Here's an example of how you can develop a simple containerized

application using Docker:

Choose an application:

•Choose a simple application that you want to containerize. For

example, a Python script that prints "Hello World".

Write a Dockerfile:

•Create a file named "Dockerfile" in the same directory as the

application.

In the Dockerfile, specify the base image, copy the application into the

container, and specify the command to run the application. Here's an

example Dockerfile for a Python script:

# Use the official Python image as the base image

FROM python:3.9

# Copy the Python script into the container

COPY hello.py /app/

# Set the working directory to /app/

WORKDIR /app/

# Run the Python script when the container starts

CMD ["python", "hello.py"]

•Build the Docker image:

Run the following command to build the Docker image:

$ docker build -t myimage .

This command builds a new Docker image using the Dockerfile and tags the

image with the name "myimage".

•Run the Docker container:

Run the following command to start a new container based on the image:

$ docker run --name mycontainer myimage

This command starts a new container named "mycontainer" based on the

"myimage" image and runs the Python script inside the container.

•Verify the output:

Run the following command to verify the output of the container:

$ docker logs mycontainer

This command displays the logs of the container and should show the "Hello

World" output.

This is a simple example of how you can use Docker to containerize an

application. In a real-world scenario, you would likely have more complex

requirements, such as running multiple containers, managing network

connections, and persisting data. However, this example should give you a

good starting point for using Docker to containerize your applications.