DAY 1:

SOFTWARE DEVELOPMENT LIFE CYCLE:

The Software Development Life Cycle (SDLC) is a structured process used for planning, creating, testing, and deploying software. It provides a systematic approach to software development, ensuring that projects are completed efficiently and meet quality standards. Here are the typical phases of SDLC:

Planning: Define the scope, objectives, and requirements of the project. This phase involves feasibility studies, resource allocation, and risk assessment.

Requirements Analysis: Gather and document user and business needs. This step forms the foundation for the software's features and functionality.

Design: Create architectural and detailed designs for the system. This includes data models, user interfaces, and technical workflows.

Implementation (Coding): Developers write the code for the software based on the design documents.

Testing: Validate the software through various tests (e.g., unit, integration, system, and user acceptance testing) to ensure quality and performance.

Deployment: Release the software to users, either all at once or in stages.

Maintenance: After deployment, address user feedback, fix issues, and update the software to adapt to changing needs or technologies.

DevOps:

DevOps is a set of practices and principles that combines **development (Dev)** and **operations (Ops)** to enhance collaboration between software developers and IT operations teams. The goal of DevOps is to streamline the software development lifecycle, improve the speed and quality of software delivery, and promote a culture of continuous improvement. Here are its key components:

Collaboration and Communication: DevOps fosters teamwork between development and operations, breaking down traditional silos.

Automation: Tasks like testing, deployment, and monitoring are automated to save time and reduce human error.

Continuous Integration (CI) and Continuous Delivery (CD): Frequent integration of code changes and automated deployment processes ensure that software can be delivered quickly and reliably.

Infrastructure as Code (IaC): Managing infrastructure using code allows for consistency, scalability, and version control.

Monitoring and Feedback: Real-time monitoring tools track system performance, and feedback loops provide insights for improvement.

Agility: DevOps embraces iterative and incremental approaches, allowing organizations to respond quickly to changes and user needs.

DOCKER:

Docker is an open-source platform that simplifies the development, deployment, and operation of applications by using containerization. Containers package an application along with all its dependencies, libraries, and configurations, ensuring it runs consistently across different environments. Here are some key points about Docker:

Lightweight Containers: Unlike virtual machines (VMs), Docker containers share the host operating system's kernel, making them faster and more efficient.

Portability: Containers can run on any system that supports Docker, whether it's your local machine, a data center, or a cloud environment.

Isolation: Each container operates independently, ensuring that changes in one container don't affect others.

Version Control: Docker allows versioning of containers, enabling easy rollback to previous versions when needed.

Ecosystem and Tools:

Docker Engine: The core runtime for building and running containers.

Docker Hub: A registry where developers can find and share container images.

Docker Compose: A tool for defining and managing multi-container applications.

Integration: Docker integrates with tools like Kubernetes for orchestration and CI/CD pipelines to streamline workflows.

Nginx:

Nginx (pronounced "engine X") is a powerful, open-source web server and reverse proxy server. It is widely used for its high performance, scalability, and ability to handle concurrent connections. Here's an overview of Nginx:

Web Server: Nginx serves static content like HTML, CSS, and JavaScript files efficiently, making it a popular choice for hosting websites.

Reverse Proxy: It acts as a gateway, forwarding client requests to backend servers (e.g., application servers) and returning the responses to the clients.

Load Balancing: Nginx distributes incoming traffic across multiple servers to ensure high availability and efficient utilization of resources.

HTTP/HTTPS Support: It supports modern web protocols and secure connections using SSL/TLS.

Scalability: Nginx is designed to handle a large number of simultaneous connections with minimal resource consumption, making it ideal for high-traffic sites.

Use Cases:

Serving static and dynamic web content.

Acting as an API gateway.

Caching and improving web performance.

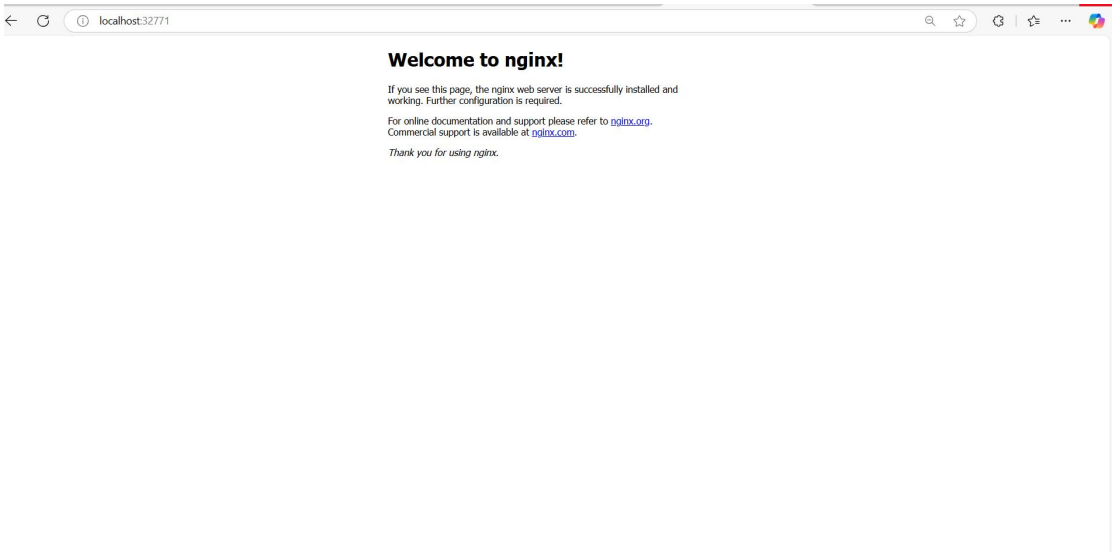
Enabling secure and scalable web architectures.

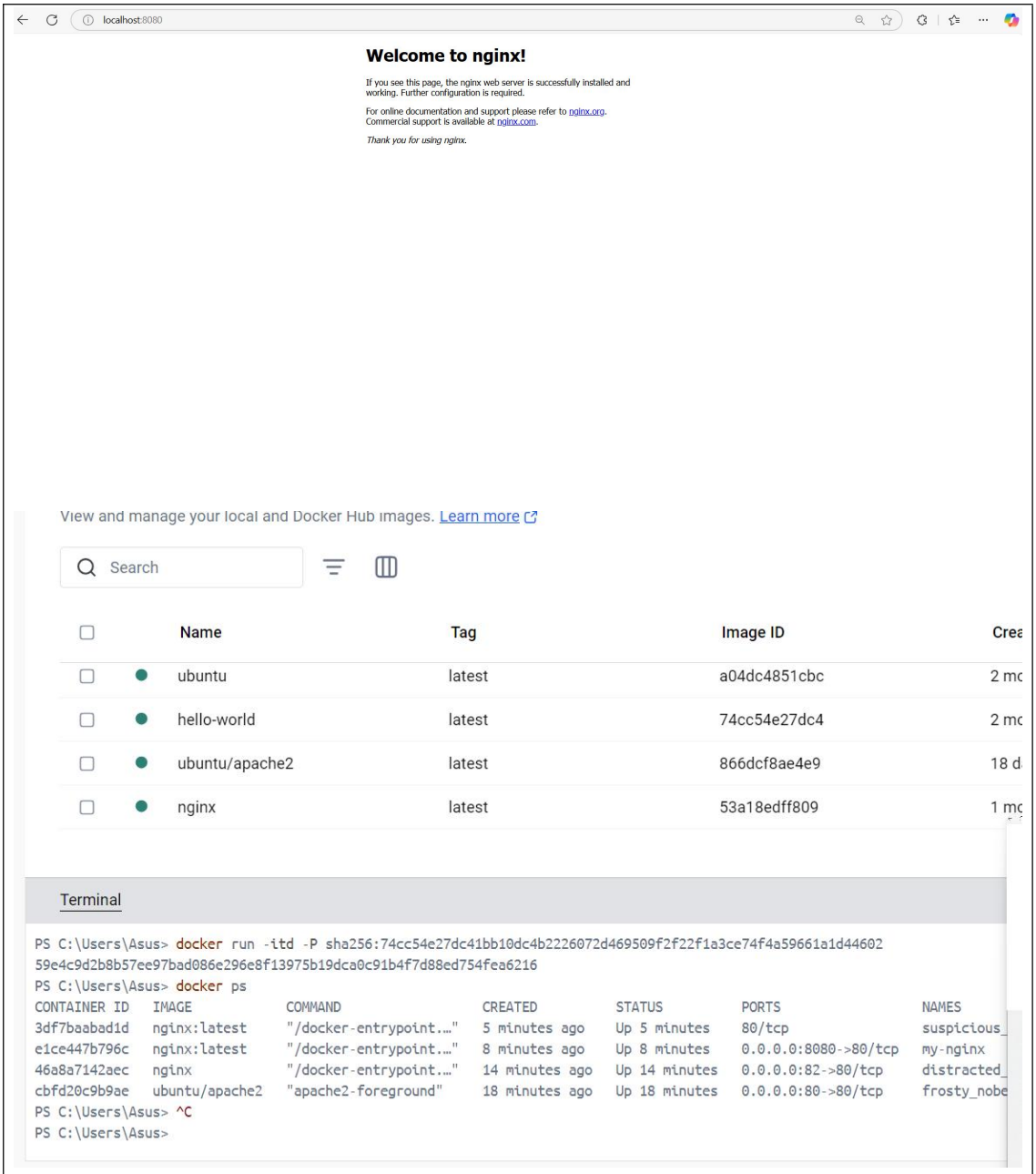
ole Output

Console Output

Download Copy View as plain text

```
Started by user ASHILIN B S
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\nginx1
[nginx1] $ cmd /c call C:\Windows\TEMP\jenkins6330323189387189312.bat

C:\ProgramData\Jenkins\.jenkins\workspace\nginx1>docker run -itd -P sha256:53a18edff8091d5faff1e42b4d885bc5f0f897873b0b8f0ace236cd5930819b0
d82deb5d3c035475332195608803ec51550598a0bef40eb6a6ccb4379f2f50f0

C:\ProgramData\Jenkins\.jenkins\workspace\nginx1>exit 0
[nginx1] $ cmd /c call C:\Windows\TEMP\jenkins2561699168541419939.bat

C:\ProgramData\Jenkins\.jenkins\workspace\nginx1>docker run -itd -P sha256:53a18edff8091d5faff1e42b4d885bc5f0f897873b0b8f0ace236cd5930819b0
a497f4432e42faee30424c78ca0f395a5313bd76f7aaa4b32dd6c5ab29565a91

C:\ProgramData\Jenkins\.jenkins\workspace\nginx1>exit 0
Finished: SUCCESS
```




localhost:8080
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.
View and manage your local and Docker Hub images. Learn more
Search
Name
Tag
Image ID
Crea
ubuntu
latest
a04dc4851cbc
2 mo
hello-world
latest
74cc54e27dc4
2 mo
ubuntu/apache2
latest
866dcf8ae4e9
18 d
nginx
latest
53a18edff809
1 mo
Terminal
PS C:\Users\Asus> docker run -itd -P sha256:74cc54e27dc41bb10dc4b2226072d469509f2f22f1a3ce74f4a59661a1d44602
59e4c9d2b8b57ee97bad086e296e8f13975b19dca0c91b4f7d88ed754fea6216
PS C:\Users\Asus> docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
3df7baabad1d nginx:latest "/docker-entrypoint.…" 5 minutes ago Up 5 minutes 80/tcp suspicious_
e1ce447b796c nginx:latest "/docker-entrypoint.…" 8 minutes ago Up 8 minutes 0.0.0.0:8080->80/tcp my-nginx
46a8a7142aec nginx "/docker-entrypoint.…" 14 minutes ago Up 14 minutes 0.0.0.0:82->80/tcp distracted_
cbfd20c9b9ae ubuntu/apache2 "apache2-foreground" 18 minutes ago Up 18 minutes 0.0.0.0:80->80/tcp frosty_nobe
PS C:\Users\Asus> ^C
PS C:\Users\Asus>