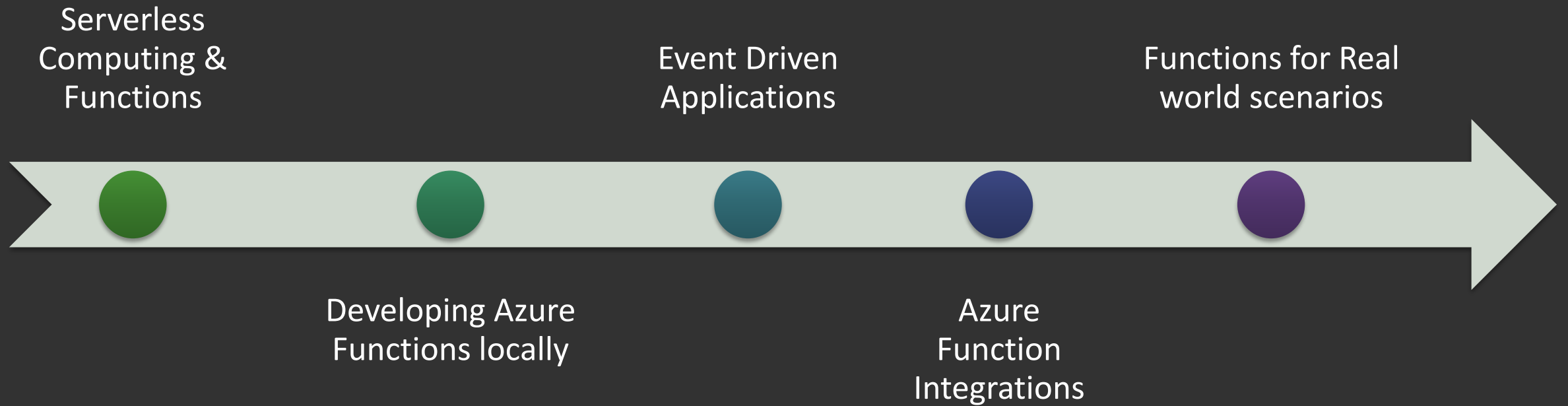






# Course Structure



# Points To Ponder

- Course Github Repository:  
<https://github.com/SainiVijayProgram/learn-serverless-functions>
- Download practice files & PDF to use as reference material
- Practice along with me. Don't simply sit and watch.
- Finish the course before start developing your own Azure Functions
- Review the course and provide your rating



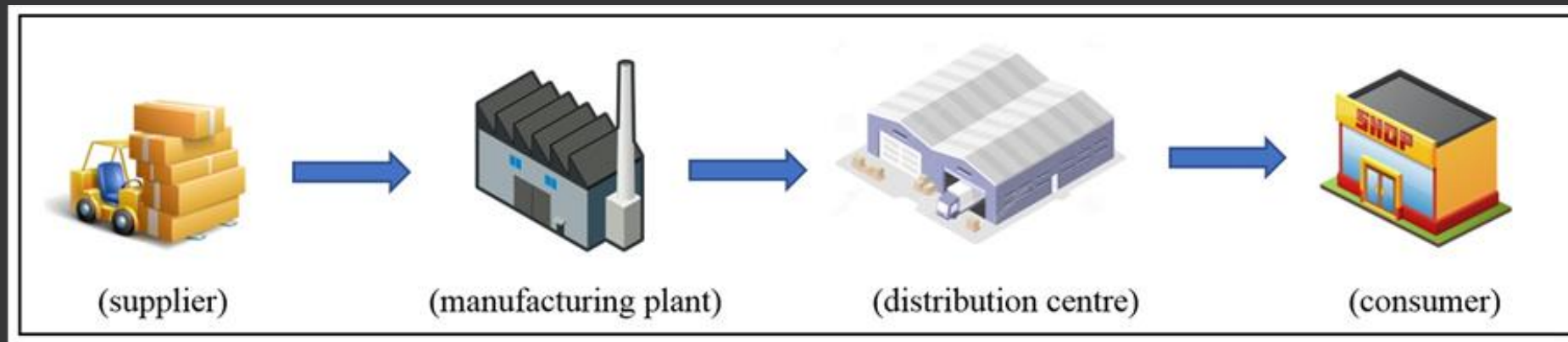
Thank You



# Serverless Computing

Vijay Saini







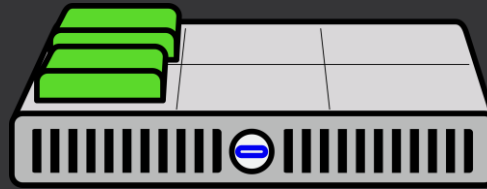
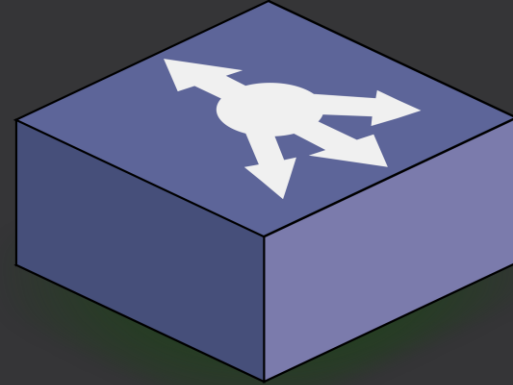
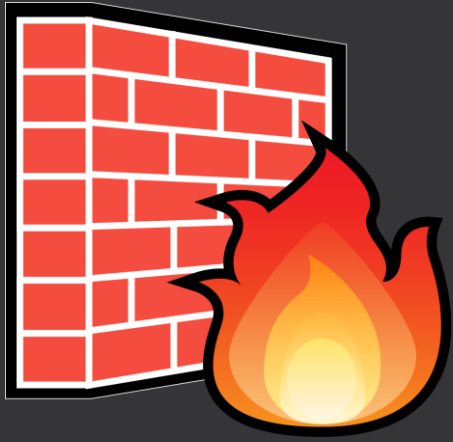
















# Azure Serverless Computing

# What is Serverless Computing

Serverless computing enables developers to build applications faster by eliminating the need for them to manage infrastructure.

With serverless applications, the cloud service provider automatically provisions, scales and manages the infrastructure required to run the code.



# What is Serverless Computing

Serverless functions accelerate development by using an event-driven model, with triggers that automatically execute code to respond to events and bindings to seamlessly integrate additional services.

A pay-per-execution model with sub-second billing charges only for the time and resources it takes to execute the code.





# Benefits of Serverless Computing

No infrastructure  
management

Faster time to  
market

Dynamic  
scalability

More efficient  
use of resources

# Azure Serverless Service

Service Categories:

<https://azure.microsoft.com/en-us/solutions/serverless/#solutions>



# Event-driven Architecture

Vijay Saini





# Event-driven Architecture

An event-driven application is a computer program that is written to respond to events.

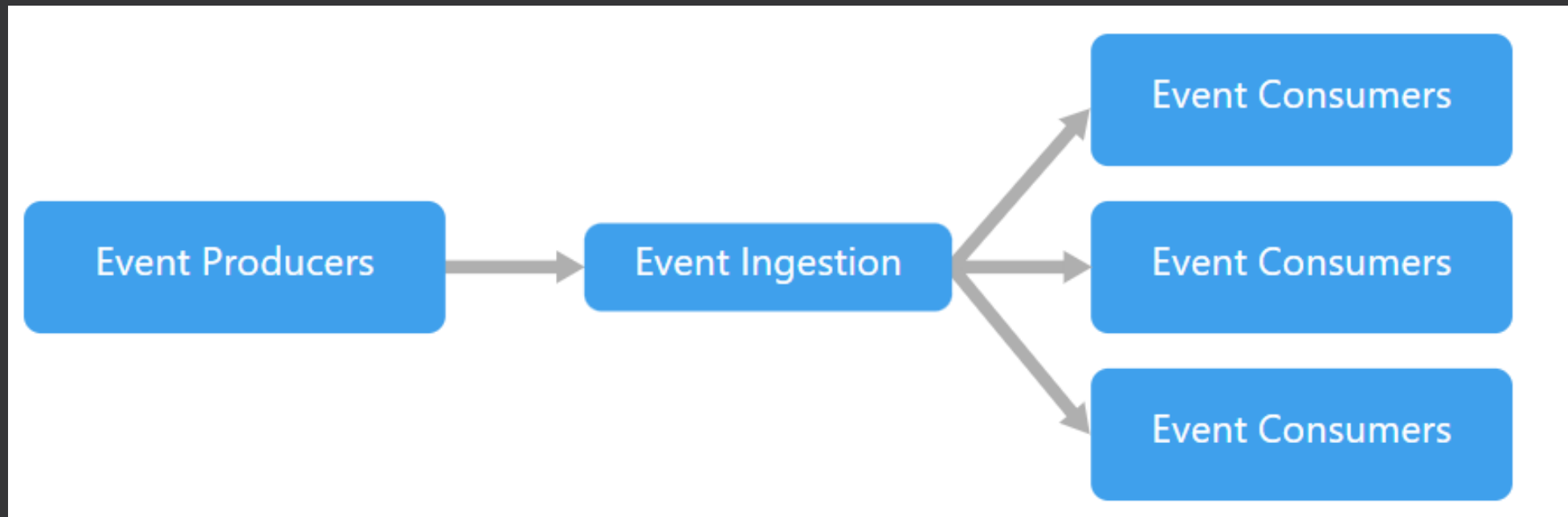
An event is any identifiable occurrence

## Few familiar events:

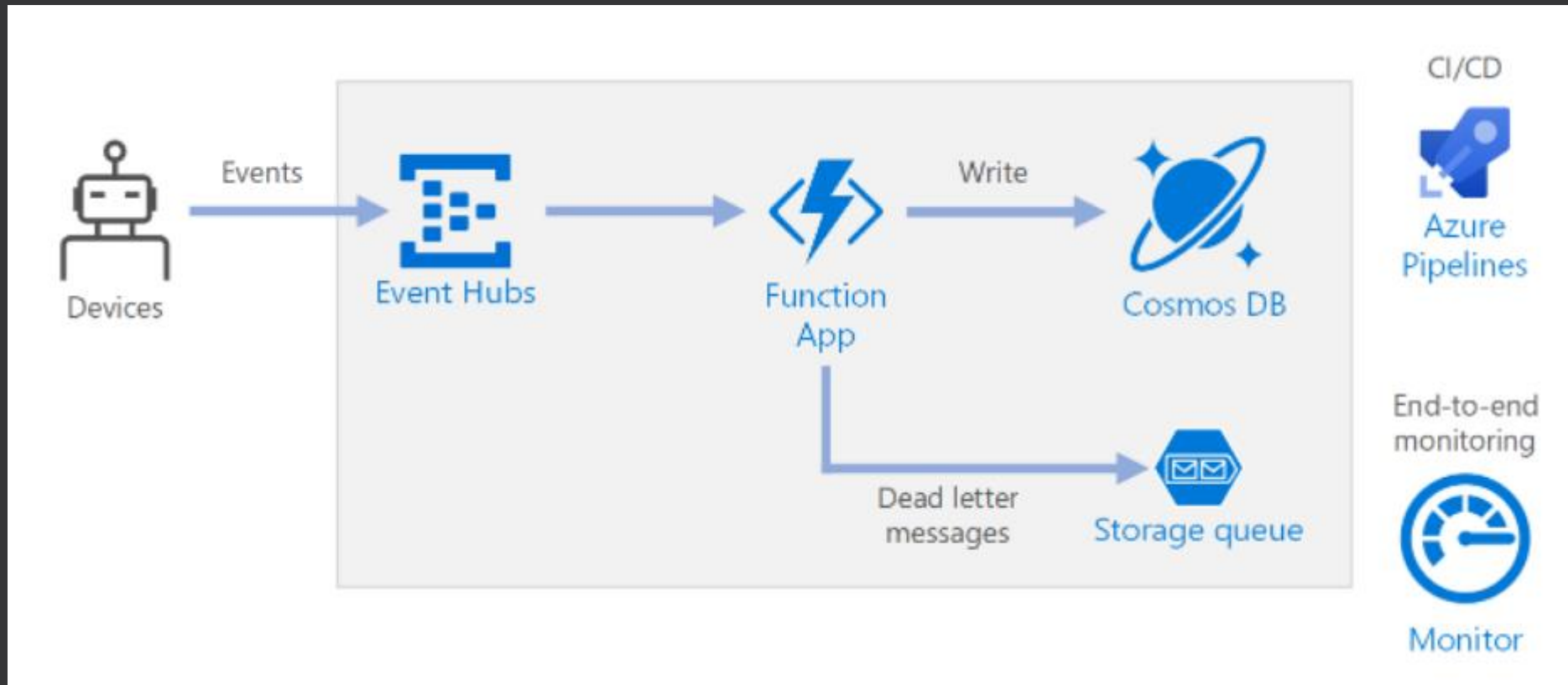
- Mouse movement & click
- Keyboard button press
- Time-triggered system
- Log message
- Program's interrupt, warning, error, crash, completion

# Event-driven Architecture

- Can help organizations achieve a flexible system that can adapt to changes and make decisions in real time.
- Consists of event producers that generate a stream of events, and event consumers that listen for the events.
- Producers are decoupled from consumers.



# Example of Event-driven Application





# Azure Function

Vijay Saini



# What is Azure Function

Azure Functions is a serverless solution that allows you to write less code, maintain less infrastructure, and save on costs.

Instead of worrying about deploying and maintaining servers, the cloud infrastructure provides all the up-to-date resources needed to keep your applications running.

It uses an event-driven model, where a piece of code is invoked by a trigger..



# What is Azure Function

functions accelerate development by using an event-driven model, with triggers that automatically execute code to respond to events and bindings to seamlessly integrate additional services.

A pay-per-execution model with sub-second billing charges only for the time and resources it takes to execute the code.



# Function app

A function app provides an execution context in Azure in which your functions run. As such, it is the unit of deployment and management for your functions. A function app is comprised of one or more individual functions that are managed, deployed, and scaled together.

All of the functions in a function app share the same pricing plan, deployment method, and runtime version. Think of a function app as a way to organize and collectively manage your functions. To learn more, see [How to manage a function app](#)



# Triggers and Bindings

Triggers cause a function to run. A trigger defines how a function is invoked and a function must have exactly one trigger. Triggers have associated data, which is often provided as the payload of the function.

Binding to a function is a way of declaratively connecting another resource to the function; bindings may be connected as input bindings, output bindings, or both. Data from bindings is provided to the function as parameters.

Bindings are optional and a function might have one or multiple input and/or output bindings.



# Hosting plan influences:

How your function app is scaled.

The resources available to each function app instance.

Support for advanced functionality, such as Azure Virtual Network connectivity.



# Azure Functions hosting options

## Consumption plan

Scale automatically and only pay for compute resources when your functions are running.

## Premium plan

Automatically scales based on demand using pre-warmed workers, which run applications with no delay after being idle, runs on more powerful instances, and connects to virtual networks.

## Dedicated plan

Run your functions within an App Service plan at regular App Service plan rates.

# Function app timeout duration

The timeout duration of a function app is defined by the `functionTimeout` property in the `host.json` project file. The following table shows the default and maximum values (in minutes) for specific plans:

Plan	Default	Maximum <sup>1</sup>
Consumption plan	5	10
Premium plan	30 <sup>2</sup>	Unlimited
Dedicated plan	30 <sup>2</sup>	Unlimited

<sup>1</sup> Regardless of the function app timeout setting, 230 seconds is the maximum amount of time that an HTTP triggered function can take to respond to a request. This is because of the [default idle timeout of Azure Load Balancer](#). For longer processing times, consider using the [Durable Functions async pattern](#) or [defer the actual work and return an immediate response](#).

<sup>2</sup> The default timeout for version 1.x of the Functions runtime is *unlimited*.

# Identifying the right Azure Service

Vijay Saini



# Choosing the right Azure Service

How much control do I need?



Where do I need my app to run?



What usage model do I need?



Which functionality do I need?

# Scenario 1

Which Azure service would you recommend for below requirement:

- A.) I want control over load balancer & Antivirus
- B.) Application should be always running
- C.) App should also support on-prem cloud and AWS
- D.) My application has a dependency on a legacy software

Answer:

Virtual Machine



# Scenario 2

Which Azure service would you recommend for below requirement:

A.) I need control over App configuration( nothing else ).

B.) I trust Azure for Auto-Scaling

C.) My application should run whenever there is a new message in service queue

D.) I might want to run the application in AWS, GCP & On-Prem

Answer:

Function App





# Creating Function in Azure Portal

Vijay Saini



# With Azure Function Core tools, how to



- Create a new Function App
- Create your functions locally
- Start your functions app
- Publish your function app to Azure

# Working in Local Development Environment

Vijay Saini



# Developing Azure Functions using VS Code

Vijay Saini



# Developing Azure Functions using VS Code

Microsoft Azure support for Visual Studio Code is provided through a rich set of extensions that make it easy to discover and interact with the cloud services that power your applications.

The Azure Functions extension provides these benefits:

- Edit, build, and run functions on your local development computer.
- Publish your Azure Functions project directly to Azure.
- Write your functions in various languages while taking advantage of the benefits of Visual Studio Code.



← → ↺ 🔒 docs.microsoft.com/en-us/python/api/azure-functions/azure.functions.httpresponse?view=azure-python

Docs / Reference / Functions / azure-functions / azure.functions.httpresponse

Version

Azure SDK for Python

🔍 Search

azure.functions.HttpRequest  
**azure.functions.HttpResponse**  
azure.functions.InputStream  
azure.functions.KafkaConverter  
azure.functions.KafkaEvent  
azure.functions.KafkaTriggerConverter  
azure.functions.OrchestrationContext  
azure.functions.Out  
azure.functions.QueueMessage  
azure.functions.ServiceBusMessage  
azure.functions.TimerRequest  
azure.functions.WsgiMiddleware  
azure-functions-durable  
> HDInsight  
> Help + support  
> Hybrid Compute  
> Hybrid Kubernetes  
> Hybrid Network  
> Identity  
> IoT

# HttpResponse Class

Reference 👍 🗨

An HTTP response object.

Inheritance `azure.functions._abc.HttpResponse` → `HttpResponse`

## Constructor

Python 📄 Copy

```
HttpResponse(body=None, *, status_code=None, headers=None, mimetype=None, charset=None)
```

## Parameters

**body** `<xref:<xref:str/bytes>>`  
default value: None  
Optional response body.

**status\_code** `int`  
Required  
Response status code. If not specified, defaults to 200.

**headers** `dict`  
Required

## Environment variables


In Functions, [application settings](#), such as service connection strings, are exposed as environment variables during execution. There are two main ways to access these settings in your code.

Method	Description
<code>os.environ["myAppSetting"]</code>	Tries to get the application setting by key name, raising an error when unsuccessful.
<code>os.getenv("myAppSetting")</code>	Tries to get the application setting by key name, returning null when unsuccessful.

Both of these ways require you to declare `import os`.

The following example uses `os.environ["myAppSetting"]` to get the [application setting](#), with the key named `myAppSetting`:

Python

 Copy

```
import logging
import os
import azure.functions as func

def main(req: func.HttpRequest) -> func.HttpResponse:

    # Get the setting named 'myAppSetting'
    my_app_setting_value = os.environ["myAppSetting"]
    logging.info(f'My app setting value:{my_app_setting_value}')
```

For local development, application settings are [maintained in the local.settings.json](#) file.

# Fix 'Port xxxx is unavailable' Issue

```
C:\delete>func start  
Found Python version 3.9.10 (py).  
Port 7071 is unavailable. Close the process using that port, or specify another port using --port [-p].
```

Run below statement in PowerShell

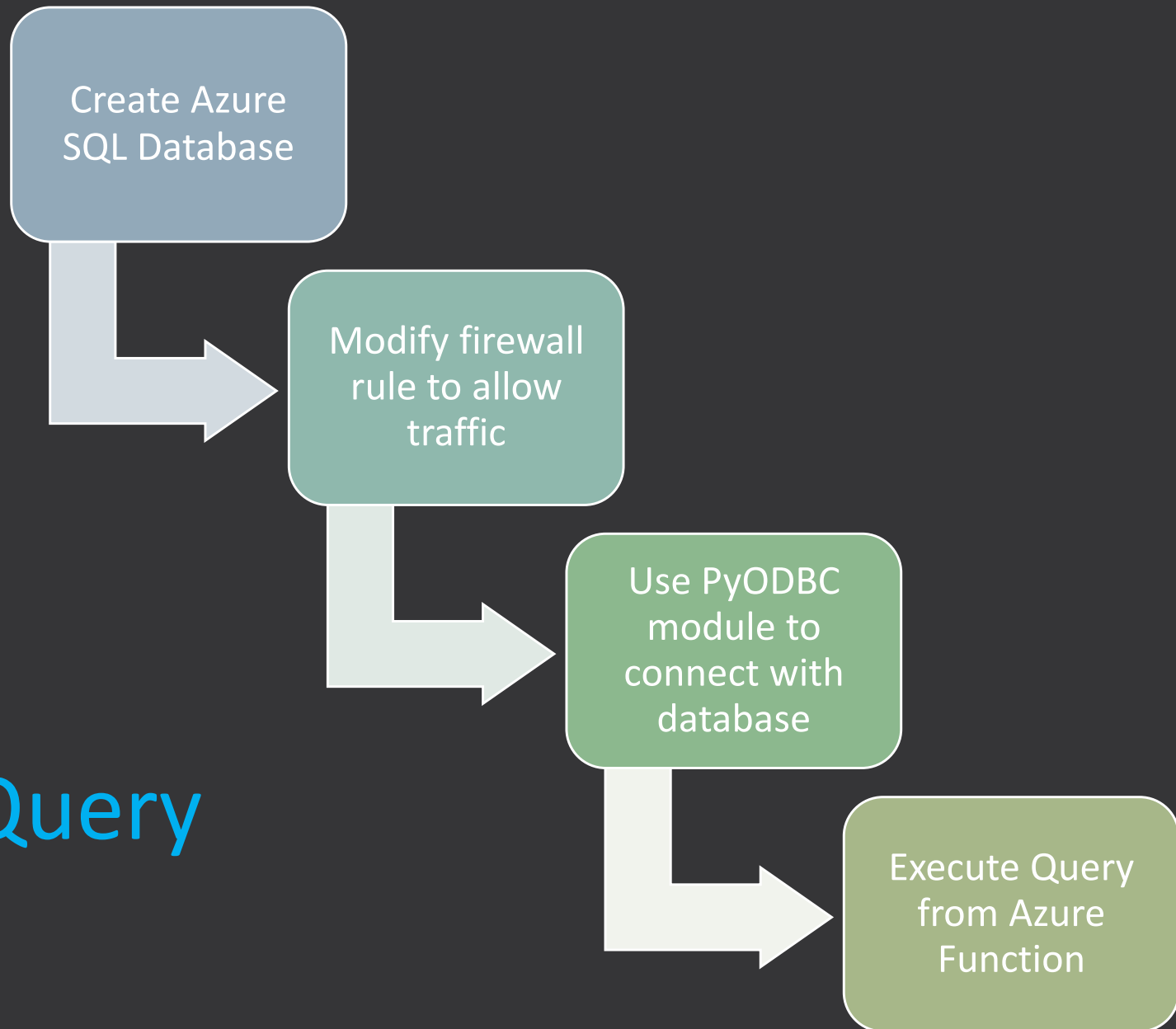
```
Get-Process -Id (Get-NetTCPConnection -LocalPort 7071).OwningProcess | kill
```

# Azure Function - Database Interaction

Vijay Saini



# Execute SQL Query on Database





# Environment Setup



# Retrieve Online Course Information

Develop an Azure Function API to connect to a SQL database and retrieve course information from the Online Courses SQL table.



Publish the formatted output in the browser and in the logs.



# Comprehension in Python

Comprehension is a technique with the help of which we can create sequences of elements in a short, concise manner.

## Syntax

```
output_list = [item for item in existing_list if (condition to be satisfied) ]
```

we store the output of the comprehension into a list and then explicitly convert it into a tuple.

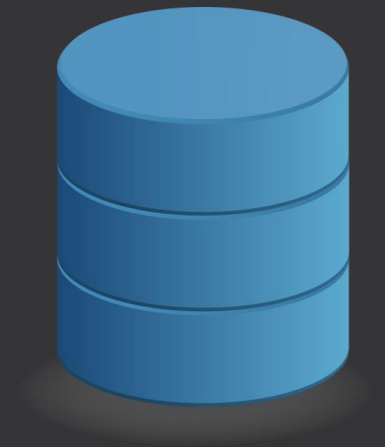
```
records = [tuple(record) for record in records]
```

# Collect & Store Feedback

Collect and store customer reviews submitted to an API hosted inside an Azure Function.



The information should be saved in a SQL database that includes a timestamp.



Insert a bad record into the same table if no review is submitted at the time of the calling function.

# Scheduled Database Cleanup

Perform SQL database cleanup by deleting the bad records from SQL table every 1 hour.



# Azure Service Bus

Vijay Saini

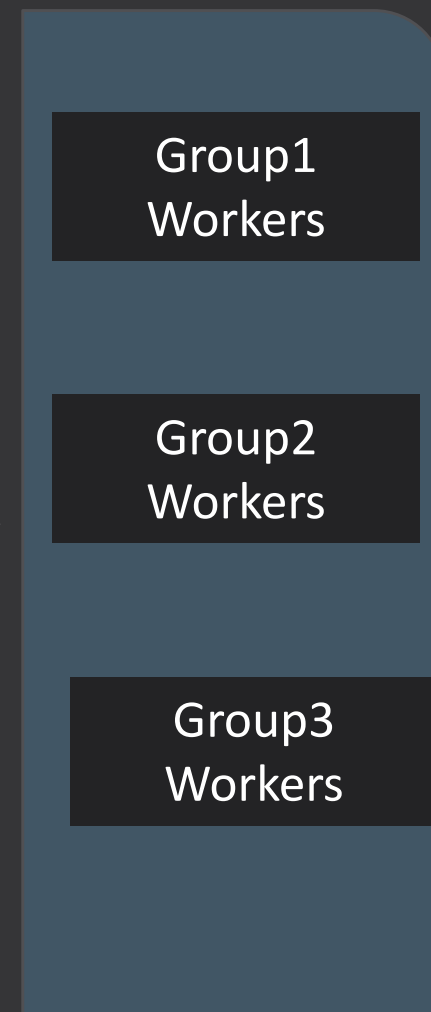
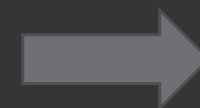
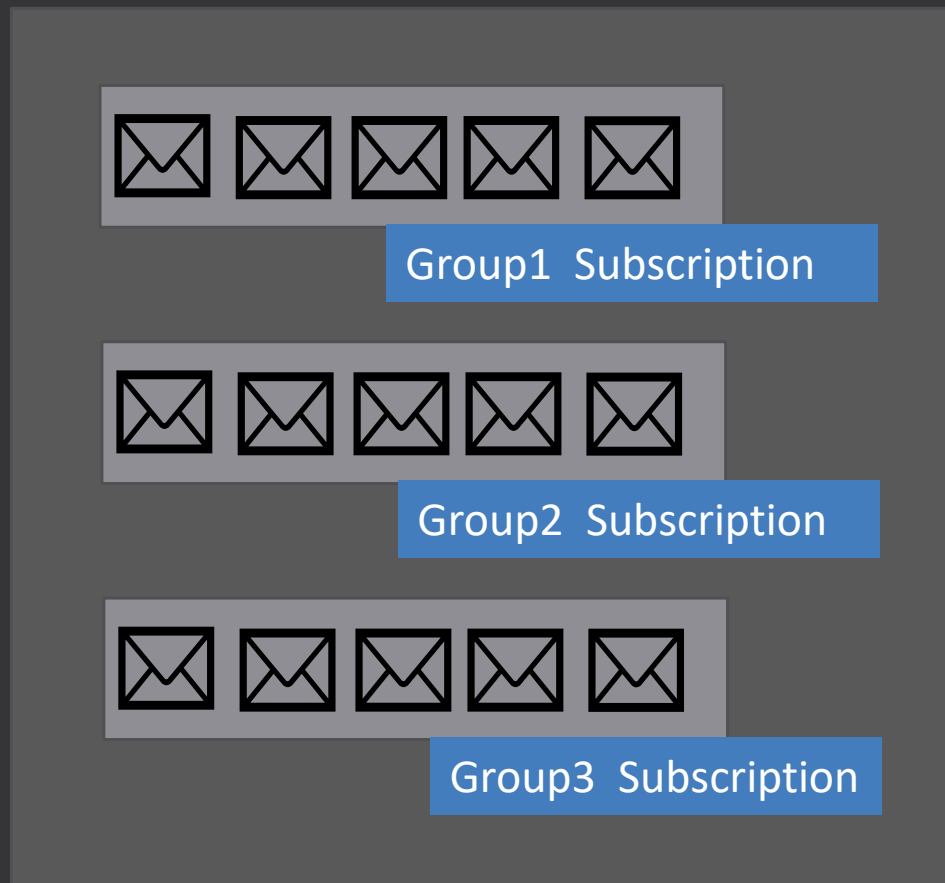


Team  
Manager

Workers



Task Queue





Web App

Mobile  
App

Service

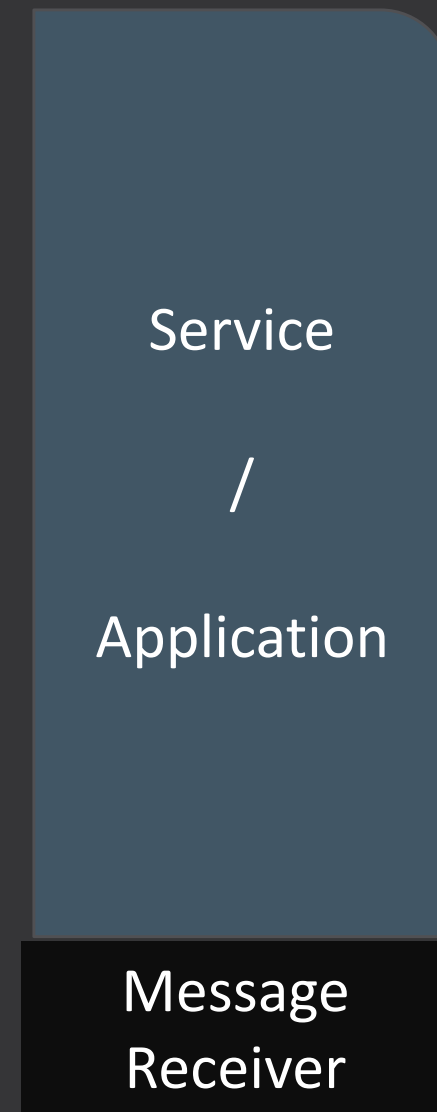
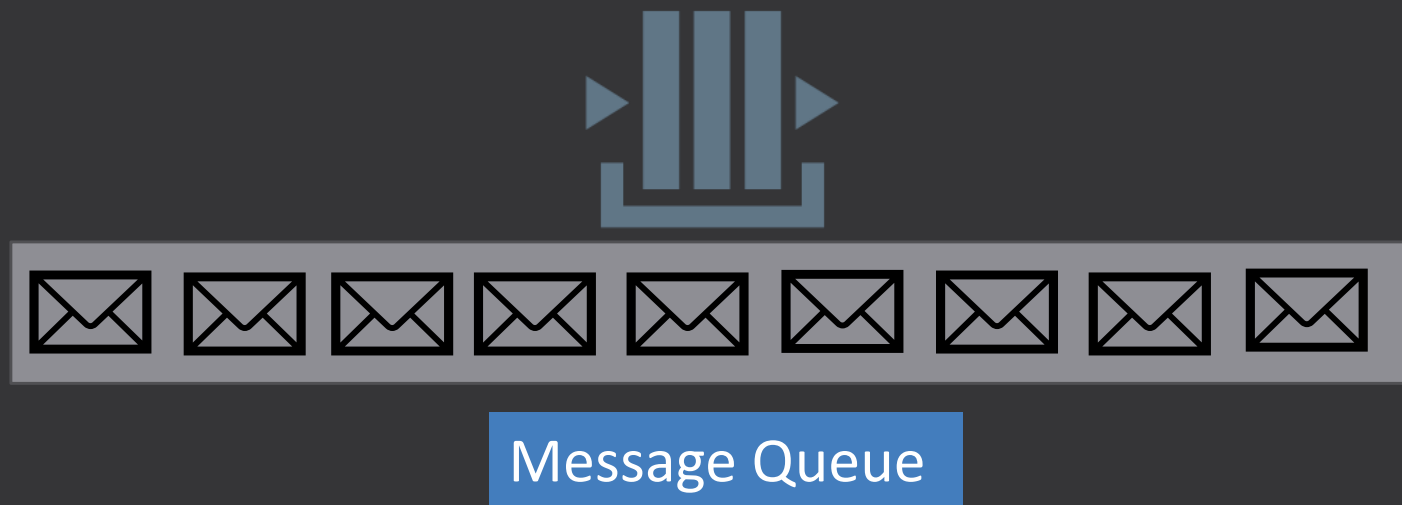
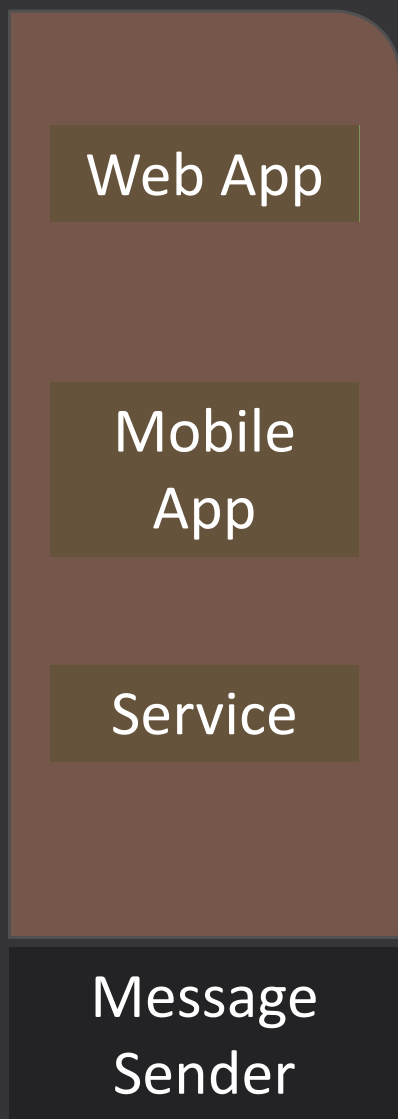
Message  
Sender

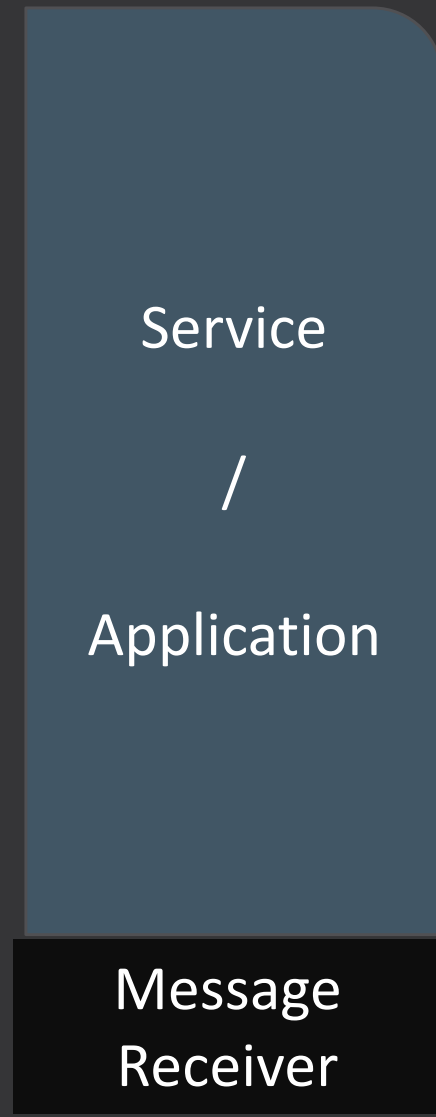
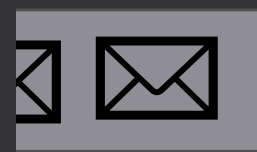
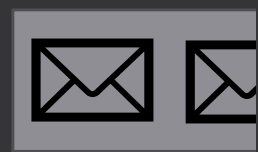
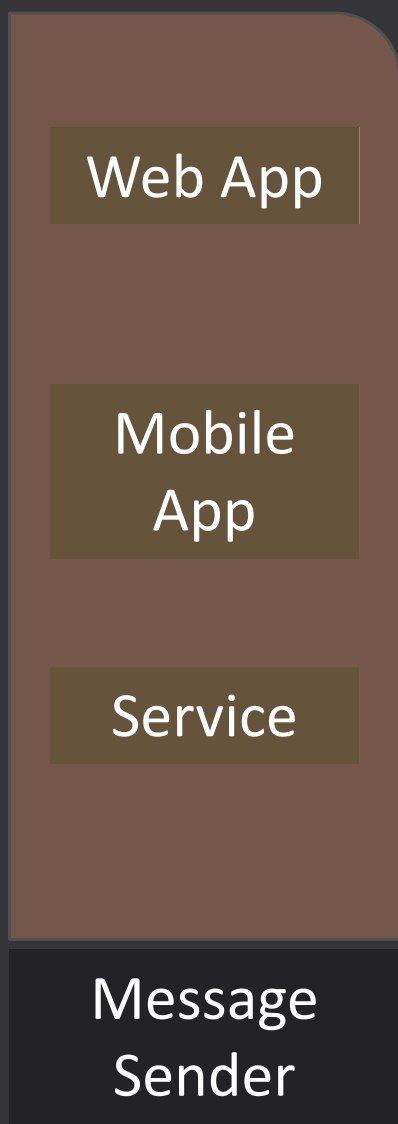
Service

/

Application

Message  
Receiver





# Azure Service Bus

Azure Service Bus is a fully managed enterprise message broker with message queues and publish-subscribe topics.

Service Bus is used to decouple applications and services from each other



# Azure Service Bus : Benefits

Load-balancing work across competing workers

Safely routing and transferring data and control across service and application boundaries

Coordinating transactional work that requires a high-degree of reliability

# Azure Service Bus

Data is transferred between different applications and services using messages.

The data can be any kind of information, including structured data encoded with the common formats such as the following ones: JSON, XML, Apache Avro, Plain Text.



# Namespaces

Namespace serves as the application container and has the ability to hold multiples queues and topics.

# Message

A message is a container decorated with metadata, and contains data.

A message is raw data produced by a service to be consumed or stored elsewhere.



# Queues

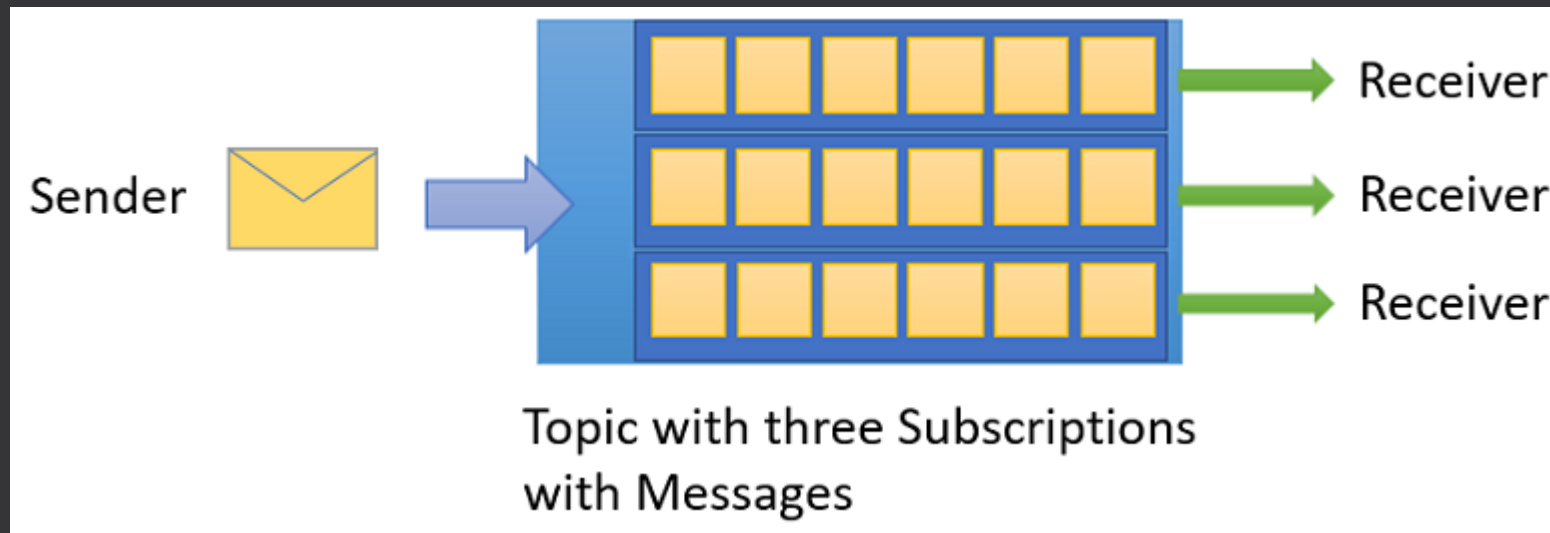
Messages are sent to and received from queues. Queues store messages until the receiving application is available to receive and process them.

Messages are delivered in pull mode, only delivering messages when requested.



# Topics

While a queue is often used for point-to-point communication, topics are useful in publish/subscribe scenarios.



# Topics

A subscriber to a topic can receive a copy of each message sent to that topic.

Subscriptions are named entities. Subscriptions are durable by default, but can be configured to expire and then be automatically deleted.

A subscription rule has a filter to define a condition for the message to be copied into the subscription and an optional action that can modify message metadata.

# Azure Queue Storage

Azure Queue Storage is a service for storing large numbers of messages.

You access messages from anywhere in the world via authenticated calls using HTTP or HTTPS. A queue message can be up to 64 KB in size.

A queue may contain millions of messages, up to the total capacity limit of a storage account.

Queues are commonly used to create a backlog of work to process asynchronously.

# Dead-lettering

Service Bus supports a dead-letter queue (DLQ) to hold messages that cannot be delivered to any receiver, or messages that cannot be processed. You can then remove messages from the DLQ and inspect them.

# Azure Function Integrations

Vijay Saini



# Triggers and Bindings

**Triggers** cause a function to run. A trigger defines how a function is invoked and a function must have exactly one trigger. Triggers have associated data, which is often provided as the payload of the function.

**Binding** to a function is a way of declaratively connecting another resource to the function; bindings may be connected as input bindings, output bindings, or both. Data from bindings is provided to the function as parameters.

Bindings are optional and a function might have one or multiple input and/or output bindings.



# Common Binding Properties

**Name** - Defines the function parameter through which you access the data (for example, in a queue input binding, this is the name of the function parameter that receives the queue message content).

**Type** - Identifies the type of binding (for example, the type of data or service we want to interact with).

**Direction** - Indicates the direction data is flowing (for example, is it an input or output binding)?

**Connection** - Provides the name of an app setting key that contains the connection string.




## Trigger file name

The `path` for a Blob trigger can be a pattern that lets you refer to the name of the triggering blob in other bindings and function code. The pattern can also include filtering criteria that specify which blobs can trigger a function invocation.

For example, in the following Blob trigger binding, the `path` pattern is `sample-images/{filename}`, which creates a binding expression named `filename`:

JSON

 Copy

```
{
  "bindings": [
    {
      "name": "image",
      "type": "blobTrigger",
      "path": "sample-images/{filename}",
      "direction": "in",
      "connection": "MyStorageConnection"
    },
    ...
  ]
}
```

The expression `filename` can then be used in an output binding to specify the name of the blob being created:

# What is a binding expression?

A binding expression is specialized text in function.json, function parameters, or code that is evaluated when the function is invoked, to yield a value.

For example, if the blob output binding path is `%Environment%/newblob.txt`, and the Environment app setting value is Development, a blob is created in the Development container.

# QnA

Scenario	Trigger	Input binding	Output binding
A new queue message arrives which runs a function to write to another queue.	?	?	?
A scheduled job reads Blob Storage contents and creates a new Cosmos DB document.	?	?	?
The Event Grid is used to read an image from Blob Storage and a document from Cosmos DB to send an email.	?	?	?
A webhook that uses Microsoft Graph to update an Excel sheet.	?	?	?

# QnA - Solved

Scenario	Trigger	Input binding	Output binding
A new queue message arrives which runs a function to write to another queue.	Queue	None	Queue
A scheduled job reads Blob Storage contents and creates a new Cosmos DB document.	Timer	Blob Storage	Cosmos DB
The Event Grid is used to read an image from Blob Storage and a document from Cosmos DB to send an email.	Event Grid	Blob Storage and Cosmos DB	SendGrid
A webhook that uses Microsoft Graph to update an Excel sheet.	HTTP	None	Microsoft Graph

# Assignment

Check if our Azure function still works if, instead of sending an HTTP request, you

- Upload a sample text file in container-1, Validate the message in Service Bus
- Upload a sample text file in container-2, Validate the message in Service Bus
- Add a message in storage (myapp-queue), Validate the message in Service Bus

Explore input binding by developing below:

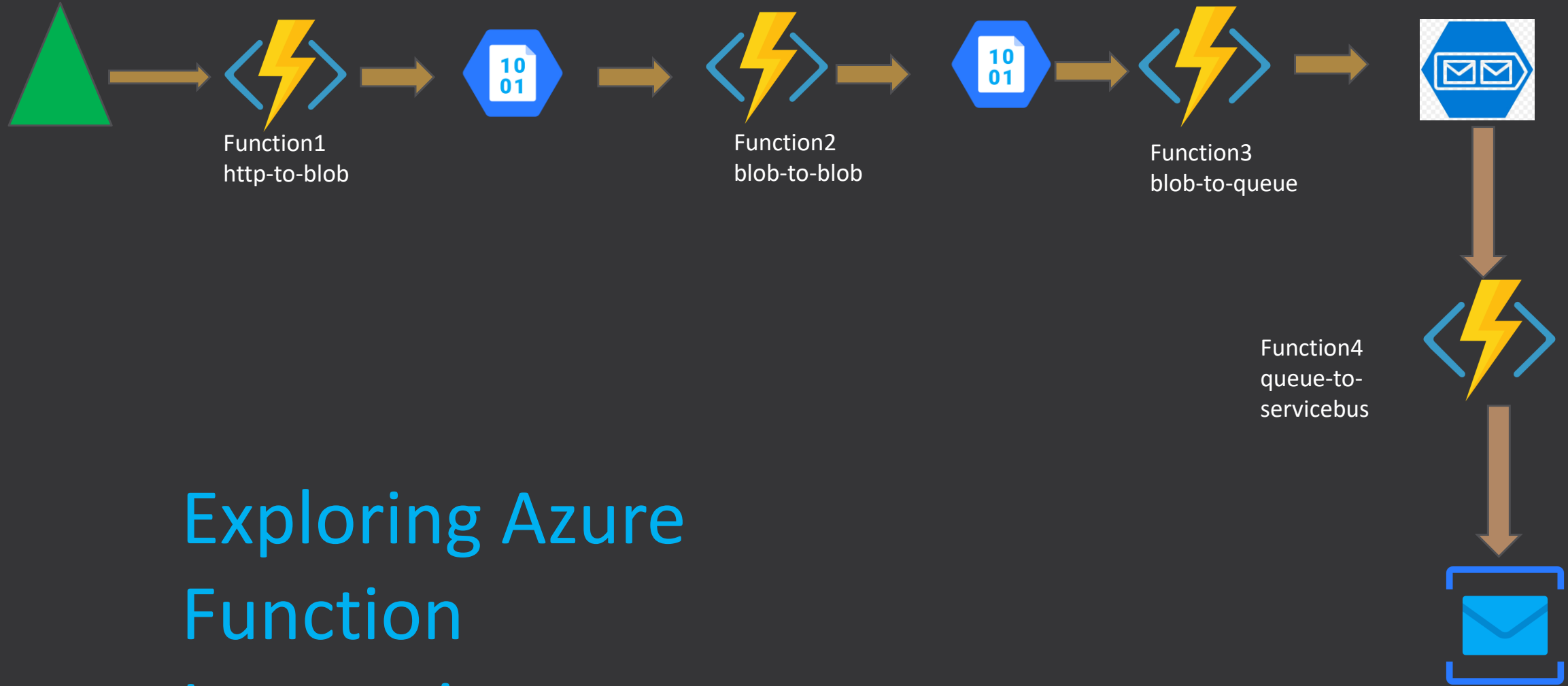
A scheduled job reads Blob Storage contents, add line “processed by myFunc” , then

- Add the text to storage queue as a message
- Delete the blob

# Exploring Azure Function Integrations

- ✓ Parse the text from a http request and save it as a blob file in an Azure storage account container
- ✓ Process each blob in a blob container and save the output into another container
- ✓ Process each blob in a blob container and queue the text as a message in Azure Storage Queue
- ✓ Process each message in the Storage Queue and set the text into a Azure Service Bus Topic





# Exploring Azure Function Integrations

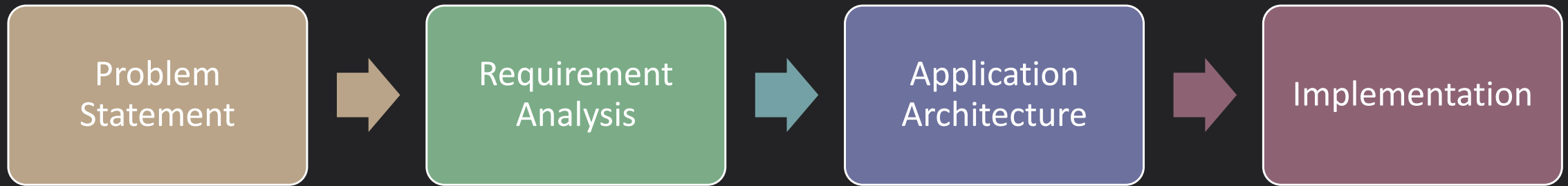
# Developing a Transaction Processing System

Vijay Saini





# Process



Highly Available

Cost Efficient

Fault Tolerant

Scalable System

# Business Requirement

Your client company TechSckool uses a third party payment gateway.

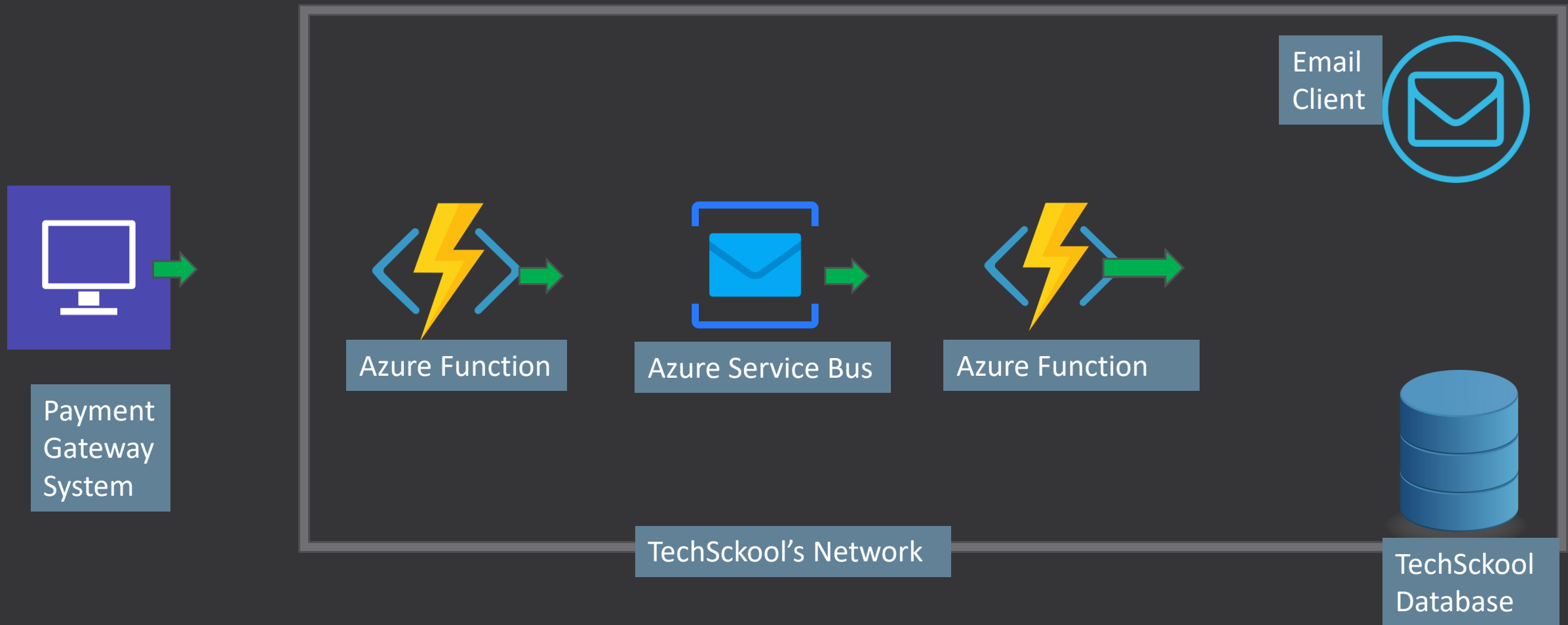
Client Business Process is documented as :

- a) Customers visit the website and place an order.
- b) The Payment Gateway receives the request.
- c) The Payment Gateway system gathers the relevant information and connects with the bank to complete the transaction.
- d) TechSckool receives transaction status from the Payment Gateway.

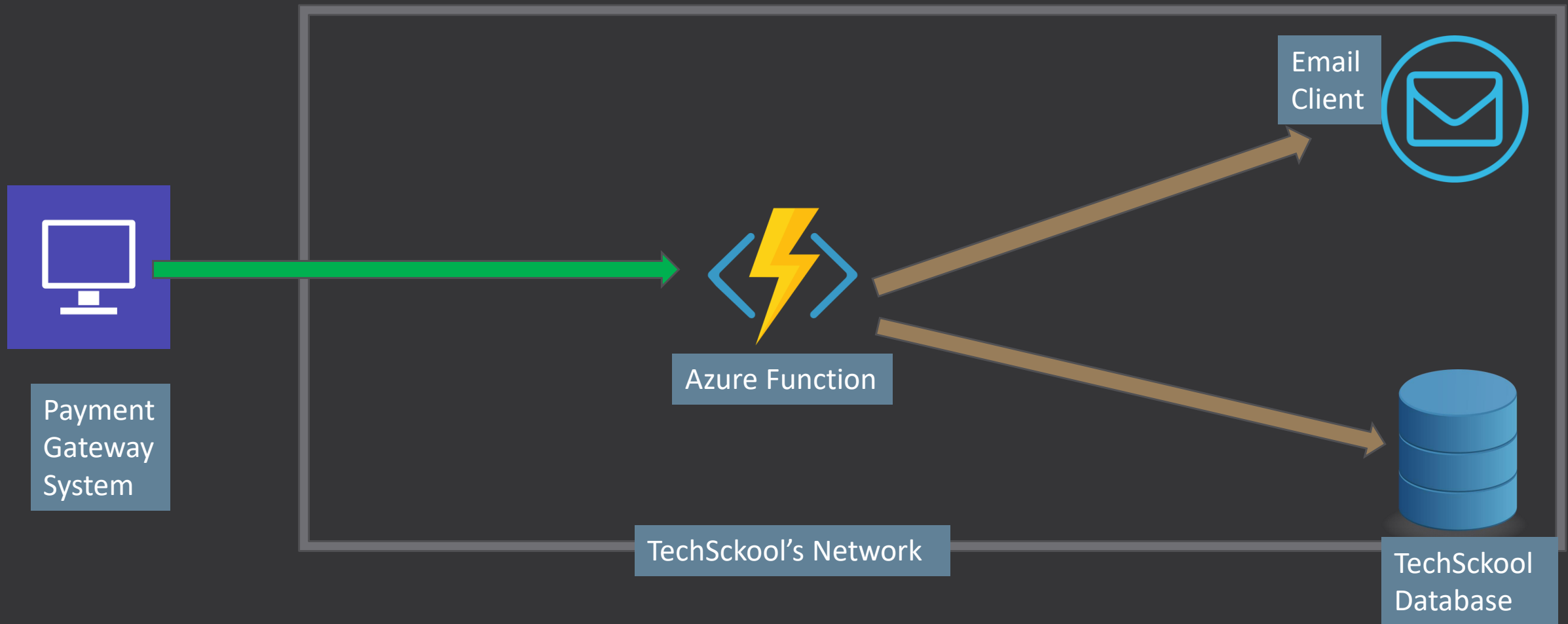
# Business Requirement

- ✓ TechSckool wants you to device a mechanism to send an email update to the end users whenever they buy their product.
- ✓ If a transaction is denied, the end user should be notified through an email as soon as possible.
- ✓ In both circumstances, the transaction success or failure should be recorded in TechSckool's database.

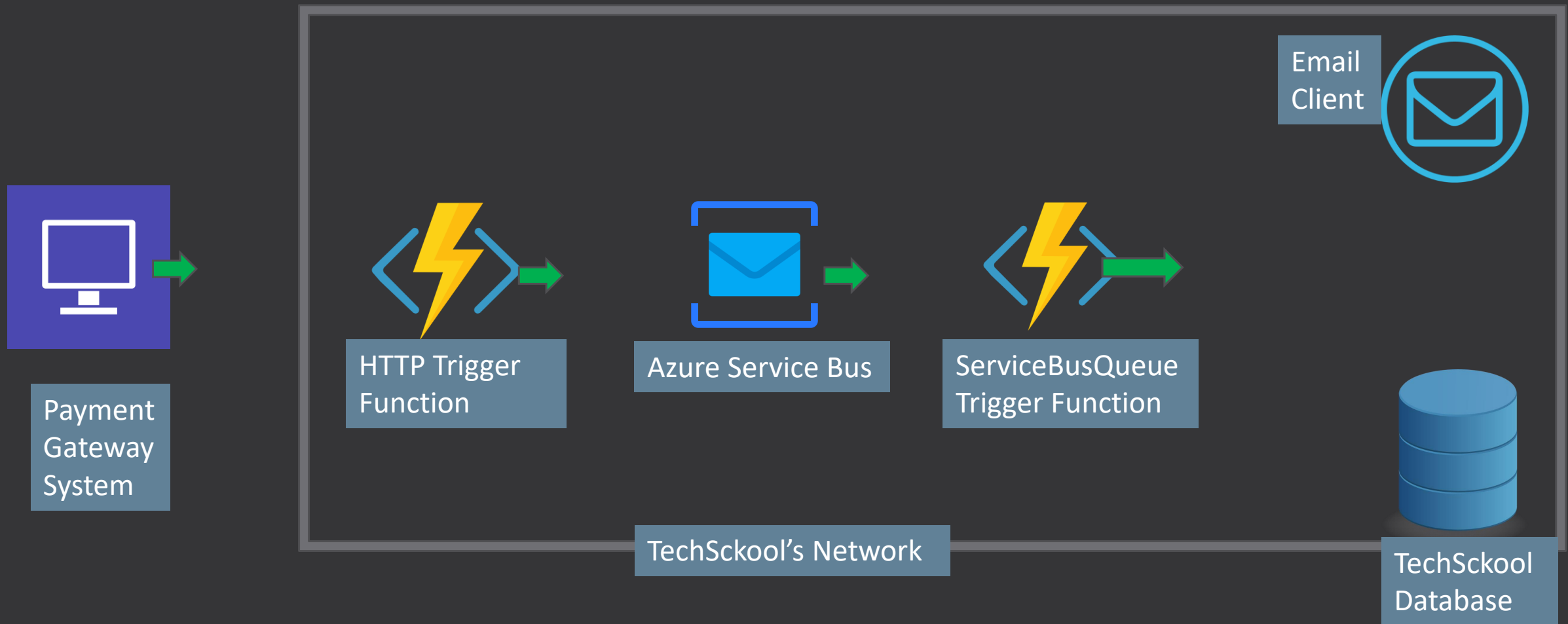
# Architecture



What are the advantages of our Architecture over a simpler architecture like this?



# Architecture



# Assignment

- ✓ Find the single point of failures and fix them
- ✓ Success Email to the end users should be based on a html template which is stored in a blob storage
- ✓ In case of failed transaction, Don't simply send a plain notification to the end user.  
Instead Begin another business process in your system by
  - send failed message to another queue
  - Raise ticket for your support/sales team
  - In email to the end user, give the ticket number and contact of team/person who is working



Thank You

