

Lab 6

BTI525 NCC

Fall 2025

INSTRUCTOR: ROBIN HUANG

Title: *Working with Docker Hub and Fragments*

Course: Web Programming for the Cloud

Student Name: Minhaz Abedin

Student ID: 171424229

Email: mabedin1@myseneca.ca

Date: October 27, 2025

Github Link: https://github.com/minhazabedin53/bti525_lab6

Docker Hub Link: <https://hub.docker.com/r/abedin53/fragments>

Github Link: https://github.com/minhazabedin53/bti525_lab6

Docker Hub Link: <https://hub.docker.com/r/abedin53/fragments>

Screenshot after:

- Pulled hello-world from Docker Hub
- Verified it appears in docker images
- Ran it and saw the "Hello from Docker!" message

```
PS C:\Users\minha\Downloads\BTI\btih525_lab6> docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
Digest: sha256:56433a6be3fd188089fb548eae3d91df3ed0d6589f7c2656121b911198df065
Status: Image is up to date for hello-world:latest
docker.io/library/hello-world:latest
● PS C:\Users\minha\Downloads\BTI\btih525_lab6> docker images
REPOSITORY      TAG          IMAGE ID       CREATED        SIZE
fragments       latest        6fd8a3ec3c38   12 days ago   1.83GB
hello-world     latest        56433a6be3fd   2 months ago  20.3kB
httpd           2.4-alpine    07b2fabb7029   3 months ago  93.2MB
● PS C:\Users\minha\Downloads\BTI\btih525_lab6> docker run --rm hello-world
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
● PS C:\Users\minha\Downloads\BTI\btih525_lab6> █
```

Screenshots of Tagging and Pushing My Image to Docker Hub

```
| ● PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker tag fragments:latest abedin53/fragments:latest
PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker tag fragments:latest abedin53/fragments:lab-6
● PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker login --username abedin53

[Info] Info → A Personal Access Token (PAT) can be used instead.  
To create a PAT, visit https://app.docker.com/settings
```

Password:

```
Login Succeeded
PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker push abedin53/fragments:latest
The push refers to repository [docker.io/abedin53/fragments]
fdf894e782a2: Pushed
5bd71677db44: Pushed
480cf7800ccb: Pushed
6f76c9e452df: Pushed
cd8cd1fac358: Pushed
460b77464f81: Pushed
551df7f94f9c: Pushed
d1b68cc99c4c: Pushed
d841a39ee66b: Pushed
ce82e98d553d: Pushed
0b8ec04fd63d: Pushed
a626dd299f99: Pushed
151024cdfe55: Pushed
latest: digest: sha256:6fd8a3ec3c38467c38145d6072b10c8de1ea18e31cddf7a27dc3cb9fe8ae size: 856
PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker push abedin53/fragments:lab-6
The push refers to repository [docker.io/abedin53/fragments]
151024cdfe55: Layer already exists
460b77464f81: Layer already exists
ce82e98d553d: Layer already exists
6f76c9e452df: Layer already exists
0b8ec04fd63d: Layer already exists
cd8cd1fac358: Layer already exists
d1b68cc99c4c: Layer already exists
fdf894e782a2: Layer already exists
a626dd299f99: Layer already exists
551df7f94f9c: Layer already exists
480cf7800ccb: Layer already exists
5bd71677db44: Layer already exists
d841a39ee66b: Already exists
lab-6: digest: sha256:6fd8a3ec3c38467c38145d6072b10c8de1ea18e31cddf7a27dc3cb9fe8ae size: 856
```

abedin53/fragments

Last pushed less than a minute by [abedin53](#)

TAG	Digest	OS/ARCH	Last pull	Compressed size
lab-6	d70756fb0218	linux/amd64	less than 1 day	442.37 MB
latest	d70756fb0218	linux/amd64	less than 1 day	442.37 MB

Screenshot: Pulling & running from Docker Hub (local):

```

lab-6: digest: sha256:0708a3ec3c38407c3814500072b10cc1e810e5100f7fb7a27dc3cb9fe8ae size: 850
PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker kill fragments 2>$null
PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker pull abedin53/fragments:lab-6
● lab-6: Pulling from abedin53/fragments
Digest: sha256:6fd8a3ec3c38467c38145d6072b10c8de1ea18e31cddfeffb7a27dc3cb9fe8ae
Status: Image is up to date for abedin53/fragments:lab-6
docker.io/abedin53/fragments:lab-6
PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker run --rm --name fragments --env-file env.basic -p 5555:8080 -d abedin53/fragments:lab-6
● 7bbdb6a4c728097cc6f2543886fb6465c71033a38d40c6dc550e1ce50e1c024e
● PS C:\Users\minha\Downloads\BTI\bt1525_lab6> curl.exe -u user:pass123 http://localhost:5555
{"status": "ok", "author": "Minhaz Abedin mabedin1@myseneca.ca", "githubUrl": "https://github.com/minhazabedin53/fragments-service", "version": "1.0.0"}
PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker logs fragments
●
> fragments-service@1.0.0 start
> node src/server.js

[dotenv@17.2.3] injecting env (0) from .env -- tip: ⚙️ load multiple .env files with { path: ['.env.local', '.env'] }
{"level":30,"time":1761656330670,"pid":20,"hostname":"7bbdb6a4c728","port":"8080","msg":"Fragments service listening"}
{"level":30,"time":1761656337852,"pid":20,"hostname":"7bbdb6a4c728","req":{"id":1,"method":"GET","url":"/","query":{},"params":{},"headers":{"host":"localhost:5555","authorization":"Basic dXNlcjpwYXNzMTIz","user-agent":"curl/8.14.1","accept":"/*/*"}, "remoteAddress":":ffff:172.17.0.1", "remotePort":48558}, "res":{"statusCode":200,"headers":{"content-security-policy": "default-src 'self';base-uri 'self';font-src 'self' https: data:;form-action 'self';frame-ancestors 'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr 'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests","cross-origin-opener-policy": "same-origin","cross-origin-resource-policy": "same-origin","origin-agent-cluster": "?1","referrer-policy": "no-referrer","strict-transport-security": "max-age=31536000; includeSubDomains","x-content-type-options": "none","x-dns-prefetch-control": "off","x-download-options": "noopener","x-frame-options": "SAMEORIGIN","x-permitte d-cross-domain-policies": "none","x-xss-protection": "0","access-control-allow-origin": "*","cache-control": "no-store, no-cache, must-revalidate, proxy-revalidate","pragma": "no-cache","expires": "0","surrogate-control": "no-store","content-type": "application/json; charset=utf-8","content-length": "145","etag": "W/\"91-3uvSxIsDN+52CQ8eU7/b3qlgqM\""}, "responseTime": 4, "msg": "request completed"}
PS C:\Users\minha\Downloads\BTI\bt1525_lab6> docker ps
● CONTAINER ID IMAGE COMMAND CREATED STATUS
PORTS NAMES
7bbdb6a4c728 abedin53/fragments:lab-6 "docker-entrypoint.s..." About a minute ago Up About a minute
0.0.0.0:5555->8080/tcp, [::]:5555->8080/tcp fragments
○ PS C:\Users\minha\Downloads\BTI\bt1525_lab6>

```

Screenshot: Docker running on EC2:

docker ps on EC2 showing 0.0.0.0:8080->8080 and Up
docker logs fragments on EC2 showing the GET request

```
[ec2-user@ip-172-31-29-218 ~]$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
NAMES
17b79983dc4d      abed1n53/fragments:lab-6   "docker-entrypoint.s..."   9 seconds ago     Up 8 seconds    0.0.0.0:8080->8080/tcp, :::8080->8080/tcp
fragments

[ec2-user@ip-172-31-29-218 ~]$ docker logs -f fragments
> fragments-service@1.0.0 start
> node src/server.js

[dotenv@17.2.3] injecting env (0) from .env -- tip: ⚙ override existing env vars with { override: true }
{"level":30,"time":1761659082910,"pid":20,"hostname":"17b79983dc4d","port":8080,"msg":"Fragments service listening"}
{"level":30,"time":1761659120321,"pid":20,"hostname":"17b79983dc4d","req":{"id":1,"method":"GET","url":"/","query":{},"params":{},"headers":{"host":"ec2-18-212-248-95.compute-1.amazonaws.com:8080","authorization":"Basic dXNlclpwYXNzMTIz","user-agent":"curl/8.14.1","accept":"*/*"}, "remoteAddress":":ffff:184.146.17.66", "remotePort":54061}, "res":{"statusCode":200,"headers":{"content-security-policy":"default-src 'self';base-uri 'self';font-src 'self' https: data:;form-action 'self';frame-ancestors 'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr 'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests", "cross-origin-opener-policy":"same-origin", "cross-origin-resource-policy":"same-origin", "origin-agent-cluster":"?1", "referrer-policy":"no-referrer", "strict-transport-security":"max-age=31536000; includeSubDomains", "x-content-type-options":"nosniff", "x-dns-prefetch-control":"off", "x-download-options":"noopener", "x-frame-options":"SAMEORIGIN", "x-permitted-cross-domain-policies":"none", "x-xss-protection":"0", "access-control-allow-origin":"*", "cache-control":"no-store, no-cache, must-revalidate, proxy-revalidate", "pragma":"no-cache", "expires":"0", "surrogate-control":"no-store", "content-type":"application/json; charset=utf-8", "content-length":145, "etag":W/"91-3uvSxIsDN+x52CQ8eU7/b3qlgqM\\"", "responseTime":6, "msg":"request completed"}}

[]
```

Screenshot: Docker call on EC2 from local

curl.exe -u user:pass123 http://ec2-18-212-248-95.compute-1.amazonaws.com:8080 returning JSON

```
● PS C:\Users\minha\Downloads\BTI\bt1525_lab6> curl.exe -u user:pass123 http://ec2-18-212-248-95.compute-1.amazonaws.com:8080
{"status":"ok", "author":"Minhaz Abedin mabedini@myseneca.ca", "githubUrl":"https://github.com/minhazabedin53", "fragments-service", "version":"1.0.0"}
○ PS C:\Users\minha\Downloads\BTI\bt1525_lab6> []
```

Screenshot: Step 21 convert working on ec2

```
● PS C:\Users\minha\Downloads\BTI\bt1525_lab6> curl.exe -u user:pass123 http://ec2-18-212-248-95.compute-1.amazonaws.com:8080/
{"status":"ok","author":"Minhaz Abedin mabedin1@myseneca.ca","githubUrl":"https://github.com/minhazabedin53/fragments-service","version":"1.0.0"}
● PS C:\Users\minha\Downloads\BTI\bt1525_lab6> curl.exe -u user:pass123 -H "Content-Type: text/markdown" --data "# Hello`n`n**bold**" -i http://ec2-18-212-248-95.compute-1.amazonaws.com:8080/v1/fragments
HTTP/1.1 201 Created
Content-Security-Policy: default-src 'self';base-uri 'self';font-src 'self' https: data:;form-action 'self';
frame-ancestors 'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr 'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: same-origin
Origin-Agent-Cluster: ?1
Referrer-Policy: no-referrer
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-DNS-Prefetch-Control: off
X-Download-Options: noopen
X-Frame-Options: SAMEORIGIN
X-Permitted-Cross-Domain-Policies: none
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
Location: http://localhost:8080/v1/fragments/642fa7d0-43f8-44d7-94af-348a6ddcd648
Content-Type: application/json; charset=utf-8
Content-Length: 256
ETag: W/"100-BS+xgSQowr/fjGz0ySwiAsbHPQ"
Date: Tue, 28 Oct 2025 15:31:19 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{"status":"ok","fragment": {"id": "642fa7d0-43f8-44d7-94af-348a6ddcd648", "ownerId": "04f8996da763b7a969b1028ee3007569eaef3a635486ddab211d512c85b9df8fb", "created": "2025-10-28T15:31:19.008Z", "updated": "2025-10-28T15:31:19.008Z", "type": "text/markdown", "size": 17}}
● PS C:\Users\minha\Downloads\BTI\bt1525_lab6> curl.exe -u user:pass123 -i http://ec2-18-212-248-95.compute-1.amazonaws.com:8080/v1/fragments/642fa7d0-43f8-44d7-94af-348a6ddcd648.html
HTTP/1.1 200 OK
Content-Security-Policy: default-src 'self';base-uri 'self';font-src 'self' https: data:;form-action 'self';
frame-ancestors 'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr 'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: same-origin
Origin-Agent-Cluster: ?1
Referrer-Policy: no-referrer
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-DNS-Prefetch-Control: off
X-Download-Options: noopen
X-Frame-Options: SAMEORIGIN
X-Permitted-Cross-Domain-Policies: none
X-XSS-Protection: 0
Access-Control-Allow-Origin: *
Content-Type: text/html; charset=utf-8
Content-Length: 44
ETag: W/"2c-ytLCOSjEQcfNfXbYZTFtuPxnS2o"
Date: Tue, 28 Oct 2025 15:31:55 GMT
Connection: keep-alive
Keep-Alive: timeout=5

<h1>Hello</h1>
<p><strong>bold</strong></p>
● PS C:\Users\minha\Downloads\BTI\bt1525_lab6>
```

Explanation of Optimizations (Step 20)

For Step 20, optimized my Dockerfile following Docker's official best-practice guidelines.

- Used official base image: node:22.11.0-alpine (lightweight, secure).
- Pinned version: used a specific version tag to ensure reproducibility.
- Multi-stage build: separated the build and runtime stages so the final image contains only the production dependencies and compiled code.
- Non-root user: switched to a dedicated node user for better security.
- Environment variables: set NODE_ENV=production and LOG_LEVEL=info to optimize runtime behavior.
- Healthcheck: added a Docker HEALTHCHECK instruction to monitor container health.
- Optimized layer caching: copied only package*.json before installing dependencies, reducing rebuild time.

Implementation of Requirement (Step 21)

For Step 21, I implemented one new Assignment 2 requirement: "GET /v1/fragments/:id.ext returns an existing fragment's data converted to a supported type."

I created a new module src/services/convert.js that uses the markdown-it library to convert Markdown fragments (text/markdown) to HTML (text/html). A new route /v1/fragments/:id.:ext was added in src/routes/convert.js and registered in app.js.

I verified this feature using:

```
curl -u user:pass123 -H "Content-Type: text/markdown"
--data "# Hello\n\nbold" http://localhost:3000/v1/fragments curl -u user:pass123
http://localhost:3000/v1/fragments/.html
```

The second command successfully returned:

```
<h1>Hello</h1>
<p><strong>bold</strong></p>
```

This proves the Markdown-to-HTML conversion works correctly in both local and EC2 deployments.