DOCKER AND KUBERNETES - Introduction to Microservices

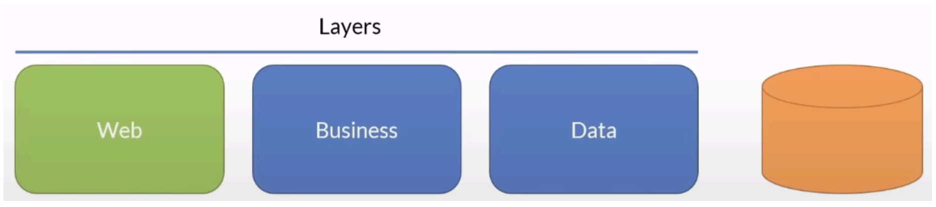
1) Docker Commands

Command	Meaning
docker version	Checks the version of our installed docker
docker info	Displays detailed information about the Docker installation, including system status, configuration, and resources.
docker pull hello-world	Download the hello-world image from Docker Hub to your local machine.
docker images	Lists all Docker images stored locally on your machine.
docker run hello-world	Runs a container from the hello-world image to verify that Docker is installed and working correctly.
docker ps	Lists all running Docker containers on your system.
docker ps -a	Lists all Docker containers, including both running and stopped ones.

2) Microservices Architecture

- A variant of the **service-oriented architecture** (SOA) structural style - arranges an application as a collection of **loosely coupled services**.
- In a microservice architecture, services are **fine-grained** (single responsibility) and the protocols are **lightweight** (They use simple, efficient communication methods).

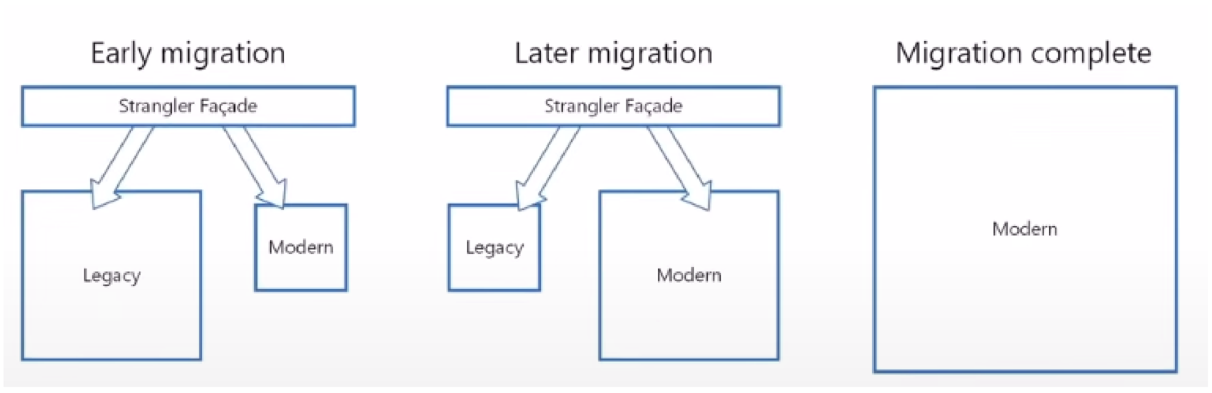
3) Monolithic Architecture

- Built as a **single unit**.
- **Deployed** as single unit
- **Duplicated on each server**.
- Ex: 3-tier apps.
- 

4) Microservices

- Segregates functionality into **smaller separate services** each with a single responsibility.
- Scales out by **deploying each service independently**.
- **Loosely coupled**.
- Enable **autonomous development** by different teams, languages and platforms.
- Can be written by smaller teams.
- Each microservice can own its **own data/database.**

5) From Monolithic to Microservices

- Break your application/system in small units.
- Use the strangler pattern.
- 

- **Early migration**:
  - You have an old system (**Legacy**) running everything.
  - You start building a new system (**Modern**).
  - A **Strangler Façade** (like a smart middle layer) sits in front.
  - It decides whether to send user requests to the **Legacy** system or the **Modern** one.
- **Later migration**:
  - You've moved more features to the **Modern** system.
  - **Legacy** system is now handling fewer things.
  - The **Strangler Façade** still decides where each request should go.
- **Migration complete**:
  - Everything now runs on the **Modern** system.
  - The **Legacy** system is gone.
  - The **Strangler Façade** is no longer needed — or can now just point to Modern directly.

DOCKER AND KUBERNETES - Introduction to Microservices

6) Microservices - Benefits

- Improved **fault isolation**.
- Eliminate **vendor or technology lock-in** (Since there is a use of open source technologies).
- **Ease of understanding**.
- **Smaller and faster deployments.**
- **Scalability**.

7) Microservices - Drawbacks

- **Complexity** is added to resolve complexity issues.
- **Testing** may appear simpler but is it?
- **Deployment** may appear simple but is it?
- Handling **multiple databases**.
- **Latency issues**.(You click a button in an app and it takes 3 seconds to respond)
- **Transient errors**. (You're using a cloud service (like a database or API), and suddenly you get an error. But then you retry after 2 seconds... and it works fine)
- **Multiple points of failures**.
- **Security issues**.