Here is a short, step-by-step procedure for your exam, covering both experiments.

Procedure for Docker Image Management and Building a Custom Image
This procedure covers the container life cycle (Experiment 6) and building a custom web server image (Experiment 7).

Experiment 6: Docker Container Life Cycle Management
Pull an Image: Download an image (e.g., ubuntu) from the Docker Hub registry.

Bash

```
docker pull ubuntu
```

List Images: Verify that the image is now stored locally.

Bash

```
docker images
```

Run a Container: Create and run a new container from the image. We use -it to run in interactive terminal mode.

Bash

```
docker run -it ubuntu bash
```

Exit Container: Inside the container's shell, type exit to stop it.

Bash

```
exit
```

List Stopped Containers: Use docker ps -a to see all containers, including those that have exited. Note the container's name or ID.

Bash

```
docker ps -a
```

Remove a Container: Clean up the stopped container using its name or ID.

Bash

```
docker rm <container_name_or_id>
```

Remove an Image: Remove the image that is no longer in use by any container.

Bash

```
docker rmi ubuntu
```

Experiment 7: Build a Custom Image using a Dockerfile
Create Project Directory: Make a new folder for the project and move into it.

Bash

```
mkdir nginx-app
cd nginx-app
```

Create Content File: Create a static web page.

Bash

```
nano index.html
```

(Inside the file, add HTML content like <h1>Hello World</h1>)

Create Dockerfile: Create the build instruction file (note the capital 'D').

Bash

```
nano Dockerfile
```

Write Dockerfile Instructions: Add the following lines to the Dockerfile:

Dockerfile

```
# Use the official Nginx image as the base
FROM nginx:alpine

# Copy the local index.html into the Nginx web root directory
COPY index.html /usr/share/nginx/html/
```

Build the Image: Build a new image from the Dockerfile and tag (-t) it with a name (e.g., my-nginx-app). The . points to the current directory.

Bash

```
docker build -t my-nginx-app .
```

Run the Custom Image: Run the newly built image as a container.

-d: Run in detached (background) mode.

-p 80:80: Map port 80 on the host to port 80 in the container.

--name nginx-web-app: Give the container a custom name.

Bash

```
docker run -d -p 80:80 --name nginx-web-app my-nginx-app
```

Verify Application: Test the running container using curl (or a web browser at http://localhost).

Bash

```
curl http://localhost
```

Stop and Clean Up: Stop and remove the running container.

Bash

```
docker stop nginx-web-app
docker rm nginx-web-app
```

Bash

```
docker login
docker tag my-nginx-app YOUR_DOCKERHUB_USERNAME/my-nginx-app:latest

docker push YOUR_DOCKERHUB_USERNAME/my-nginx-app:latest
```