Here's a step-by-step guide to learning Docker by building, running, and pushing a simple FastAPI application Docker image:

1. Prepare a Simple FastAPI Application

Create a minimal FastAPI app with these files in a project directory:

- **main.py** (FastAPI app)

```
from fastapi import FastAPI

app = FastAPI()

@app.get("/")
def read_root():
    return {"message": "Hello, Docker with FastAPI!"}
```

- **requirements.txt** (dependencies)

```
fastapi
uvicorn[standard]
```

2. Create a Dockerfile to Build the Image

In the same directory, create a file named `Dockerfile` with the following content:

```
FROM python:3.9

WORKDIR /code

COPY ./requirements.txt /code/requirements.txt

RUN pip install --no-cache-dir --upgrade -r /code/requirements.txt

COPY ./main.py /code/

CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "80"]
```

Explanation:

- Use official Python 3.9 image as base.
- Set working directory inside container to `/code`.
- Copy `requirements.txt` and install dependencies.
- Copy FastAPI app file.
- Run the app with Uvicorn on port 80, binding to all interfaces[1][2].

3. Build the Docker Image

Open a terminal, navigate to your project directory (where the Dockerfile is), and run:

```
docker build -t myfastapiimage .
```

- `-t myfastapiimage` tags the image with the name `myfastapiimage`.
- The `.` means use current directory as build context[1][2][3].

4. Run the Docker Container

Start a container from the image with port mapping:

```
docker run -d --name myfastapicontainer -p 80:80 myfastapiimage
```

- `-d` runs container in detached mode (background).
- `--name` assigns a name to the container.
- `-p 80:80` maps container port 80 to host port 80, so you can access the app at http://localhost[1][2][4].

Check running containers with:

```
docker ps
```

To stop the container:

```
docker stop myfastapicontainer
```

5. Login to Docker Hub

If you don't have a Docker Hub account, create one at hub.docker.com.

Log in via terminal:

```
docker login
```

Enter your Docker Hub username and password when prompted[5][6][3].

6. Tag Your Image for Docker Hub

Docker Hub images are named as `<username>/<repository>:<tag>`. Tag your local image before pushing:

```
docker tag myfastapiimage yourdockerhubusername/myfastapiimage:latest
```

Replace `yourdockerhubusername` with your actual Docker Hub username[5][3].

7. Push the Image to Docker Hub

Push your tagged image:

```
docker push yourdockerhubusername/myfastapiimage:latest
```

After a successful push, your image will be available on Docker Hub for others to pull and run[5][3].

8. Pull and Run Your Image on Another Machine (Optional)

To test, on any machine with Docker, run:

```
docker pull yourdockerhubusername/myfastapiimage:latest
docker run -d -p 80:80 yourdockerhubusername/myfastapiimage:latest
```

Summary of Essential Docker Commands Used

Command	Purpose
`docker build -t name .`	Build Docker image from Dockerfile
`docker run -d -p host:container name`	Run container in background with port mapping
`docker ps`	List running containers
`docker stop container_name`	Stop a running container
`docker login`	Login to Docker Hub
`docker tag local_image repo:tag`	Tag image for Docker Hub
`docker push repo:tag`	Push image to Docker Hub

***

1. https://fastapi.tiangolo.com/deployment/docker/
2. https://www.restack.io/p/fastapi-answer-deployment-docker
3. https://www.incredibuild.com/blog/docker-101-a-comprehensive-tutorial-for-beginners
4. https://dev.to/hasanelsherbiny/top-10-docker-commands-you-need-to-learn-47cj
5. https://noted.lol/build-and-publish-to-docker-hub/
6. https://hub.docker.com/login