CI/CD y DevSecOps con Github Actions y Docker

1. **Núcleo de Github Actions:** motor de automatización integrado en el repositorio (.Yaml)
   * Un evento (Push - pull) dispara un job el cual tiene una serie de steps que ejecutan comandos o acciones predefinidas
   * actua como orquestador, permitiendo compilar código, ejecutar pruebas y gestionar secretos de forma segura

2. **Actions y Docker:** el contenedor actua como unidad mínima de entrega
   * El pipeline se encarga de construir imagenes optimizadas y generar evidencia

3. **Runners y Base para Kubernets:** El runner es el servidor donde se ejecuta el Workflow
   * Son MV gestionadas por Github, no tiene acceso a las redes privadas

4. **Tipos de CI y flujo genérico:** Un pipeline de devsecops combina múltiples disciplinas en una sola ejecución secuencial.
   a. **Integración:** Compilacion y pruebas unitarias
   b. **Calidad:** Análisis estático y formateo de código
   c. **Seguridad:** Escaneos SAST (codigo), SCA (dependencias) y DAST (App en ejecución)
   d. **Empaquetado:** Generacion de la imagen final del contenedor

5. **Preparacion para despliegue:** antes de liberar el software, se debe generar confianza y evidencias como:
   - **Artefactos:** imagenes de contenedor inmutables y firmadas
   - **Evidencias:** Generacion de SBOM y reportes de vulnerabilidades
   - **Publicación:** solo si pasa todo (Test + seguridad)

6. **Entrega Continua CD y estrategias de despliegue:** transporta artefactos aprobados en CI hacia entornos reales.
   - **Estrategias de despliegue:**
     - **zero-downtime:** Actualizar sin interrumpir el servicio
     - **Ring-based:** Despliegue gradual por grupos de usuario para limitar el impacto de errores

7. **Seguridad en el Pipeline y Supply Chain:**
   * Aplicar principio del menor privilegio
   * no basta con construir, se debe probar que se construyó

8. Gobernanza, trazabilidad y Revisión :

- **Trazabilidad :** capacidad de llegar a la línea de código del problema y el pipeline que lo generó
- **Four-Eyes principles :** nadie probara sus propios combios, se usa revicion por los codeowners
- **Workflows obligatorios :** Configurar gates de seguridad

9. Obtimizacion y Coste : cuando el equipo crece, los pipelines se hacen lentos y caros

- **caché:** Reutilizar descargas de dependencias y capas docker para asceleror tiempos.
- **concurrencia :** cancelar ejecucionas obsoletas
- **Gestión de artefactos :** No guardar reportes o binarios indefinidamente