FQR

Tuesday, April 9, 2024 12:00 PM

Hello Cx,

Thank you for contacting Microsoft Azure Support.

My name is Anya, I am the support engineer working with you on this Service Request - 2212xxxxxxxxxxx. To ensure all communications reach the appropriate contact and that the incident is kept up to date, please reply to this email thread with "Reply All".

From your email, I understand that you are having difficulties creating an App Service Domain as a result of insufficient payment history. Having reviewed your subscription, I see that you do not have sufficient payment history to purchase a domain.

As a result of the new App Service Domain policy introduced which requires sufficient payment history prior to purchasing a domain, your current subscription with its payment history will not pass the check that will allow you to purchase any domain and that is the reason you cannot create a custom domain.

However, I would recommend that you continue actively using your Azure resources, incur more charges, and you will be able to purchase an App services domain in the future.

I understand this may not be the response you might have been expecting, but this is due to the recent App services domain policy.

My sincere apologies for any inconvenience this may have caused as I would have loved to assist if this was within my capacity.

Please let me know if you have further concerns. I eagerly await your response.

LQR || Closure Email

Tuesday, April 9, 2024 12:01 PM

Hi Cx,

Thank you very much for your time on this case - 22xxxxxxxxxxxx.

With the confirmation in your previous email, we will be proceeding to archive this support request.

Please find below a quick summary of this support request.

Issue.

Resolution.

As there are no further questions or concerns at this time and regarding your email confirmation, I am going ahead to archive this ticket.

If you would like to rate the support you will be receiving a separate email after this one that has a brief questionnaire. Your feedback is extremely important to us and helps improve the service we provide. The questionnaire is three questions and only should take a moment of your time.

If you have any feedback or concerns on the handling of your case, please feel free to reach out to my manager, as listed below in my signature.

Thank you for choosing Microsoft.

Have a lovely day.

Hi Cx,

Thank you very much for your time on this case - 22xxxxxxxxxxxx.

With the confirmation in your previous email, we will be proceeding to archive this support request.

Please find below a quick summary of this support request.

Issue.

Resolution.

As there are no further questions or concerns at this time and regarding your email confirmation, I am going ahead to archive this ticket.

Your feedback is important to us. After this interaction, you will receive a separate closure email

with the opportunity to tell us about your experience.

If you have any feedback or concerns on the handling of your case, please feel free to reach out to my manager, as listed below in my signature.

Thank you for choosing Microsoft.

Have a lovely day.

==========

If you would like to rate the support you will be receiving a separate email after this one that has a brief questionnaire. Your feedback is extremely important to us and helps improve the service we provide. The questionnaire is three questions and only should take a moment of your time.

In addition to that, if you would like to leave feedback that is related to the product, you may do so here Ideas · Community (azure.com)

Sev A - Unresponsive

Tuesday, April 9, 2024 12:02 PM

While we will still handle your case with high priority and great care, I will be temporarily lowering the severity to B, since as per our official support terms, severity A is reserved for situations that involve a critical real-time service degradation that severely affects production or profitability.

Unresolved - Cx wants to sadly close

Tuesday, April 9, 2024 12:03 PM

I hope this email finds you well. I truly appreciate your patience and understanding throughout this process, and I want to extend my sincerest apologies for the inconvenience you've experienced with our Azure services.

Your feedback is invaluable to us, and I understand the frustration that arises when encountering issues with stability.
Rest assured, we take your concerns seriously, and we are committed to continuously improving our services to provide a more reliable experience for all our customers.

In light of the challenges you've faced, I completely understand your request to close the ticket. While we may not have an immediate solution from Azure's perspective, please know that we are actively working to address these issues internally and with our partners.

We will proceed with closing the ticket as per your request, but please don't hesitate to reach out if you encounter any further issues or if there's anything else we can assist you with. Your satisfaction is our top priority, and we are here to support you every step of the way.

Your feedback is important to us. After this interaction, you will receive a separate closure email with the opportunity to tell us about your experience.

If you have any feedback or concerns on the handling of your case, please feel free to reach out to my manager, as listed below in my signature.

Once again, I apologize for any inconvenience this situation has caused. Thank you for your patience and understanding.

Warm regards,

Unresponsive Cx

Tuesday, April 9, 2024 12:03 PM

Hello Hari

Unfortunately, we have been unable to contact you regarding your support request.

Perhaps this is not a good time for you due to other obligations or you no longer need assistance.

If your issue has been resolved, we always appreciate anything you can share as to the resolution as we can update our records and documentation to help us in future cases.

If you do want to continue working on this request, please respond to this email or contact me or my manager using the contact information in my signature.

Issue

Action Plan

Your feedback is important to us. After this interaction, you will receive a separate closure email with the opportunity to tell us about your experience.

If you have any feedback or concerns on the handling of your case, please feel free to reach out to my manager, as listed below in my signature.

Thank you for choosing Microsoft.

Have a lovely day.

We hope this message finds you well. We have been trying to reach you regarding your recent support request but haven't been successful.

We understand that you might be busy or perhaps your issue has already been resolved. If your issue is resolved, we would greatly appreciate it if you could share the details with us. This helps us improve our records and documentation for future cases.

If you still need assistance, please reply to this email or contact me or my manager directly using the contact information in my signature.

Issue:

Action Plan:

Your feedback is important to us. After this interaction, you will receive a separate closure email with the opportunity to tell us about your experience.

Should you have any feedback or concerns about how your case was handled, please feel free to reach out to my manager, whose contact details are listed below in my signature.

Thank you for choosing Microsoft. Have a wonderful day!

Best regards,

Untitled

Wednesday, April 10, 2024 10:53 AM

SR	Issue	Action Plan/Engineer
2403240050000280	on-prem database connectivity	Developer team has given feedback on the dumps. It seems that there is an issue with the name being used as the FQDN on the server. I recommended that the cx use the FQDN of the on-prem server **SqlConnection.ConnectionString Property (System.Data.SqlClient) | Microsoft Learn** **Just follow up on the 11th if the recommendation worked or in case he has further questions - Timi**
2402010030005244	There is high CPU usage on the Azure App Service plan	Cx has his own his own clients and they are yet to give him consent to close this case. The cx was impacted by the HIGH CPU outage that happened a while back. Although the issue is resolved, the cx and his clients have been monitoring the multiple app they have. I called him and we agreed to reach out to him next week to get updates from their end. **Follow up on the 15th to know updates from his end** - Muna
2404040050004368001	Logs verification	I have updated the collab case note with the necessary information. Reach out to main case owner in case of further assistance. If note close the collab - Biliki
2404080030001777	Action recommended: Reminder to implement disaster recovery strategies for your Azure App Service web apps by 31 March 2025	Please see the case note +++Next Action Plan Check if cx requires further assistance - Sarah
2403280050003206	Paying for double the normal usage (duplicate)	See case notes (I already spoke to you about it) - Yomi
2404030030003526	failed request with 499 error code	Follow up with the cx on the 12th and on the 16 (just like we discussed) - Chizaram
2403090050000657	Renew my domain	Follow up on the 12th just like we discussed - Ifeoma
2404040050004368001	Logs verification	Given the MCO the requested information on the collab case note. Reach out to him telling him that if he wants anything else or we can close the collab - abayomi
2403100030000467	disk utilization is high	Follow up with AVA and Saviour (ask savior to help check up with Maria) - Chika

Begging cx LQR

Wednesday, September 4, 2024 8:03 PM

Thank you very much for your time on this case - 240708xxxxxxxx. I appreciate your patience and cooperation as we worked through the ongoing investigation of the issue you presented to us.

With the confirmation in our Teams chat, we will be proceeding to temporarily archive this support request seeing as you are quite occupied at the moment.

Please find below a quick summary of this support request.

- **Issue Description:** Function app unable to reach On-Prem data gateway
- **Action Plan:** We were in the midst of investigation with the express route engineer

As discussed, and per your confirmation, pending your availability to continue the investigation, we will proceed to temporarily archive this ticket to avoid overwhelming you with emails. I understand that the delay in finding a permanent resolution may be frustrating, rest assured we are dedicated to resolving the issue you brought forward whenever you are free to continue the support.

When you are ready, please feel free to reach out to me so we can reopen the support request and complete the process. In the meantime, should you encounter any further issues or have additional concerns, please do not hesitate to reach out to me directly.
Your feedback is important to us. After this interaction, you will receive a separate closure email with the opportunity to tell us about your experience.
If you have any feedback or concerns on the handling of your case, please feel free to reach out to my manager, as listed below in my signature.

Thank you for choosing Microsoft.

Application Engineering team

Thursday, September 5, 2024 2:47 PM

The engineering team is the highest level of escalation from a technical perspective, which means that the issue is properly handled and investigated. They are performing a deep evaluation of the issue based on a number of factors which depending on the situation may include time to fix, time to test, risk of destabilization to the product and so on.
Kindly understand that we are actively working on this request and due to the nature of the escalated issues, time to fix may vary (sometimes 5-10 business days), hence the delay in resolution that may arise.

Perhaps you want us to only communicate with you when we get a resolution from the Product Group, kindly let us know. For now, we shall continue to follow up with the Product Group, in a bid to expedite actions and update you (**set frequency - 2 business days or every Tuesday, etc).**

Idle

Tuesday, November 26, 2024 2:16 PM

I hope this email finds you well. I wanted to follow up regarding ticket #[Ticket Number]. We haven't received any recent updates or feedback from your end, and the ticket has been idle for some time.

If you require further assistance or have any updates to provide, please don't hesitate to reach out to us. We're here to help and want to ensure your issue is resolved satisfactorily.

Thank you for your attention to this matter, and I look forward to hearing from you soon.

Product Limitation

Wednesday, March 12, 2025 1:26 PM

Hi Cx,

Thank you very much for your time on this case - 22xxxxxxxxxxxx.

With the confirmation in your previous email, we will be proceeding to archive this support request.

Please find below a quick summary of this support request.

Issue.

Resolution.

As there are no further questions or concerns at this time and regarding your email confirmation, I am going ahead to archive this ticket.

If you would like to rate the support you will be receiving a separate email after this one that has a brief questionnaire. Your feedback is extremely important to us and helps improve the service we provide. The questionnaire is three questions and only should take a moment of your time.

In addition to that, if you would like to leave feedback that is related to the product, you may do so here Ideas · Community (azure.com)

If you have any feedback or concerns on the handling of your case, please feel free to reach out to my manager, as listed below in my signature.

Thank you for choosing Microsoft.

Have a lovely day.

App Service Fundamentals

Wednesday, June 11, 2025 10:10 AM

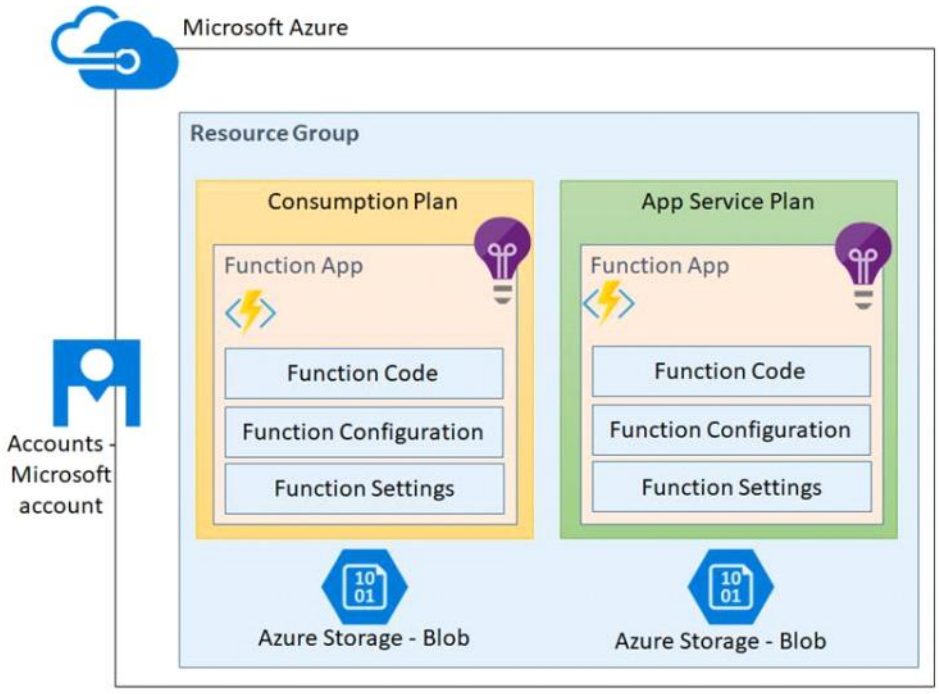
[Azure - Inside the Azure App Service Architecture | Microsoft Learn]

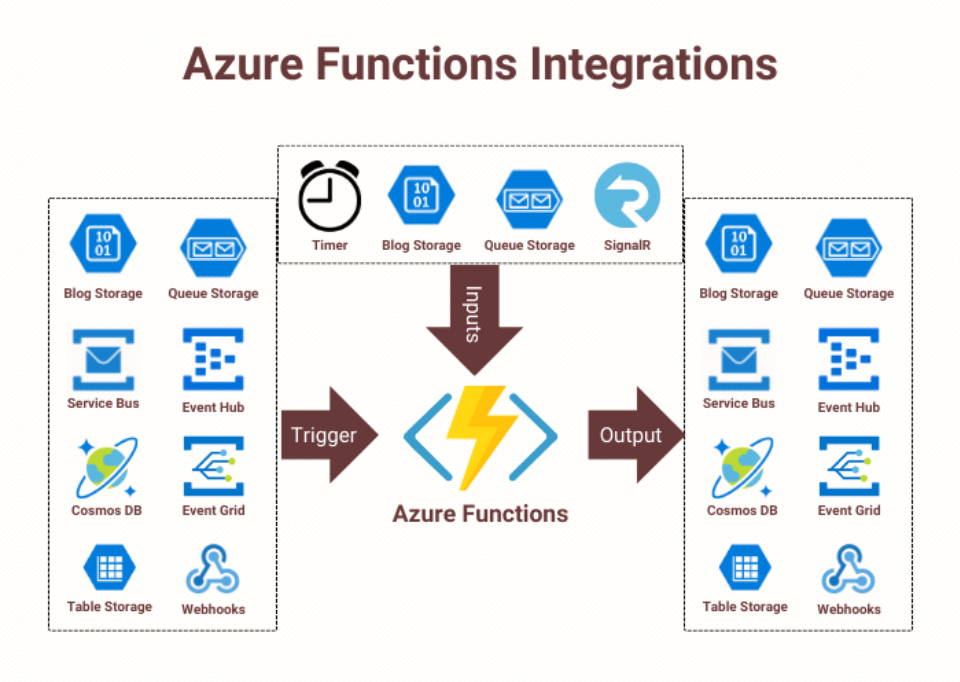
Function Apps

The main difference between app service and function apps -

Web apps are currently running on a server
Even if you turn off a web app you still pay for the instance, because the instance is allocated to you.

Function apps are only available on demand (a block of code that executes when there is requests)
The function app is stored on the storage account





https://learn.microsoft.com/en-gb/azure/azure-functions/functions-overview

Instance
An instance is an representation(or a component) of a portion of a physical machine (Worker)

Worker
The worker is the machine where the instance is extracted from. The worker's role is to load your code (The worker is what loads your code).
***Sliced Bread

Plans for function app

1. Dynamic/Elastic (scale out is handled automatically depending on the load) - Consumption (known for not being suitable for prod environment) and Flex Consumption. (Function premium is also known as Elastic Premium)
2. Dedicated/App Service plan - Function app can be run on a Dedicated plan. E.g. In case of stable IP requirement.

For function app/Serverless you are only billed only when your code is executed

C#, JavaScript, Python, NodeJS, Rust

Durable Function - These are long running events. They are used to run stateful functions - Durable Functions Overview - Azure | Microsoft Learn

FUNCTION KEYS
Every function in a function app has its own function key. To allows individual functions authenticate requests
├── Host Keys
├── System Keys (Do not touch)

BINDING MECHANISMS
Input and Output bindings - This is how the function app interacts with other

resources/Environments
Custom Handlers - Allows you to use other languages

App Service Identity Management

Wednesday, June 11, 2025 11:57 AM

Tenants handles all your subscription

Active Directory -is now Entra ID

RBAC - What is Azure role-based access control (Azure RBAC)? | Microsoft Learn - You are only give access based on your role and what you need. (controlling access)

Active Directory -is now Entra ID - > A central platform to manage users and resources

1. Authentication (second layer - MFA)
2. Authorization
3. Conditional Access (applying policies)
4. RBAC
5. SSO - Single Sign Ons
6. Principle of least privileged

Managed System Identity - helps give an unique resource identity

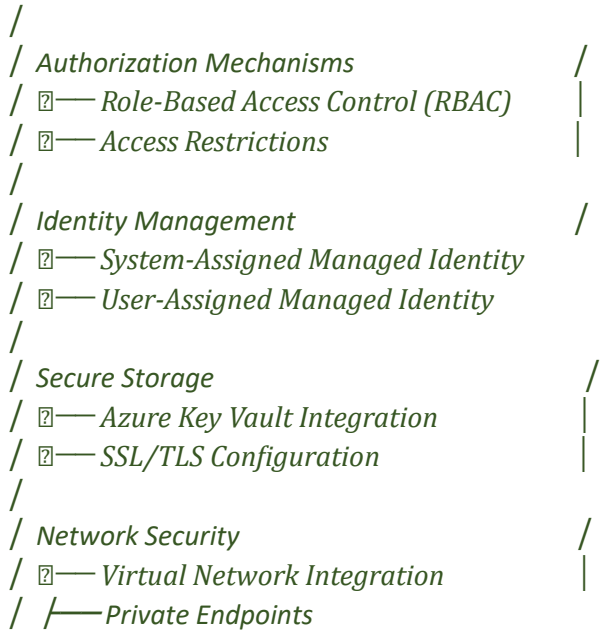
Annie's note

Key Concepts:

Identity and Access Management (IAM): The process of managing who (users, applications, services) can access what resources (applications, data, storage) in a secure and controlled manner.
Authentication: The process of verifying the identity of a user or application attempting to access a resource.
Authorization: The process of determining what level of access (read, write, delete) a user or application has to a resource after successful authentication.

Available Authentication and Authorization Mechanisms in Azure App Service:

Azure Active Directory (AAD):
A central cloud-based IAM service for Microsoft Azure.
Allows you to integrate user authentication with your Azure App Service using industry-standard protocols like OAuth 2.0 and OpenID Connect.
Enables you to implement role-based access control (RBAC) where users are assigned specific roles with defined permissions to access application resources.



user request -- app service -- identity provider -- token validation -- access granted

1. **Unauthenticated Request**: User accesses protected resource
2. **Redirect**: App Service redirects to configured identity provider
3. **Authentication**: User authenticates with identity provider
4. **Token Exchange**: Identity provider returns authentication token
5. **Validation**: App Service validates token
6. **Session Creation**: App Service creates authenticated session

Access Granted: User gains access to protected resources

RBAC - Access assigned based on roles. You can edit already built roles based on preference.

Access Restriction - Set Up Access Restrictions - Azure App Service | Microsoft Learn - You can also block service tag. Meaning access restriction applies to a client or resource.

When trying to assign a permission, you can assign to a scope (e.g. Resource Group or Subscription)

Key Vault can be used to provide granular security and management to Secrets/Keys/Credentials

INBOUND ACCESS RESTRICTION

- Deny or allow IPS
- VNET Related restriction
  - Subnet Based Access
  - Private Endpoint
  - Service Endpoint/Service Tag
- HTTPs Headers can be used for restrictions
- Network Security Group

OUTBOUND ACCESS RESTRICTON

- VNET Integration
- Network Security Group

- User Defined Groups
- Hybrid Network/Connection
- NAT Gateway
- WAF Policy

App Service Monitoring

Wednesday, June 11, 2025 2:58 PM

Metrics on the app gives you quantitative data of the application

Logs gives you qualitative data // measurement

Application Insights (the quantitative and qualitative values)

Annie's Note

Application Insights:
A service that collects detailed telemetry data from your application, including performance metrics, exceptions, user actions, and custom events.
Offers a robust set of features for analyzing, visualizing, and diagnosing application issues.
https://learn.microsoft.com/en-us/azure/azure-monitor/app/app-insights-overview

Benefits of Application Insights:
Detailed Performance Monitoring: Gain insights beyond basic request metrics. Drill down to identify slow code sections, database calls, or external API interactions impacting performance.
User Behavior Analysis: Track user actions within your application, understand user journeys, and identify areas for improvement.
Exception Tracking and Diagnostics: Pinpoint exceptions, errors, and crashes within your application code and diagnose root causes efficiently.
Custom Telemetry: Collect and analyze metrics specific to your application's functionalities for comprehensive monitoring.

Implementing Application Insights:
Install the Application Insights SDK for your chosen programming language (.NET, Node.js, Java, etc.) into your Azure App Service application code.
Configure the Application Insights resource within Azure Monitor and connect it to your application.
Utilize the SDK to instrument your code and collect telemetry data (performance counters, exceptions, custom events).
https://learn.microsoft.com/en-us/azure/azure-functions/functions-monitoring

Application Insights Features:
Interactive Dashboards: Create custom dashboards to visualize key performance indicators (KPIs) and application health metrics.
Log Analytics Queries: Utilize powerful Kusto Query Language (KQL) to analyze telemetry data, identify trends, and troubleshoot issues.
Alerts and Notifications: Set up alerts based on specific telemetry data to proactively address potential problems.
Funnels and User Flows: Visualize user journeys through your application and identify drop-off points.
Anomaly Detection: Leverage built-in anomaly detection capabilities to identify unusual spikes or drops in performance metrics.

Logs in Azure App Service:
Records of events and activities that occur within your application or its environment.
They provide detailed information for troubleshooting issues, debugging code, and analyzing application behavior.

Types of Logs in Azure App Service:

HTTP Logs: Record details of user requests received by your application, including timestamps, URLs, HTTP status codes, and response sizes.

Platform Logs: Logs generated by the Azure platform itself, providing details about events related to your App Service instance, such as deployments, restarts, and scaling operations.

Custom Application Logs: Logs written by your application code using logging libraries or frameworks. These logs can contain detailed information about application logic execution, errors encountered, or user actions.

Benefits of Utilizing Logs:

Troubleshooting and Debugging: Logs help pinpoint the root cause of application errors and exceptions by analyzing specific events and error messages.

Security Monitoring: Logs can be used to identify security incidents or suspicious activity within your application.

Performance Analysis: Analyze logs to understand how specific code sections impact application performance and identify bottlenecks.

Audit Trails: Logs can be used to maintain historical records of user actions and application modifications for auditing purposes.

Working with Logs in Azure Monitor:

Azure Monitor Logs is a service that collects, stores, and analyzes log data from various Azure resources, including App Service applications.

You can access logs through the Azure portal or utilize Log Analytics workspaces for advanced querying and analysis.
https://learn.microsoft.com/en-us/azure/app-service/troubleshoot-diagnostic-logs

Key Features of Azure Monitor Logs for App Service:

Log Search: Search through collected logs based on keywords, timestamps, and specific log categories.

Log Analytics Queries (KQL): Utilize KQL, a powerful query language, to filter, aggregate, and analyze log data for deeper insights.

Diagnostics Logs: Access detailed diagnostic logs from your app service instances for advanced troubleshooting.

Alerting on Log Events: Configure alerts based on specific log entries to be notified of critical events or errors.

Alerts in Azure Monitor:

Rules defined based on specific thresholds or conditions for monitoring data collected from your App Service application (metrics, logs).

When a condition defined in an alert is triggered, Azure Monitor initiates a notification to inform you of the potential issue.

Benefits of Configuring Alerts:

Proactive Response: Alerts notify you of potential problems before they escalate into critical outages or impact users.

Faster Troubleshooting: Early notification allows you to investigate and address issues swiftly, minimizing downtime.

Improved Application Reliability: By proactively addressing potential problems, you enhance the overall reliability of your application.

Reduced Operational Costs: Proactive issue resolution through alerts can prevent costly downtime and resource disruptions.

Types of Alerts in Azure Monitor:

Metric Alerts: Triggered when a monitored metric (e.g., CPU utilization, request latency)

exceeds or falls below a predefined threshold.

Log Alerts: Activated when specific log entries containing keywords or error messages are identified within your application logs.

Application Insights Alerts: Defined based on conditions set within Application Insights, such as exceeding a specific threshold for exceptions or failed transactions.

Configuring Alerts in Azure Monitor:

Access Azure Monitor within the Azure portal.

Choose the resource you want to monitor (your App Service application).

Select "Alerts" and define a new alert rule.

Choose the type of alert (metric, log, Application Insights).

Configure the alert condition based on specific metric thresholds, log queries, or Application Insights criteria.

Set up notification channels for receiving alerts (e.g., email, SMS, webhooks).

Defining Monitoring Goals:

Clearly define the goals you aim to achieve with your monitoring strategy.

Examples:

Ensure application availability and uptime.

Monitor user experience and identify performance bottlenecks.

Proactively detect and address potential issues before impacting users.

Optimize resource utilization for cost-efficiency.

Establishing a Baseline:

Before implementing changes, collect baseline data for key metrics (e.g., request latency, CPU utilization) to establish a reference point.

This baseline helps you identify deviations and potential anomalies later.

Monitoring Throughout the Application Lifecycle:

Integrate monitoring throughout the entire application lifecycle, from development and testing to deployment and ongoing operation.

Monitor code changes during development to identify potential issues early.

Utilize Application Insights for detailed performance analysis during testing.

Implement continuous monitoring in production for proactive problem detection.

Leveraging Logs Effectively:

Go beyond simple log collection – establish a log management strategy for efficient storage, access, and analysis.

Utilize Kusto Query Language (KQL) to filter and analyze logs for deeper insights.

Correlate logs from different sources (App Service, databases, etc.) for comprehensive troubleshooting.

Alerting and Notification Strategies:

Configure meaningful alerts based on your monitoring goals, avoiding alert fatigue with excessive notifications.

Set up multi-layered alerting with varying severity levels for critical, warning, and informational alerts.

Integrate alerts with notification channels like email, SMS, or team collaboration tools for timely response.

Continuous Monitoring Improvement:

Regularly review monitoring data and identify areas for improvement.

Refine your monitoring strategy based on application usage patterns and identified bottlenecks.

Integrate monitoring data with DevOps pipelines for continuous feedback and performance

optimization.

WORKER PROCESSES and APPLICATION POOL

Wednesday, June 11, 2025 3:22 PM

WORKER PROCESSES and APPLICATION POOL

.NET is Microsoft is an Enterprise Microsoft framework
.NET Core is an open source framework

Introduction to IIS Architectures | Microsoft Learn
IIS has two main modes
├── Kernel Mode- The part that allows native functionality (e.g. HTTP/Protocol listener - request queueing, caching, security filtering, throttling)
├── User Mode- Focuses on handling the user code (e.g. WAS and WWW [contains configuration])

The Application Pool - When the code has been loaded on the worker instance from the storage account, for the code to start executing a worker process will be created. Make sure that the processes run independent of each other

A worker process (w3wp) is a process that is activated to run your code. The worker process does this by creating application pool to run a process

WAS (windows activation Service) is responsible for creating application pool and worker process

WWW Service is an intermediary that between the protocol listener and the WAWS. This translates request from Protocol listener to the WAS

Workflow of a requests ==== > Protocol/Http Listener (http.sys) ---> WWW3 ---> WAS

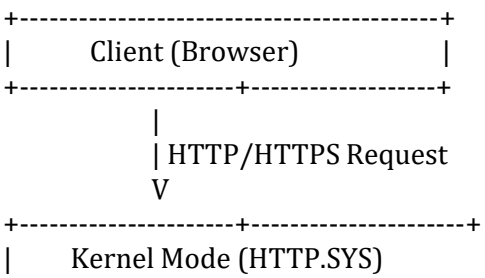
WORKER PROCESS: Every app pool is running its own worker process

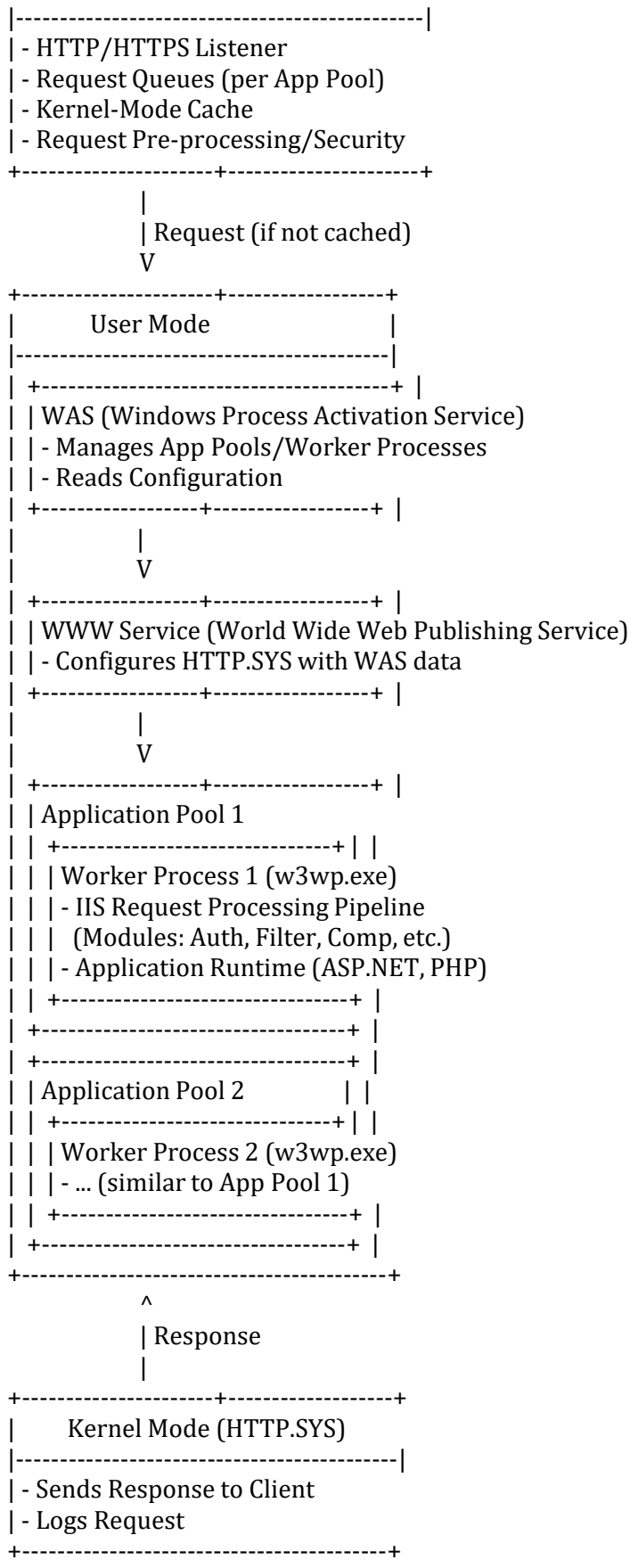
APP POOL: is like a child process in the worker process. Prevents one thread from affecting another thread

ISS modular? What does it really mean

Dynamic concurrency is handled by Compression module (app/process request)
Static Concurrency

When a client makes




- HTTP/HTTPS Listener
- Request Queues (per App Pool)
- Kernel-Mode Cache
- Request Pre-processing/Security
Request (if not cached)
User Mode
WAS (Windows Process Activation Service)
- Manages App Pools/Worker Processes
- Reads Configuration
WWW Service (World Wide Web Publishing Service)
- Configures HTTP.SYS with WAS data
Application Pool 1
Worker Process 1 (w3wp.exe)
- IIS Request Processing Pipeline
(Modules: Auth, Filter, Comp, etc.)
- Application Runtime (ASP.NET, PHP)
Application Pool 2
Worker Process 2 (w3wp.exe)
- ... (similar to App Pool 1)
Response
Kernel Mode (HTTP.SYS)
- Sends Response to Client
- Logs Request

ANNIE'S NOTE

Key Architectural Components of IIS

The IIS architecture is primarily divided into two layers: **Kernel Mode** and **User Mode**. This separation enhances security and performance by allowing critical components to operate with higher privileges in the kernel and application-specific processes to run with fewer privileges in user mode.

1. Kernel Mode

The kernel mode in IIS primarily consists of `HTTP.SYS`.

* **HTTP.SYS (HTTP Protocol Stack):**
  * **Role:** This is the core component in the kernel mode. It's a kernel-mode device driver that listens for HTTP and HTTPS requests, performs preliminary processing, and efficiently routes them to the appropriate user-mode processes.
  * **Key Functions:**
    * **Request Queuing:** `HTTP.SYS` maintains separate request queues for each application pool, ensuring that requests are buffered and delivered to the correct worker processes.
    * **Kernel-mode Caching:** It can cache static content and some dynamic content responses directly in the kernel, significantly improving performance by serving frequently accessed content without the overhead of context switching to user mode.
    * **Request Pre-processing and Security Filtering:** It performs initial validation and filtering of requests, enhancing security by dropping malicious or malformed requests early.
    * **Quality of Service (QoS):** Includes features like connection limits, connection timeouts, queue-length limits, and bandwidth throttling.

2. User Mode

The user mode handles the bulk of the web server's logic and application processing. It comprises several key services and processes:

* **WAS (Windows Process Activation Service):**
  * **Role:** WAS is the central component responsible for managing application pools and worker processes. It's a crucial part of IIS 7.0 and later, extending the process model introduced in IIS 6.0.
  * **Key Functions:**
    * **Process Management:** Starts, stops, and recycles worker processes based on configuration and incoming requests.
    * **Configuration Management:** Reads configuration information from `applicationHost.config` (the main IIS configuration file) and passes it to the WWW Service.
    * **Application Pool Management:** Monitors the health and state of application pools and their associated worker processes.

* **WWW Service (World Wide Web Publishing Service):**
  * **Role:** This service acts as an interface between WAS and `HTTP.SYS`. It uses the configuration information provided by WAS to configure `HTTP.SYS` for specific websites and application pools.
  * **Key Functions:**
    * **Configuration Integration:** Integrates configuration settings with `HTTP.SYS` to ensure proper request routing and processing.
    * **Site Management:** Manages the lifecycle of websites and their bindings (e.g., port numbers, host headers).

* **Worker Process (w3wp.exe):**
  * **Role:** This is where your web applications actually run. Each application pool typically runs

in its own worker process (`w3wp.exe`), providing isolation between different applications.
   * **Key Functions:**
      * **Request Processing Pipeline:** Contains a series of ordered modules that process incoming HTTP requests and generate responses. This pipeline is highly extensible, allowing developers to add custom modules.
      * **Application Hosting:** Loads and executes application code (e.g., ASP.NET, PHP) based on the request URL.
      * **Resource Management:** Manages memory, CPU, and other resources consumed by the hosted applications.
      * **Isolation:** By running applications in separate worker processes, IIS ensures that a crash or error in one application does not affect others.

* **Application Pools:**
   * **Concept:** Application pools are a fundamental concept in IIS for isolating web applications. An application pool is a group of one or more URLs that are served by a single worker process (or multiple worker processes in a "web garden" scenario).
   * **Benefits:**
      * **Isolation:** Prevents one application from affecting another if it crashes or has a memory leak.
      * **Security:** Allows different applications to run under different security identities.
      * **Management:** Simplifies managing and recycling applications independently.

* **Modules:**
   * **Concept:** IIS 7.0 and later versions have a modular architecture, meaning its functionalities are broken down into discrete components called modules.
   * **Benefits:**
      * **Extensibility:** Allows for adding or removing specific functionalities, reducing the attack surface and improving performance by only loading necessary modules.
      * **Customization:** Developers can create custom modules to extend IIS functionality.
   * **Examples:**
      * **Authentication Modules:** Anonymous, Basic, Windows Authentication.
      * **Security Modules:** Request Filtering, URL Authorization.
      * **Content Modules:** Static File Handler, Default Document, Directory Listing.
      * **Compression Modules:** Static and Dynamic Content Compression.
      * **Logging Modules:** HTTP Logging.

How IIS Processes an HTTP Request (Simplified Flow)

Here's a step-by-step breakdown of how IIS handles an HTTP request:

1. **Client Request:** A client (e.g., a web browser) sends an HTTP/HTTPS request to the IIS server.
2. **HTTP.SYS Interception:** `HTTP.SYS` in kernel mode intercepts the request on the configured port (e.g., 80 for HTTP, 443 for HTTPS).
3. **Static Content Check (Kernel Cache):** `HTTP.SYS` first checks if the request is for static content and if that content is already in its kernel-mode cache. If so, it serves the content directly from the cache, bypassing user mode, and sends the response back to the client.
4. **Request Routing (Dynamic Content):** If the request is for dynamic content or not found in the cache, `HTTP.SYS` determines which application pool should handle the request based on the URL and site bindings. It then places the request into the appropriate application pool's request queue.
5. **WAS Notification:** If no worker process is currently active for that application pool, `HTTP.SYS` notifies WAS to start one.

6. **WAS Configuration:** WAS retrieves the configuration information for the requested application pool and site from `applicationHost.config` and passes it to the WWW Service.
7. **WWW Service Configuration:** The WWW Service uses this information to configure `HTTP.SYS` for the worker process.
8. **Worker Process Activation:** WAS starts a new worker process (`w3wp.exe`) for the application pool (if one isn't already running or available).
9. **Request Processing in Worker Process:** The worker process retrieves the request from the queue. Inside the worker process, the request goes through a series of ordered events, known as the **IIS Request Processing Pipeline**. Various modules (authentication, authorization, logging, compression, static file handler, ISAPI/ASP.NET modules, etc.) process the request at different stages.
10. **Application Execution:** If the request requires application code execution (e.g., an ASP.NET page or a PHP script), the relevant ISAPI extension (like `aspnet_isapi.dll` for older ASP.NET versions or the ASP.NET Core Module for modern ASP.NET Core applications) or FastCGI handler takes over, loads the application's runtime, and executes the code.
11. **Response Generation:** The application code or static file module generates the appropriate response (HTML, JSON, image, etc.).
12. **Response Back to HTTP.SYS:** The worker process sends the response back to `HTTP.SYS`.
13. **Client Response:** `HTTP.SYS` sends the response back to the client and, if configured, logs the request.

WORKER PROCESSES and APPLICATION POOL

Wednesday, June 11, 2025 3:22 PM

Perfect, Pelumi. Let's now go step by step, with each concept **first explained like you're a child**, and then followed by a **technical breakdown** — all from the angle of someone **troubleshooting performance issues in Azure App Service**.

◇ 1. What is a Worker Process and Application Pool?

🧒 **Child-like Explanation:**

Imagine you're in a school, and every classroom is a different web app. Each class has a teacher (the **worker process**) that teaches and handles questions. The school assigns each classroom their own space (this is like the **application pool**), so one noisy class doesn't disturb the others.

🧠 **Technical Explanation:**

- A **worker process** (w3wp.exe) runs your web application code inside IIS.
- An **application pool** manages worker processes and isolates them from other apps — like virtual boundaries.
- Azure App Service uses a **per-app isolation model** similar to application pools for security and performance.

💡 **Troubleshooting Insight:**

If your app is slow or crashes:

- Check if the worker process is consuming too much CPU or memory.
- Look for app pool recycles (auto-heal, memory thresholds, config issues).

◇ 2. What Are WAS and W3SVC Services?

🧒 **Child-like Explanation:**

Think of a school receptionist (W3SVC) who gets all visitors (HTTP requests) and tells them which classroom (app) to go to. There's also a janitor (WAS) who turns on the lights and unlocks the class before lessons start — so the teacher can do their job.

🧠 **Technical Explanation:**

- **W3SVC (World Wide Web Publishing Service):** Accepts incoming HTTP requests and maps them to the correct IIS site.
- **WAS (Windows Process Activation Service):** Starts and stops worker processes as needed.

💡 **Troubleshooting Insight:**

- If WAS fails → App pool won't start → App won't load → 503 Service Unavailable.
- Use "Diagnose and Solve Problems" or App Service diagnostics to check process start errors.

◇ 3. What is Kernel Mode vs User Mode?

🧒 **Child-like Explanation:**

Your computer has two "work areas":

- **Kernel Mode** is like a secure, behind-the-scenes control room.
- **User Mode** is where apps (like games or browsers) run — closer to what people see.

🧠 **Technical Explanation:**

- **Kernel Mode**: Low-level OS components like HTTP.sys run here. It listens for web requests before IIS touches them.
- **User Mode**: IIS, your app, and w3wp.exe run here. Most crashes, memory leaks, and slowness happen here.

💡 **Troubleshooting Insight:**

- If requests don't reach your app → check **network stack or access restrictions**.

- If requests reach your app but are slow → focus on **User Mode diagnostics** (App Insights, Kudu).

◇ 4. Classic vs Integrated Mode (IIS Execution Modes)

🧒 **Child-like Explanation:**
Think of two ways to organize a team:

- **Classic Mode**: You have two separate teams (IIS and ASP.NET), and they only talk at certain points.
- **Integrated Mode**: One big team works together from the beginning.

🧠 **Technical Explanation:**

- **Classic Mode**: ASP.NET modules can only handle requests with managed extensions (.aspx, .cshtml).
- **Integrated Mode** (used in Azure): Modules run as part of IIS pipeline, so they can inspect and modify every request type.

💡 **Troubleshooting Insight:**
Integrated mode enables consistent diagnostics, easier custom modules, and fewer surprises in behavior.

◇ 5. Static vs Dynamic Concurrency

🧒 **Child-like Explanation:**
Imagine you're baking cookies.

- **Static concurrency** is like having a fixed number of ovens — if all are busy, new cookies wait.
- **Dynamic concurrency** is like magically adding more ovens as more people show up.

🧠 **Technical Explanation:**

- **Static Concurrency (Web Apps):** Number of threads/requests is fixed based on the plan size.
- **Dynamic Concurrency (Functions Premium):** Azure adjusts the number of parallel function executions based on load and telemetry.

💡 **Troubleshooting Insight:**

- Web Apps: Look at thread pool starvation, queue length, and long-running requests.
- Functions: Monitor cold starts, throughput, scaling delays.

◇ 6. How They All Work Together – Internal Flow in Azure App Service

🧒 **Child-like Story:**

1. A person knocks at your school's front door (client makes a request).
2. The receptionist (W3SVC) checks who they want to see.
3. The janitor (WAS) ensures the class is open.
4. The teacher (worker process) hears the question and answers.
5. The visitor leaves with the answer (response is sent back).

🧠 **Technical Flow:**

1. Client sends HTTP(S) request.
2. **Azure Load Balancer** routes it to a front-end.
3. If **Access Restrictions** or **Private Endpoint** apply, the request is filtered.
4. **HTTP.sys** in Kernel Mode receives and queues it.
5. **W3SVC** routes to the right app.
6. **WAS** starts w3wp.exe if not already running.
7. **Worker Process** executes code via the ASP.NET pipeline (middleware, routes, DB calls).
8. Response travels back the same way.

💡 **Troubleshooting Insight:**

- 403/503 before app is hit → Front-end, access rules, WAS issues.
- Slow requests after app is hit → code, database, thread pool, memory.

☑ Summary Diagram (Simple Terms)

Client → Azure Front-End → HTTP.sys (Kernel)
→ W3SVC → WAS → App Pool → Worker Process (User Mode)
→ App Pipeline → Response → Client

☑ Bonus Tip: When Troubleshooting

Problem	What to Check	Tools
App Won't Start	WAS / App Pool startup	Diagnose & Solve
Slowness	Thread starvation, high CPU	Kudu → Process Explorer
Crashes / Memory	GC pressure, leaks	Memory dumps, App Insights
Cold Start	Dynamic concurrency	Premium Plan, Pre-warm
Not reachable	Access restrictions, Private Link	Networking tab, NSG logs

Would you like this formatted into a **cheat sheet**, or should we build a **case-based walkthrough** (e.g., "App is slow," "Function times out," etc.) next?

App Service Networking

Thursday, June 12, 2025 2:00 PM

In regards to Networking in app service, it is divided in to two
- Single-Tenant (Isolated Network) - App Service Environment
- Multi-Tenant (Shared)

INBOUND ACCESS RESTRICTION
- Deny or allow IPS
- VNET Related restriction
  - Subnet Based Access
  - Private Endpoint
  - Service Endpoint/Service Tag
- HTTPs Headers can be used for restrictions
- Network Security Group

****Private Endpoint create an NIC in the subnet of the VNET

OUTBOUND ACCESS RESTRICTON
- VNET Integration
- Network Security Group
- User Defined Groups - all the traffic leaving my App should go through a tunnel
- Hybrid Network/Connection - TCP
- NAT Gateway
- WAF Policy

Unlike traditional virtual machines where you have full control over the network interface, App Service is a Platform-as-a-Service (PaaS) offering, meaning Microsoft manages the underlying infrastructure. Therefore, its networking features are designed as abstractions that integrate with Azure's virtual networking capabilities.

There are two primary deployment types for Azure App Service, each with distinct networking characteristics:

1. **Multi-tenant App Service:** This is the default, shared public service (Free, Shared, Basic, Standard, Premium, PremiumV2, PremiumV3, and Elastic Premium SKUs). Apps in this environment share underlying infrastructure but are logically isolated.
2. **Single-tenant App Service Environment (ASE):** This is a fully isolated and dedicated environment (Isolated SKU) that runs directly within your Azure Virtual Network (VNet).

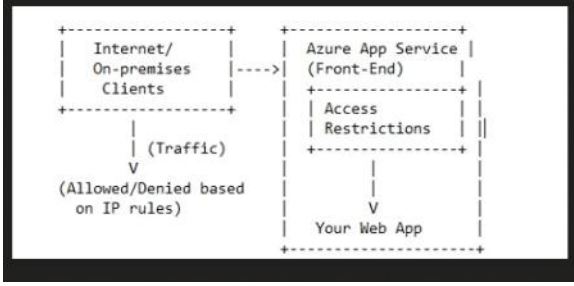
Let's explore the key networking features:

Networking Features for Multi-tenant App Service

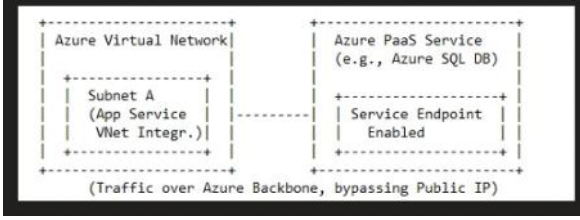
For multi-tenant App Service plans, you have several features to control network access:

Inbound Networking Features

* **Access Restrictions (IP Restrictions):**
  * **Role:** This feature allows you to define a list of IP addresses or IP ranges that are allowed or denied access to your App Service app.
  * **Mechanism:** You configure allow/deny rules based on IPv4 or IPv6 addresses. You can also specify HTTP headers to filter requests.
  * **Use Case:** Securing your app to be accessible only from your corporate network, specific client IPs, or Azure services with known outbound IPs.
  * **Diagram Snippet:**

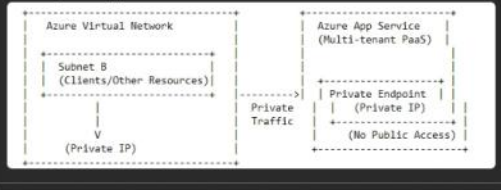


* **Service Endpoints:**
  * **Role:** Extends your Azure Virtual Network's private address space over the Azure backbone network. It allows you to secure access to Azure PaaS services (like Azure SQL Database, Storage Accounts, Key Vault) from specific subnets in your VNet.
  * **Mechanism:** When you enable a service endpoint for a subnet and a PaaS service, traffic from that subnet to the PaaS service is routed directly over the Azure backbone, bypassing the public internet. The PaaS service can then be configured to only allow connections from that specific subnet.
  * **Use Case:** Providing secure and optimized connectivity from your App Service (using VNet Integration) to other PaaS services within Azure, ensuring traffic never traverses the public internet.
  * **Diagram Snippet:**



Aniekan Udoh 6/12/2025 2:32 PM

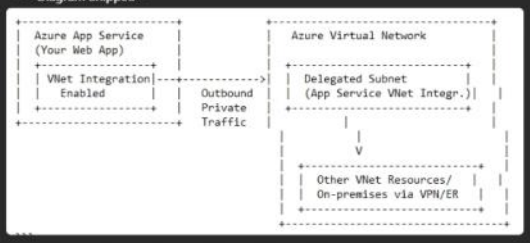
* **Private Endpoint:**
  * **Role:** Provides a private IP address for your App Service app within your Azure Virtual Network, making your app accessible only from your VNet and connected networks (on-premises via VPN/ExpressRoute). It completely removes public internet exposure for your app.
  * **Mechanism:** Creates a Network Interface Card (NIC) in a subnet of your VNet, which is assigned a private IP address. DNS resolution for your App Service's public hostname is then configured to resolve to this private IP within your VNet (using Azure Private DNS Zones).
  * **Use Case:** Hosting internal-only applications, enhancing security by ensuring no public internet access to your app, and compliance requirements.
  * **Key Difference from VNet Integration:** Private Endpoint is for **inbound** private access to your app, while VNet Integration (discussed below) is for **outbound** access from your app. You can use both together for fully private inbound and outbound communication.
  * **Diagram Snippet:**



Aniekan Udoh 6/12/2025 2:34 PM

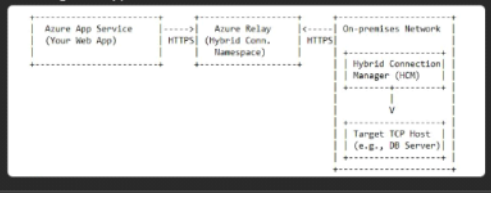
Outbound Networking Features

* **Regional Virtual Network (VNet) Integration:**
  * **Role:** Allows your App Service app to make **outbound** calls to resources within your Azure Virtual Network, including resources in peered VNets or on-premises networks connected via VPN/ExpressRoute.
  * **Mechanism:** Your App Service workers are "virtually injected" into a dedicated subnet within your VNet by mounting virtual interfaces with addresses from that subnet. This makes your App Service behave as if it's part of your VNet for outbound traffic.
  * **Key Points:**
    * Only for outbound traffic from your app.
    * Does not provide inbound private access to your app (use Private Endpoint for that).
    * Requires a dedicated subnet (at least a /28 block) for integration.
    * Allows your app to use DNS servers configured in your VNet.
  * **Use Case:** Accessing Azure SQL Database, Azure Storage, custom VMs, or on-premises resources that are VNet-connected and have private IP addresses.
  * **Diagram Snippet:**



Aniekan Udoh 6/12/2025 2:34 PM

* **Hybrid Connections:**
  * **Role:** Enables your App Service app to access TCP resources (host and port) in on-premises networks or other networks that can make outbound calls to Azure over port 443.
  * **Mechanism:** Built on Azure Relay. You install a "Hybrid Connection Manager" (HCM) agent on a server within your target network (e.g., on-premises). Both the App Service infrastructure and the HCM establish outbound connections to an Azure Relay endpoint over port 443. This creates a secure "tunnel" for TCP traffic between your app and the specific target host:port.
  * **Use Case:** Connecting to legacy databases, mainframes, or other TCP endpoints on-premises without requiring a full site-to-site VPN. Ideal for scenarios where only a few specific endpoints need to be accessed.
  * **Limitations:** Only TCP, no UDP. Cannot mount drives or join domains.
  * **Diagram Snippet:**



Aniekan Udoh 6/12/2025 2:35 PM

Network Security Groups (NSGs) and User-Defined Routes (UDRs) with App Service

* **NSGs (Network Security Groups):**
  * **Role:** For ASEs, you can apply NSGs directly to the ASE's subnet to control inbound and outbound traffic at the network level.
  * **Multi-tenant App Service:** When using VNet Integration or Private Endpoint, NSGs can be applied to the integrated subnet or the subnet where the Private Endpoint is located, controlling traffic to/from your app.
* **UDRs (User-Defined Routes):**
  * **Role:** For ASEs or VNet Integrated apps, UDRs allow you to force all outbound traffic from your App Service through a specific next hop (e.g., an Azure Firewall, a Network Virtual Appliance) for centralized inspection and control. This is often called "forced tunneling."

Static Web App

Monday, June 16, 2025 10:19 AM

Static Web App - use case is for hosting single page application (Login page). ← JavaScript/React/Vue
Azure Static Web Apps - Training | Microsoft Learn || What is Azure Static Web Apps? | Microsoft Learn ||
Quickstart: Building your first static site with the Azure Static Web Apps | Microsoft Learn

They are popular because they are globally distributed - No Region

The back end is on the Function app?

Static web app is created alongside with a function app (managed function). You can also create your own function for the static web app

It uses client side rendering.

Limitations
No Dynamic content
Client side rendering
No complex authentication procedure

Authentication and Routing

Azure Function Introduction

Tuesday, June 17, 2025 11:17 AM

Functions Introduction
Implement Azure Functions - Training | Microsoft Learn
Execute an Azure Function with triggers - Training | Microsoft Learn
Chain Azure Functions together by using input and output bindings - Training | Microsoft Learn
Create a long-running serverless workflow with Durable Functions - Training | Microsoft Learn

You only pay for "***execution time***"

An azure function - majorly one of the characteristics is event driven. A serverless resources, that runs app code in
Response to events.

Key Things
├── Serverless
├── Event Driven
├── Only pay for execution

Some Events
├── Blob Trigger - input binding
├── Output Binding

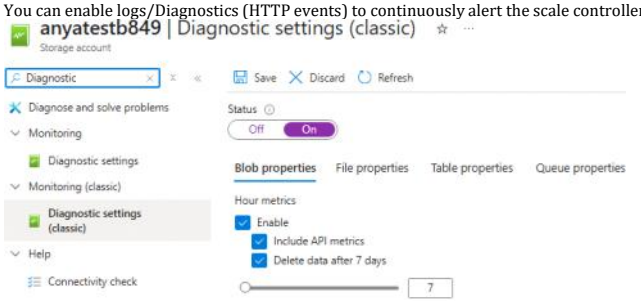
Consumption plan
There is a certain amount of time for executions for consumption which is free. If your function execution exceed this allocated time you will get charged>>>time or number of executions is free

- scale controller is responsible for allocating and deallocation of instances to a functions.

Cold Start - No available instances to run instance. It should only take seconds. Because the scale controller knows that an event has been triggered an allocates the instance.

If you are in a consumption plan you don't own the worker

You can enable logs/Diagnostics (HTTP events) to continuously alert the scale controller



Elastic Plan/Elastic Premium -
No cold starts because there are readily available instances. This allows functions allow you to run functions on individual instances.
├── Using Auto Scaling

Concurrency allows a particular instance to run a certain amount of instances. Instances picks up functions that need

NB: Elastic Scaling -> Automatic allocation and deallocation of instances

Scaling -> Horizontal (adding instance) and Vertical (adding compute) Scaling

Function and Logic - Both event driven but the difference
├── Logic app teams require API triggers
├── Function app are app code triggers

├── Webjob (run on kudu) -> Continuous web job and is also triggered by the Events. It uses the compute resources of the app service plan.
├──

RUNTIME required for function app to run. Runtime is like a kernel for the function app

RUNTIME3 is no longer supported. Because security patches

In other to support cx in RUNTIME 3, cx has to move to RUNTIME 4 - Chimezie

CONNECTION STRING - is a string that specifies information about a data source and how to connect to it. It's commonly used to connect app to databases message

Ensure your function runs locally FIRST

Second Goal:

ZIP deploy - package application code and put in the storage account. If you are doing this you will be needing website Run your functions from a package file in Azure | Microsoft Learn

Functions can be running in a container app

Triggers and bindings in Azure Functions | Microsoft Learn

Troubleshoot MOSTLY from Diagnose and solve

Managed Identity - Security and communication -> In **Azure**, a **Managed Identity** is used to authenticate to Azure services without storing credentials in code.
├── Used when we want to connect to various applications/DB (a connector)

Major different
├── System identity is created with the resources. If the resource deleted the identity is also deleted
├── User identity can be used on multiple resources. And it is not deleted alongside a resources

Blob Trigger

Tuesday, June 17, 2025 12:29 PM

Triggers

Wednesday, June 18, 2025 10:03 AM

Triggers are what invokes a functions
Every function must have at least one trigger to run

Common use of Trigger

- Https
- Timer
- blob

Dead letter queue - A resource ….ChatGPT explain
Service Bus (Explain ChatGPT)

https://www.serverlesslibrary.net/sample/c8ca9ad7-552c-4807-9f77-1d88266895db
Optional

https://learn.microsoft.com/en-us/azure/azure-functions/functions-event-grid-blob-trigger?pivots=programming-language-python
Task 1

https://learn.microsoft.com/en-us/azure/event-grid/blob-event-quickstart-portal?toc=%2Fazure%2Fazure-functions%2Ftoc.json
Task 2

Durable Functions

Thursday, June 19, 2025 12:10 PM

Durable Functions

[Durable Functions Overview - Azure | Microsoft Learn]

[Quickstart: Create a Python Durable Functions app | Microsoft Learn]

App Service Deployment

Monday, June 23, 2025 11:03 AM

Getting your code to be executable to the host machine

Making a SW App available for use, Installing, configuring and running (making it executable) it on a server

There should be a host that runs the machine.

Key Stages of Application deployment

- Development - code is written, code is tested
- Build and packaging - compilation
- Testing
- Deployment execution

Deployment makes your application accessible for users

Deployment best practices - Azure App Service | Microsoft Learn

Deployment Mechan

Deployments Process

- Continuous
- Single phase

Deployments Methods

ZIP/WAR - Deploy Files - Azure App Service | Microsoft Learn

Run from Package (Major Advantage)

- allows fast startup of the site
- Prevents file locking
- Reduces cold start
- Lighter

LOCAL GIT

Docker - Container

Friday, June 27, 2025 11:00 AM

Container - Your application is packaged in a way that it runs in isolation. It does not have to interact with the internla

A standard unitf

Wordpress

Tuesday, July 1, 2025 12:10 PM

WordPress on App Service

Plugin: Code bases that adds functionality to you app

Wordpress.org

App URL/wp-admin

Logging (always enable logging), Also
WP_DEBUG, true > enable error on the webpage

wp-content

wp-config.php > for managing configuration setting (including password, routing)

WordPress - Anie

Tuesday, July 8, 2025 11:13 AM

Performance Optimization

Configuration -> phpinit.php

Manage
Windows - [HTTP Request] → [IIS] → [FastCGI] → [PHP](ini) → [WordPress]
↑
[ARR (Application Request Routing)]
↑
[Web.config] (Rewrite rules)

IIS -> does caching, redirect

Perma links handled by web,config

/site/wwwroot/ (WordPress root)
|- /wp-content/
|- /wp-admin/
|- /wp-includes/
|- web.config

Linux
[HTTP Request] → [Nginx] (Reverse Proxy)
↓
[php-fpm] (PHP FastCGI Process Manager)
↓
[WordPress]

Detailed Components

1. **Web Server Layer**
   - **Nginx** (default for Linux App Service):
     - Handles static files directly
     - Proxies PHP requests to php-fpm via FastCGI
     - Configuration in /etc/nginx/sites-enabled/default
   - **Apache** (optional configuration):
     - Uses mod_php or php-fpm
     - .htaccess support (though disabled by default for performance)
     - httpd.conf customizations via Startup Command
2. **PHP Integration**
   - php-fpm (PHP FastCGI Process Manager) is standard
   - Pool configuration in /etc/php/{version}/fpm/pool.d/www.conf
   - Process model example:

5. Key Differences Between Windows (IIS) and Linux (Nginx/Apache)

FEATURE	WINDOWS (IIS)	LINUX (NGINX/APACHE)
Web Server	IIS	Nginx or Apache
PHP Execution	FastCGI with php-cgi.exe	PHP-FPM
Configuration File	web.config	Nginx: nginx.conf, Apache: .htaccess
Performance	Slightly slower due to IIS overhead	Faster with Nginx
Flexibility	Limited	Highly customizable

Components that are shared across the WordPress app

Everything on the WordPress is stored on MySQL

Database Interaction

- WordPress uses the wp-config.php file to connect to the MySQL database.
- The database stores posts, pages, user data, and plugin configurations.

File Storage

- WordPress uploads (media files) are stored in the /wp-content/uploads directory.
- Azure App Service provides persistent storage for these files.

Environment Variables

- Azure App Service allows you to configure environment variables via the Azure Portal or CLI.
- These variables can be accessed in PHP using getenv().

Logging and Monitoring

- Logs are stored in /home/LogFiles.

Performance Optimization

To optimize performance of WordPress on Azure App Service:

1. Caching

- Use full-page caching plugins like W3 Total Cache or WP Super Cache.
- Leverage object caching (e.g., using Redis via Azure Cache for Redis).

2. CDN and Front Door

- Use Azure CDN or Azure Front Door to cache static assets at edge locations.
- Minimize latency and reduce load on the App Service.

3. Persistent Storage

- Store media assets in Azure Blob Storage or use Azure Files with NFS mounting.
- Prevents data loss during horizontal scaling.

4. Database Optimization

- Index frequently queried columns.
- Use optimized queries and reduce autoloaded options.
- Enable slow query logs for diagnostics.

5. Scaling

- Use App Service Plans with higher compute capabilities (e.g., PremiumV3).
- Enable autoscale rules based on CPU, memory, or request count.

6. Application Insights

- Integrate with Azure Application Insights for performance telemetry.
- Identify slow transactions, plugin bottlenecks, and SQL inefficiencies.

7. Minification and Compression

- Use plugins to minify HTML/CSS/JavaScript.
- Enable GZIP compression for faster delivery.

8. Disable Unused Plugins and Themes

- Reduce PHP execution time by cleaning up inactive components.

9. Keep WordPress Core and Plugins Updated

- Performance improvements are often included in updates.

10. Use PHP 8.x

- Ensure you're running a supported and performant PHP version.

High CPU and High Memory - > Scale up
↑ ↑
|---------- |
|
Traffic Induced---↑-------->Scale Out

Redis Cache

https://supportability.visualstudio.com/AzureAppService/_wiki/wikis/AzureAppService/188504/Azure-App-Service-OSS-WordPress
Azure DevOps Services | Sign In

https://supportability.visualstudio.com/AzureAppService/_wiki/wikis/AzureAppService/334796/Azure-App-Service-Linux-OSS-WordPress-Deployment
Azure DevOps Services | Sign In

- Azure Monitor and Application Insights provide detailed telemetry and diagnostics.

Azure App Config

Wednesday, July 9, 2025 12:23 PM

Azure App Configuration
What is Azure App Configuration? | Microsoft Learn

- **Purpose:** Stores frequently accessed database queries, results, and complex objects in fast in-memory cache, preventing repeated database lookups.
- **Azure Redis Cache:** A fully managed, highly scalable, and secure Redis service.
    1. **Create Redis Cache:** Provision an Azure Cache for Redis instance in the **same Azure region** as your App Service. Choose a SKU (C1, C2, etc.) based on your memory and throughput needs.
    2. **WordPress Plugin:** Use the **Redis Object Cache** plugin (preferred) or W3 Total Cache's Redis integration.
    3. **Configure wp-config.php:** Add the following to your wp-config.php, sourcing credentials from App Service Application Settings for security:

```
// Azure Redis Cache settings
define('WP_REDIS_HOST', getenv('REDIS_HOST')); // Set this as an App Setting
define('WP_REDIS_PORT', getenv('REDIS_PORT')); // Set this as an App Setting
define('WP_REDIS_PASSWORD', getenv('REDIS_KEY')); // Set this as an App Setting
define('WP_CACHE_KEY_SALT', 'your-unique-salt-prefix'); // Optional: unique prefix for cache keys

// Enable object caching
define('WP_CACHE', true);
```

Note: You'll need to set REDIS_HOST, REDIS_PORT, and REDIS_KEY as "Application settings" in your App Service configuration.

https://supportability.visualstudio.com/AzureAppService/_wiki/wikis/AzureAppService/414133/Azure-App-Config

Start/Stop VMs v2 overview | Microsoft Learn

Azure DevOps

Wednesday, July 9, 2025 11:06 AM

Build Pipeline
Release Pipeline

Azure DevOps is a standalone resource

Components of Azure DevOps

- Azure board (kanban board)
- Azure repo
- Azure Test Plan
- Azure artifact

Managing

Servers used for DevOps are called agents

Managed Agent
Self-Hosted Agent

Agent Pools

Service Connections

Variable Groups

Task Groups are reusable

CI/CD

YAML and Classic Pipelines

YAML -> Allows you to do IAAS, version control,

Stages -> Steps --> Task) all being run an agent

```yaml
# Example Multi-Stage Pipeline Structure
stages:
- stage: Build
  jobs:
  - job: BuildApplication
    steps:
    - task: Build
    - task: Test
    - task: PublishArtifacts

- stage: DeployDev
  dependsOn: Build
  jobs:
  - deployment: DeployToDevEnvironment
    environment: 'Development'
    strategy:
      runOnce:
        deploy:
          steps:
          - task: Deploy
```

Azure DevOps does not have a feedback mechanism. It just sends the operation

App Service does not have a wait queue (it does only deploy one task at a time)

Blue - Prod
Green - Stage

Canary

Rolling Deployment

Stages

- **Concept:** A stage is a logical division within a pipeline, representing a major phase of your CI/CD process. Stages are executed sequentially by default, but can also run in parallel with specified dependencies.
- **Purpose:** Stages help organize your pipeline into distinct, manageable parts, often corresponding to different environments or logical steps.
- **Examples:** Common stages include Build, Test, Dev Deployment, QA Deployment, Staging Deployment, and Production Deployment.
- **Technical Detail:** Each stage typically runs on a separate agent or pool of agents and can have its own set of variables, conditions, and approvals.

Service Connections

- **Concept:** A secure way to connect Azure DevOps to external services or resources. They encapsulate authentication and authorization details.
- **Purpose:** To allow your pipelines to interact with services like Azure subscriptions, Kubernetes clusters, GitHub, or generic HTTP endpoints without embedding credentials directly in your pipeline definition.
- **Technical Detail:**
  - **Types:** Azure Resource Manager (for Azure services), Generic, GitHub, Kubernetes, Service Fabric, etc.
  - **Authentication:** For Azure, Service Principals (recommended) or Managed Identities are typically used. A Service Principal is an identity created in Microsoft Entra ID (formerly Azure Active Directory) that can be assigned permissions to Azure resources.
  - **Security:** Service connections are stored securely in Azure DevOps, and access to them can be controlled via role-based access control (RBAC).

Environments

- **Examples:** Common stages include Build, Test, Dev Deployment, QA Deployment, Staging Deployment, and Production Deployment.
- **Technical Detail:** Each stage typically runs on a separate agent or pool of agents and can have its own set of variables, conditions, and approvals.

Jobs

- **Concept:** A job is a sequence of steps that run together on a single agent. Jobs are the smallest unit of work that can be scheduled to run.
- **Purpose:** Jobs group related tasks. Multiple jobs can exist within a stage and can run in parallel or sequentially.
- **Technical Detail:** Each job runs in its own context and workspace on the agent. Variables and files created in one job are isolated from others unless explicitly passed as outputs or artifacts. Jobs can have dependencies on other jobs within the same stage or across different stages.
- **Example:** Within a "Build" stage, you might have one job for "Build Application" and another for "Run Unit Tests."

Steps

- **Concept:** A step is the smallest building block in a pipeline, representing an individual task or script that performs a specific action.
- **Purpose:** Steps perform the actual work of the pipeline.
- **Technical Detail:** Steps are executed in the order they are defined within a job. They can be:
  - **Built-in Tasks:** Pre-defined tasks provided by Azure DevOps or the Marketplace (e.g., DotNetCoreCLI@2, AzureWebAppDeploy@4, PowerShell@2). These tasks abstract away much of the underlying complexity.
  - **Scripts:** Inline scripts written in PowerShell, Bash, Python, or command-line scripts. These offer maximum flexibility for custom operations.

created in Microsoft Entra ID (formerly Azure Active Directory) that can be assigned permissions to Azure resources.
  - **Security:** Service connections are stored securely in Azure DevOps, and access to them can be controlled via role-based access control (RBAC).

Environments

- **Concept:** An environment in Azure DevOps represents a collection of resources (e.g., Azure App Service, Virtual Machine, Kubernetes cluster) targeted by a deployment pipeline.
- **Purpose:**
  - **Resource Grouping:** Logical grouping of deployment targets.
  - **Approvals and Checks:** Enables defining pre-deployment and post-deployment approvals and automated checks directly on the environment. This ensures that deployments to critical environments (like production) are gated by necessary validations and human approvals.
  - **Deployment History:** Provides a centralized view of deployment history to that specific environment.
- **Technical Detail:** Environments are defined in the "Environments" section under "Pipelines" in Azure DevOps. You associate specific resources with an environment. For example, an "Azure App Service" resource linked to a "Production" environment. When a pipeline deploys to this environment, the configured approvals and checks are automatically enforced.

Error Handling

Wednesday, July 9, 2025 2:39 PM

Pipeline > Run > Logs

Step	Action
1. Check Logs	Go to **Pipelines > Runs > Logs**. Expand the failed step.
2. Examine Error Output	Look for keywords like Exception, Exit Code, Timeout.
3. Check Agent Status	Verify if the agent was online and had correct capabilities.
4. Validate Secrets	Ensure service connections, variables, and secrets are correctly configured.
5. Re-run with Debug	Enable system.debug: true to get verbose output.
6. Check Resource Availability	Ensure Azure resources (App Service, DB, etc.) are healthy.
7. Use Deployment Logs	If deploying to Azure, check logs in the **Azure Portal** under App Service logs

Error Message	Cause	Solution
Authentication failed	Invalid or expired service connection	Refresh or reauthorize service connection in Azure DevOps
No package found	Artifact not properly published or downloaded	Ensure artifact path is correct and the CI pipeline completed successfully
Timeout expired	Slow deployment or unresponsive service	Increase timeout in YAML or verify service health
403 Forbidden	Missing permissions	Grant correct access to the service principal or managed identity
Zip deployment failed	File corruption or size limit exceeded	Validate the ZIP package and check Azure App Service file limits
Invalid YAML	Syntax or logic error	Use Azure DevOps YAML validator or format checker
Resource not found	Missed environment or app name typo	Double-check target resource names and ensure they exist

Azure Container App

Thursday, July 10, 2025 10:34 AM

[Azure Container Apps environments | Microsoft Learn]

Difference between app service and azure container app

ACA that is built to take advantage of containerized apps. Allows you to run containerized app.

The underlying infrastructure is azure Kubernetes service.

https://learn.microsoft.com/en-us/azure/aks/keda-about

Azure Container app uses KEDA scaling

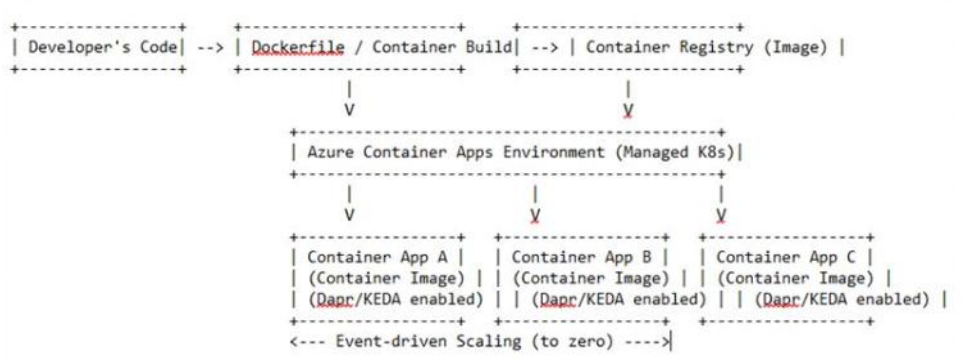
Dapr is like a runtime template that contains all the required run times

You can deploy multiple containers in a container app

Feature	**Container app**	**regular app service**
Infrastructure Management	Fully managed, serverless	Partially managed, requires app hosting plan
Container Support	Native support for containers	Limited container support (via Web App for Containers)
Scaling	Auto-scaling with KEDA	Manual or rule-based scaling
Event-Driven	Built-in support for event-driven architectures	Requires additional configuration
Multi-container Deployments	Yes	No
Dapr Integration	Yes	No
Use Case	Microservices, event-driven apps, serverless	Web apps, APIs, lightweight backend services

Container Image - > azure registry -> KEDA rules are applied

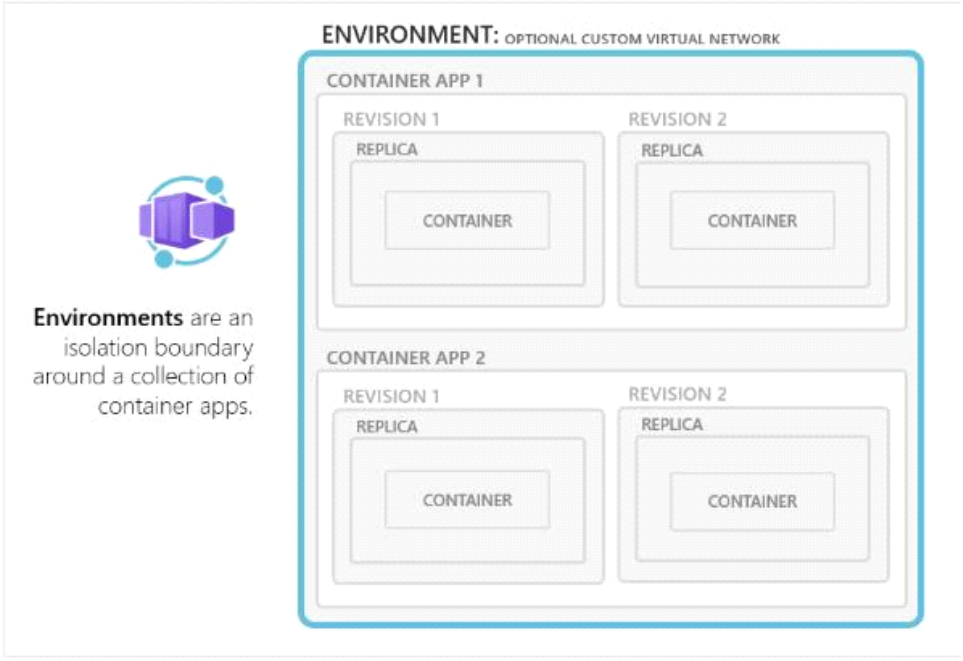
Most of the core technology used in azure container apps are open source

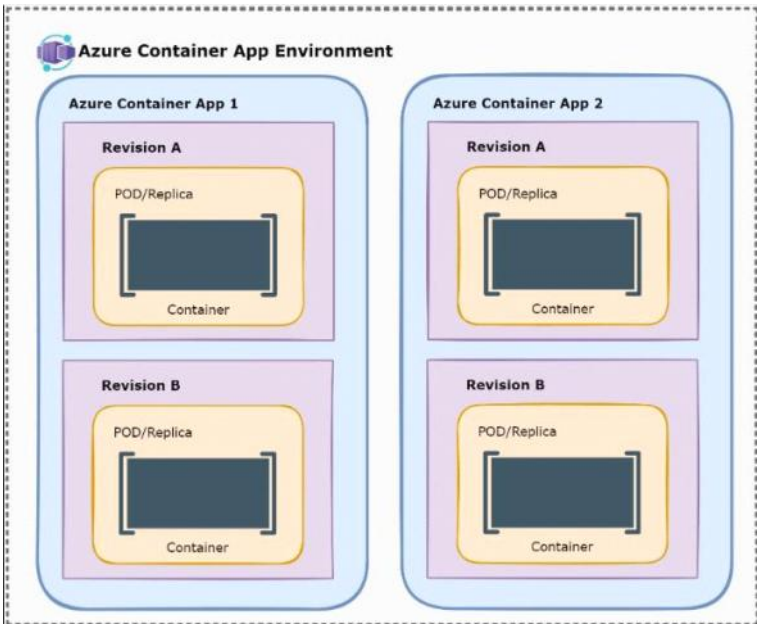


[Compute and billing structures in Azure Container Apps | Microsoft Learn]

[Comparing Container Apps with other Azure container options | Microsoft Learn]

Container App Environment





Compare container apps and jobs

There are two types of compute resources in Azure Container Apps: apps and jobs.

Apps are services that run continuously. If a container in an app fails, it restarts automatically. Examples of apps include HTTP APIs, web apps, and background services that continuously process input.

Jobs are tasks that start, run for a finite duration, and exit when finished. Each execution of a job typically performs a single unit of work. Job executions start manually, on a schedule, or in response to events. Examples of jobs include batch processes that run on demand and scheduled tasks.

[Jobs in Azure Container Apps | Microsoft Learn]