriptions is controlled through Shared Access Signature (SAS) tokens, providing granularcontrol over permissions. This security model is also integrated with Azure Active Directory forrobust identity management.



The Service Bus architecture includes a feature for Duplication Detection, preventing theprocessing of duplicate messages sent within a particular time span. In scenarios wheremessages cannot be processed, a dead-lettering feature moves undeliverable messages toa separate dead-letter queue for later inspection. Moreover, to ensure consistency acrossoperations, the Service Bus supports transactions, allowing multiple operations to be groupedinto a single atomic operation.

In conclusion, the Azure Service Bus provides a comprehensive and reliable messaging brokerservice for integrating diverse applications. With its robust architecture that includesnamespaces, queues, topics, subscriptions, and a secure model, it stands as a pivotal tool indeveloping sophisticated cloud-native, hybrid, and on-premises applications.



CONCEPTS

Azure Service Bus is a managed service provided by Azure formessage delivery, and it includes several important concepts:

Namespace

This is the top-level container for messaging components. Thenamespace contains queues and topics. It provides an applicationcontainer and isolation boundary in the Service Bus. Each namespacehas a unique FQDN (Fully Qualified Domain Name).

Queue

Queues provide First In, First Out (FIFO) message delivery to one or morecompeting consumers. This means that messages are received andprocessed by the receivers in the order they were added to the queue,and each message is received by only one message receiver.

Topic

Topics are a publish/subscribe mechanism. While queues are suitablefor point-to-point communication, topics are useful in publish/subscribescenarios. A sender sends a message to a topic, and then any numberof subscriptions can be used to receive that message.

Subscription

Subscriptions are associated with a topic. Each subscription can use afilter to receive only specific types of messages published to a topic.Multiple subscribers can receive a message from a topic, each with theirown settings.

Namespace

This is the top-level container for messaging components. Thenamespace contains queues and topics. It provides an applicationcontainer and isolation boundary in the Service Bus. Each namespacehas a unique FQDN (Fully Qualified Domain Name).

Message

This is the data that is sent from the sender application and received bythe receiver application. It's composed of two parts, the header and thebody. The header contains standard and custom properties, while thebody contains the application-specific data.

Relays

Azure Service Bus Relays allow on-premises services to project publicendpoints. They facilitate bi-directional communication without requiringintrusive changes to the corporate network infrastructure.

Security

Azure Service Bus supports a secure model for access with SAS (SharedAccess Signatures) tokens. These tokens provide specific rights toqueues, topics, and subscriptions. This is integrated with Azure ActiveDirectory for identity management.

Event Hubs
They provide event and telemetry ingress at massive scale, as a part ofthe Azure messaging ecosystem.

Duplication Detection
The Service Bus provides a feature that removes duplicate messagesposted within a certain time span.

Dead-lettering
Undeliverable messages can be moved to a separate queue, known asthe dead-letter queue, where they can be inspected later.

Transactions
Service Bus supports grouping multiple operations into a single atomicoperation.

QUEUES VS TOPICS

Azure Service Bus provides two types of communication mechanisms: Queues andTopics. They serve different use cases and have distinct characteristics.

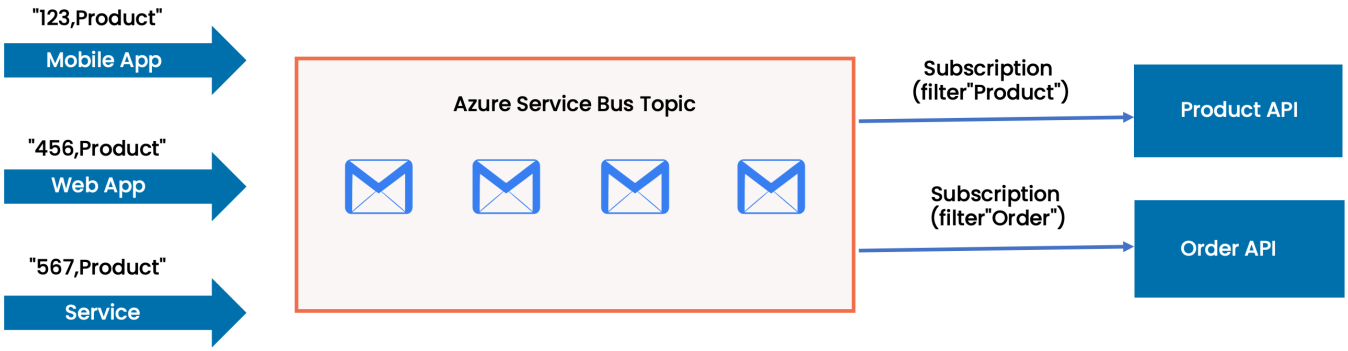
Azure Service Bus Queues

One-to-One Communication
Messages are processed in the order they are received. This is alsoknown as First-In-First-Out (FIFO) processing.

Order Guarantee
Messages are processed in the order they are received. This is alsoknown as First-In-First-Out (FIFO) processing.

Pull-based Communication
The receiver actively retrieves the message from the queue whenit's ready to process it.

Single Delivery
Each message is received and processed by only one messagereceiver.



Azure Service Bus Topics

One-to-One Communication

Topics are useful for publish/subscribe scenarios. Multiple receiverscan subscribe to a topic and receive a copy of the message.

Filters

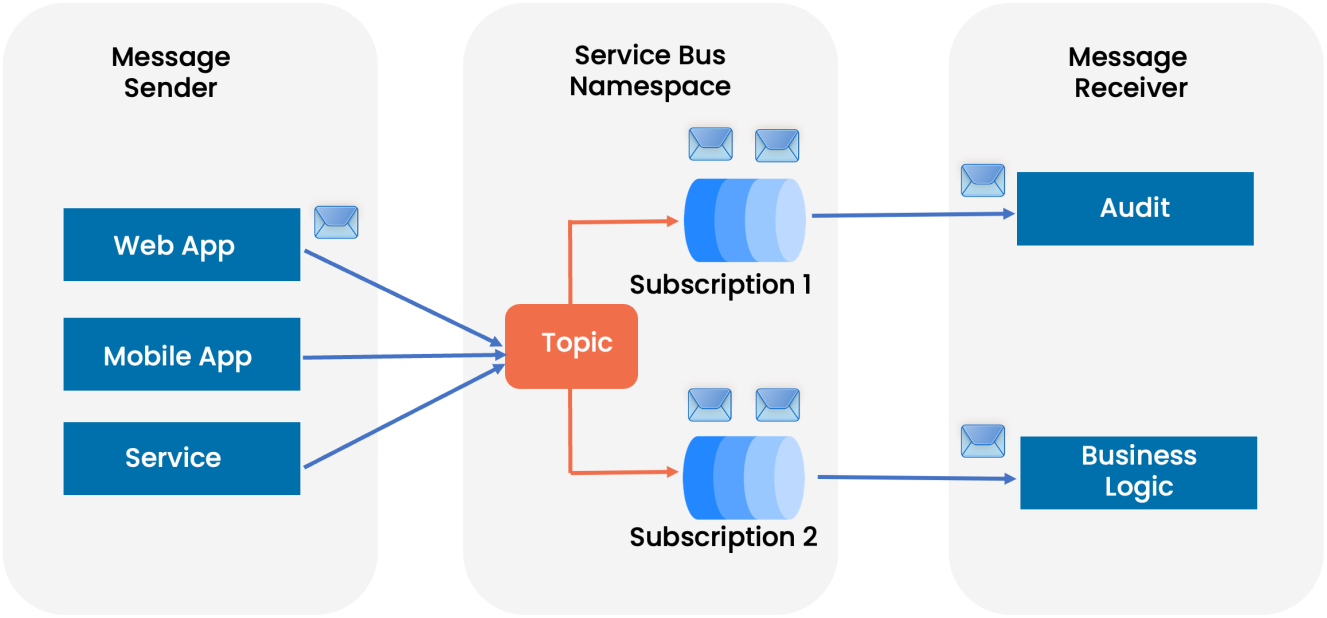
Subscriptions (receivers) can define filters based on the propertiesof incoming messages, ensuring that they only receive messagesof interest

Multiple Subscriptions

A single topic can have multiple subscriptions, allowing onemessage to be received by multiple independent receivers.

Single or Multiple Delivery

Depending on the number of subscriptions, the message isreceived and processed by one or many receivers.



MESSAGE PROPERTIES

Properties in Azure Service Bus provide metadata and control structures thatcan be used to handle and route messages. They are divided into threecategories: system properties, custom user properties, and message headers.

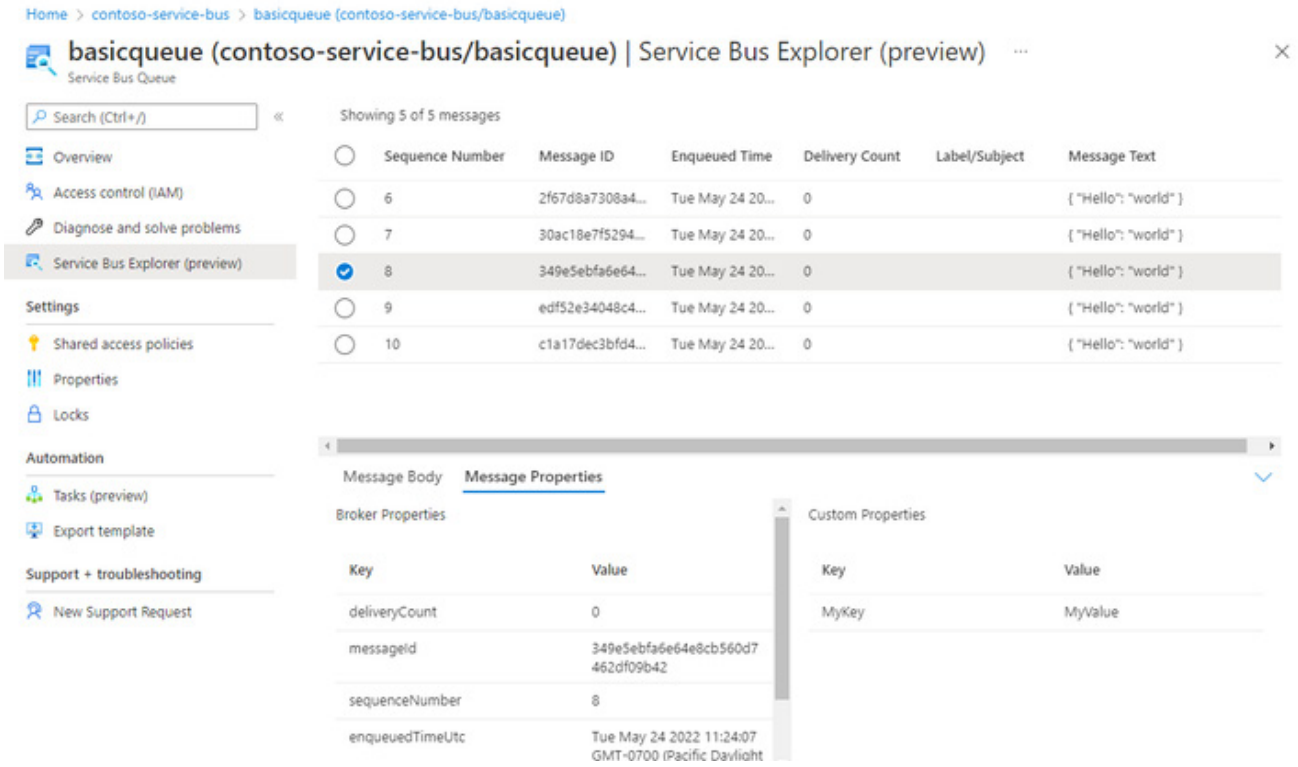
Azure Service Bus Queues

System Properties:
These are built-in properties that are set by Azure ServiceBus itself and provide information about the state of the message. They cannotbe changed. System properties include:

- **MessageId:** A unique identifier for the message.
- **SequenceNumber:** A unique identifier assigned by Service Bus when amessage is accepted and stored.
- **EnqueuedTimeUtc:** The time at which the message was stored in the queueor topic.
- **DeliveryCount:** The number of times this message was delivered to clients.
- **To:** The address of the queue to which the message is sent.
- **ReplyTo:** The address of the queue to which reply messages should be sent.
- **Label:** An application-specific identifier for filtering purposes.

Custom User Properties:
These properties can be set by the user for anypurpose. They provide a way to attach metadata to the message that can beused by the receiving application.

Message Headers:
These include standard HTTP headers such as ContentTypeand CorrelationId. The ContentType property is used to specify the type of thecontent, and the CorrelationId property can be used to group relatedmessages.



MESSAGE DETAILS

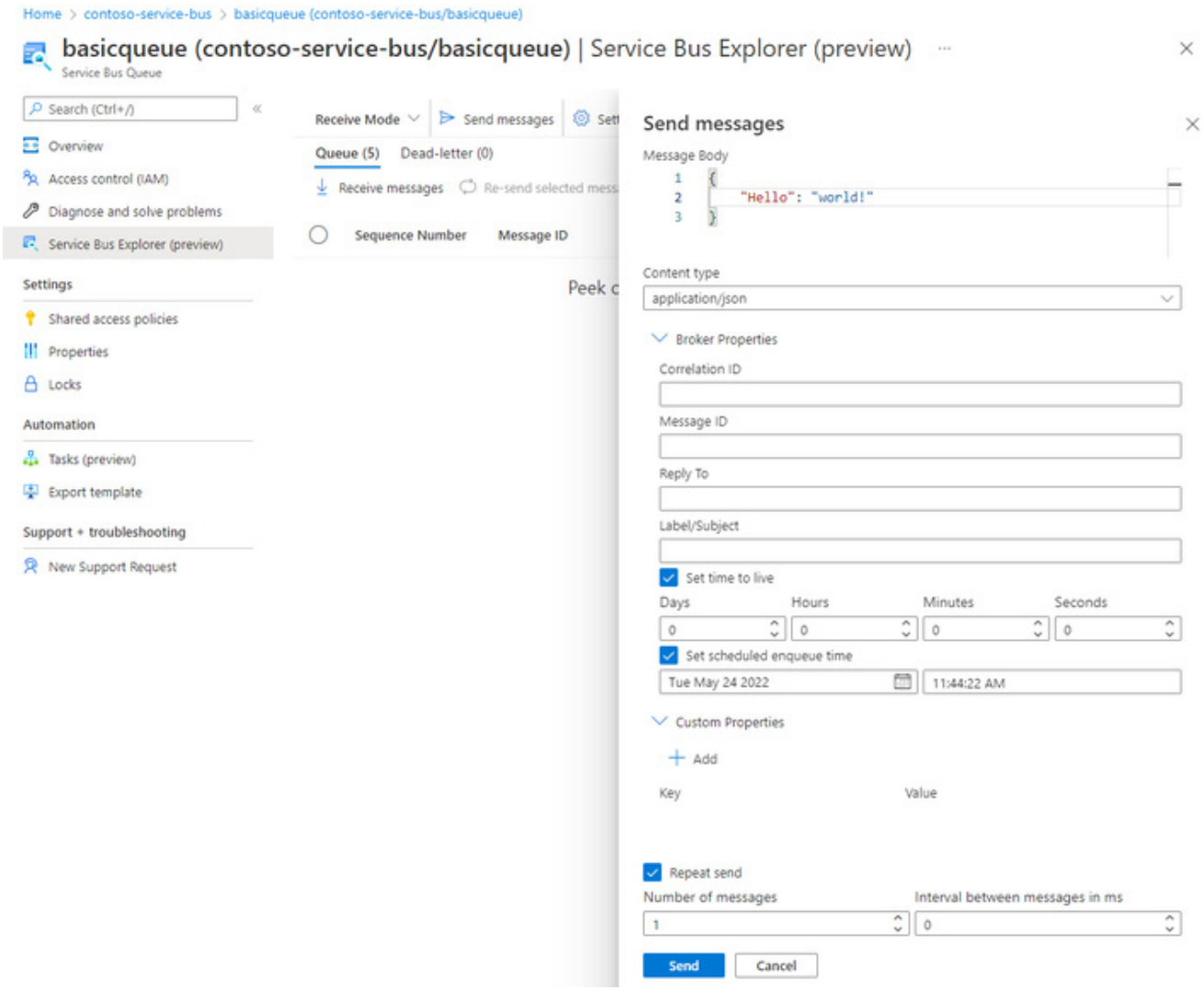
In Azure Service Bus, a message is a binary format of data that can be XML,JSON, text, or any other type of data that any service can understand. Itconsists of specific parts including body, properties, and headers.

Body:
This is the main part of the message that contains the application-specific data. The body format is application-specific and Service Bus does notinterpret this data.

Properties:
The properties are name-value pairs used to provide additionalinformation about the message. There are two types of properties:

- **System Properties:** These are built-in properties that provide informationabout the state of the message. They are set by Azure Service Bus andcannot be changed. Examples of system properties are MessageId,SequenceNumber, To, ReplyTo, Label, etc.
- **User Properties:** These are custom properties that can be defined by thesender application and can be used for any purpose. These properties canbe used by receivers for filtering and processing messages.

Headers: Headers carry information required for processing the message, suchas the ContentType and CorrelationId



METRICS

Azure Service Bus metrics provide insights into the behavior,performance, and health of your Service Bus namespaces, queues,topics, and subscriptions. These metrics are essential formonitoring your Service Bus resources and troubleshooting issues.

Here are some of the key Azure Service Bus metrics you should know:

Incoming Messages
The number of messages sent to a queue or topic over aspecific period.

Outgoing Messages
The number of messages retrieved from a queue orsubscription over a specific period.

Active Messages
The current number of messages in a queue or subscriptionthat are yet to be retrieved.

Dead-lettered Messages
The number of messages moved to the Dead Letter Queue.A high number of dead-lettered messages can indicate aproblem with the handling of these messages.

REQUESTS METRICS

Azure Service Bus provides metrics that let you monitoryour Service Bus namespaces at a granular level, givingyou insights into the behavior, performance, and health ofyour messaging system.

In terms of requests, here are some important metrics you should understand:

Successful Requests:
This metric measures the total numberof successful requests to your ServiceBus entities, including sending andreceiving messages, and managingresources.

Throttled Requests:
Azure Service Bus throttles requestswhen a threshold (like number ofconnections, size of entities, ormessage rate) is exceeded. ThrottledRequests counts the number ofrequests that were limited due to theseconstraints.

Server Errors:
This metric counts the number ofrequests resulting in HTTP 5xxresponses. This could indicate issueswith the Service Bus service itself.

User Errors:
This metric measures the total numberof requests resulting in HTTP 4xxresponses, indicating client-side errors.This could be caused by an attempt toretrieve a message from an emptyqueue, or a request for a non-existententity.

Requests Per Second:
This is an important performancemetric that indicates the number ofrequests (both successful and failed)made per second to your Service Busentities.

AZURE SERVICE BUS - APPLICATION TELEMETRY

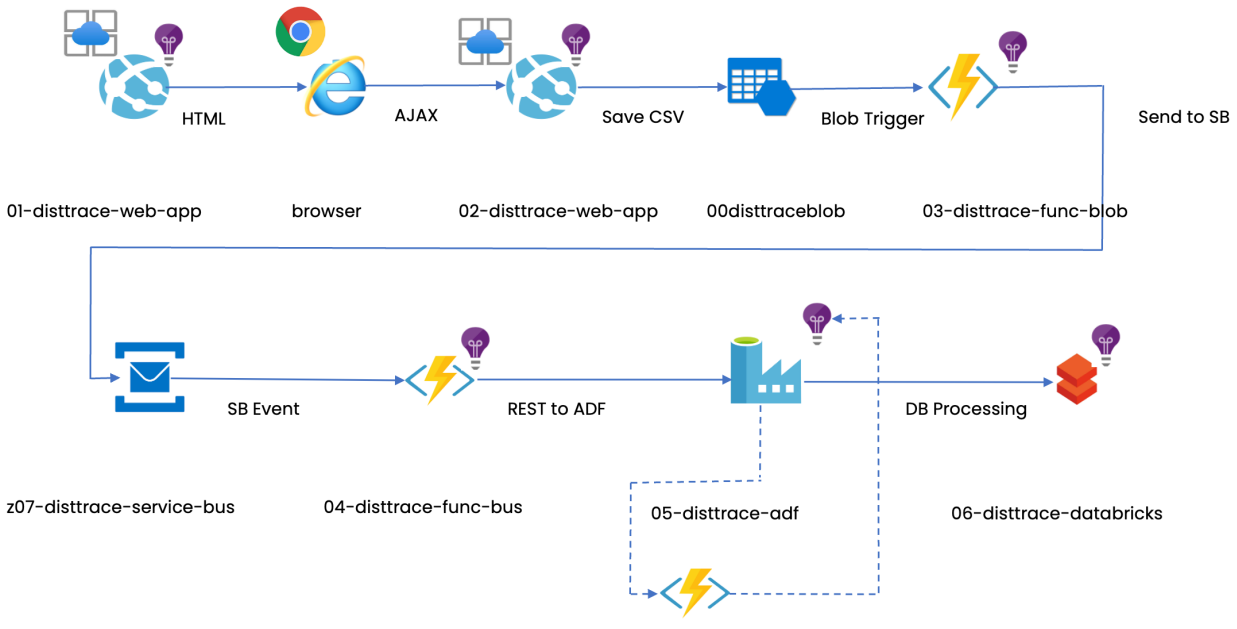
Azure Monitor:
This service allows you to collect, analyze, and act on telemetrydata from your Azure resources. With Azure Monitor, you can visualize real-timeperformance metrics of your Service Bus namespace, including metrics relatedto messages, requests, connections, and more. These metrics can be used tounderstand the behavior of your applications, identify bottlenecks, and troubleshoot issues.

Diagnostic Logs:
Azure Service Bus supports Azure Diagnostic Logs, whichprovide detailed logging of operations performed by Service Bus. These logscan be streamed to various endpoints for analysis, such as Azure Monitor logs,Azure Event Hubs, or Azure Storage.

Integration with Application Insights:
Application Insights is a feature of AzureMonitor and can be used to monitor your live applications. It automaticallydetects performance anomalies, and includes powerful analytics tools to helpyou diagnose issues and to understand what users actually do with your app.

Azure Monitor Alerts:
With Azure Monitor, you can set up alerts based onmetrics or logs, so you get notified when an anomaly occurs. For example, youmight want to be alerted when there are too many active messages in a queue,or when the rate of incoming requests exceeds a certain threshold.

Metrics Explorer:
Metrics Explorer in Azure Monitor can be used to visualize andquery the metrics collected from your Service Bus.



Tracing Architecture

REQUESTS METRICS

Azure Service Bus provides metrics that let you monitoryour Service Bus namespaces at a granular level, givingyou insights into the behavior, performance, and health ofyour messaging system.

In terms of requests, here are some important metrics you should understand:

Successful Requests:
This metric measures the total numberof successful requests to your ServiceBus entities, including sending andreceiving messages, and managingresources.

Throttled Requests:
Azure Service Bus throttles requestswhen a threshold (like number ofconnections, size of entities, ormessage rate) is exceeded. ThrottledRequests counts the number ofrequests that were limited due to theseconstraints.

Server Errors:
This metric counts the number ofrequests resulting in HTTP 5xxresponses. This could indicate issueswith the Service Bus service itself.

User Errors:
This metric measures the total numberof requests resulting in HTTP 4xxresponses, indicating client-side errors.This could be caused by an attempt toretrieve a message from an emptyqueue, or a request for a non-existententity.

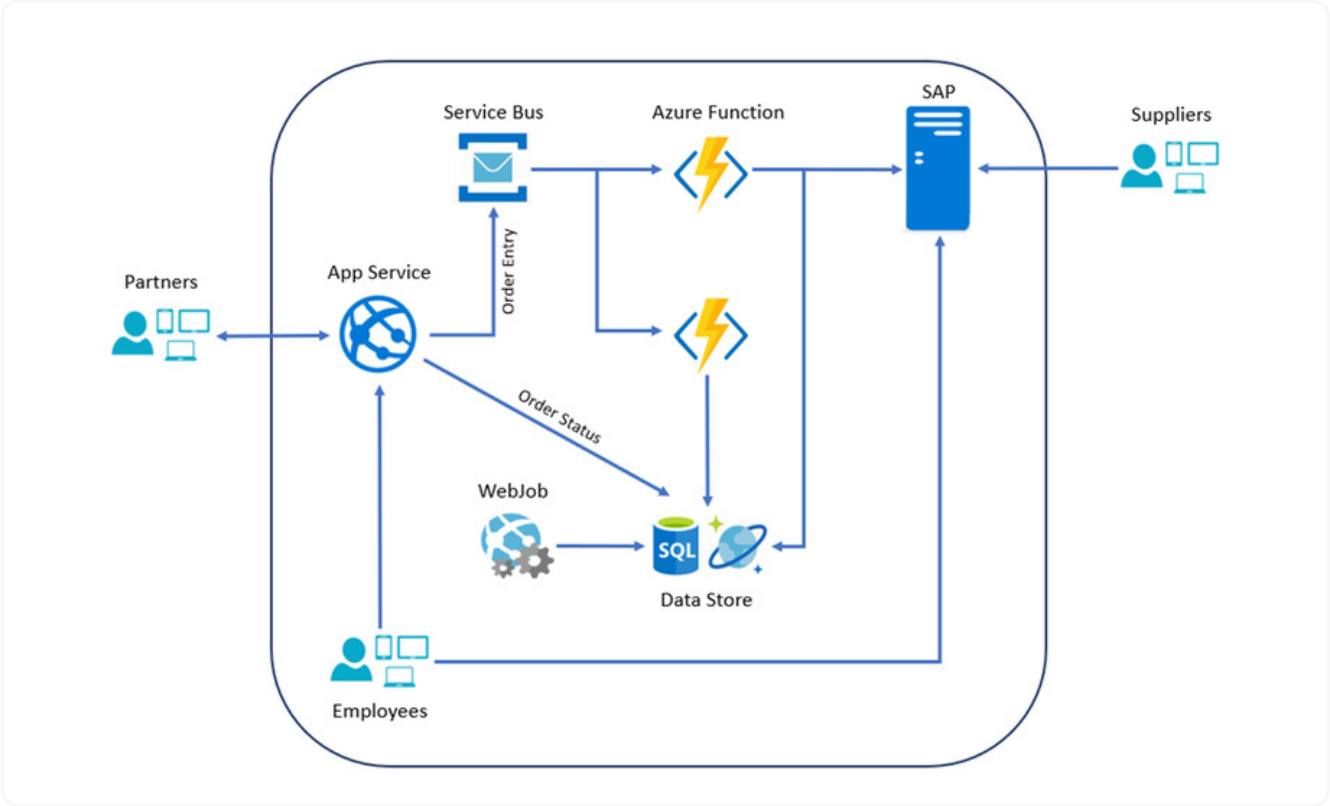
Requests Per Second:
This is an important performancemetric that indicates the number ofrequests (both successful and failed)made per second to your Service Busentities.

AZURE SERVICE BUS

USECASES

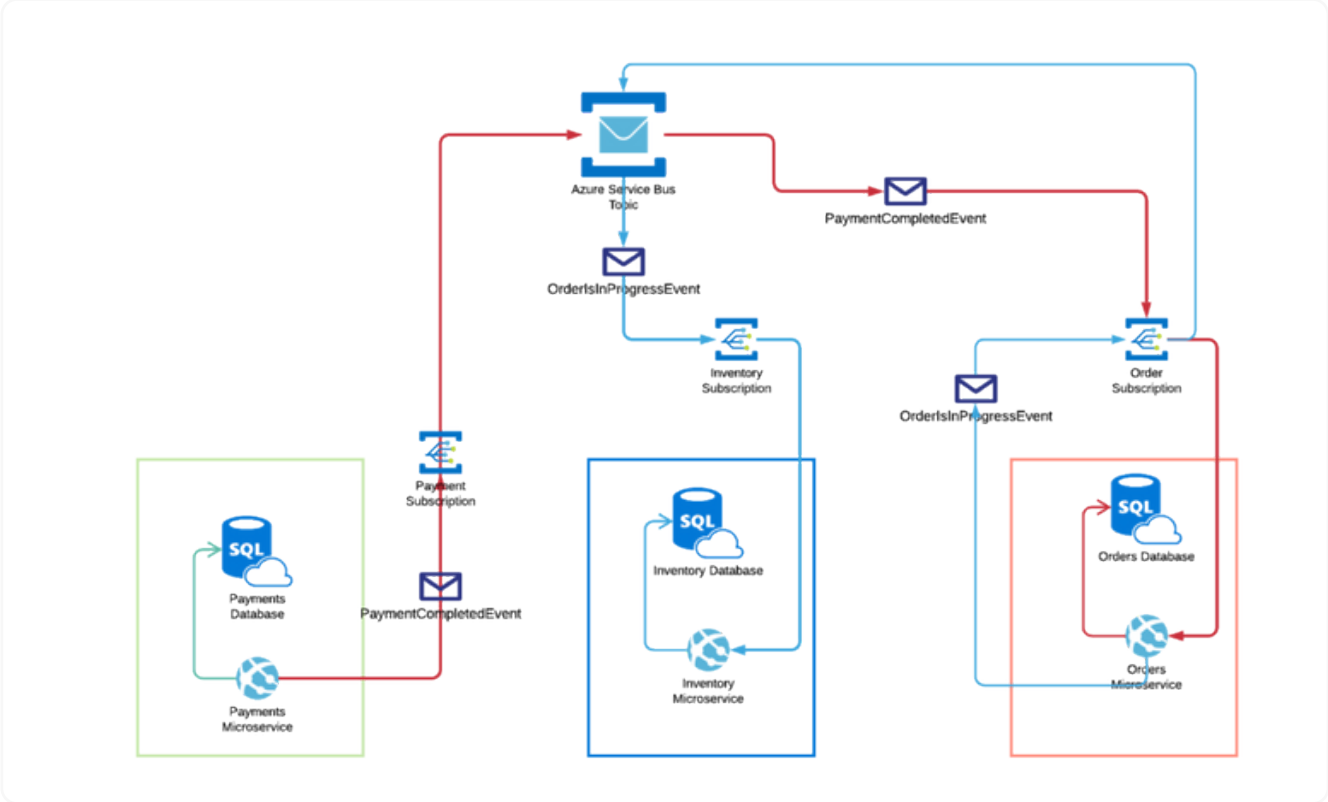
DECOUPLING SYSTEMS

Azure Service Bus can be used to decouple different parts of an application,allowing them to interact via messages. This can make the application morescalable and resilient.



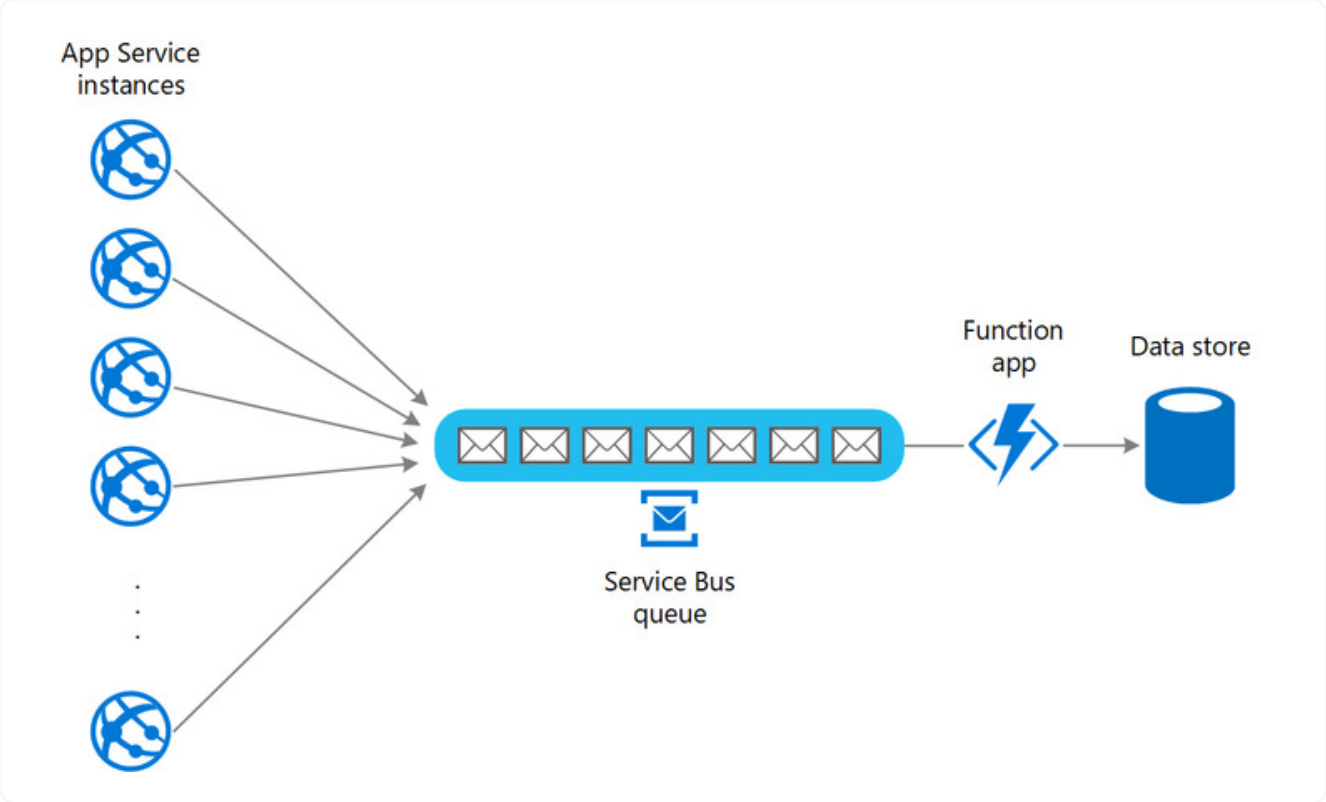
ASYNCHRONOUS MESSAGING

Azure Service Bus provides a reliable and durable platform for asynchronousmessaging, allowing tasks to be performed in the background or at a later time.



LOAD BALANCING

Messages sent to a Service Bus queue can be processed by multiple receivers. Thisallows for load balancing where workloads can be distributed across multipleprocessing units.



Ensar

Wheeling, IL
18 Schoenbeck Rd,
Sute#101, Wheeling,
Illinois, 60090

Naperville, IL
4132, Heartleaf Ln,
Naperville,
Illinois, 60564

Naperville, IL
50 South main street,
Suite#200, Naperville,
Illinois, 60540

India
MIG-64, KPHB 7th Phase,
Hyderabad - 500072
Telangana, India

Become Customer
sales@ensarsolutions.com

Join our Team
hr@ensarsolutions.com

Any Information
info@ensarsolutions.com