Fondamenti per DevOps (16 ore)

Obiettivo: Fornire le basi necessarie per comprendere e lavorare con i principi e gli strumenti DevOps.

Modulo 1: VM, Linux, Git (4 ore)

- Intro Fondamenti
- VirtualBox & Vagrant: VM vs container
- Linux basics: CLI, gestione file, vi/vim
- Pacchetti e servizi: apt, systemd
- Git & GitHub: repo, PR Esercizi:
- VM con NAT + host-only e port forwarding
- Script bash che verifica servizi e logga
- Fork repo, branch feature, PR

Modulo 2: Networking, Web server, TLS (4 ore)

- IP, subnet, gateway, DNS, dig, netstat/ss, ufw
- NGINX static e reverse proxy
- Apache overview
- SSL/TLS: chiavi, CSR, self-signed
- JSON, YAML, jq, yq, JSONPath
- Quiz breve Esercizi:
- NGINX su 8080, log access/error
- Self-signed cert; test curl -k
- jq/yq con esempi

Modulo 3: App & Database (4 ore)

- Python venv, pip, requirements
- App Flask: /health, /users, config YAML
- setup PostgreSQL: ruoli, DB, schema, seed Esercizi:
- Collegare Flask a Postgres
- Logging e configurazione via YAML/ENV

Modulo 4: Integrazione end-to-end (4 ore)

- Integrazione NGINX reverse proxy verso app
- Sicurezza base: ufw, least privilege, secrets
- Systemd unit: restart, env file, logging
- DNS pratico: /etc/hosts, app.local
- Progetto finale, demo e retrospettiva Criteri valutazione:
- Automazione (30%), qualità repo (20%), app (20%), integrazione (15%), DB (10%), troubleshooting (5%)

Materiale Didattico e Requisiti

- Computer con 8 GB di RAM e buona connettività Internet
- Software richiesti:
  - Terminal, Git, Python, Virtualbox, Vagrant

Laboratorio di DevOps (24 ore)

Obiettivo: Fornire una formazione pratica sull'implementazione e gestione di un flusso di lavoro DevOps. Include l'installazione degli strumenti necessari e l'uso di piattaforme come GitHub e Microsoft Azure.

Modulo 1: Introduzione e Configurazione Ambiente (4 ore)

Obiettivi: Familiarizzare con DevOps, installare e configurare l'ambiente di sviluppo.

Sessione 1: Concetti Base di DevOps (1 ora)

- Introduzione a DevOps:
  - Definizione, principi e vantaggi
  - Differenze tra metodologie tradizionali e DevOps
- Strumenti fondamentali: Git, CI/CD, containerizzazione
- Overview delle piattaforme (GitHub, Azure)

Sessione 2: Setup dell'Ambiente di Lavoro (3 ore)

- Prerequisiti:
  - Installazione Terminal app e Winget package manager
  - Installazione Git, Bash, Azure CLI, Terraform e VS Code
  - Installazione WSL2 e Ubuntu app
  - Installazione di Docker Desktop
- Piattaforma di sviluppo:
  - GitHub:
    - Configurazione SSH e autenticazione
  - Microsoft Azure:
    - Introduzione al portale Azure
    - Introduzione a Microsoft Learn
- Strumenti DevOps:
  - Configurazione di Terminal e GitHub
  - Configurazione di VS Code

Modulo 2: Versionamento e Collaborazione con GitHub (4 ore)

Obiettivi: Apprendere l'uso di Git e GitHub per il controllo del codice e la collaborazione, con focus su GitHub Projects e GitHub Classroom.

Sessione 1: Git (2 ore)

- Concetti di base:
  - Configurazione di un repository
  - Comandi principali: `add`, `commit`, `push`, `pull`, `merge`
  - Creazione e gestione di branch
  - Strategie di branching (GitHub Flow, Trunk-Based, GitOps)

Sessione 2: GitHub avanzato: Projects e Classroom + Azure Boards (2 ore)

- Workflow collaborativo
- GitHub Projects:
  - Configurazione di progetti Agile (kanban, timeline, board)
  - Issue, label, assignee, milestone
  - Campi personalizzati, viste, filtri e automazioni
  - Roadmapping e collegamento con pull request
  - Esempi di template per gestione backlog e sprint

- GitHub Classroom (per le scuole):
  - Creazione di classi e compiti (assignments)
  - Integrazione con template repository e test automatici (GitHub Actions)
  - Valutazione e feedback, gestione plagi tramite similarity checks
  - Provisioning automatico di repository per studenti
- Azure Boards:
  - Collegamento repo GitHub con Boards
  - Sincronizzazione work items, PR e commit
- Pull request e Code Review:
  - Branch protection rules, required reviews e status checks
  - Code owners, draft PR, templates

Modulo 3: Automazione e CI/CD (8 ore)

Obiettivi: Configurare pipeline CI/CD e integrare strumenti DevOps.

Sessione 1: Containerizzazione con Docker e Kubernetes (4 ore)

- Introduzione a Docker:
  - Creazione di immagini Docker
  - Gestione di container
  - Introduzione a un container registry (Docker Hub o GHCR)
- Introduzione a Kubernetes:
  - Concetti base: Pod, Deployment, Service, Ingress
  - Configurazione di un cluster con Docker Desktop o Docker Swarm/Stack

Sessione 2: CI/CD Pipeline (4 ore)

- Concetti chiave:
  - Continuous Integration e Continuous Deployment
  - Principali strumenti: Jenkins, GitHub Actions, Azure DevOps
- Creazione di una pipeline:
  - Test del codice
  - Build automatizzata con Docker
  - Scan delle immagini e dei container (SAST/Container scanning)
- Esempio pratico:
  - Esecuzione di una pipeline CI/CD su GitHub Actions e Azure Pipelines
  - Deployment su ambiente di test/staging

Modulo 4: Site Reliability & Platform Engineering (4 ore)

Obiettivi: Infrastructure as Code, monitoraggio e ottimizzazione del flusso DevOps.

Sessione 1: IaC e Monitoraggio (3 ore)

- Gestione delle risorse con Infrastructure as Code/Software:
  - CloudFormation, HCL/Terraform, Pulumi, CDK, ARM, Bicep
- Monitoraggio:
  - Azure Monitor e Azure Service Diagnostics
  - Metriche, log, alert e dashboard
- Introduzione SRE e Internal Developer Platforms

Sessione 2: Feedback e Miglioramento Continuo (1 ora)

- Analisi delle metriche della SDLC DevOps (DORA Metrics)

- Ottimizzazione del metodo DevOps:
  - Automazione dei flussi di lavoro con piattaforme self-service per sviluppatori
  - Riduzione dei rischi e della duplicazione dei carichi di lavoro
  - Semplificazione e standardizzazione
  - FinOps e Cybersecurity

Modulo 5: Progetto Finale e Debriefing (4 ore)

Obiettivo: Applicare le conoscenze acquisite in un progetto pratico.

Sessione 1: Progetto DevOps (3 ore)

- Scenario proposto:
  - Sviluppo nel GitHub Repo di un deployment workflow con Terraform
  - Creazione di una pipeline end-to-end multi-environment su GitHub Actions
  - Pubblicazione di un modulo Terraform su registry (GHCR come artifact)
  - Deployment continuo sicuro

Sessione 2: Debriefing e Q&A (1 ora)

- Presentazione del progetto
- Discussione su sfide affrontate e soluzioni adottate
- Domande e retrospettiva

Materiale Didattico e Requisiti

- Computer con 8 GB di RAM e veloce connessione Internet
- Accesso agli account GitHub e Azure
- Software richiesti:
- VS Code, Docker Desktop, Terminal, Git, Terraform/OpenTofu