Introduction to Docker

Main goals:

- Explain the circumstances in which to use containerization and Docker specifically
- Explain how the Dockerfile works
- Use Docker to run a web server such as nginx or apache to display a static HTML page (Something as simple as Hello World)
- Use Dockerfile to create the above demo

Course Outline

1. **Explain the circumstances in which to use containerization and Docker**
    - Background: What is Docker?
        - New technology launched in 2013.
        - Builds on top of Linux kernel features for OS level virtualization.
        - Isolated containers where processes can run without impacting each other (process, memory, filesystem isolation).
        - Shipping code is hard. One of the goals of Docker is to solve this problem.
        - When things break in production, most common developer excuse is "Works on my laptop".
        - Docker provides a standardized container that works for shipping software from local to staging to production.
    - When to use Docker?
        - Anywhere where you are deploying software between multiple environments.
        - Docker provides an easy to use and portable way to run same software anywhere.
        - You can run same container on your laptop, same in CI environment and same in production environment.
        - Docker containers are order of magnitude faster than VMs.
        - Caveats:
            - Docker ecosystem is changing very fast as of right now, so

        if you want something mature or stable this is not it.
        - Fully leveraging Docker containers does require rethinking how to refactor or redesign code deployment pipelines.

   - How does Docker relate to the rest of container ecosystem
     - Docker is the defacto container engine and format right now.
     - The more lower level container primitives like cgroups & LXC are harder to use. Docker provides a really nice API on top of those lower level primitives.
     - A lot of ecosystem around containers is trying to solve problems of container scheduling and orchestration.
   - How does Docker affect DevOps?
     - Less sysadmin work.
     - Less variance between developer local environment and prod environment.
     - Enables more DevOps, since developers can easily package their own code!
     - Allows easier integration and end to end testing.

2. **Docker Building Blocks**
   - Containers
   - Images
   - Image Repository
     - DockerHub
   - Concept of layers in images
3. **Docker basic CLI commands (demo)**
   - `docker` (docker CLI for docker engine)
   - `docker ps`
   - `docker run`
   - `docker rm`
   - `docker images`
   - `docker run ubuntu ls` (simple demo of docker run)
4. **Use Docker to run a web server (demo)**
   - `sudo docker run --name docker-nginx -p 80:80 nginx`
   - Access webserver via browser
   - Explain concept of port mapping
5. **Explain how the Dockerfile works**
   - Purpose: to define our own image
   - Basic Dockerfile syntax and commands
     - FROM

        - RUN
        - ADD
    - More Dockerfile commands
        - ENV
        - EXPOSE
        - VOLUME
6. **Use Dockerfile to create the above demo**
    - Create the Dockerfile
    - `sudo docker build -t my-nginx .`
    - `sudo docker run -name my-nginx-instance my-nginx`