

Containerizing the application using docker

1. Dockerfile

```
Dockerfile X
1  FROM openjdk:22-oracle
2
3  # Set the working directory inside the container
4  WORKDIR /app
5
6  # Copy the JAR file into the container
7  COPY AMS.jar /app/AMS.jar
8
9  # Specify the command to run the JAR file
10 CMD ["java", "-jar", "AMS.jar"]
```

2. Building image

```
D:\ams>docker build -t prathik008/ams:1.0 .
[+] Building 3.2s (9/9) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile              0.0s
=> => transferring dockerfile: 278B                             0.0s
=> [internal] load metadata for docker.io/library/openjdk:22-oracle 3.1s
=> [auth] library/openjdk:pull token for registry-1.docker.io    0.0s
=> [internal] load .dockerignore                                0.0s
=> => transferring context: 2B                                    0.0s
=> [1/3] FROM docker.io/library/openjdk:22-oracle@sha256:08b2d714025cbb08c787f5395d931bae89345a856e4ab1be20891b 0.0s
=> [internal] load build context                                0.0s
=> => transferring context: 30B                                    0.0s
=> CACHED [2/3] WORKDIR /app                                    0.0s
=> CACHED [3/3] COPY AMS.jar /app/AMS.jar                       0.0s
=> exporting to image                                           0.0s
=> => exporting layers                                           0.0s
=> => writing image sha256:d8d0d27e01feff90aad373e54e8fb64a15b09bbf7d0742b99ef68d9e61731f2d 0.0s
=> => naming to docker.io/prathik008/ams:1.0                   0.0s
```

View build details: <docker-desktop:///dashboard/build/desktop-linux/desktop-linux/r64du87n6891sdp92dvrhjawn>

What's next:

View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

3. Running image

```
D:\ams>docker run -it prathik008/ams:1.0
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 1
Admin Operations
1. Add Admin
2. Remove Admin
3. Update Admin
4. Get Admin
5. List All Admins
6. Back to Main Menu
1
Enter Admin ID: 1
Enter Admin Name: Prathik
Admin added successfully.
Asset Management System
1. Admin Operations
2. Hardware Asset Operations
3. Employee Operations
4. Hardware Assigned Operations
5. Exit
Choose an option: 5
Exiting...
```

4. Creating tag and push the image to remote repository

```
D:\ams>docker push prathik008/ams:1.0
The push refers to repository [docker.io/prathik008/ams]
c0325bf12cdb: Pushed
0f4ad5ddd5d4: Layer already exists
6acaaba9e97a: Layer already exists
cf3ce83da20a: Layer already exists
0a628c3f1dfa: Layer already exists
1.0: digest: sha256:894c1eba4348b9a80b22bce1aec684d2f6211029c964362757e0ead03b4145be size: 1369
```

The screenshot shows the Docker Hub interface. At the top, there's a navigation bar with 'dockerhub', 'Explore', 'Repositories' (selected), and 'Organizations'. A search bar is on the right. Below the navigation bar, there's a filter section with 'prathik008' selected, a search input, and 'All Content'. A 'Create repository' button is on the right. The main content area lists two repositories: 'prathik008 / ams' and 'prathik008 / gadgets'. The 'ams' repository is highlighted, showing it contains an image, was last pushed 1 minute ago, has 0 stars, 0 downloads, is public, and has a Scout icon that is inactive. The 'gadgets' repository is also listed, showing it contains an image, was last pushed 5 days ago, has 0 stars, 4 downloads, is public, and has a Scout icon that is inactive.

