1. Develop Azure Compute Solutions

└ 1.1 Implement Containerized Solutions

└ 1.1.1 Create and Manage Container Images for Solutions

1. What is the correct structure of a Dockerfile?
2. How do you optimize a Dockerfile for size and performance?
3. How do multi-stage builds work?
4. What commands are used to build an image locally?
5. How do you tag versions appropriately?
6. What is the correct format for image names (registry/repository:tag)?
7. How do tags work and how are they used in CI/CD pipelines?
8. How do you create and configure an ACR?
9. What are the SKU tiers and when do you use them
10. How do you push/pull images using Docker CLI?
11. How do you authenticate to ACR (admin account, service principal, managed identity)?
12. What is az acr login and when is it required?
13. How does Azure App Service, AKS and ACI consume images from ACR?
14. What are the permission requirements for pulling from ACR?
15. How do you scan images for vulnerabilities?
16. What tools or services are used to harden container images?
17. How do you manage image versions across environments?
18. How do you clean up unreferenced or old images in ACR?
19. How can ACR tasks automate image builds?
20. How do you use az acr task to create scheduled or event-driven builds?
21. What are the pros and cons of storing images publicly vs privately?
22. How do you configure access control for image repositories?

---

1. What is the correct structure of a Dockerfile?

```
FROM <base_image>
[ENV <key>=<value> ...]
[WORKDIR <directory>]
[COPY <src> <dest>]
[RUN <command>]
[EXPOSE <port>]
[CMD ["executable", "param1", ...]]
```

Key Points:

- FROM must be first (defines base image).
- WORKDIR sets working directory inside container.
- COPY adds files to the image from build context.
- RUN executes shell commands (e.g., install packages).
- EXPOSE is optional metadata; doesn't actually open ports.
- CMD defines default container startup command (only one allowed; last one wins).

2. How do you optimize a Dockerfile for size and performance?

1. Use slim/minimal base images:
   Prefer FROM mcr.microsoft.com/dotnet/aspnet:7.0-alpine over full images.
3. Leverage multi-stage builds:
   Build in one stage, copy only final output to runtime stage to reduce size.
4. Minimize layers:
   Group related RUN commands and clean up temp files in the same layer:
4. Avoid unnecessary files:
   Use .dockerignore to exclude files (e.g., .git, node_modules).
5. Set only needed environment variables and permissions:
   Avoid excessive ENV or USER changes unless required.

---

3. How do multi-stage builds work?

Reduce final image size by separating build and runtime stages.

- Define multiple FROM statements in one Dockerfile.
- Use an alias for the build stage (AS build).
- Copy only needed artifacts from the build stage into the final image.

---

4. What commands are used to build an image locally?

docker build -t <name>:<tag> <path>

e.g. docker build -t myapp:latest .

---

5. How do you tag versions appropriately?

Use semantic tags like:

latest, v1.0.0, dev, staging

e.g. docker build -t myapp:v1.0.0

Tag meaning should reflect version or environment for clarity and traceability.

---

6. What is the correct format for image names?

<registry>/<repository>:<tag> *:*

myregistry.azurecr.io/myapp:v1.0.0

Registry is optional for local images. Tag defaults to latest if omitted.

---

7. How do tags work and how are they used in CI/CD pipelines?

Tags identify image versions. Pipelines use tags to pull, test, and deploy specific builds:

docker push myapp:staging → used in staging environment.

latest often used in dev, versioned tags in prod.

---

8. How do you create and configure an ACR?

- Create ACR: az acr create --name <acr-name> --resource-group <rg> --sku Basic
- Enable admin access: az acr update -n <acr-name> --admin-enabled true
- Login: az acr login --name <acr-name>

---

9. What are the SKU tiers and when do you use them?

- Basic – Dev/test, low-cost, limited features
- Standard – Prod-ready, geo-replication support
- Premium – High-scale, content trust, private endpoints, more throughput

10. How do you push/pull images from/to ACR using Docker CLI?

Push:

1. docker tag myapp myacr.azurecr.io/myapp
2. docker push myacr.azurecr.io/myapp

Pull:

docker pull myacr.azurecr.io/myapp

---

11. How do you authenticate to ACR (admin, service principal, managed identity)?

- Admin account:
  Enable with az acr update --admin-enabled true, then use provided username/password.
- Service principal:
  Assign AcrPush/AcrPull role, login with docker login using SP credentials.
- Managed identity:
  Grant role to identity, Azure services (e.g., App Service) authenticate automatically.

---

12. What is az acr login and when is it required?

az acr login --name <acr-name>

Authenticates Docker CLI with ACR.
Required for manual Docker pushes/pulls. Not needed for Azure services using managed identity.

---

13. How do App Service, Kubernetes Service (AKS), and Container Instances consume ACR images?

- App Service: Configure container settings with ACR URL; use managed identity or admin acc.
- AKS: Enable ACR integration via az aks update or use imagePullSecrets.
- ACI: Reference image with full ACR path; grant access via identity or admin credentials.

---

14. What are the permission requirements for pulling from ACR?

The identity must have **AcrPull** role on the ACR.
Can be assigned to:

- User
- Service principal
- Managed identity (App Service, AKS, etc.)

---

15. How do you scan images for vulnerabilities?

Use Microsoft Defender for Cloud with ACR integration. It scans images on push and shows CVEs in the portal.
Enable under Defender plans > Container registries.

---

16. What tools or services are used to harden container images?

- Microsoft Defender for Cloud – vulnerability scanning
- Dockerfile best practices – minimize layers, use minimal base images
- Content trust – ensure image integrity
- Private ACR – restrict access
- ACR Tasks – automate secure builds

---

17. How do you manage image versions across environments?

Use consistent tagging strategy (e.g., dev, staging, v1.0.0).
Promote images by re-tagging and pushing to ACR for each stage.

18. How do you clean up unreferenced or old images in ACR?

Use ACR Tasks with retention policies or manual cleanup via:

```
az acr repository delete --name <acr> --image <repo>:<tag>
```

---

19. How can ACR tasks automate image builds?

ACR Tasks can auto-build images on source code or base image changes:

`az acr task create` with `--source` and `--cmd "docker build"`

Supports triggers (e.g., Git push) and scheduling.

---

20. How do you use az acr task to create scheduled or event-driven builds?

Event-driven:

```
az acr task create \
  --name mytask \
  --registry myacr \
  --image myapp:{{.Run.ID}} \
  --context https://github.com/org/repo.git \
  --file Dockerfile \
  --git-access-token <token>
```

Scheduled:

```
az acr task create \
  --name mytask \
  --registry myacr \
  --schedule "0 2 * * *" \
  --image myapp:nightly \
  --context https://github.com/org/repo.git \
  --file Dockerfile
```

---

21. What are the pros and cons of storing images publicly vs privately?

Public: Easy access, no auth needed — but insecure, no access control.

Private: Secure, controlled access — but needs auth, may cost more.

---

22. How do you configure access control for image repositories?

Assign `AcrPull` or `AcrPush` roles to users, service principals, or managed identities using Azure RBAC on the ACR resource.

Develop Azure Compute Solutions

└ 1.1 Implement Containerized Solutions

└ 1.1.2 Publish an image to Azure Container Registry

1. What is Azure Container Registry?
2. What are the core capabilities and use cases?
3. How does managed identity authentication work for ACR?
4. How do you authenticate ACR access in CI/CD workflows (e.g., GitHub Actions, Azure DevOps)?
5. How do you list repositories and images in ACR?
6. How do you delete/update images using Azure CLI or Portal?
7. How do you validate if a service (App Service, AKS, ACI) is authorized to pull from ACR?
8. What happens if permissions are missing?
9. How do you use ACR with GitHub Actions or Azure Pipelines?
10. What are best practices for secure build pipelines pushing to ACR?
11. How do you configure image signing and content trust in ACR?
12. What security best practices apply to private ACR use?

---

1. What is Azure Container Registry (ACR)?

A managed, private Docker registry in Azure used to store container images and artifacts like Helm charts. Supports geo-replication, image signing, ACR Tasks, and CI/CD integration.

---

2. What are the core capabilities and use cases?

Capabilities:

- Private image storage with RBAC
- ACR Tasks for build automation
- Integration with App Service, AKS, ACI
- Image scanning with Defender for Cloud

Use Cases:

- Hosting internal images
- Serving images to AKS
- Automating builds/pushes
- Enforcing image compliance

---

3. How does managed identity authentication work for ACR?

Azure services (e.g., App Service, AKS) use their managed identity to access ACR. You assign the `AcrPull` role to the identity on the ACR resource—no credentials needed.

---

4. How do you authenticate ACR access in CI/CD workflows?

- GitHub Actions: Use `azure/login` and `docker/login-action` with a service principal or OIDC.
- Azure DevOps: Use built-in service connection or Docker task with a service principal. Ensure the principal has AcrPush or AcrPull as needed.

---

5. How do you list repositories and images in ACR?

Use Azure CLI:

- List repos: `az acr repository list --name <acr-name>`
- List tags/images: `az acr repository show-tags --name <acr-name> --repository <repo>`

6. How do you delete or update images in ACR?

- Delete image: az acr repository delete --name <acr-name> --image <repo>:<tag>
- To update, re-tag and push a new version.
  Use retention policies or manual cleanup to manage old images.

---

7. How do you validate if a service is authorized to pull from ACR?

Check that the service's managed identity has the AcrPull role on the ACR.
Verify access using:

```
az role assignment list --assignee <identity> --scope <acr-resource-id>
```

---

8. How do you use ACR with GitHub Actions or Azure Pipelines?

GitHub Actions:

- Authenticate with azure/login and docker/login-action
- Push using docker/build-push-action

Azure Pipelines:

- Use Docker or container tasks
- Authenticate via service connection

Both require proper role assignment (e.g., AcrPush).

---

9. What happens if permissions are missing?

The service (e.g., App Service, AKS, ACI) will fail to start or pull the image, typically with an authentication or 403 error in logs.

---

10. What are best practices for secure build pipelines pushing to ACR?

- Use service principals or workload identity (OIDC)
- Grant least privilege (only AcrPush)
- Avoid storing secrets in plain text; use Key Vault or pipeline secrets
- Enable image signing and scanning post-push

---

11. How do you configure image signing and content trust in ACR?

Enable content trust to verify image integrity:

- Use Docker's DOCKER_CONTENT_TRUST=1 for signing and verification
- For ACR, use Microsoft Defender for Containers to enforce policies

---

12. What security best practices apply to private ACR use?

- Use private endpoints or firewall rules
- Disable anonymous pull access
- Require authentication via managed identity or service principal
- Enable Defender for vulnerability scanning
- Assign roles using RBAC (AcrPull/AcrPush only as needed)

Develop Azure Compute Solutions

└ **1.1 Implement Containerized Solutions**

└ **1.1.3 Run Containers by Using Azure Container Instances (ACI)**

1. What is Azure Container Instances (ACI) and when should you use it?
2. How do you create and deploy a container using ACI via Azure CLI?
3. What are the key configuration parameters for az container create?
4. How do you pull images from Azure Container Registry to ACI?
5. How does managed identity authentication work with ACI?
6. How do you assign and verify roles for ACI to access ACR?
7. What are the options for exposing containers to the internet or VNETs in ACI?
8. How do you mount Azure Files or secrets (Key Vault) into containers?
9. How do you monitor logs, metrics, and container status in ACI?
10. What are the restart policies and lifecycle options in ACI?
11. How do you use YAML to define ACI deployments?
12. What are common use cases and limitations of ACI?

---

1. What is Azure Container Instances (ACI) and when should you use it?

ACI is a serverless container platform allowing fast, isolated container runs without VM management.
Use cases:

- Short-lived jobs or batch processing
- Event-driven container execution
- Lightweight API hosting without orchestration overhead

---

2. How do you create and deploy a container using ACI via Azure CLI?

```
az container create \
    --resource-group <rg> \
    --name <container-name> \
    --image <image-name> \
    --cpu 1 \
    --memory 1 \
    --restart-policy OnFailure \
    --dns-name-label <unique-label> \
    --ports 80
```

This deploys a public-facing container running on port 80.

---

3. What are the key configuration parameters for az container create?

--image: Container image to run (e.g., from ACR or Docker Hub)
--cpu / --memory: Resource limits
--environment-variables: Inject app settings
--ports: Exposed ports
--dns-name-label: For public IP
--restart-policy: Options: Always, OnFailure, Never
--vnet and --subnet: Attach to virtual network
--secrets and --secrets-mount-path: Mount secrets

4. How do you pull images from Azure Container Registry to ACI?

ACI can pull private images from ACR by granting ACI's managed identity the **AcrPull** role on the ACR.

Ensure image format is:

```
<acr-name>.azurecr.io/<repository>:<tag>
```

Example ACI deployment with an image from ACR:

```
az container create \
  --name <container-name> \
  --resource-group <rg> \
  --image <acr-name>.azurecr.io/app:latest \
  --registry-login-server <acr-name>.azurecr.io \
  --assign-identity \
  --cpu 1 --memory 1
```

---

5. How does managed identity authentication work with ACI?

ACI supports user-assigned and system-assigned managed identities.

Steps:

1. Enable managed identity with --assign-identity.
2. Assign AcrPull role to the identity at ACR scope.
3. ACI uses this identity to authenticate and pull private images—no credentials needed.

---

6. How do you assign and verify roles for ACI to access ACR?

Use Azure CLI to assign roles:

```
az role assignment create \
  --assignee <principal-id> \
  --role AcrPull \
  --scope /subscriptions/<sub-id>/resourceGroups/<rg>/providers/Microsoft.ContainerRegistry/registries/<acr-name>
```

Verify with:

```
az role assignment list --assignee <principal-id> --scope <acr-resource-id>
```

---

7. What are the options for exposing containers to the internet or VNETs in ACI?

- Public IP (default): Use --dns-name-label and --ports to expose over the internet.
- Private IP (VNET): Use --vnet & --subnet to deploy into a virtual network for internal-only access.

ACI supports:

- Inbound public access
- Private IP in VNET (for secure inter-service traffic)
- No ingress (headless jobs)

---

8. How do you mount Azure Files or secrets (Key Vault) into containers?

Mount Azure Files:

```
az container create \
  --azure-file-volume-share-name <share> \
  --azure-file-volume-account-name <storage-account> \
  --azure-file-volume-account-key <key> \
  --azure-file-volume-mount-path /mnt/data
```

Mount Key Vault secrets:

```
az container create \
  --secrets key1=value1 key2=value2 \
  --secrets-mount-path /mnt/secrets
```

9. How do you monitor logs, metrics, and container status in ACI?

View logs:

```
az container logs --name <container-name> --resource-group <rg>
```

Get status:

```
az container show --name <container-name> --resource-group <rg> --query instanceView.state
```

---

10. What are the restart policies and lifecycle options in ACI?

Available values for --restart-policy:

- Always: Container restarts on exit (default).
- OnFailure: Restarts only on non-zero exit code.
- Never: One-time execution, used for jobs.

Lifecycle:

- No native job scheduling—combine with Logic Apps, Functions, or Event Grid for automation.
- ACI auto-deletes after manual az container delete or TTL implementation logic.

---

11. How do you use YAML to define ACI deployments?

Example aci.yaml:

```
apiVersion: 2018-10-01
location: eastus
name: mycontainer
properties:
 containers:
  - name: myapp
    properties:
     image: myacr.azurecr.io/myapp:latest
     resources:
       requests:
         cpu: 1
         memoryInGb: 1.5
     ports:
       - port: 80
 osType: Linux
 restartPolicy: OnFailure
 ipAddress:
  type: Public
  dnsNameLabel: mycontainerdemo
  ports:
    - port: 80
type: Microsoft.ContainerInstance/containerGroups
```

Deploy with:

```
az deployment group create \
  --resource-group <rg> \
  --template-file aci.yaml
```

---

12. What are common use cases and limitations of ACI?

Use Cases:

- Lightweight API/backend services
- Batch jobs
- Temporary compute (build/test)
- Event-driven processing

Limitations:

- No built-in autoscaling
- No service mesh or ingress controller
- Not suited for complex orchestration—use AKS instead

Develop Azure Compute Solutions

└ 1.1 Implement Containerized Solutions

└ 1.1.4 Create solutions by using Azure Container Apps

1. What is Azure Container Apps and when should it be used over AKS or App Services?
2. What components define an Azure Container App?
3. How do revisions work in Azure Container Apps?
4. What are the prerequisites for deploying a container to Azure Container Apps?
5. How do you deploy a container from Azure Container Registry using Azure CLI?
6. How is a YAML manifest used to deploy a container app?
7. How do you configure ingress and expose ports in Azure Container Apps?
8. How do you configure authentication for private container registries?
9. How are environment variables added to a container app?
10. How are secrets stored and injected into container apps?
11. What is KEDA and how does it apply to Azure Container Apps?
12. What scaling rules are supported in Azure Container Apps?
13. How do you configure HTTP-based autoscaling?
14. How do you configure scaling based on Azure Service Bus or Azure Queue Storage?
15. What are minReplicas and maxReplicas and how are they configured?
16. How does revision mode affect app behavior in Azure Container Apps?
17. How is traffic splitting configured across revisions?
18. How do you perform A/B testing using revisions?
19. How do you roll back to a previous revision?
20. What is Dapr and how is it used with Azure Container Apps?
21. What Dapr capabilities are supported in Azure Container Apps?
22. How do you enable and configure the Dapr sidecar?
23. How do container apps communicate using Dapr?
24. How is state management and pub/sub handled with Dapr?
25. What monitoring and logging features are built into Azure Container Apps?
26. How do you enable and access diagnostics logs?
27. How do you view application logs and container output?
28. What tools can be used to troubleshoot container app issues?
29. How does integration with Azure Monitor and Log Analytics work?
30. How do you integrate Azure Container Apps with Event Grid or Service Bus?
31. How do you connect a container app to Azure Storage queues?
32. How do you securely access Azure services from a container app?
33. How do you use managed identities in Azure Container Apps?
34. What role does VNET integration play in accessing private resources?

---

1. What is Azure Container Apps and when should it be used over AKS or App Services?

Azure Container Apps is a fully managed serverless container service for microservices, APIs, and background processing. Use it when:

- You need event-driven or HTTP-based workloads
- You prefer serverless scaling (including scale to zero)
- You want built-in Dapr and KEDA support without managing orchestration

Prefer AKS for orchestration or App Services for traditional web apps or minimal container needs.

---

2. What components define an Azure Container App?

- Container App: The deployed app instance
- Environment: A shared context for apps (networking, logging)
- Revision: An immutable version of the app
- Ingress: Controls public/private HTTP access
- Scaling Rules: Define autoscaling behavior (HTTP, KEDA, etc.)

---

3. How do revisions work in Azure Container Apps?

- Each deployment creates a new immutable revision
- Revisions can run concurrently
- Traffic can be split between revisions
- You can pin a revision or roll back
- Revision mode can be single (default) or multiple

---

4. What are the prerequisites for deploying a container to Azure Container Apps?

- A container image in ACR or public registry
- A Container Apps environment
- Azure CLI with the containerapp extension installed
- App image must expose the correct HTTP port.

---

5. How do you deploy a container from Azure Container Registry using Azure CLI?

```
az containerapp create \
  --name myapp \
  --resource-group myrg \
  --environment myenv \
  --image myacr.azurecr.io/myimage:tag \
  --target-port 80 \
  --ingress external \
  --registry-server myacr.azurecr.io \
  --registry-username <username> \
  --registry-password <password>
```

---

6. How is a YAML manifest used to deploy a container app?

Define app configuration in a .yaml file (image, ports, scaling, secrets, etc.)

- Deploy using:
  `az containerapp create --resource-group myrg --name myapp --yaml app.yaml`

---

7. How do you configure ingress and expose ports in Azure Container Apps?

- Use `--ingress external` or `internal` in CLI or `ingress:` block in YAML
- Set targetPort to match container's exposed port
- Public ingress automatically provisions HTTPS endpoint

---

8. How do you configure authentication for private container registries?

- Use `--registry-username` and `--registry-password` in CLI
- In YAML:

```
registryCredentials:
  - server: myacr.azurecr.io
    username: <username>
    passwordSecretRef: acr-password
```

- Store password as a secret and reference it

9. How are environment variables added to a container app?

- In CLI:
  --env-vars VAR1=value1 VAR2=value2
- In YAML:

```
env:
 - name: VAR1
   value: value1
 - name: VAR2
   value: value2
```

---

10. How are secrets stored and injected into container apps?

- Define secrets in CLI
  --secrets key1=value1 key2=value2
- or YAML:

```
secrets:W
 - name: key1
   value: value1
```

- Reference in env vars:

```
env:
 - name: SEC_VAR
   secretRef: key1
```

---

12. What is KEDA and how does it apply to Azure Container Apps?

KEDA (Kubernetes Event-driven Autoscaler) enables event-based scaling. In Azure Container Apps, it's integrated to scale apps based on metrics like:

- HTTP traffic
- Queue length (e.g., Service Bus, Storage Queues)
- Custom metrics

---

13. What scaling rules are supported in Azure Container Apps?

- HTTP request concurrency
- CPU utilization
- KEDA-based triggers (e.g., Azure Service Bus, RabbitMQ, Redis, Kafka)
- Cron-based schedules

---

14. How do you configure HTTP-based autoscaling?

In YAML:

```
scale:
 rules:
   - name: http-scaler
     http:
       concurrentRequests: 50
```

App will scale based on the number of concurrent HTTP requests.

---

15. What are minReplicas and maxReplicas and how are they configured?

- minReplicas: minimum number of app instances
- maxReplicas: cap on autoscaling
  In YAML:

```
scale:
 minReplicas: 1
 maxReplicas: 10
```

---

16. How do you configure scaling based on Azure Service Bus or Azure Queue Storage?

Define a KEDA trigger in YAML:

```
scale:
  rules:
    - name: sb-scaler
      azureServiceBus:
        queueName: myqueue
        connection: sb-connection
        messageCount: 100
```

*"connection" references a secret holding the Service Bus connection string

---

17. How does revision mode affect app behavior in Azure Container Apps?

- Single revision mode (default): only one revision is active; new deployments kill the previous
- Multiple revision mode: multiple revisions can run concurrently; useful for traffic splitting
- Set via CLI or YAML:

```
revisionMode: multiple
```

---

18. How is traffic splitting configured across revisions?

Assign percentage of traffic to each revision

- In CLI:

```
az containerapp revision set-trafficsplit \
  --name myapp \
  --resource-group myrg \
  --revision-weight revisionA=80 revisionB=20
```

- In YAML:

```
traffic:
 - revisionName: revisionA
   weight: 80
 - revisionName: revisionB
   weight: 20
```

---

19. How do you perform A/B testing using revisions?

- Deploy a new revision in multiple revision mode
- Split traffic between revisions (e.g., 90/10)
- Monitor metrics and logs for both
- Adjust traffic weights or rollback based on results

---

20. How do you roll back to a previous revision?

- Set traffic weight to 100% for the target revision
- Optionally disable the newer revision
- CLI:

```
az containerapp revision set-trafficsplit --name myapp --revision-weight oldrev=100
```

---

21. What is Dapr and how is it used with Azure Container Apps?

Dapr (Distributed Application Runtime) provides building blocks for microservices (e.g., service discovery, state management). Azure Container Apps has built-in Dapr support. Enable by setting daprEnabled: true. No additional setup is required for the Dapr sidecar.

---

22. What Dapr capabilities are supported in Azure Container Apps?

- Service invocation over HTTP/gRPC
- State management (e.g., Redis, Cosmos DB)
- Pub/sub messaging
- Secrets integration
- Middleware and observability tools
  Note: Components are defined via Dapr-compatible configuration files.

---

23. How do you enable and configure the Dapr sidecar?

In YAML:

```
dapr:
  enabled: true
  appId: myapp
  appPort: 80
```

---

24. How do container apps communicate using Dapr?

- Service A calls Service B via http://<appId>.dapr
- Dapr handles service discovery and routing
- Requires both apps to have dapr.enabled: true and unique appId

---

25. What monitoring and logging features are built into Azure Container Apps?

- Integrated Log Streaming via Azure CLI
- Application logs, revision logs, and system logs
- Container stdout/stderr collection
- Azure Monitor and Log Analytics integration for metrics and centralized logging

---

26. How do you enable and access diagnostics logs?

- Enable diagnostics when creating the Container App Environment
- Logs are sent to Azure Monitor (Log Analytics workspace)
- Use Azure CLI:
  `az containerapp logs show --name myapp --resource-group myrg`

---

27. How do you view application logs and container output?

Via Azure CLI:

```
az containerapp logs show --name myapp --follow
```

- Logs include stdout and stderr from the container
- For historical logs, query via Log Analytics using Kusto Query Language (KQL)

---

28. What tools can be used to troubleshoot container app issues?

- `az containerapp` logs show for live logs
- Log Analytics queries for historical data
- Metrics in Azure Monitor (CPU, memory, HTTP throughput)
- Azure CLI/Portal for revision status and health
- Re-deploy with `--debug flag` to get CLI diagnostics

---

29. How does integration with Azure Monitor and Log Analytics work?

- When enabled, diagnostics are sent to a specified Log Analytics workspace
- Use KQL to query logs (e.g., ContainerAppConsoleLogs_CL)
- Metrics surface in Azure Monitor for alerting and dashboarding

30. How do Azure Container Apps integrate with Event Grid or Service Bus?

- Use KEDA triggers to scale based on Event Grid or Service Bus messages
- Event Grid: typically triggers external logic that posts to app HTTP endpoint
- Service Bus: KEDA listens and scales app based on queue/topic message count
- Connection strings are passed as secrets and referenced in scaling rules

---

31. How do you connect a container app to Azure Storage queues?

- Use KEDA with azureQueue scaler
- Define queueName, connection, and queueLength threshold
- Store Storage Account connection string as a secret and reference it in scaling config
- App logic must poll the queue if not using an event trigger

---

32. How do you securely access Azure services from a container app?

- Use **Managed Identity** to authenticate to Azure services like Key Vault, Storage, or SQL
- Avoid hardcoding credentials
- Access tokens are obtained via Azure SDK or HTTP call to IMDS endpoint

---

33. How do you use managed identities in Azure Container Apps?

- Enable system-assigned or user-assigned identity at app level
- Assign proper RBAC role to the identity
- Access Azure services using Azure SDKs with default credential chain
- Example (Azure SDK):
  ```
  from azure.identity import DefaultAzureCredential
  from azure.keyvault.secrets import SecretClient
  ```

---

35. What role does VNET integration play in accessing private resources

Enables access to private endpoints, databases, or internal APIs

- Configure internal ingress and associate the Container Apps environment with a VNET
- Required for scenarios needing outbound traffic restrictions or private-only dependencies

Develop Azure Compute Solutions

└ 1.2 Implement Azure App Service Web Apps

└ 1.2.1 Create an Azure App Service Web App

1. What is Azure App Service and when should it be used?
2. How do you create an App Service Web App using the Azure CLI?
3. How do you create an App Service Plan and what are the pricing tiers?
4. What runtime stacks and OS options are supported?
5. How do you configure deployment credentials?
6. How do you deploy code to App Service (e.g., ZIP, GitHub Actions, Azure DevOps)?
7. How do you configure environment variables and app settings?
8. How do you assign a custom domain and configure HTTPS?
9. What is the role of App Service managed identity?
10. What are the basic scaling options available (manual, autoscale)?

---

1. What is Azure App Service and when should it be used?

A PaaS for hosting web apps and APIs. Supports .NET, Node.js, Python, Java, PHP.
Use when: You need quick deployment, scaling, custom domains, or managed identity.

---

2. How do you create an App Service Web App using Azure CLI?

```
az group create --name myRG --location eastus
az appservice plan create --name myPlan --resource-group myRG --sku B1 --is-linux
az webapp create --resource-group myRG --plan myPlan --name mywebapp123 --runtime "DOTNET|7.0"
```

Creates a Linux-based .NET 7 Web App in Basic tier.

---

3. What is an App Service Plan and how do tiers differ?

Defines compute resources for Web Apps.
Tiers:

- F1 (Free): Shared, limited
- B1 (Basic): Dedicated, no autoscale
- S1 (Standard): Autoscale, staging slots
- P1V3 (Premium): VNET, better scaling
- I1 (Isolated): Private VNET, high compliance

---

4. What OS and runtime stacks are supported?

OS: Linux or Windows
Runtimes: .NET, Node.js, Java, Python, PHP, Ruby, Static HTML
Selected via `--runtime` (e.g., "DOTNET|6.0" or "NODE|18-lts")

---

5. How do you deploy code to Web Apps?

- ZIP Deploy: `az webapp deployment source config-zip`
- GitHub Actions: `az webapp deployment github-actions add`
- Azure DevOps: Use App Service deploy task
- Local Git: Configure via Portal or CLI

6. How do you configure app settings and connection strings?

```
az webapp config appsettings set --name <app-name> --resource-group <rg> --settings KEY=VALUE
az webapp config connection-string set --name <app-name> --resource-group <rg> --settings connStr=... --connection-string-type SQLAzure
```

Stored securely and available as environment variables.

---

7. How do you bind a custom domain and enable HTTPS?

- Add domain via `az webapp config hostname add`
- Enable HTTPS: `az webapp update --https-only true`
- Use App Service-managed certs or upload custom certs

---

8. What is the purpose and setup of managed identity in App Service?

Grants the app access to Azure resources without secrets.

```
az webapp identity assign --name <app-name> --resource-group <rg>
```

Assign roles to this identity for access (e.g., Key Vault).

---

9. How does scaling work in App Service?

- Manual: Set instance count (Standard+)
- Autoscale: Use Azure Monitor rules (CPU, schedule)
  Configured via Portal or ARM/CLI (az monitor autoscale)

---

10. What diagnostic/logging options are available?

- App logs: `az webapp log config`
- Live logs: `az webapp log tail`
- App Insights: Enable via Portal or `az monitor app-insights component create`

Develop Azure Compute Solutions

└ 1.2 Implement Azure App Service Web Apps

└ 1.2.2 Configure and Implement Diagnostics and Logging

1. What types of logs are available in Azure App Service?
2. How do you enable diagnostics logging via CLI?
3. How do you stream logs from App Service in real-time?
4. How do you integrate Application Insights with App Service?
5. What's the difference between App Service logs and App Insights?
6. How do you query logs in Application Insights using KQL?

---

1. What types of logs are available in Azure App Service?

- Application Logs: stdout/stderr from app
- Web Server Logs: HTTP access logs
- Detailed Error Logs: 500-level errors
- Failed Request Tracing Logs (FREB): IIS-level tracing
- App Insights Logs: Telemetry (requests, exceptions, dependencies)

---

2. How do you enable diagnostics logging via CLI?

```
az webapp log config --name <app> --resource-group <rg> \
--application-logging filesystem --web-server-logging filesystem
```

Stores logs in local file system (Basic+) or blob storage.

---

3. How do you stream logs from App Service in real-time?

```
az webapp log tail --name <app> --resource-group <rg>
```

Streams stdout/stderr and App Logs live to console.

---

4. How do you integrate Application Insights with App Service?

```
az monitor app-insights component create --app <name> --location <region> --resource-group <rg>
az webapp config appsettings set --name <app> --resource-group <rg> --settings
APPINSIGHTS_INSTRUMENTATIONKEY=<key>
```

.NET apps may auto-integrate; others need SDK.

---

5. What's the difference between App Service logs and App Insights?

- App Service Logs: Platform logs (stdout, web server logs)
- App Insights: Full telemetry—requests, exceptions, dependencies, metrics, traces

---

6. How do you query logs in Application Insights using KQL?

Use Log Analytics (or Logs tab in App Insights):

Example query:

```
requests
| where timestamp > ago(1h)
| summarize count() by resultCode
```

Develop Azure Compute Solutions

└ 1.2 Implement Azure App Service Web Apps

└ 1.2.3 Deploy Code and Containerized Solutions

1. What deployment options exist for App Service (code and container)?
2. How do you deploy a ZIP package via Azure CLI?
3. How do you configure GitHub Actions deployment?
4. How do you deploy a containerized app to App Service?
5. How do you configure deployment slots and swap?
6. What are best practices for zero-downtime deployments?

---

1. What deployment options exist for App Service (code and container)?

- Code: ZIP Deploy, GitHub Actions, Azure DevOps, FTP, Local Git
- Container: Docker Hub, Azure Container Registry (ACR), custom registry

App type determines supported methods.

---

2. How do you deploy a ZIP package via Azure CLI?

```
az webapp deployment source config-zip \
  --resource-group <rg> \
  --name <app> \
  --src <path-to-zip>
```

---

3. How do you configure GitHub Actions deployment?

```
az webapp deployment github-actions add \
  --repo <user/repo> \
  --branch main \
  --name <app> \
  --resource-group <rg> \
  --login-with-github
```

Generates CI/CD workflow for App Service.

---

4. How do you deploy a containerized app to App Service?

```
az webapp create \
  --resource-group <rg> \
  --plan <plan> \
  --name <app> \
  --deployment-container-image-name <acr-name>.azurecr.io/app:tag
```

Ensure the Web App uses a Linux plan and has AcrPull access.

---

5. How do you configure deployment slots and swap?

- Create slot:

```
az webapp deployment slot create --name <app> --resource-group <rg> --slot staging
```

- Swap slots:

```
az webapp deployment slot swap --name <app> --resource-group <rg> --slot staging
```

Slots isolate staging, testing, and production deployments.

---

6. What are best practices for zero-downtime deployments?

- Use deployment slots
- Run warm-up tests before slot swap
- Enable auto-swap with health checks
- Avoid in-place updates on production slot

Develop Azure Compute Solutions

└ **1.2 Implement Azure App Service Web Apps**

└ **1.2.4 Configure settings including TLS,API, ServiceConnections**

1. How do you enforce HTTPS (TLS) for an App Service?
2. How do you configure minimum TLS version?
3. How do you set custom domains and bind SSL certificates?
4. How do you configure API settings such as CORS?
5. How do you connect to backend services using managed identity?
6. How do you restrict outbound traffic using VNET integration?

---

1. How do you enforce HTTPS (TLS) for an App Service?

```
az webapp update --name <app> --resource-group <rg> --https-only true
```

Redirects all HTTP traffic to HTTPS.

---

2. How do you configure minimum TLS version?

```
az webapp config set --name <app> --resource-group <rg> --min-tls-version 1.2
```

Options: 1.0, 1.1, 1.2. Use 1.2 for compliance.

---

3. How do you set custom domains and bind SSL certificates?

- Add domain:

```
az webapp config hostname add --webapp-name <app> --resource-group <rg> --hostname <custom-domain>
```

- Upload and bind cert:

```
az webapp config ssl upload --certificate-file cert.pfx --certificate-password <pwd> --name <app> --resource-group <rg>
az webapp config ssl bind --name <app> --resource-group <rg> --ssl-type SNI --certificate-thumb
```

---

4. How do you configure API settings such as CORS?

```
az webapp cors add --name <app> --resource-group <rg> --allowed-origins https://example.com
```

Use cors remove or cors show to manage rules.

---

5. How do you connect to backend services using managed identity?

1. Enable identity:

```
az webapp identity assign --name <app> --resource-group <rg>
```

2. Grant role to identity (e.g., Key Vault Reader):

```
az role assignment create --assignee <principal-id> --role Reader --scope <resource-id>
```

---

6. How do you restrict outbound traffic using VNET integration?

- Integrate with VNET (Linux apps, Standard+):

```
az webapp vnet-integration add --name <app> --resource-group <rg> --vnet <vnet-name> --subnet <subnet-name>
```

Restricts outbound traffic and enables access to private services.

Develop Azure Compute Solutions

└ 1.2 Implement Azure App Service Web Apps

└ 1.2.5 Implement Auto-Scaling

1. What scaling options are available in Azure App Service?
2. How do you configure autoscale rules using CLI?
3. What metrics can be used for autoscaling?
4. How do you set instance count manually?
5. What are best practices for autoscaling App Service?

---

1. What scaling options are available in Azure App Service?

- Manual scale: Fixed instance count
- Autoscale: Rule-based via CPU, memory, schedule
  Available in Standard, Premium, Isolated tiers only.

---

2. How do you configure autoscale rules using CLI?

```
az monitor autoscale create \
  --resource <app-service-plan-id> \
  --resource-group <rg> \
  --name autoscale-rule \
  --min-count 1 --max-count 5 --count 2
az monitor autoscale rule create \
  --resource-group <rg> \
  --autoscale-name autoscale-rule \
  --condition "CpuPercentage > 70 avg 5m" \
  --scale out 1
```

---

3. What metrics can be used for autoscaling?

- CPUPercentage
- MemoryPercentage (Premium+ plans)
- HttpQueueLength
- Schedule-based triggers

---

4. How do you set instance count manually?

```
az appservice plan update \
  --name <plan> --resource-group <rg> \
  --number-of-workers <count>
```

Used for fixed scaling when autoscale is not enabled.

---

5. What are best practices for autoscaling App Service?

- Use min/max limits to control scale boundaries
- Combine CPU and schedule rules for stability
- Enable App Insights to monitor autoscale behavior
- Use Premium tiers for memory-based and faster scaling

Develop Azure Compute Solutions

└ 1.2 Implement Azure App Service Web Apps

└ 1.2.6 Configure deployment slots

1. What are deployment slots and when should you use them?
2. How do you create a deployment slot via CLI?
3. How do you swap slots in App Service?
4. What settings can be cloned or configured per slot?
5. What are best practices for using deployment slots?

---

1. What are deployment slots and when should you use them?

Slots are live app environments (e.g., staging, testing) under the same App Service.
Use for: zero-downtime deployments, staged testing, A/B validation.

---

2. How do you create a deployment slot via CLI?

```
az webapp deployment slot create \
  --name <app> --resource-group <rg> --slot staging
```

Inherits settings from the production slot by default.

---

3. How do you swap slots in App Service?

```
az webapp deployment slot swap \
  --name <app> --resource-group <rg> \
  --slot staging --target-slot production
```

Promotes staging to production with no downtime.

---

4. What settings can be cloned or configured per slot?

- Cloned: App settings, connection strings (default)
- Slot-specific settings: Mark as "deployment slot setting" to isolate
  Examples: DB connection strings, API keys

---

5. What are best practices for using deployment slots?

- Use staging slot for validation
- Enable auto-swap with health checks
- Keep secrets slot-specific
- Minimize downtime by warming up the slot before swap

Develop Azure Compute Solutions

└ 1.3 Implement Azure Functions

└ 1.3.1 Create and Configure an Azure Functions App

1. What hosting plans are available for Azure Functions, and when should each be used?
2. How do you create an Azure Function App using Azure CLI?
3. What are the key configuration settings for a Function App?
4. How do you configure the runtime stack and version?
5. What are the authentication options for securing Function Apps?
6. How do you configure application settings and connection strings?
7. How do you set and manage function-level timeouts?
8. What are the deployment options for Azure Functions?
9. How do you configure CORS for a Function App?
10. How do you configure monitoring and diagnostics?

---

1. What hosting plans are available for Azure Functions, and when should each be used?

- Consumption Plan: Auto-scales and is cost-effective for sporadic workloads. Limited to 5 mins (default) or 10 mins (max) execution.
- Premium Plan: Supports VNETs, unlimited execution time, and pre-warmed instances. Use for high-load or enterprise apps.
- Dedicated (App Service) Plan: Use if consolidating services on the same App Service plan or for always-on requirements with full control.

---

2. How do you create an Azure Function App using Azure CLI?

```
az functionapp create \
  --resource-group <rg> \
  --consumption-plan-location <region> \
  --runtime <dotnet|node|python|java|powershell> \
  --functions-version <version> \
  --name <app-name> \
  --storage-account <storage-name>
```

---

3. What are the key configuration settings for a Function App?

- Runtime Stack & Version
- Platform Architecture (32-bit/64-bit)
- Always On (Premium/Dedicated only)
- Application Settings (APPINSIGHTS_INSTRUMENTATIONKEY, etc.)
- CORS
- Authentication

---

4. How do you configure the runtime stack and version?

Use CLI or Portal:

```
az functionapp config set \
  --name <app-name> \
  --resource-group <rg> \
  --linux-fx-version "DOTNET|6.0"
```

Use "linux-fx-version" or "windows-fx-version" depending on OS.

5. What are the authentication options for securing Function Apps?

- App-level authentication: Use Microsoft Entra ID, Facebook, Google, etc.
- Function key/token: For per-function access control.
- Anonymous access: Only if explicitly enabled.
- Recommended: Use Entra ID for secure, enterprise-grade auth.

---

6. How do you configure application settings and connection strings?

Use Azure CLI or Portal. Example with CLI:

```
az functionapp config appsettings set \
  --name <app-name> \
  --resource-group <rg> \
  --settings "MySetting=value" "MyConnStr=connectionstring"
```

Connection strings should be stored as app settings and referenced via `Environment.GetEnvironmentVariable()` in code.

---

7. How do you set and manage function-level timeouts?

- Consumption Plan: Max 5 min by default, extendable to 10 min via host.json:

```
{
 "functionTimeout": "00:10:00"
}
```

- Premium/Dedicated Plan: No timeout limit.

---

8. What are the deployment options for Azure Functions?

- Zip deployment: az functionapp deployment source config-zip
- Local Git or GitHub: Continuous integration via DevOps or GitHub Actions
- VS Code / Azure CLI: Direct push
- Azure DevOps Pipelines: Full CI/CD with environment config

---

9. How do you configure CORS for a Function App?

```
az functionapp cors add \
  --name <app-name> \
  --resource-group <rg> \
  --allowed-origins https://example.com
```

Use * only for public or development APIs — avoid in production.

---

10. How do you configure monitoring and diagnostics?

- Enable Application Insights during creation or via:

```
az monitor app-insights component create \
--app <app-name> --location <region> --resource-group <rg>
```

- Link to Function App:

```
az functionapp config appsettings set \
--name <app-name> --resource-group <rg> \
--settings "APPINSIGHTS_INSTRUMENTATIONKEY=<key>"
```

Logs and metrics are available in App Insights, Azure Monitor, or via az monitor.

Develop Azure Compute Solutions

└ 1.3 Implement Azure Functions

└ 1.3.2 Implement input and output bindings

1. What are input and output bindings in Azure Functions?
2. What types of bindings are most commonly used?
3. How do you define bindings in function.json?
4. How are bindings configured in C# using attributes?
5. What is the difference between trigger and input bindings?
6. How do you bind to Azure Blob Storage?
7. How do you bind to Azure Queue Storage?
8. How do you bind to Azure Cosmos DB?
9. How do you use output bindings to return data from a function?
10. How do you manage binding configuration via app settings?

---

1. What are input and output bindings in Azure Functions?

Bindings abstract external data sources into declarative configurations.

- Input bindings: Bring external data into the function.
- Output bindings: Send function output to an external service.

---

2. What types of bindings are most commonly used?

- Azure Blob Storage
- Azure Queue Storage
- Azure Cosmos DB
- HTTP (Trigger + Output)
- Azure Table Storage

---

3. How do you define bindings in function.json?

Bindings are defined as JSON objects. Example:

```
{
 "bindings": [
  {
   "name": "myBlob",
   "type": "blobTrigger",
   "direction": "in",
   "path": "samples/{name}",
   "connection": "AzureWebJobsStorage"
  }
 ]
}
```

---

4. How are bindings configured in C# using attributes?

Example (Queue input, Blob output):

```
public static void Run(
   [QueueTrigger("queue-name")] string input,
   [Blob("output-container/{rand-guid}", FileAccess.Write)] out string outputBlob)
            {
               outputBlob = input.ToUpper();
            }
```

Use specific attributes like [BlobTrigger], [Queue] , [CosmosDB] .

5. What is the difference between trigger and input bindings?

- Trigger: Initiates function execution (only one per function).
- Input: Supplies data for processing (can have multiple).
  Example: A blobTrigger starts the function; a Cosmos DB input binding provides extra data.

---

6. How do you bind to Azure Blob Storage?

Input (read):

```
[BlobTrigger("input-container/{name}", Connection = "AzureWebJobsStorage")] Stream inputBlob
```

Output (write):

```
[Blob("output-container/{name}", FileAccess.Write, Connection = "AzureWebJobsStorage")] Stream outputBlob
```

---

7. How do you bind to Azure Queue Storage?

Trigger:

```
[QueueTrigger("queue-name", Connection = "AzureWebJobsStorage")] string msg
```

Output:

```
[Queue("output-queue", Connection = "AzureWebJobsStorage")] out string outputQueueMsg
```

---

8. How do you bind to Azure Cosmos DB?

Input (read):

```
[CosmosDB(
   databaseName: "db",
   collectionName: "coll",
   Id = "id",
   PartitionKey = "key",
   Connection = "CosmosDBConnection")] dynamic item
```

Output (write):

```
[CosmosDB(
   databaseName: "db",
   collectionName: "coll",
   Connection = "CosmosDBConnection")] out dynamic outputItem
```

---

9. How do you use output bindings to return data from a function?

- Use the return statement for HTTP output or single output bindings:

```
[FunctionName("HttpExample")]
public static IActionResult Run([HttpTrigger(...)] HttpRequest req, [Queue("myqueue")] out string msg)
{
   msg = "Queue message";
   return new OkObjectResult("Success");
}
```

---

10. How do you manage binding configuration via app settings?

- Use placeholders in function.json or attribute Connection:

```
"connection": "AzureWebJobsStorage"
```

- Then define the value in Configuration > Application settings in the portal or via:

```
az functionapp config appsettings set --name <app-name> --settings AzureWebJobsStorage=<conn-string>
```

Develop Azure Compute Solutions

└ 1.3 Implement Azure Functions

└ 1.3.3 Implement function triggers by using data operations, timers, and webhooks

1. What are function triggers and how do they differ from bindings?
2. What are the main trigger types?
3. How do you implement an HTTP trigger?
4. How do you implement a Timer trigger and define its schedule?
5. How do you implement a Blob trigger?
6. How do you implement a Queue Storage trigger?
7. How do you implement a Cosmos DB trigger?
8. How do you implement a Service Bus trigger?
9. How do you handle errors and retries in triggered functions?
10. What are best practices for managing trigger configuration?

---

1. What are function triggers and how do they differ from bindings?

Triggers start the function — only one allowed per function.
Bindings are for data input/output — zero or more allowed.
Example: HttpTrigger, TimerTrigger, BlobTrigger.

---

2. What are the main trigger types tested on the exam?

- HttpTrigger
- TimerTrigger
- BlobTrigger
- QueueTrigger
- CosmosDBTrigger
- ServiceBusTrigger

---

3. How do you implement an HTTP trigger?

```
[FunctionName("HttpExample")]
public static IActionResult Run(
   [HttpTrigger(AuthorizationLevel.Function, "get", "post")] HttpRequest req)
{
   return new OkObjectResult("Hello from HTTP trigger");
}
```

Supports get, post, etc. Use AuthorizationLevel to control access.

---

4. How do you implement a Timer trigger and define its schedule?

```
[FunctionName("TimerExample")]
public static void Run([TimerTrigger("0 */5 * * * *")] TimerInfo myTimer)
{
   // Executes every 5 minutes
}
```

CRON format: "{second} {minute} {hour} {day} {month} {day-of-week}".

5. How do you implement a Blob trigger?

```
[FunctionName("BlobExample")]
public static void Run(
[BlobTrigger("container/{name}", Connection = "AzureWebJobsStorage")] Stream myBlob, string name)
{
   // Reacts to new/updated blobs
}
```

---

6. How do you implement a Queue Storage trigger?

```
[FunctionName("QueueTriggerExample")]
public static void Run([QueueTrigger("my-queue", Connection = "AzureWebJobsStorage")] string queueItem)
{
   // Processes messages from Azure Queue Storage
}
```

---

7. How do you implement a Cosmos DB trigger?

```
[FunctionName("CosmosTriggerExample")]
public static void Run(
[CosmosDBTrigger(
databaseName: "db",
collectionName: "coll",
ConnectionStringSetting = "CosmosDBConnection",
LeaseCollectionName = "leases")] IReadOnlyList<Document> input)
{
   // Reacts to inserts and updates in Cosmos DB
}
```

Leases track change feed progress.

---

8. How do you implement a Service Bus trigger?

```
[FunctionName("ServiceBusExample")]
public static void Run(
   [ServiceBusTrigger("queue-name", Connection = "ServiceBusConnection")] string message)
{
   // Handles messages from Azure Service Bus Queue
}
```

Use "queue-name" or "topic-name", "subscription-name" for topics.

---

9. How do you handle errors and retries in triggered functions?

- Retry policies can be set in host.json:

```
{
  "retry": {
    "strategy": "fixedDelay",
    "maxRetryCount": 3,
    "delayInterval": "00:00:05"
  }
}
```

- Unhandled exceptions automatically trigger retries (except for HTTP).

---

10. What are best practices for managing trigger configuration?

- Use app settings for all connection strings
- Abstract trigger values (e.g., queue name, CRON) into settings
- Use dependency injection for service management
- Limit execution time (set functionTimeout)
- Avoid long-running or blocking operations