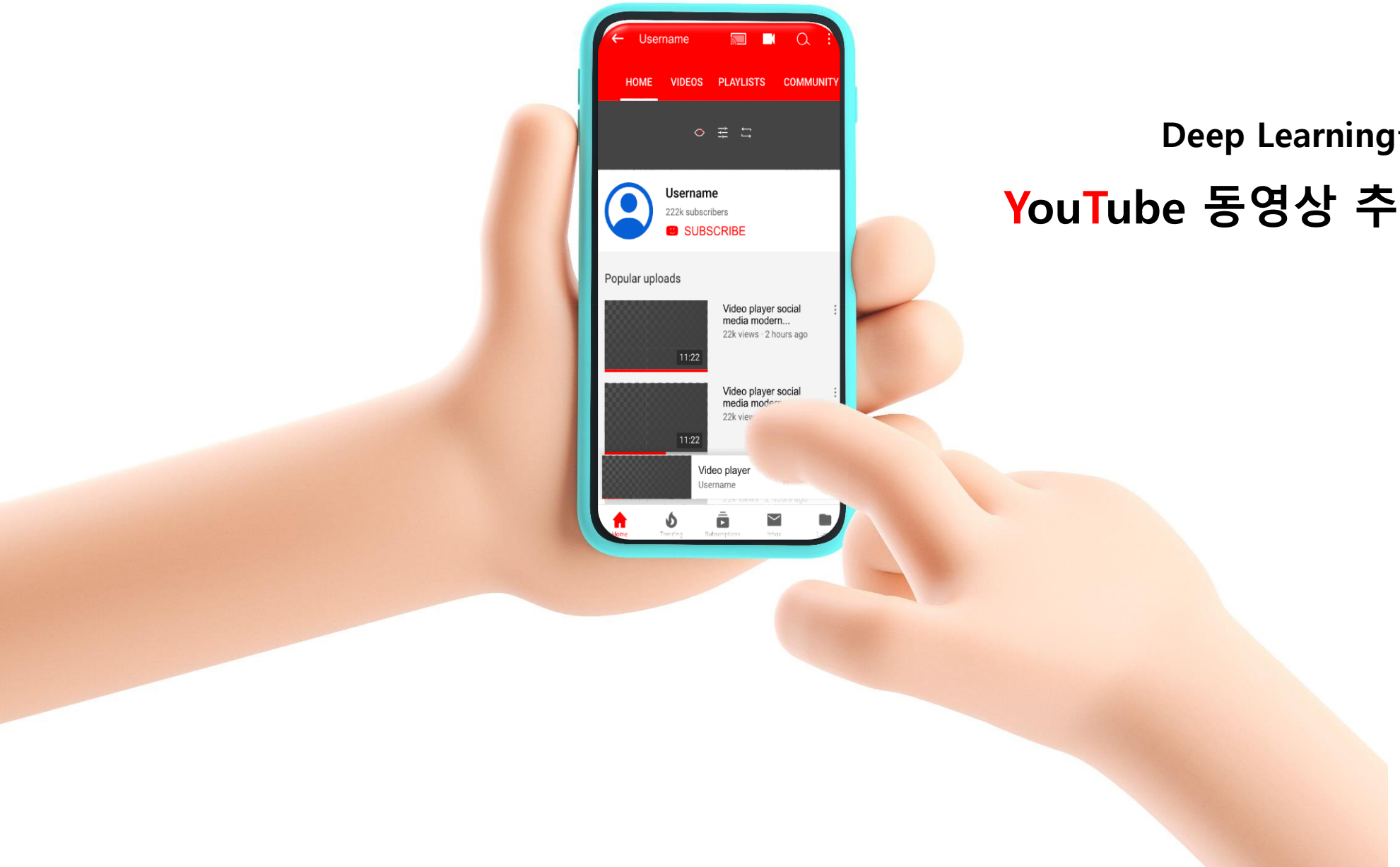


Deep Learning을 활용한

YouTube 동영상 추천 서비스 구현

Team MaruBro

김승주, 황윤지

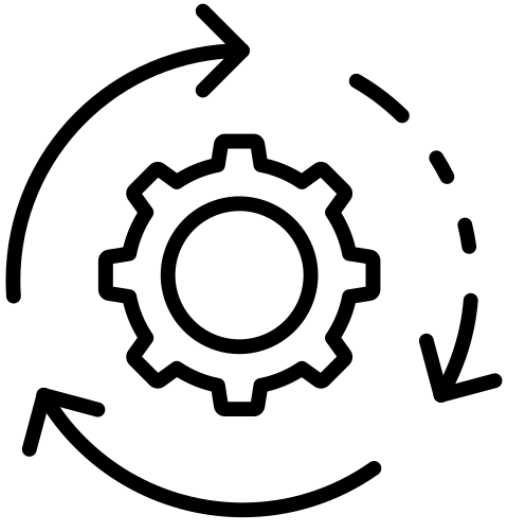
Mento

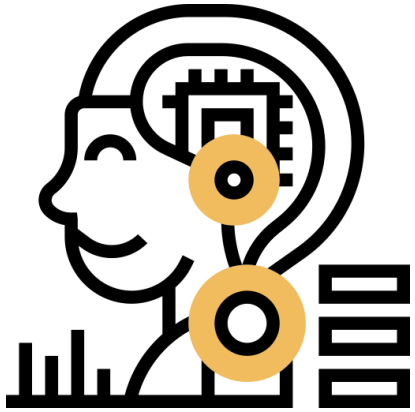
이찬우

작업기간 3개월

Contents

Team
MaruBro



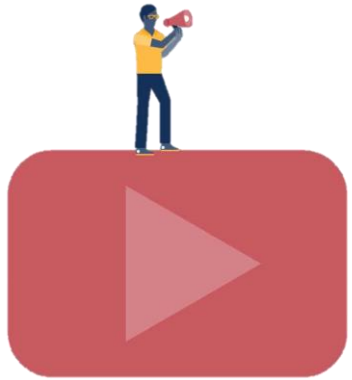


MaruBro

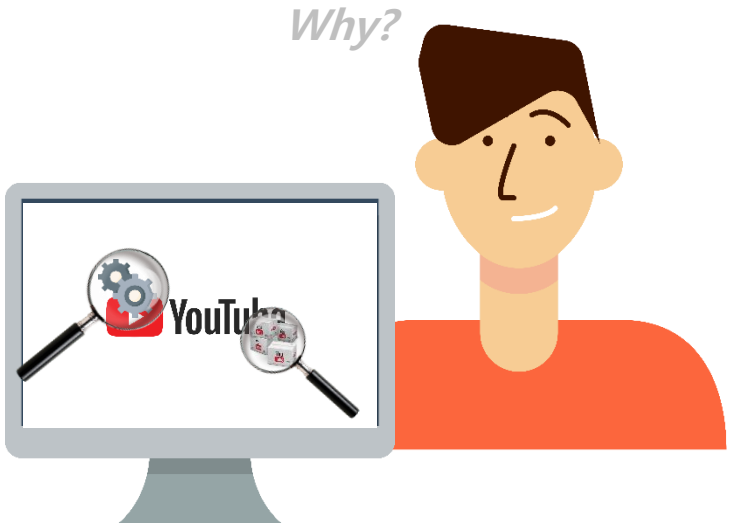
Deep Learning Production

Team
MaruBro

Idea Review



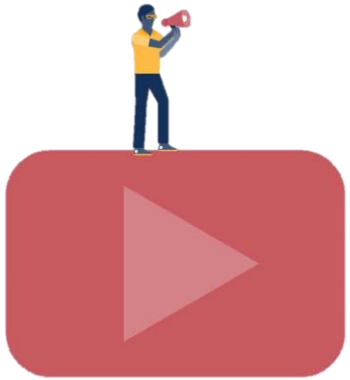
Why & How | Idea



- ✓ **Content-based Recommendation System** — Computer vision
- ✓ **Deep Learning** — Video/image data
- ✓ **Servitization** — production/deploy

Team MaruBro

About Our System Structure



System Structure

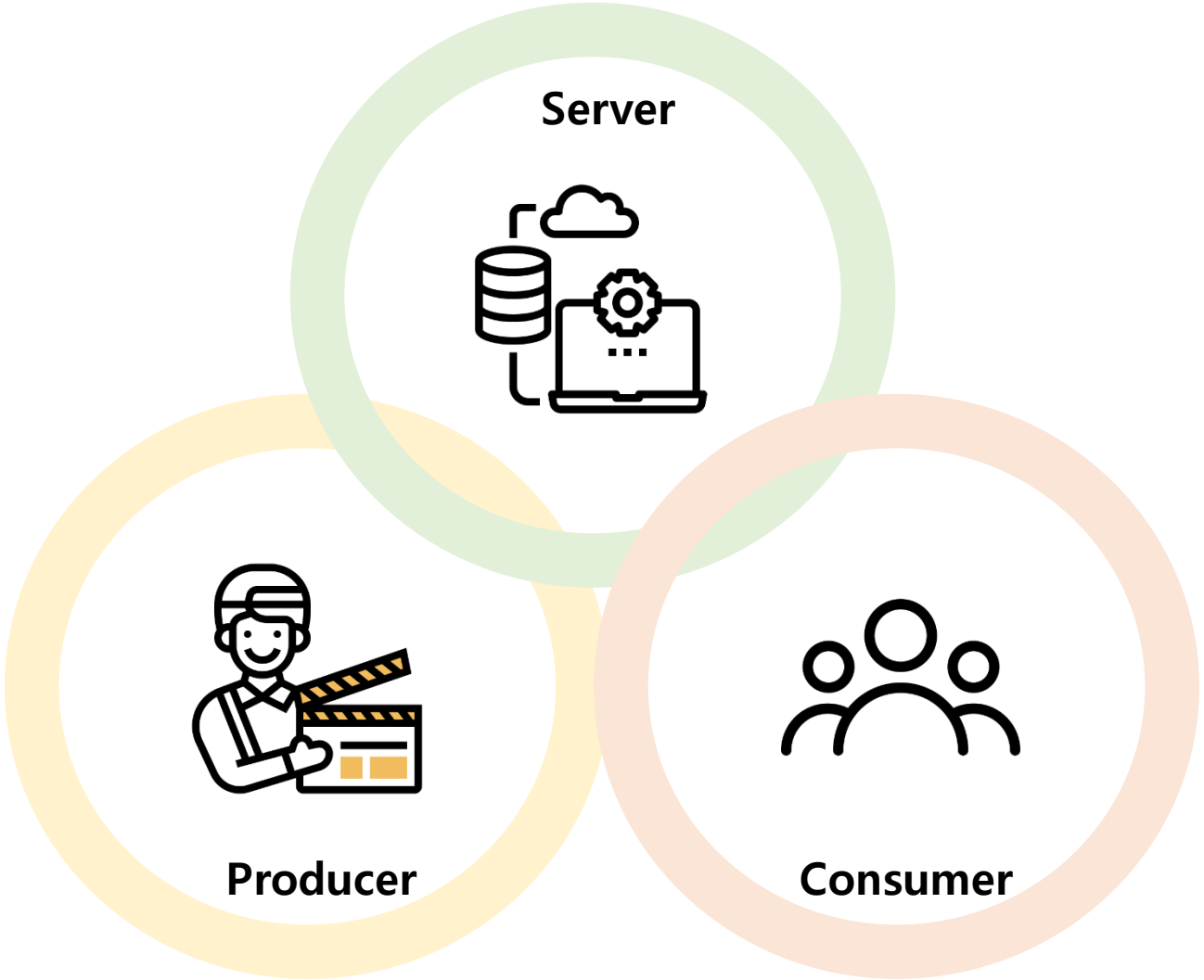
*appendix

productivity

Lambda

DynamoDB

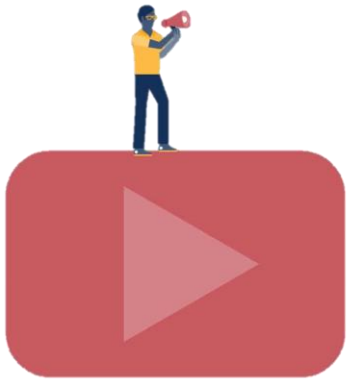
System Structure



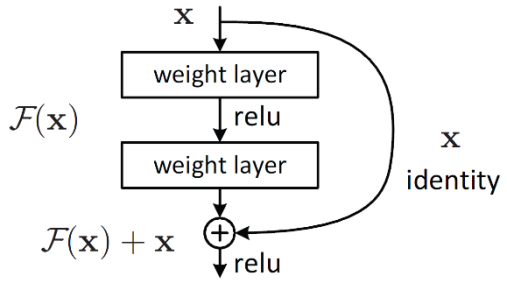
Team
MaruBro

1. Server System

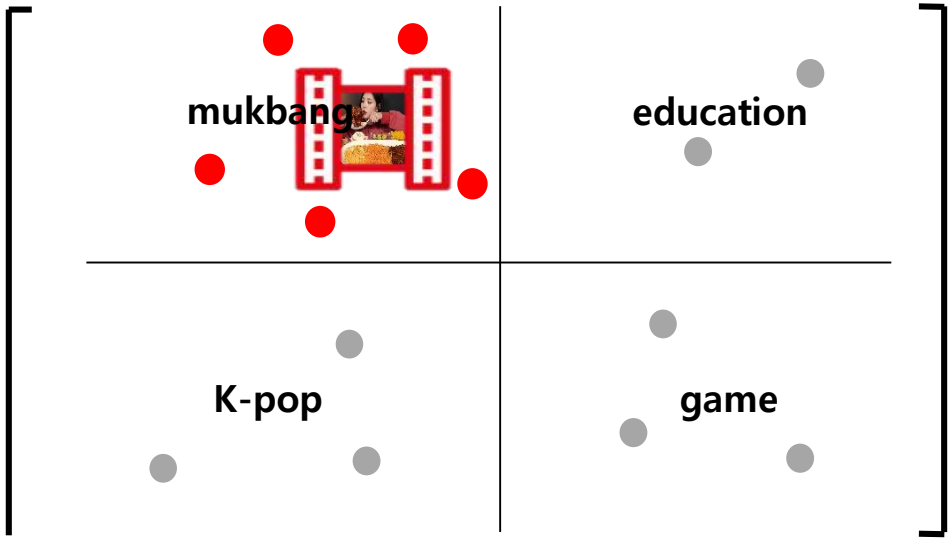
: Deep Learning Model



System | Our Model

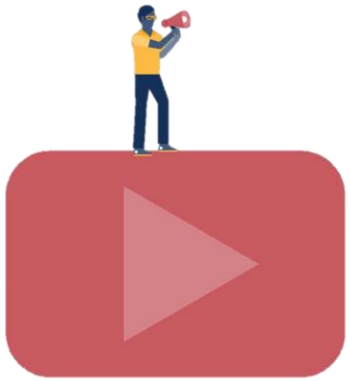


K-POP EDUCATION MUKBANG GAME

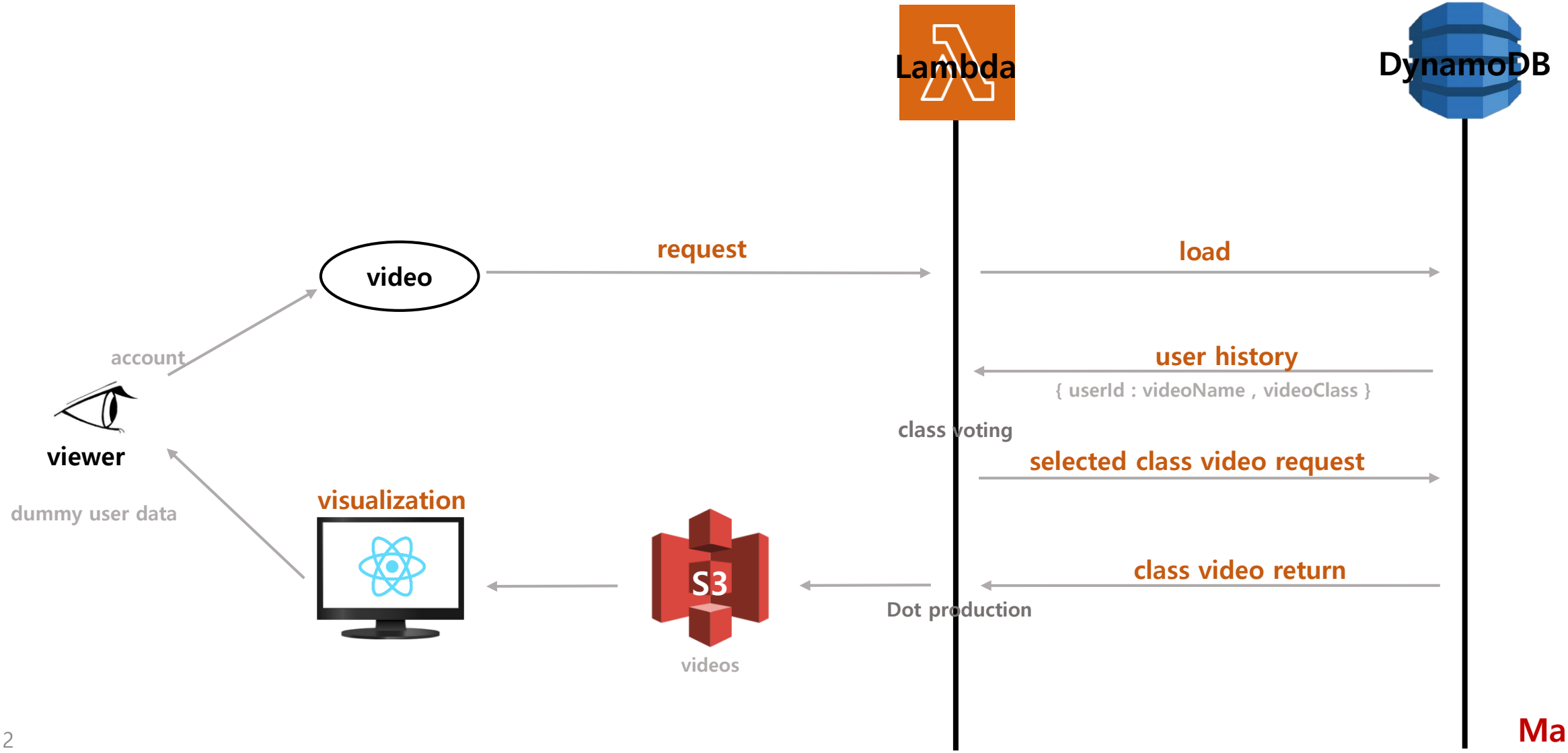


Team MaruBro

2. Consumer-side System

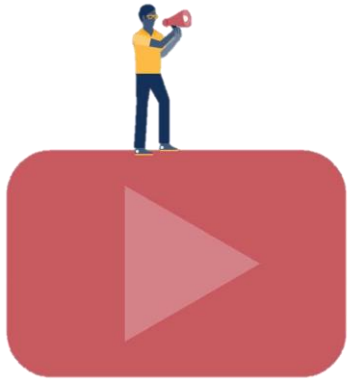


Consumer-side | System

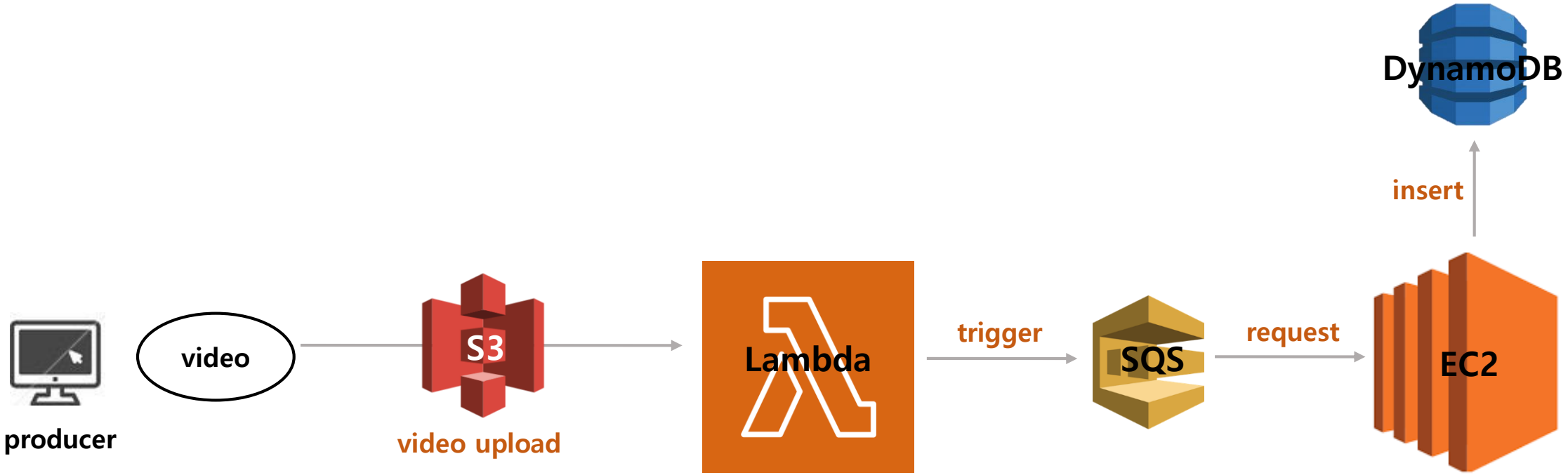


Team MaruBro

3. Producer-side System

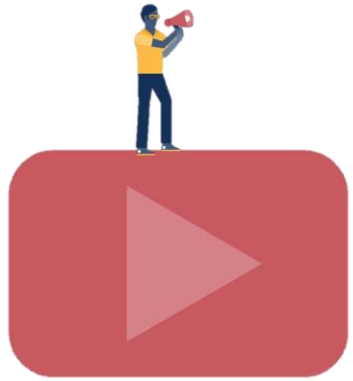


Producer-side | System



Team
MaruBro

Prototype



Service Prototype

Consumer web service

추천된 영상 리스트

AWS 추천 시스템

Endpoint

/recommend?userid=[ ]

다운로드 기능

Team MaruBro

Service Prototype | Producer web service

예시) Python *appendix

```
[36] import os
     import requests

[37] api_url1 = 'https://ynx9aa5u3j.execute-api.ap-northeast-2.amazonaws.com/presigned_for_upload'
     api_url2 = '/upload/'
     bucket = 'youtubepj-v3'
     filename = 'Tteokbokki.webm'

     url = api_url1 + api_url2 + bucket + '?filename=' + filename
     url

     'https://ynx9aa5u3j.execute-api.ap-northeast-2.amazonaws.com/presigned_for_upload/upload/youtubepj-v3?filename=Tteokbokki.webm'

[38] # 파이썬이 아닌 Browser에서의 접근으로 정보 전달
     request_headers = {
                        'User-Agent' : ('Mozilla/5.0 (Windows NT 10.0;Win64; x64)#
                        AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98#
                        Safari/537.36'),
                         }


     response_url = requests.get(url, headers = request_headers)

     presinged_url = response_url.text
     print(presinged_url)

     https://youtubepj-v3.s3.amazonaws.com/Tteokbokki.webm?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=ASIAZ067KBQH4RGI

[39] new_video = '/content/drive/MyDrive/PROJECT/sample_data/for_test/SUB)엽떡 신메뉴🔥 크림떡볶이 & 분모자 떡볶이 먹방 (ft

     with open(new_video, 'rb') as object_file:
       object_text = object_file.read()

[40] requests.put(presinged_url, data = object_text)

     <Response [200]>
```

Completed Status on Shell

```
docker.io/seungju0608/onnxmodel:latest
Success!
```

DB

Items returned (96)

pk	video	vector	video_info
class#mukb...	Tteokbokki....	sg0xPzup8j...	
class#educa...	#9	AAAAYLDN...	
class#kpop	#9	AAAAgIm8...	
class#educa...	#8	AAAA4Gqu...	

S3

mukbang/

Objects | Properties

Objects (21)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant

Copy S3 URI | Copy URL | Download | Open | Delete | Actions | Create folder | Upload

Find objects by prefix

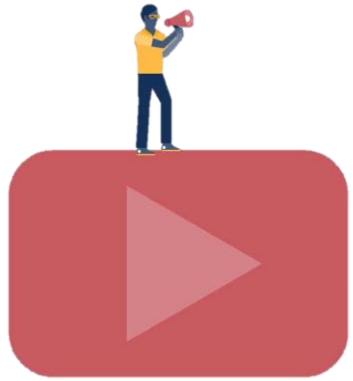
Name	Type	Last modified
mukbang-Tteokbokki.webm	webm	November 12, 2021, 17:51:16 (UTC+09:00)
mukbang-39.webm	webm	October 26, 2021, 17:31:37 (UTC+09:00)
mukbang-38.webm	webm	October 26, 2021, 17:31:36 (UTC+09:00)

Team MaruBro

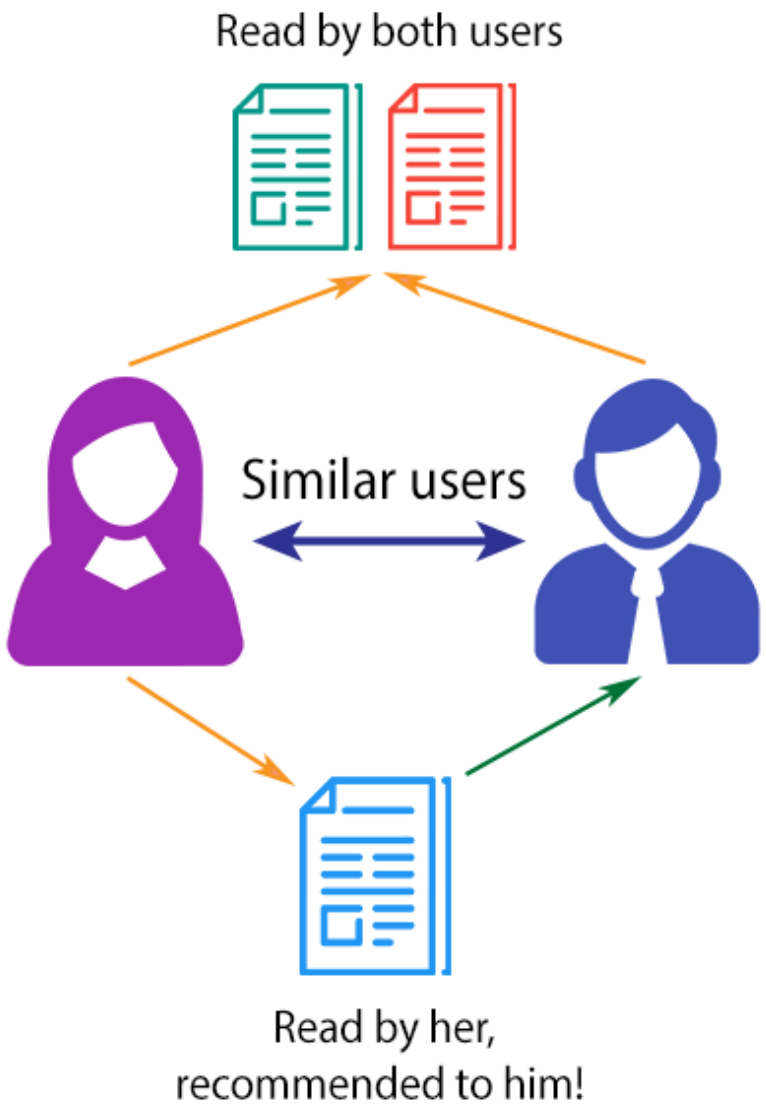
감사합니다

Q & A

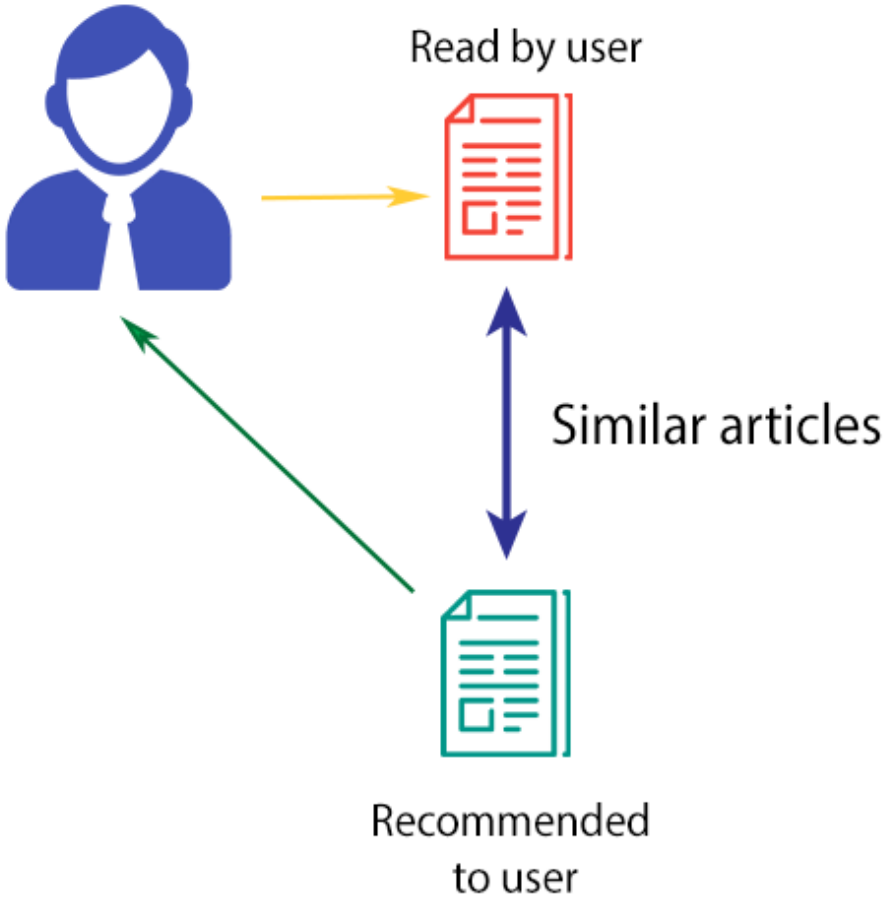
Appendix



COLLABORATIVE FILTERING

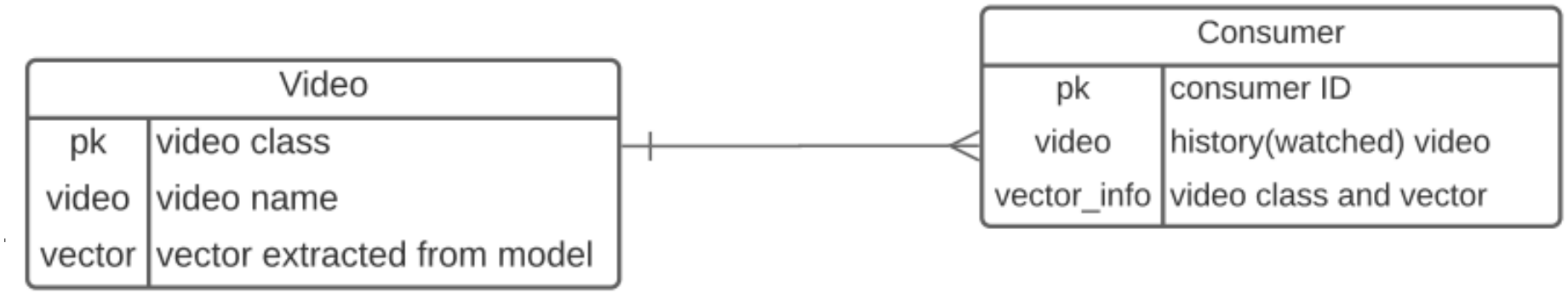


CONTENT-BASED FILTERING



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$
	1×1	~~average pool, 1000-d fc, softmax~~ Feature extraction				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

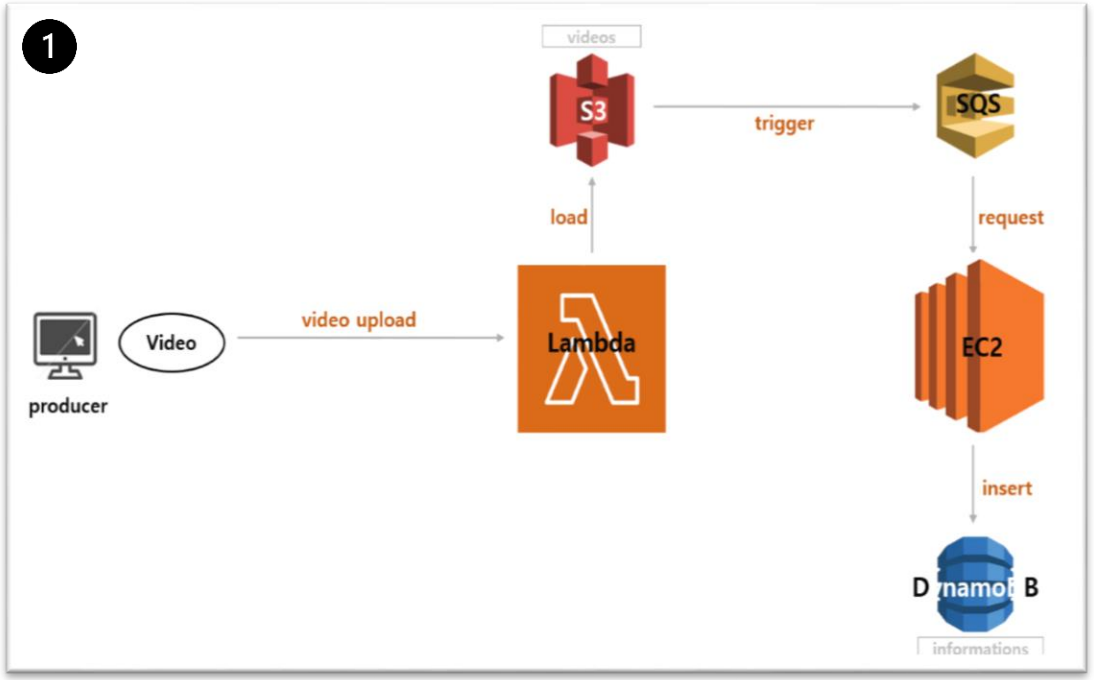
ures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of block

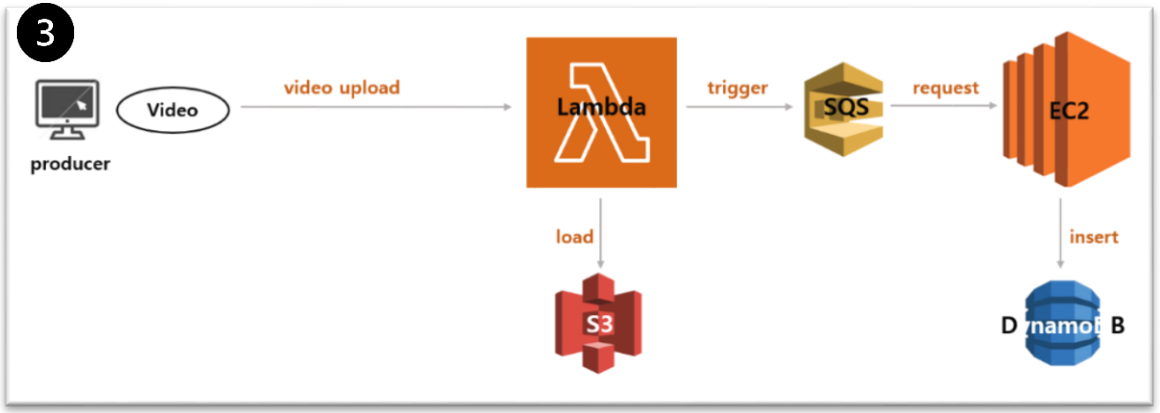


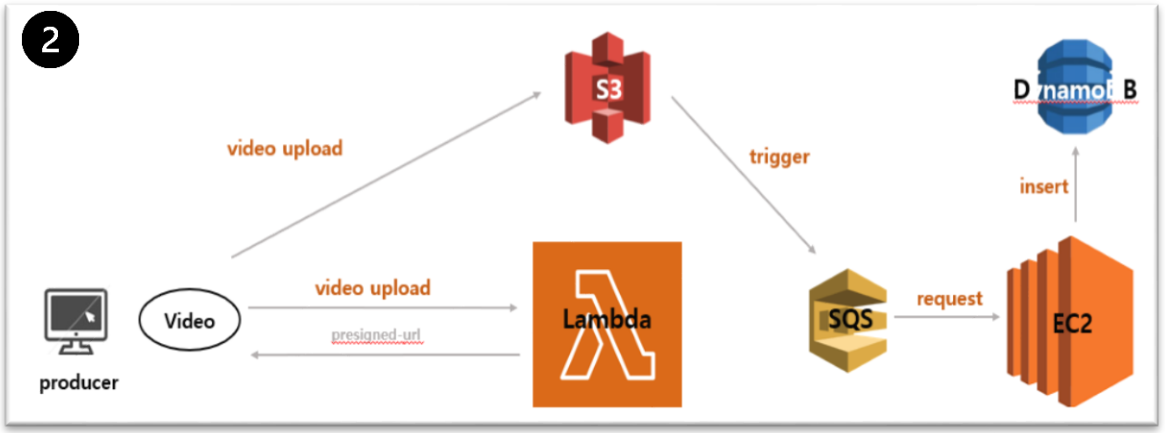
Team MaruBro

3번 시안의 경우 중심이 되는 Lambda를 잘 보여 주지만 비디오 용량을 수용할 수 있는 한계가 있다. 즉, 부하분산 구조를 구현하고자 하였다.

1과 2의 시안은 Event Driven으로 중심이 되는 Lambda의 역할이 부각되지 않는다.







사용 메뉴얼

- Authorized URL for Video Upload

내용	비디오를 업로드할 수 있는 권한이 주어진 url을 조회할 수 있습니다.
형식	https://ynx9aa5u3j.execute-api.ap-northeast-2.amazonaws.com/presigned_for_upload/upload/youtubepj-v3

Parameters

항목명 (영문)	항목명 (국문)	입력형태	항목설명	부가설명
Filename	파일이름	String	동영상 파일의 이름과 확장자명	사용가능한 확장자 : webm

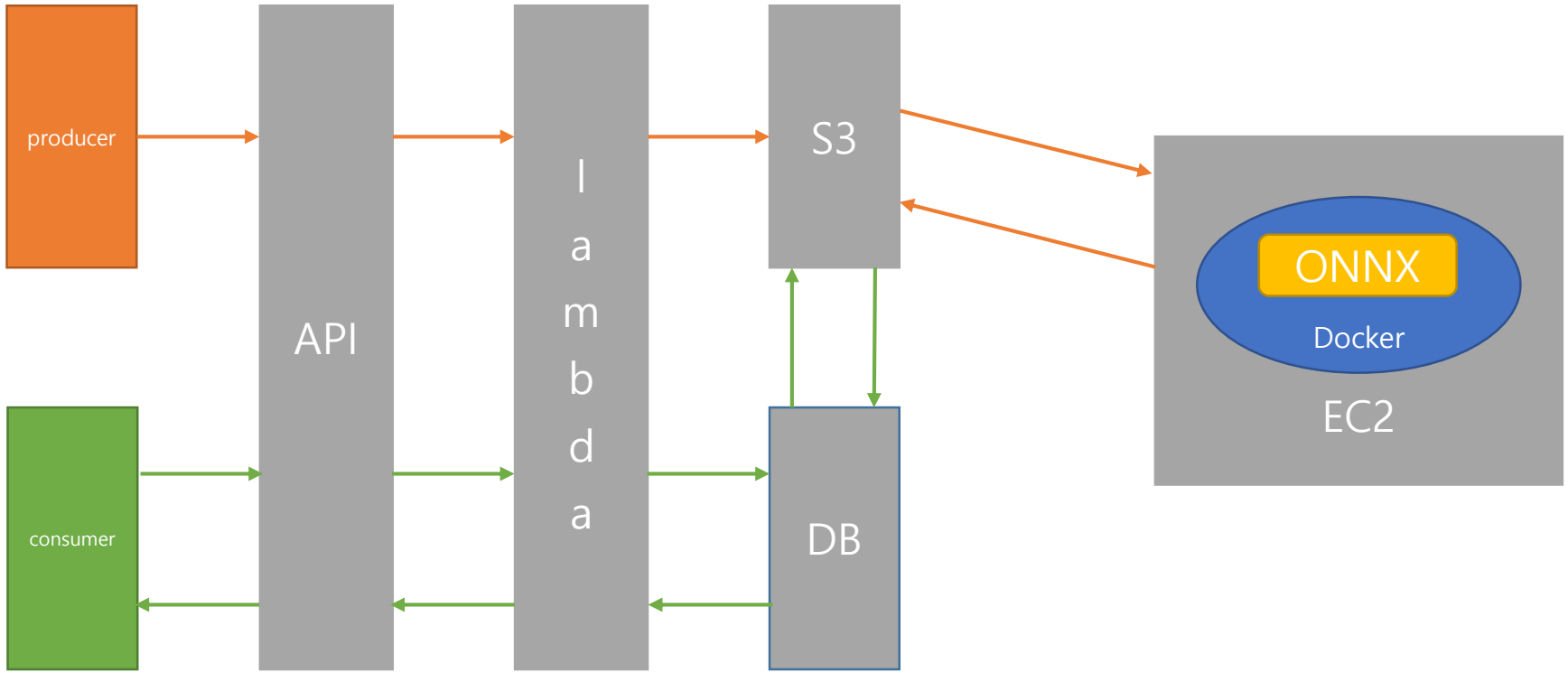
Response

PresignedURL : 저장소에 PUT 권한이 부여된 주소

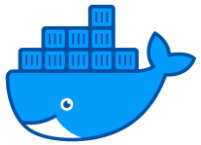
사용 예시

https://ynx9aa5u3j.execute-api.ap-northeast-2.amazonaws.com/presigned_for_upload/upload/youtubepj-v3?filename=Tteokbokki.webm

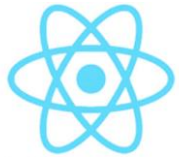
Team MaruBro


producer
consumer
API
lambda
S3
DB
ONNX
Docker
EC2


Git


Docker


Git action




Boto3


API Gateway


Lambda


DynamoDB


S3


EC2