**1. Code Commit and Version Control (Repository)**

* The process begins when developers push their code to a version control system (e.g., **GitHub**, **Azure Repos**).
* The code is stored in a **repository** and is the latest version of the application.

**2. CI Pipeline Trigger**

* When a change is committed (e.g., a push to a branch), it triggers the **Continuous Integration (CI) pipeline**.
* The CI pipeline is configured in **Azure DevOps**, **GitHub Actions**, or other CI tools like **Jenkins**.
* The pipeline typically performs several steps:
  1. **Code Build**: The source code is built, dependencies are installed (e.g., using **Maven**, **npm**, **dotnet**).
  2. **Unit Tests**: Automated tests are run to ensure the code behaves as expected.
  3. **Static Analysis**: Security scans (e.g., using **SonarQube**) or code quality checks are performed.
  4. **Docker Image Creation**: A **Dockerfile** is used to create a Docker image that packages the application.
  5. **Push to Container Registry**: The created Docker image is pushed to a container registry, such as **Azure Container Registry (ACR)** or **Docker Hub**.

**3. CD Pipeline (Continuous Deployment)**

* Once the Docker image is pushed, a **Continuous Deployment (CD) pipeline** is triggered.
* The CD pipeline deploys the application to **Azure Kubernetes Service (AKS)** using Kubernetes manifests and Helm charts.

**4. Deployment to Azure Kubernetes Service (AKS)**

* **Kubernetes Manifests**: These are YAML files that define the deployment configurations, including:
  + **Deployments**: Define how the application should be deployed (e.g., number of replicas, resource requests/limits).
  + **Services**: Expose the application to other services or the internet (e.g., LoadBalancer or ClusterIP).
  + **Ingress**: Defines the routing of traffic to services, especially for external access.
* **Helm Charts**: An alternative to Kubernetes manifests, **Helm** is a package manager for Kubernetes that simplifies complex deployments with predefined charts (templates).

**5. Azure DevOps or CI/CD Tool Deployment**

* The **Azure DevOps** pipeline (or other tools) deploys the Docker image and Kubernetes manifests to the AKS cluster.
* **kubectl** (Kubernetes CLI) or Helm commands are run to apply the configurations in AKS.

**6. Kubernetes Schedules and Manages Containers**

* AKS receives the deployment request and schedules the Docker containers based on the available resources in the cluster.
* Kubernetes ensures the application is running by maintaining the desired number of replicas and managing pods and services.
* If a container crashes, Kubernetes will restart it automatically, ensuring high availability.

**7. Monitoring and Logging**

* After deployment, monitoring and logging tools (e.g., **Azure Monitor**, **Azure Application Insights**) are used to observe the application’s health, performance, and logs.
* **Prometheus** and **Grafana** can also be used for detailed monitoring in AKS.
* Alerts are triggered for issues like high CPU usage, memory leaks, or failed health checks.

**8. Scaling and Autoscaling**

* AKS can scale the number of pods (containers) automatically based on CPU, memory usage, or custom metrics (via **Horizontal Pod Autoscaler**).
* The **Azure Monitor Autoscale** feature can also help scale AKS clusters based on metrics like CPU utilization or network traffic.

**Key Tools and Concepts:**

* **Azure DevOps** or **GitHub Actions** (for CI/CD)
* **Docker** (for containerization)
* **Azure Container Registry (ACR)** (for storing Docker images)
* **Kubernetes** (for orchestrating containers)
* **Helm** (for packaging Kubernetes applications)
* **kubectl** (for interacting with Kubernetes clusters)
* **Azure Monitor** / **Prometheus** / **Grafana** (for monitoring)

**Summary:**

The code is committed to a repository, triggering the CI pipeline, which builds the application, runs tests, and packages it into a Docker image. This image is then deployed to Azure Kubernetes Service (AKS) via Kubernetes manifests or Helm. AKS handles the orchestration of the application containers, while monitoring tools track its health and performance.

This flow can be adjusted based on your environment, but this is the general process used in deploying applications to AKS.