Day 12: Cloud Deployment – GCP & Serverless Platforms

This day focuses on **multi-cloud awareness and practical deployment skills** beyond AWS. You'll cover **GCP**, **cost-effective VPS hosting**, and **modern serverless platforms**, then conclude with a **clear comparison** to choose the right platform per use case.

---

1. GCP Deployment (Compute Engine + Cloud SQL)

1.1 GCP Architecture Overview

```
User
 ↓
Cloud Load Balancer (optional)
 ↓
Compute Engine (VM)
 ↓
Cloud SQL (PostgreSQL)
```

Key philosophy: - Strong managed services - Excellent data & AI ecosystem - Competitive pricing for sustained workloads

---

1.2 Compute Engine (VM Deployment)

VM Configuration - Machine type: `e2-micro` (low cost) - OS: Ubuntu 22.04 - Disk: 10–20 GB standard

Firewall Rules - Allow HTTP (80) - Allow HTTPS (443) - Allow SSH (22) from your IP

Application Setup

```
sudo apt update
sudo apt install python3-pip nginx -y
pip install fastapi uvicorn gunicorn
```

Run application:

```
gunicorn -k uvicorn.workers.UvicornWorker app:app --bind 0.0.0.0:8000
```

✔️ Checklist: GCP VM deployment complete

1.3 Cloud SQL (Managed PostgreSQL)

Configuration - Engine: PostgreSQL 15 - Instance type: db-f1-micro / db-g1-small - Private IP enabled

Connectivity - Use Cloud SQL Auth Proxy **or** private VPC peering

```
psql -h <cloud-sql-ip> -U app_user -d app_db
```

Run migrations:

```
alembic upgrade head
```

✔️ Checklist: Cloud SQL connected & tested

---

2. DigitalOcean (Cost-Effective VPS)

Why DigitalOcean

- Simple UI
- Predictable pricing
- Ideal for startups, side projects, MVPs

Droplet Setup

- Plan: Basic $5/month
- OS: Ubuntu 22.04
- Region: Closest to users

Deployment Steps

```
sudo apt update
sudo apt install docker-compose nginx -y
pip install fastapi uvicorn
```

Run app:

```
uvicorn app:app --host 0.0.0.0 --port 8000
```

Optional: - Managed PostgreSQL - Spaces (S3-compatible storage)

✔️ Checklist: DigitalOcean deployment operational

---

3. Serverless Platforms (Frontend & Lightweight APIs)

Serverless platforms trade **control for speed and simplicity**.

3.1 Vercel

Best For - Frontend (React, Next.js) - Edge functions - Preview deployments

Deploy Steps

```
npm install -g vercel
vercel login
vercel
```

Features: - Automatic preview URLs for every PR - Global CDN - Zero-config HTTPS

✔️ Checklist: Vercel deployment with preview URLs

3.2 Render

Best For - Backend APIs - Cron jobs - Managed databases

Features: - Git-based deploys - Free tier available

3.3 Netlify

Best For - Static sites - Jamstack - Serverless functions

Features: - Form handling - CDN by default

---

4. Cost & Performance Comparison

4.1 Cloud Providers

Provider	Strengths	Weaknesses	Best Use Case
AWS	Service breadth	Complex	Enterprise systems
GCP	Data & AI	Smaller ecosystem	AI-heavy workloads
Azure	MS ecosystem	Cost complexity	.NET enterprises
DigitalOcean	Simplicity	Fewer services	MVPs & startups

4.2 Serverless Platforms

Platform	Cost	Control	Ideal For
Vercel	Medium	Low	Frontend apps
Render	Low–Medium	Medium	APIs
Netlify	Low	Low	Static sites

4.3 Monthly Cost Snapshot (Small App)

Platform	Approx Cost
AWS (EC2 + RDS)	$25–30
GCP (VM + Cloud SQL)	$20–25
DigitalOcean	$5–10
Vercel	$0–20

✔️ Checklist: Cost & performance comparison completed

Day 12 Completion Status

- ☑️ GCP Compute Engine + Cloud SQL deployed
- ☑️ DigitalOcean droplet operational
- ☑️ Vercel deployment with preview URLs
- ☑️ Cost & performance comparison documented

Key Interview Insight

> The best cloud provider is **context-dependent**. Strong engineers choose platforms based on **cost, scale, and operational complexity** — not hype.

Next strong steps: - **Day 13: CI/CD & Infrastructure as Code** - Convert Days 6–12 into a **multi-cloud system design case study** - Build a **cloud-agnostic deployment checklist**