TOP 250+

INTERVIEW QUESTIONS

ROHATASH KUMAR

ABOUT THE BOOK

This book contains **Top 250+ Microsoft Azure** interview questions designed to help you prepare effectively for Azure and Cloud technical interviews.

The questions are carefully selected to cover **Azure fundamentals**, core services, real-time scenarios, an **advanced** concepts that are most commonly asked in real interviews.

Whether you are a **beginner**, developer, or **experienced** professional, this book will help you **strengthen your Azure knowledge, build confidence**, and succeed in interviews.

ABOUT THE AUTHOR

Rohatash Kumar has over **15 years of experience** in software development. He has helped many candidates succeed in technical interviews at well-known tech companies by sharing **his knowledge and guidance.**

Azure Chapters

1. Introduction to Cloud & Azure

Q. What is Cloud computing? Why Cloud Computing? V.IMP.

Q. On-Premises Computing vs Cloud Computing V.IMP.

Q. What are Azure Functions? Have you ever used them in your project? Why?

Q. What are Triggers in Azure Functions? Can you name a few types?

Q. How Azure Logic Apps are different from Azure Functions?

Q. What is Azure Durable functions? Why we use Durable Function? Give any Real-World Example

Q. What is Docker? How to use it in .Net?

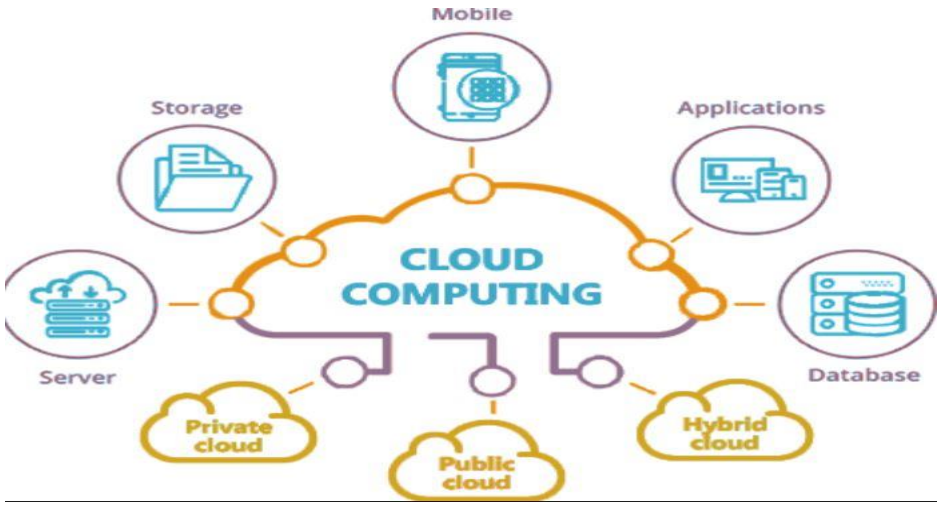
Q. What is Azure Service Bus? Share a Real-world example of how it is used?

Q. What is Topic in Azure Service Bus? What is the difference between Topic and Queue?

Q. What is Cloud computing? Why Cloud computing? V. IMP.

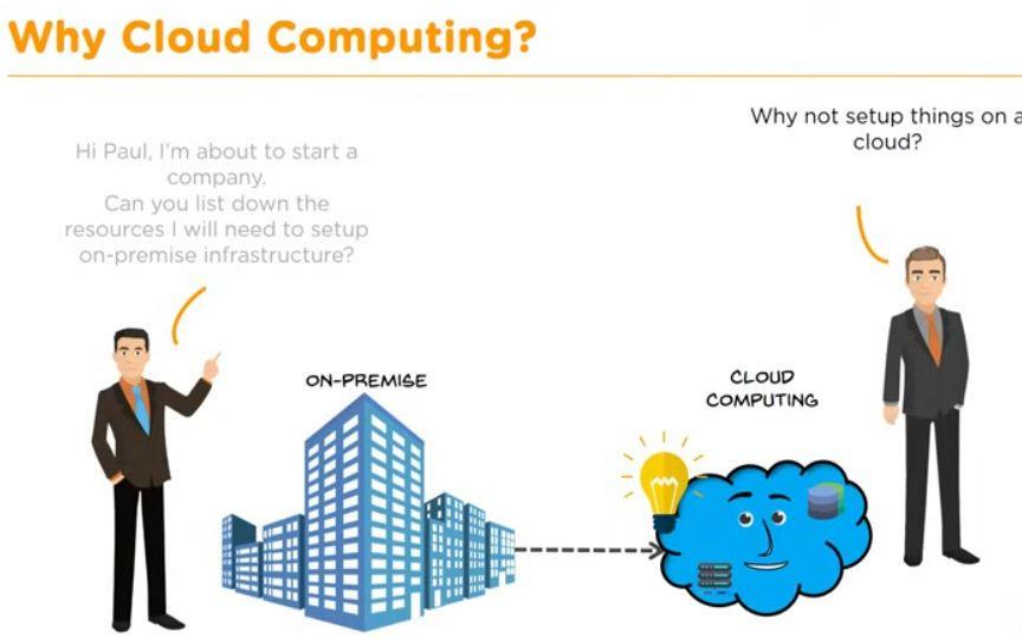
Cloud Computing

Cloud computing is the delivery of IT resources - like servers, storage, databases, and software - over the internet on a pay-as-you-go basis.



Real-world example

Just like you use electricity from the power grid instead of installing your own power plant, companies use cloud resources instead of buying their own servers. You pay only for what you use.



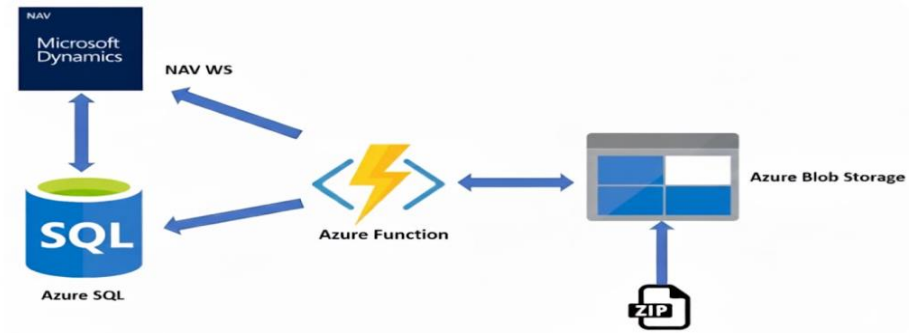
Q. On-Premises Computing vs Cloud Computing V.IMP.

On-Premises vs Cloud Computing

Feature	On-Premises	Cloud Computing
Ownership	Company owns hardware	Cloud provider owns hardware
Cost Model	High upfront capital cost	Pay-as-you-go (OPEX)
Scalability	Slow, manual	Instant, automatic
Maintenance	Managed by company	Managed by provider
Availability	Depends on local setup	Built-in high availability
Disaster Recovery	Expensive to implement	Built-in and cost-effective
Deployment Speed	Weeks to months	Minutes to hours
Security	Full control	Shared responsibility model

Q. What are Azure Functions? Have you ever used them in your project? Why?

Azure Functions is a **serverless compute service** in Azure that lets you run code in response to events without managing infrastructure(servers).

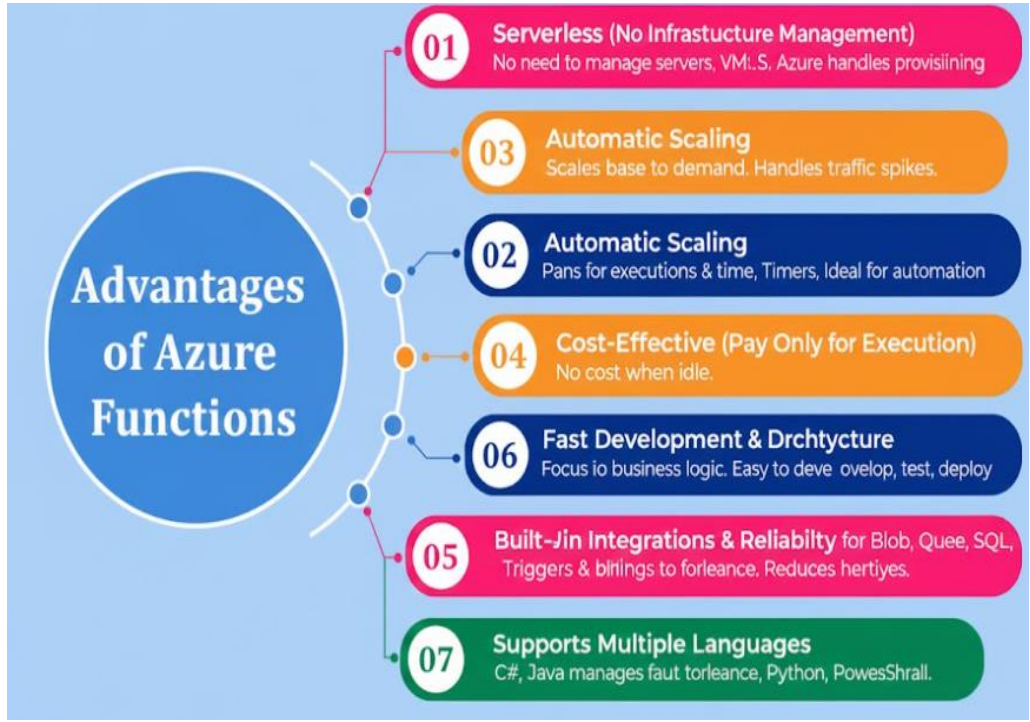


Flow Explanation (End-to-End)

1. **ZIP file is uploaded to Azure Blob Storage**
   - This can trigger an **Azure Function** automatically.
2. **Azure Function processes the ZIP file**
   - Extracts data or documents.
   - Validates or transforms the data.
3. **Azure Function writes structured data to Azure SQL**
   - Stores extracted information in database tables.
4. **Azure Function communicates with Dynamics NAV**
   - Uses **NAV Web Services** to send or retrieve business data.
   - Keeps NAV in sync with processed data.
5. **NAV and SQL stay synchronized**
   - NAV ↔ SQL data flow ensures consistency across systems.

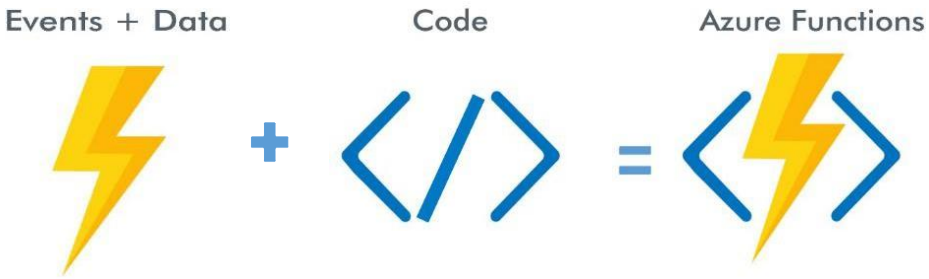
Advantages

Azure Functions provide a serverless, event-driven, and cost-effective way to run code with automatic scaling and minimal infrastructure management.



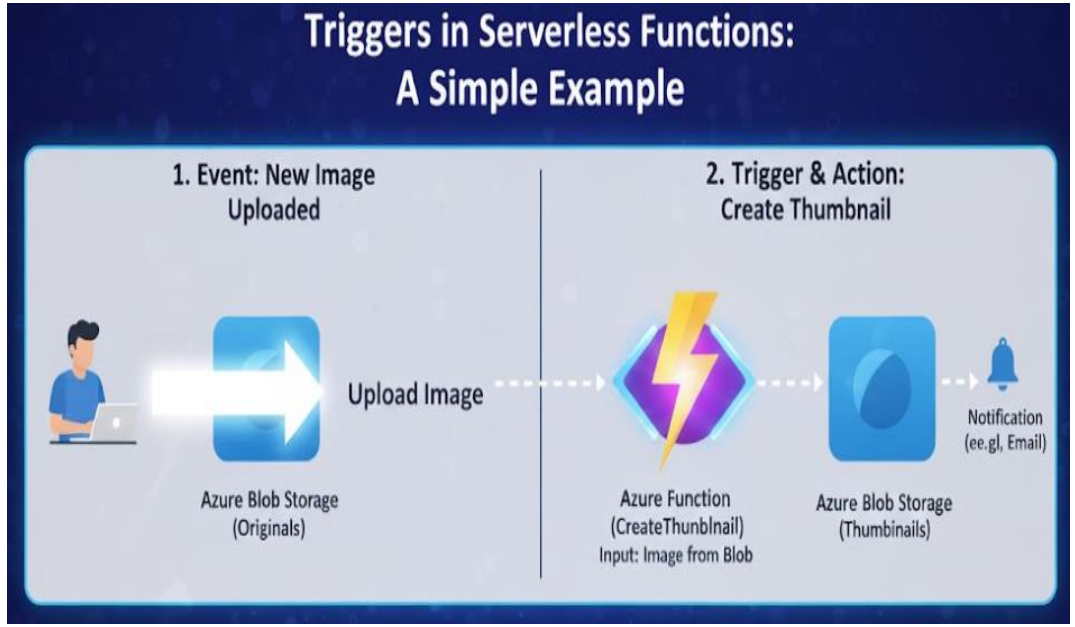
Q. What are Triggers in Azure Functions? Can you name a few types?

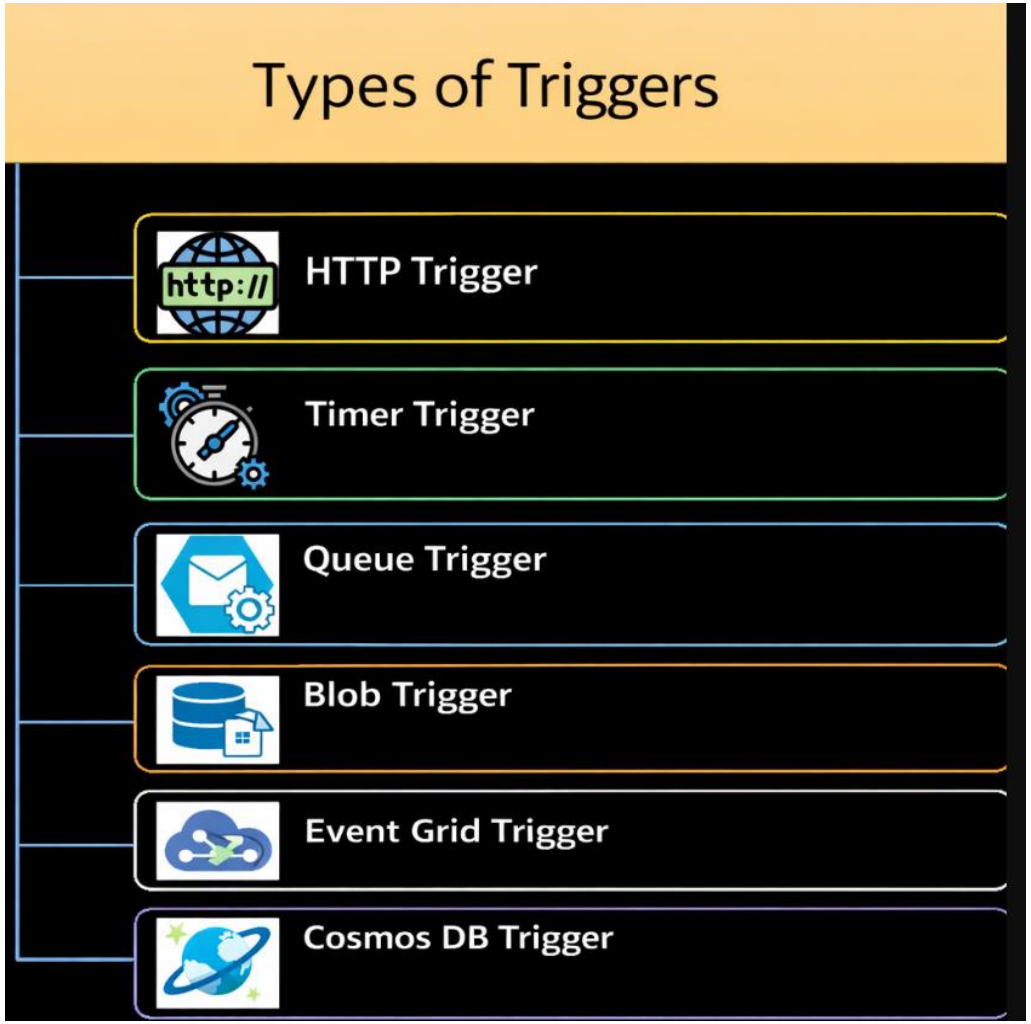
- **Triggers** are what start or invoke an Azure Function.
- When an event occurs, it triggers the Azure functions which then automatically executes, performs the task, and stops when done.



The Thumbnails Process Flow

1. **The Trigger:** A user (or a system) uploads a high-resolution "Main Image" into a specific container in **Azure Blob Storage** (e.g., a folder named original-images).
2. **The Event:** Azure Blob Storage "pokes" the **Azure Function** to wake it up because a new file has arrived.
3. **The Action:** The Function runs its code to resize that large image into a small **Thumbnail**.
4. **The Output:** The Function saves that new small Thumbnail back into a *different* container in **Azure Blob Storage** (e.g., a folder named thumbnails).





1. **HTTP Trigger** - Runs the function when an HTTP request (GET, POST, etc.) is received.
2. **Timer Trigger** - Runs the function automatically on a schedule using a CRON expression.
3. **Queue Trigger** - Runs the function when a new message is added to Azure Queue Storage.
4. **Blob Trigger** - Runs the function when a file is created or updated in Azure Blob Storage.
5. **Event Grid Trigger** - Runs the function when an event occurs in Azure services (like blob created, resource changed).
6. **Cosmos DB Trigger** - Runs the function when data is inserted or modified in Azure Cosmos DB.

Q. How Azure Logic Apps are different from Azure Functions?

Logic App vs Function App

Feature	Azure Logic App	Azure Function App
Type	Workflow-based automation	Code-based execution
Coding	Low / No code	Full code required
Designed for	Business process automation	Custom logic & processing
Visual Designer	✅ Yes (drag & drop)	❌ No
Triggers	Many built-in (email, HTTP, events)	Triggers via bindings
Connectors	100s of connectors (Office, SQL, SAP)	Limited (via code)
Scalability	Automatic	Automatic
Error Handling	Built-in	Manual in code
Deployment	Portal / ARM	CI/CD / Code
Best For	Integrations & workflows	High-performance logic

Q. What is Azure Durable functions? Why we use Durable Function? Give any Real-World Example

Azure Durable Functions are an extension of **Azure Functions** that help you build **stateful workflows** in a **serverless** way.
They are mainly used when your logic is **long-running**, **multi-step**, **needs retries**, or **must remember state** between steps.

In short:

Azure Functions = Stateless
Durable Functions = Stateful (remembers progress)

When Use Durable Functions in a Project?

Use them when:

- ✓ Workflow has **multiple dependent steps**
- ✓ Process is **long-running**
- ✓ You need **retries, timers, or approvals**

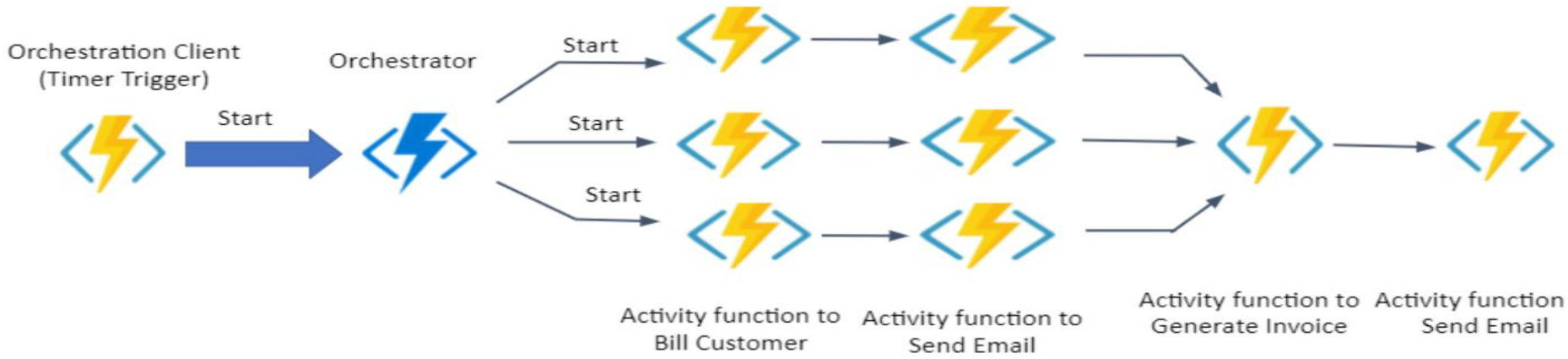
Why do we use Azure Durable Functions?

Normal Azure Functions are great for **single tasks**, but they struggle when:

- A process runs for **minutes, hours, or days**
- Multiple steps depend on each other
- You need **retry**, **waiting**, or **human approval**
- The function might restart and must **resume from where it stopped**

Durable Functions **solve** this by:

- Automatically saving state
- Handling retries and failures
- Resuming execution without re-running completed steps

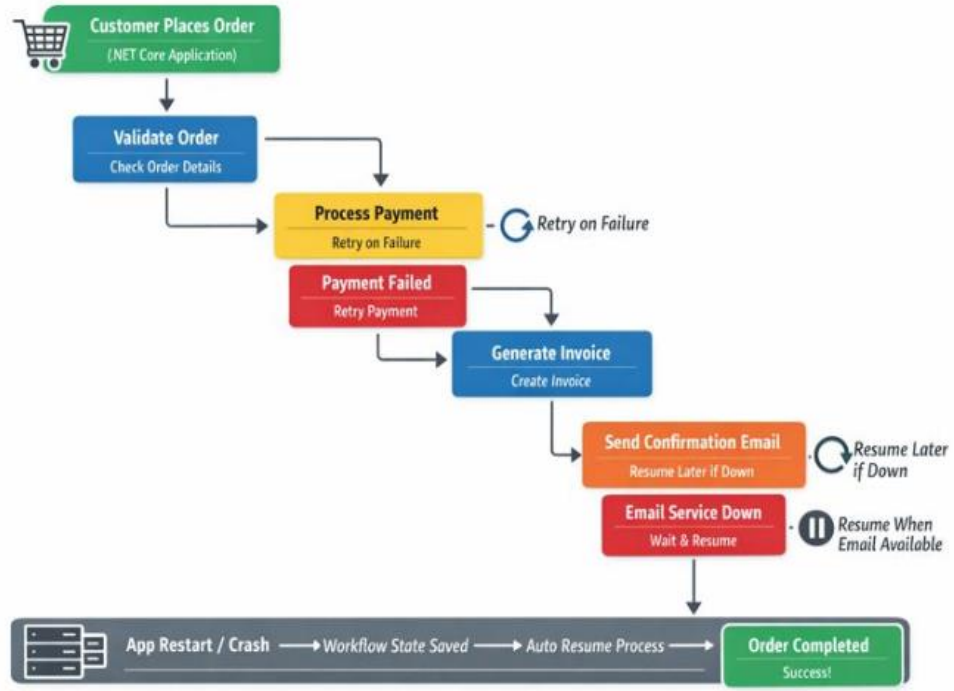


Simple Real-World Example - Order Processing System (E-commerce)

When a customer places an order:

1. Validate order
2. Reserve inventory
3. Process payment
4. Generate invoice
5. Send confirmation email

If **payment fails**, retry
If **email service is down**, resume later
If app restarts, workflow continues automatically



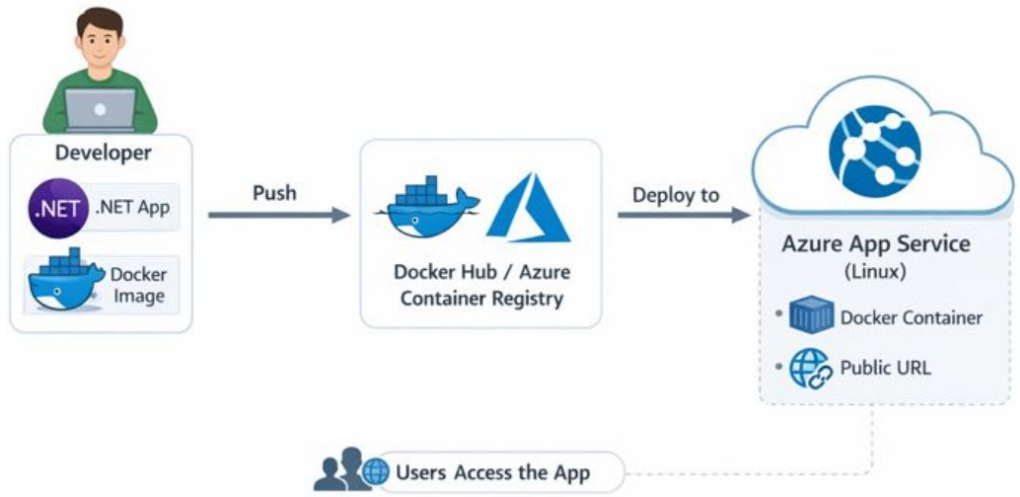
Q. What is Docker? How to use it in .Net?

Docker is a platform that allows you to package your .NET application and everything it needs (runtime, dependencies, configuration) into a single lightweight container.

👉 “Build once, run anywhere”

Think of it like this:

A container is a portable box that contains your app + all its setup, so it can run anywhere - without "it works on my machine" issues.



Why Docker is Needed (Problem → Solution)

❌ Without Docker

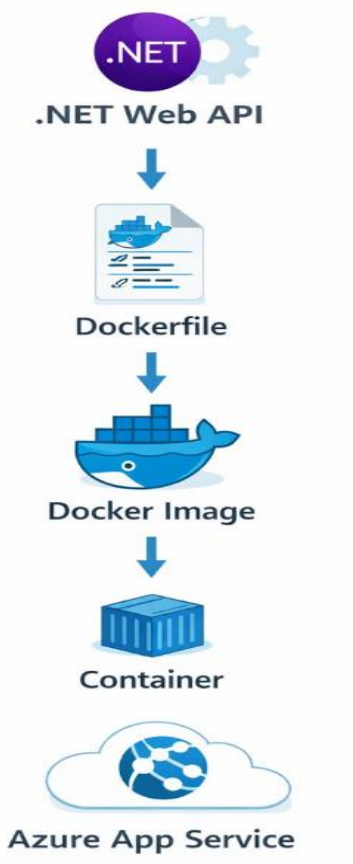
- App works on **developer machine**
- Fails on **QA / Production**
- Different OS, SDK versions, libraries

✅ With Docker

- Same container runs on:
  - Laptop
  - QA server
  - Azure Cloud
- No environment mismatch

When Should You Use Docker?

✓ Microservices
✓ Cloud deployments
✓ CI/CD pipelines
✓ Multiple environments
✓ Azure App Service / AKS



Sample Dockerfile for .NET Application

```
# Base image
FROM mcr.microsoft.com/dotnet/aspnet:8.0
WORKDIR /app

# Copy published files
COPY . .

# Expose port
EXPOSE 80

# Start app
ENTRYPOINT ["dotnet", "MyApp.dll"]
```

Build & Run Docker Image (Locally)

```
docker build -t mydotnetapp .
docker run -p 8080:80 mydotnetapp
```

Q. What is Azure Service Bus? Share a Real-world example of how it is used?

Azure Service Bus is a **fully managed enterprise message broker** in Azure that enables **reliable, asynchronous communication** between different applications, services, or microservices.

In simple words:

👉 It helps **one system send messages** and **another system receive them later**, **without being directly connected**.

Why Azure Service Bus is needed

In real systems:

- Services should **not depend on each other being online**
- Traffic may **spike suddenly**
- Systems should **process messages reliably and in order**

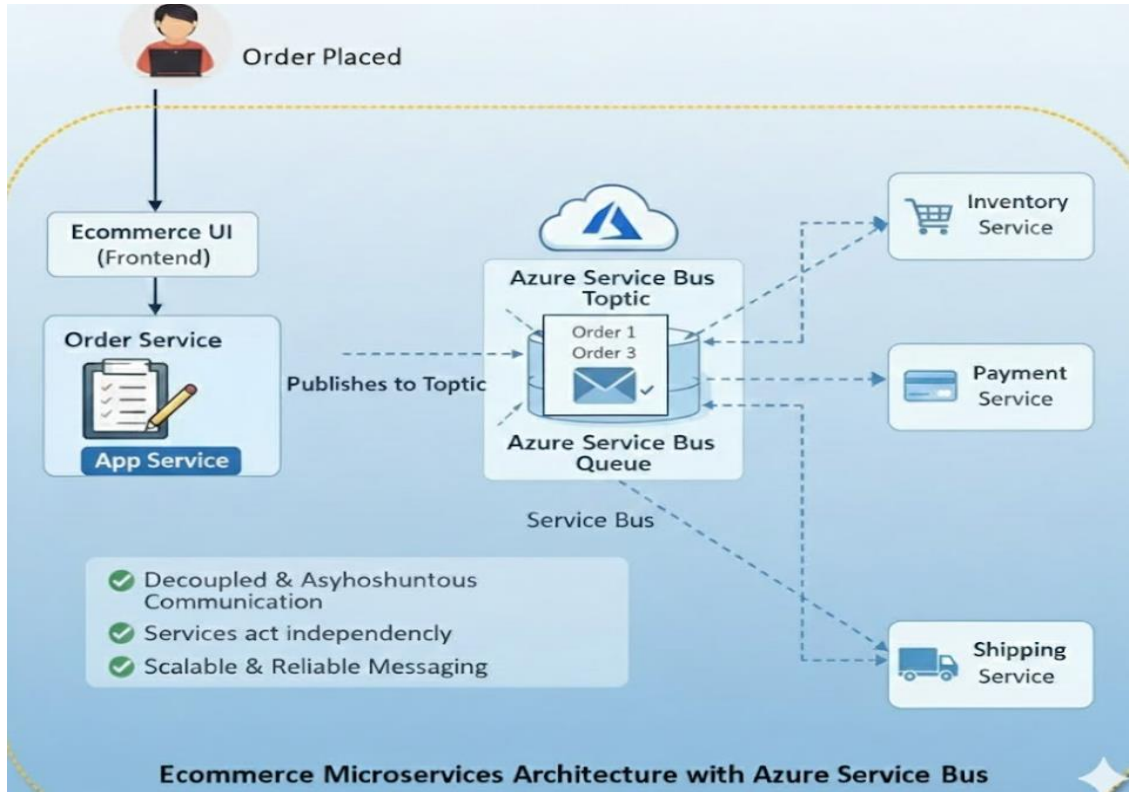
Azure Service Bus solves these problems by acting as a **middleman (message broker)**.

Where Azure Service Bus is commonly used

- Microservices communication
- Order processing systems
- Banking & financial systems
- Inventory management
- Event-driven architectures

services.

Example



Q. What is Topic in Azure Service Bus? What is the difference between Topic and Queue?

A **Topic** is like a **message broadcasting station** - one message goes in, and many subscribers can receive it independently.

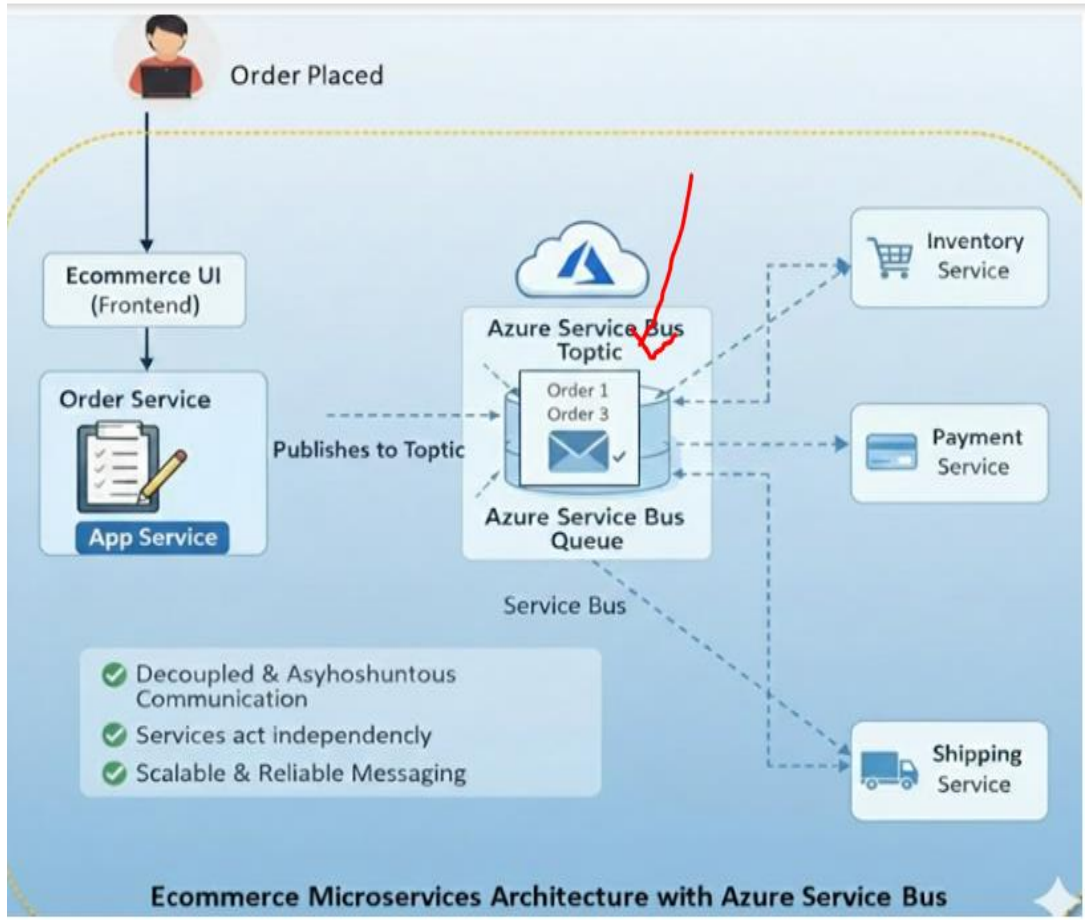
A **Topic** in **Azure Service Bus** is used for **publish–subscribe (Pub/Sub) messaging**.

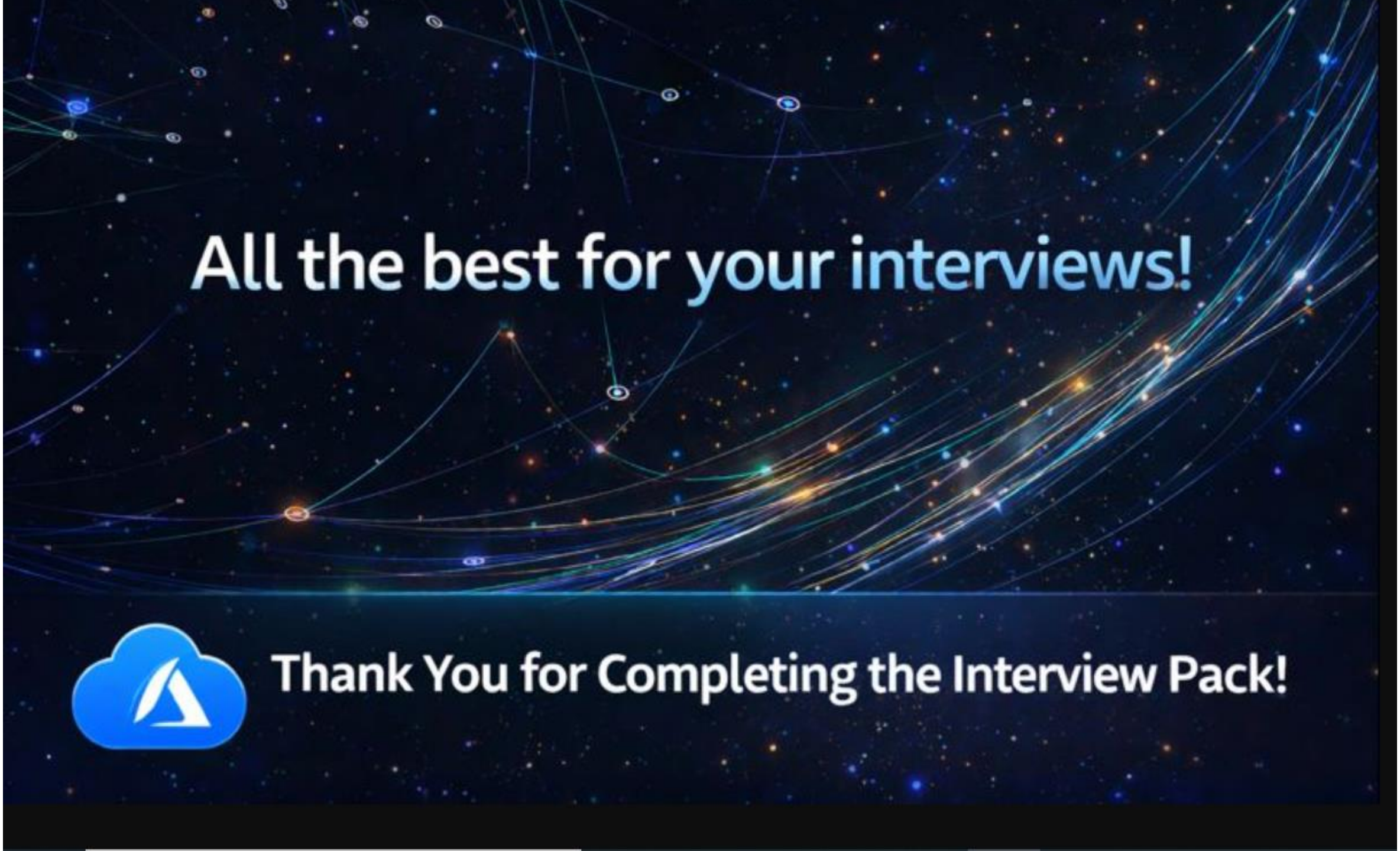
- A **publisher (sender)** sends a message to a **Topic**
- The message is **copied and delivered to multiple Subscriptions**
- Each **Subscription** acts like an independent queue
- Different subscribers can receive **the same message** and process it independently

Key idea: One message → many receivers

Difference between Topic and Queue

- Topic = One sender (publisher) -> Multiple receivers (subscribers)
- Queue = One sender -> one receiver




All the best for your interviews!
Thank You for Completing the Interview Pack!