

클컴 - 실습과제 2 보고서

2021320308 박한준

공통 - docker hub repository

pizzazoa/frontend ⓘ

Last pushed 3 minutes ago · Repository size: 49 MB · ⭐0 · ⌂9

Add a description ⓘ ⓘ
Add a category ⓘ ⓘ

General Tags Image Management BETA Collaborators Webhooks Settings

Sort by Newest Filter tags Delete

| TAG | Digest | OS/ARCH | Last pull | Compressed size |
|-----|--------------|-------------|-----------------|-----------------|
| v2 | 070b2c43e020 | linux/amd64 | less than 1 day | 48.98 MB |

| TAG | Digest | OS/ARCH | Last pull | Compressed size |
|-----|--------------|-------------|-----------------|-----------------|
| v1 | 689f7b7844fb | linux/amd64 | less than 1 day | 48.98 MB |

pizzazoa/backend ⓘ

Last pushed 10 minutes ago · Repository size: 47.4 MB · ⭐0 · ⌂8

Add a description ⓘ ⓘ
Add a category ⓘ ⓘ

General Tags Image Management BETA Collaborators Webhooks Settings

Sort by Newest Filter tags Delete

| TAG | Digest | OS/ARCH | Last pull | Compressed size |
|-----|--------------|-------------|-----------------|-----------------|
| v2 | 40e627e80934 | linux/amd64 | less than 1 day | 47.4 MB |

| TAG | Digest | OS/ARCH | Last pull | Compressed size |
|-----|--------------|-------------|-----------------|-----------------|
| v1 | dd8623c13ce2 | linux/amd64 | less than 1 day | 47.4 MB |

v1 screenshot

docker ps (실행 중인 컨테이너)

```
ubuntu@VM-2-96-ubuntu:~/assign2$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
3802cbe8af4        frontend:v1      "python app.py"    26 seconds ago   Up 25 seconds   0.0.0.0:5000->5000/tcp, [::]:5000->5000/tcp   frontend
44fac0926557       backend:v1       "python app.py"    About a minute ago   Up About a minute  0.0.0.0:5001->5001/tcp, [::]:5001->5001/tcp   backend
ubuntu@VM-2-96-ubuntu:~/assign2$
```

volume content

```
[ubuntu@VM-2-96-ubuntu:~/assign2$ sudo docker exec backend cat /data/message.txt
hello! this is the v1 version message!ubuntu@VM-2-96-ubuntu:~/assign2$
```

frontend webpage

The image shows two side-by-side browser windows. Both windows have a title bar 'Frontend Service' and a URL bar showing '43.128.136.212:5000'. The left window displays a form with a text input field containing 'Type new message' and a button labeled 'Update'. The right window displays the same form with the message 'hello! this is the v1 version message!' already entered in the input field.

- 왼쪽이 초기화면, 오른쪽이 메시지 입력 후 업데이트 버튼을 누른 화면이다.

browser hitting backend API

The image shows a browser window with the URL '43.128.136.212:5001/api/message'. Below the URL bar, there is a checkbox labeled 'pretty print 적용'. The main content area displays the JSON response: {"message": "hello! this is the v1 version message!"}.

Network appnet

```
[ubuntu@VM-2-96-ubuntu:~/assign2$ sudo docker exec backend cat /data/message.txt
[hello! this is the v1 version message!ubuntu@VM-2-96-ubuntu:~/assign2$ sudo docker network inspect appnet
[
  {
    "Name": "appnet",
    "Id": "5c058b3b3f0780d4c0952b0119e94f0a08fbcb8cfbc86d611d97588b91d9dc77",
    "Created": "2025-11-21T16:18:08.750907218+08:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv4": true,
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "IPRange": "",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Options": {},
    "Labels": {},
    "Containers": {
      "3802cbbe8af476d36e66846fd843974ce0c4425a8f3aedeb0e3c433241f380c4": {
        "Name": "frontend",
        "EndpointID": "dd777db28a916e9ace2e71fa47d60ec7d82924853fb7ada19b650f3d69fbb4c9",
        "MacAddress": "72:c9:74:7d:89:21",
        "IPv4Address": "172.18.0.3/16",
        "IPv6Address": ""
      },
      "44fac0926557c9780d1f3c00b856e109dd8bc83aca61f23f6576a9f58ffc452e": {
        "Name": "backend",
        "EndpointID": "e31114de9131ca8fbca72f222aceb7e63d22c7a81834ef9189d107ecbb1d3a01",
        "MacAddress": "16:5b:5b:f9:2a:73",
        "IPv4Address": "172.18.0.2/16",
        "IPv6Address": ""
      }
    },
    "Status": {
      "IPAM": {
        "Subnets": {
          "172.18.0.0/16": {
            "IPsInUse": 5,
            "DynamicIPsAvailable": 65531
          }
        }
      }
    }
  }
]
ubuntu@VM-2-96-ubuntu:~/assign2$
```

v2 screenshot

docker ps (실행 중인 컨테이너)

```
ubuntu@VM-2-96-ubuntu:~/assign2$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED      STATUS      PORTS          NAMES
f54147d812e4   frontend:v2   "python app.py"   About a minute ago   Up About a minute   0.0.0.0:5000->5000/tcp, [::]:5000->5000/tcp   frontend
[6daf4d0696b1] backend:v2   "python app.py"   2 minutes ago   Up 2 minutes   0.0.0.0:5001->5001/tcp, [::]:5001->5001/tcp   backend
[ubuntu@VM-2-96-ubuntu:~/assign2$
```

volume content

```
[ubuntu@VM-2-96-ubuntu:~/assign2$ sudo docker exec backend cat /data/message.txt
hello! this is the v2 version message! (updated at 2025-11-21 09:05:10)ubuntu@VM-2-96-ubuntu:~/assign2$ ]
```

frontend webpage

The image shows two side-by-side browser windows. Both windows have the URL `43.128.136.212:5000` in the address bar. The left window has a header "주의 요함" and the right window has a header "주의 요함". Below the address bar, there are tabs for NAVER, 넷플릭스, YouTube, Y Music, and Google. The main content area of both windows displays the text "Frontend Service v2". Underneath, there is a section titled "Current Message:" with the text "hello! this is the v1 version message!". Below this is a section titled "Update Message" with a form containing a text input field "Type new message" and a button "Update". The right window also contains the same "Current Message:" and "Update Message" sections.

- 왼쪽이 초기화면, 오른쪽이 메시지 입력 후 업데이트 버튼을 누른 화면이다.

browser hitting backend API

The image shows a single browser window with the URL `43.128.136.212:5001/api/message` in the address bar. The header "주의 요함" is present. Below the address bar, there are tabs for NAVER, 넷플릭스, YouTube, Y Music, Google Gemini, ChatGPT, Claude, and Co. A checkbox labeled "pretty print 적용" is checked. The main content area displays the JSON response: `{"message":"hello! this is the v2 version message! (updated at 2025-11-21 09:05:10)"}`.

The image shows a single browser window with the URL `43.128.136.212:5001/api/health` in the address bar. The header "주의 요함" is present. Below the address bar, there are tabs for NAVER, 넷플릭스, YouTube, Y Music, Google Gemini, ChatGPT, Claude, and Co. A checkbox labeled "pretty print 적용" is checked. The main content area displays the JSON response: `{"status":"healthy"}`.

Network appnet

```
[ubuntu@VM-2-96-ubuntu:~/assign2$ sudo docker exec backend cat /data/message.txt
hello! this is the v2 version message! (updated at 2025-11-21 09:05:10)ubuntu@VM-2-96-ubuntu:~/assign2$ sudo docker network inspect appnet
[{"Name": "appnet",
 "Id": "5c058b3b3f0780d4c0952b0119e94f0a08fbcb8cfbc86d611d97588b91d9dc77",
 "Created": "2025-11-21T16:18:08.750907218+08:00",
 "Scope": "local",
 "Driver": "bridge",
 "EnableIPv4": true,
 "EnableIPv6": false,
 "IPAM": {
   "Driver": "default",
   "Options": {},
   "Config": [
     {
       "Subnet": "172.18.0.0/16",
       "IPRange": "",
       "Gateway": "172.18.0.1"
     }
   ]
 },
 "Internal": false,
 "Attachable": false,
 "Ingress": false,
 "ConfigFrom": {
   "Network": ""
 },
 "ConfigOnly": false,
 "Options": {},
 "Labels": {},
 "Containers": {
   "6ae65ffc5c711fa80cc40d6b1c79b987119fa750e2eee9ed424327f4296231cb": {
     "Name": "frontend",
     "EndpointID": "bab37b8134c9af03db7e56651d9447eec63f12b47125728e6e51e9ee81d88dc",
     "MacAddress": "f2:7e:96:de:be:48",
     "IPv4Address": "172.18.0.3/16",
     "IPv6Address": ""
   },
   "6daf4d0696b113742ac42e7236715261a8a72f1a4e14ec56fa341e567f32a4e6": {
     "Name": "backend",
     "EndpointID": "2c19a6f25e83a13d8f4d19cf521a263c7ac9a718fa550318bba375a5f144ac85",
     "MacAddress": "9a:06:09:34:15:0f",
     "IPv4Address": "172.18.0.2/16",
     "IPv6Address": ""
   }
 },
 "Status": {
   "IPAM": {
     "Subnets": {
       "172.18.0.0/16": {
         "IPsInUse": 5,
         "DynamicIPsAvailable": 65531
       }
     }
   }
 }
}
ubuntu@VM-2-96-ubuntu:~/assign2$
```

Test output (part C)

```
[ubuntu@VM-2-96-ubuntu:~$ curl http://43.128.136.212:5000
<!DOCTYPE html>
<html>
<head>
  <title>Frontend Service v2</title>
</head>
<body>
  <h1>Frontend Service v2</h1>

  <h2>Current Message:</h2>
  <p id="current-message">hello! this is the v2 version message!</p>

  <h2>Last Updated At:</h2>
  <p>2025-11-21 09:05:10</p>

  <h2>Update Message</h2>
  <form action="/update" method="post">
    <input type="text" name="new_message" placeholder="Type new message" required>
    <button type="submit">Update</button>
  </form>
</body>
```

```
[ubuntu@VM-2-96-ubuntu:~/assign2$ curl http://43.128.136.212:5001/api/message
>{"message":"hello! this is the v2 version message! (updated at 2025-11-21 09:05:10)"}
[ubuntu@VM-2-96-ubuntu:~/assign2$ curl http://43.128.136.212:5001/api/health
{"status":"healthy"}
ubuntu@VM-2-96-ubuntu:~/assign2$
```

Short Explanation

1. How the frontend communicates with the backend

API를 통해 소통한다. 특히 프론트엔드가 파이썬 라이브러리인 `requests`를 사용해 백엔드에 요청을 보낸다. `GET` 요청을 통해 데이터를 받아오고, `POST` 요청을 통해 프론트에서 백으로 데이터를 보내기도 한다. 각 주소(ip+port)가 목적지가 되어 메시지 패싱을 하는데, 이때 도커 네트워크 상에서 우리가 설정한 컨테이너명(backend, frontend)이 일종의 DNS로 동작한다.

2. Why Docker needs a shared network

도커 컨테이너는 기본적으로 격리된 환경에서 실행되며, 서로 다른 bridge 네트워크에 속한 컨테이너들은 직접 통신이 불가능하다. 따라서 프론트엔드와 백엔드가 서로 데이터를 주고 받기 위해서 `appnet`과 같은 동일한 네트워크를 공유해야 한다. 이 네트워크를 통해 두 컨테이너는 같은 서브넷에 속하게 되며, ip가 아닌 컨테이너 이름을 통한 DNS 조회 및 통신이 가능해진다.

3. What the volume is used for

도커 컨테이너는 stateless 환경이기 때문에 컨테이너가 삭제, 재시작, 교체되면 내부 데이터도 함께 사라진다. 이때 데이터를 영구적으로 보존하기 위해 볼륨을 사용한다. 위 스크린샷에서 v2의 초기화면에 v1에서의 메시지가 그대로 남아있는 것은 볼륨을 이용해 보존했기 때문이다.

4. What you changed for v2

v2의 백엔드에선 `datetime` 라이브러리의 `now()` 함수를 통해, 메시지 업데이트 시 타임스탬프를 "(updated at ...)"의 형식으로 추가되게 수정했고, 서버 상태를 확인하는 헬스체크 API를 추가했다. 프론트엔드의 `app.py`에선 `GET` 요청으로 백엔드에 저장된 메시지를 불러올 때, "(updated at"이라는 문자열을 구분자로 하여 오른쪽에서부터 `rsplit` 해서 타임스탬프를 읽어올 수 있게끔 했다. 또한 `templates/index.html`에서, 타임스탬프가 존재하면 `Last Updated At:`이라는 헤더를 추가하고, 그 시간을 표기하게끔 수정했다.