AtyponTube

Done By

Othman Maher

Instructors

Dr.Motasem Aldiab

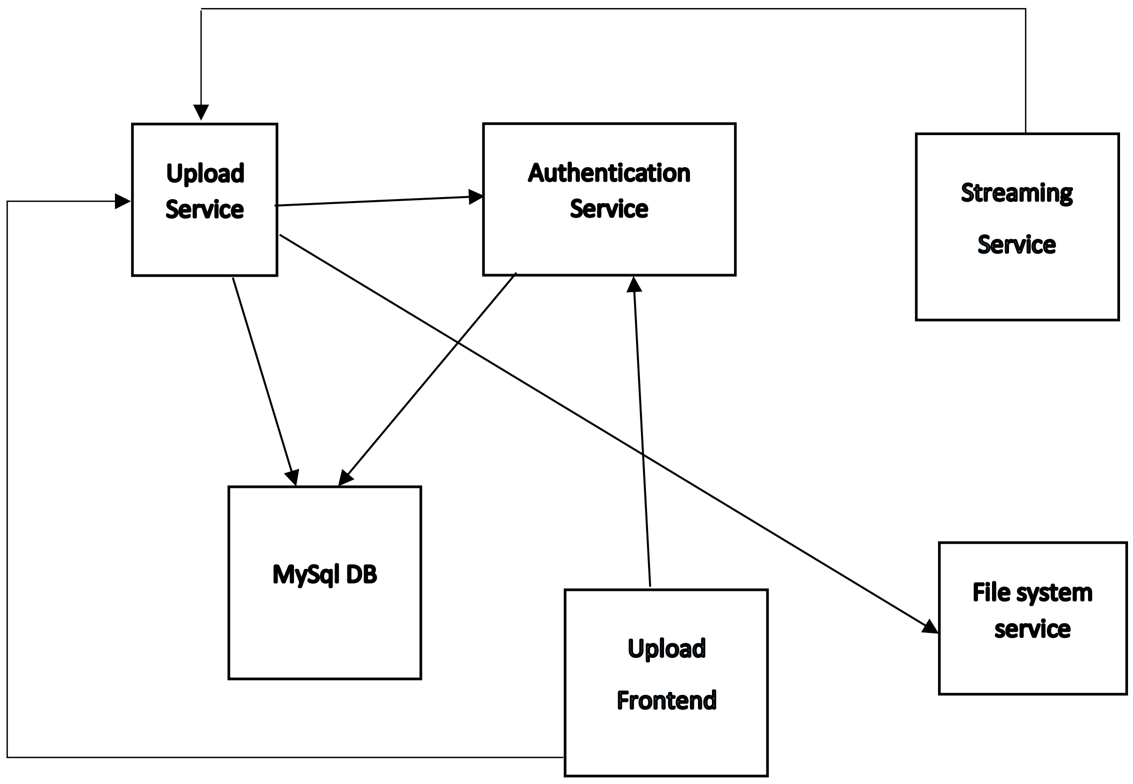
Dr.Fahed Jubair

Containerization Project

Contents

Introduction

-In this assignment, we've got a choice to choose between build a containerized microservices about Analytics System or Video Streaming System, and I chose a Video Streaming System but I made my own changes on the system, here is my own implementation.



-Streaming Service **and** Upload Frontend are images built using Vue.js (front-end framework).

Streaming Service: to stream and show the uploaded videos.

Upload Frontend: to upload the videos based on the users in the front-end.

-Authentication Service, File System Service **and** Upload Service are images built using Spring Boot (back-end framework).

Authentication Service: to create users and generate JWT after successful logging after talking to the database.

File System Service: to upload files to the storage and return the links, I used S3 Buckets.

Upload Service: to save the video information in the database.

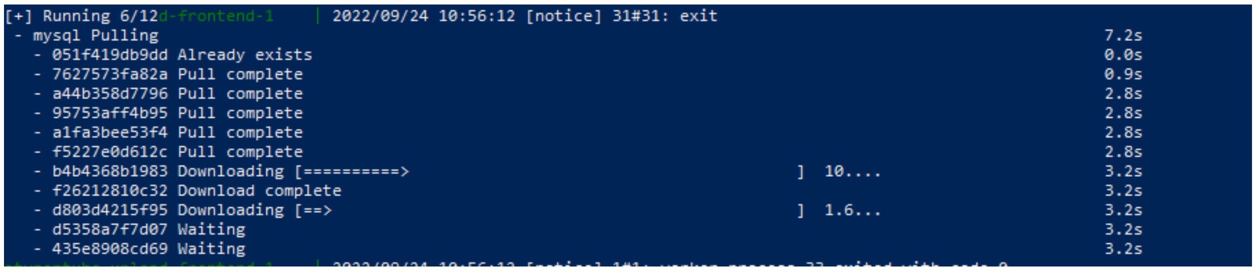
-MySql DB: database image that has the users and videos information.

Run the Project

-First of all, you need to download Docker Desktop from their website (https://docs.docker.com/get-docker/), then run it in your device.

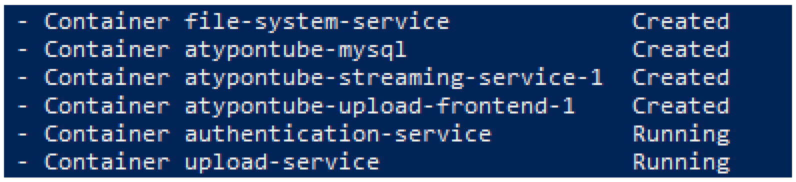
-After running Docker open the Microservice Project folder in vs code or open the terminal and change directory to the project folder and run (**docker-compose up**).

-It will first create the images if they're not exist



```
[+] Running 6/12d-frontend-1     | 2022/09/24 10:56:12 [notice] 31#31: exit
 - mysql Pulling                                                                       7.2s
   - 051f419db9dd Already exists                                                       0.0s
   - 7627573fa82a Pull complete                                                        0.9s
   - a44b358d7796 Pull complete                                                        2.8s
   - 95753aff4b95 Pull complete                                                        2.8s
   - a1fa3bee53f4 Pull complete                                                        2.8s
   - f5227e0d612c Pull complete                                                        2.8s
   - b4b4368b1983 Downloading [==========>                              ]  10....      3.2s
   - f26212810c32 Download complete                                                    3.2s
   - d803d4215f95 Downloading [==>                                      ]  1.6...      3.2s
   - d5358a7f7d07 Waiting                                                              3.2s
   - 435e8908cd69 Waiting                                                              3.2s
```

Then it will create containers from the images then run it

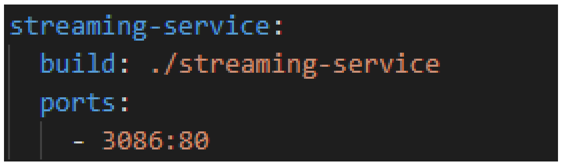


```
 - Container file-system-service              Created
 - Container atypontube-mysql                 Created
 - Container atypontube-streaming-service-1   Created
 - Container atypontube-upload-frontend-1     Created
 - Container authentication-service           Running
 - Container upload-service                   Running
```

-After the run finishes, go to one of your browsers and type this link to see the Streaming image (http://localhost:3086/) to see users videos or the Upload Frontend image (http://localhost:3087/) to create an account if you don't have one and logged in the system to upload your videos.

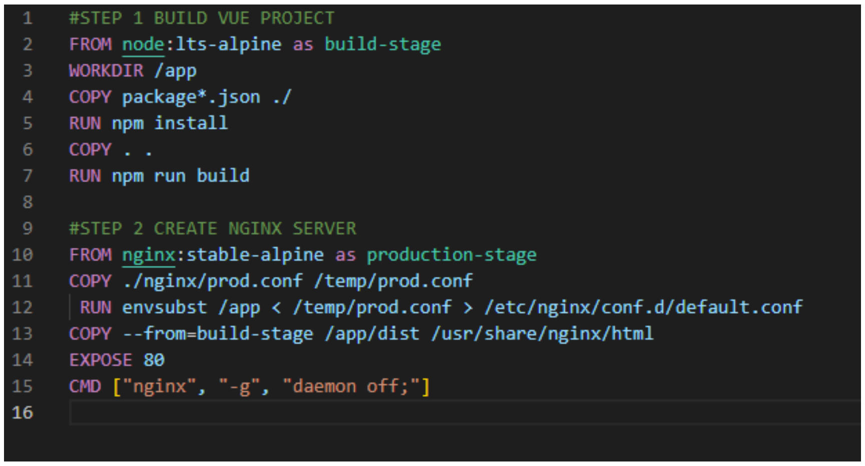
Streaming Service

- The Implementation for this image in the Docker Compose file



```
streaming-service:
  build: ./streaming-service
  ports:
    - 3086:80
```

- The Docker file



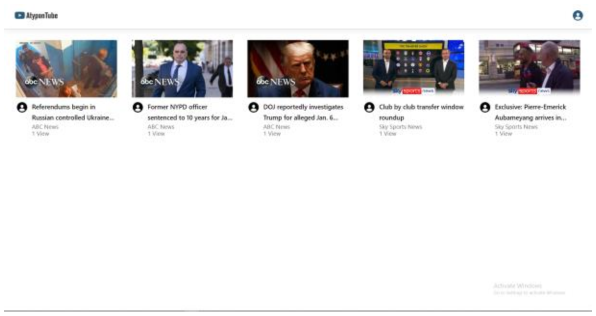
```
#STEP 1 BUILD VUE PROJECT
FROM node:lts-alpine as build-stage
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build

#STEP 2 CREATE NGINX SERVER
FROM nginx:stable-alpine as production-stage
COPY ./nginx/prod.conf /temp/prod.conf
 RUN envsubst /app < /temp/prod.conf > /etc/nginx/conf.d/default.conf
COPY --from=build-stage /app/dist /usr/share/nginx/html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

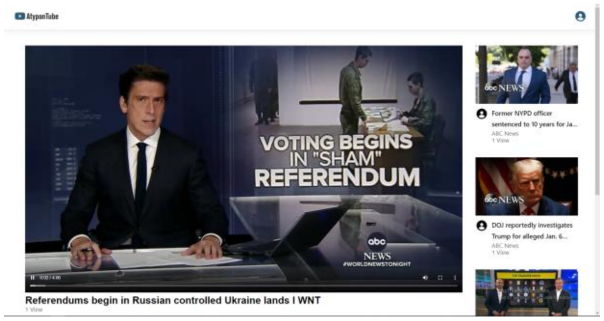
Description

This image is built using Vue.js (front-end framework) and it has two pages.

-Home Page that has all the videos in the System that got it from the Upload Service.

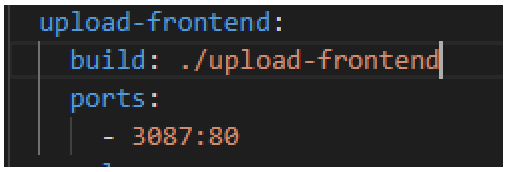

AtyponTube
Referendums begin in Russian controlled Ukraine...
ABC News
1 View
Former NYPD officer sentenced to 10 years for Ja...
ABC News
1 View
DOJ reportedly investigates Trump for alleged Jan. 6...
ABC News
1 View
Club by club transfer window roundup
Sky Sports News
1 View
Exclusive: Pierre-Emerick Aubameyang arrives in...
Sky Sports News
1 View

-Video Page that stream the video and get the video details from Upload Service.


AtyponTube
VOTING BEGINS IN "SHAM" REFERENDUM
Referendums begin in Russian controlled Ukraine lands I WNT
1 View
Former NYPD officer sentenced to 10 years for Ja...
ABC News
1 View
DOJ reportedly investigates Trump for alleged Jan. 6...
ABC News
1 View

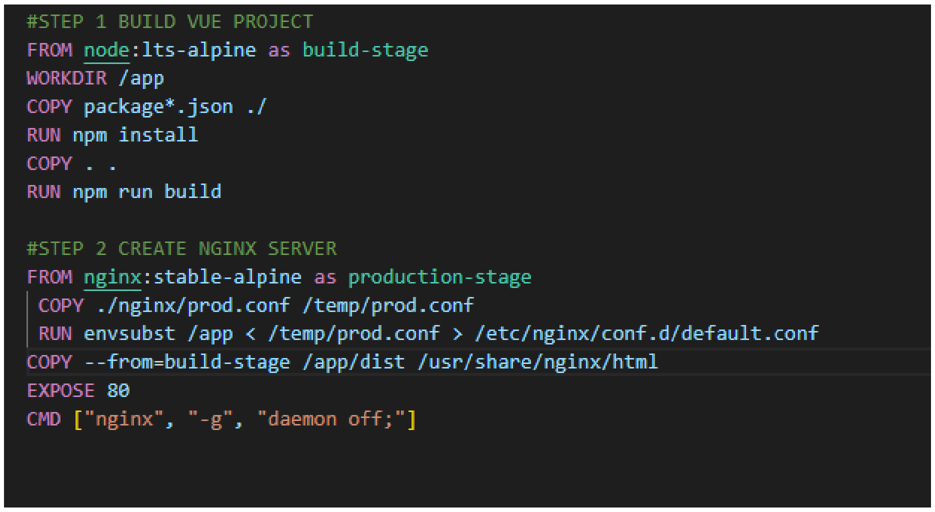
Upload Frontend

- The Implementation for this image in the Docker Compose file



```
upload-frontend:
  build: ./upload-frontend
  ports:
    - 3087:80
```

- The Docker file



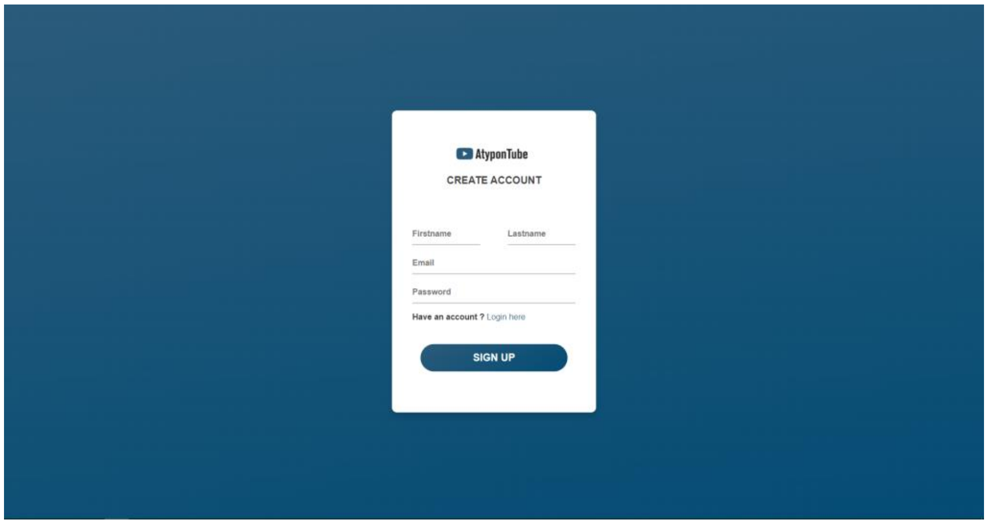
```
#STEP 1 BUILD VUE PROJECT
FROM node:lts-alpine as build-stage
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build

#STEP 2 CREATE NGINX SERVER
FROM nginx:stable-alpine as production-stage
 COPY ./nginx/prod.conf /temp/prod.conf
 RUN envsubst /app < /temp/prod.conf > /etc/nginx/conf.d/default.conf
COPY --from=build-stage /app/dist /usr/share/nginx/html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

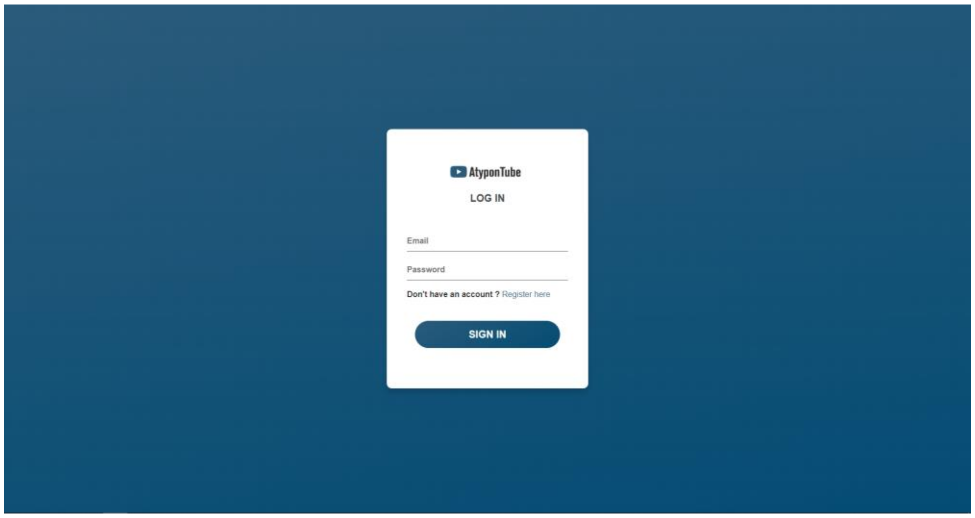
Description

This image is built using Vue.js (front-end framework) and it has three pages.

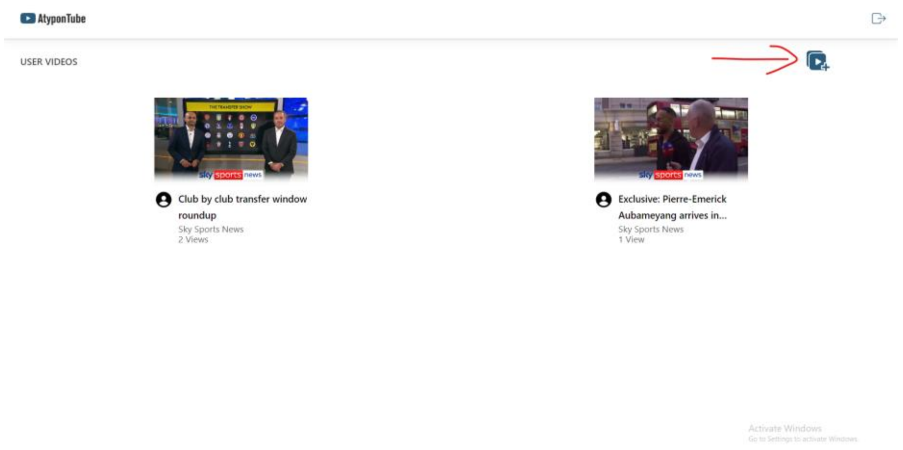
-Register Page that contact with the Authentication Service to create users.

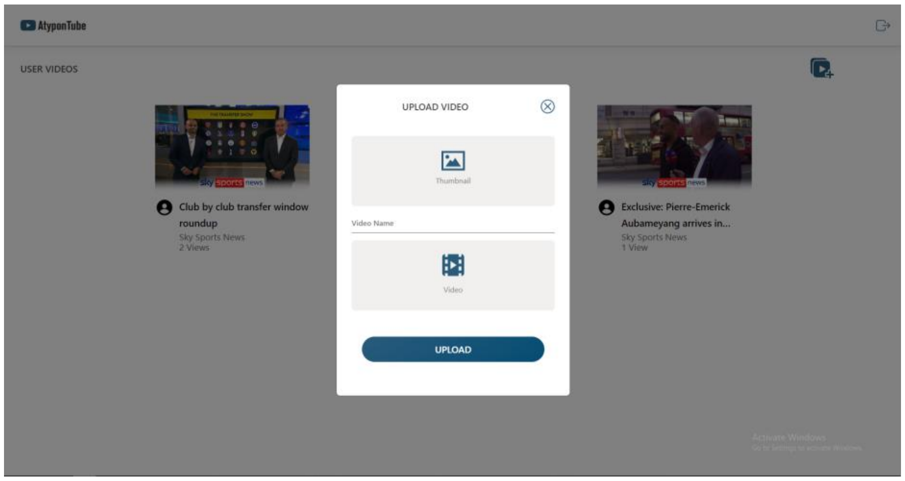


-Login Page that contact with the Authentication Service to generate JWT and logged the users.



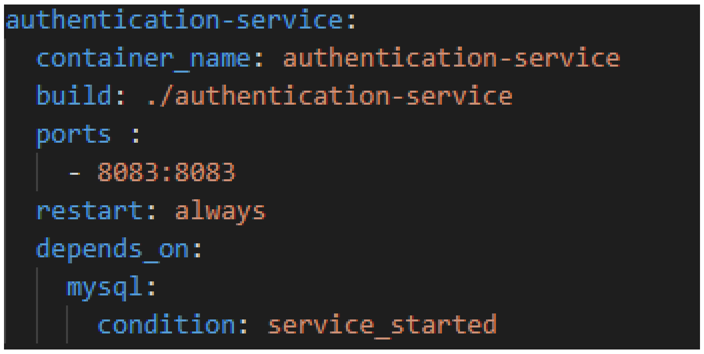
-Home Page that has all the user videos that came from Upload Service and a button triggers a window that takes video information to upload it.


AtyponTube
USER VIDEOS
Club by club transfer window roundup
Sky Sports News
2 Views
Exclusive: Pierre-Emerick Aubameyang arrives in...
Sky Sports News
1 View



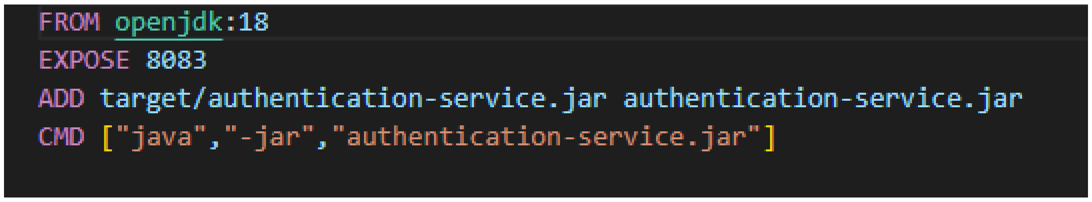
Authentication Service

- The Implementation for this image in the Docker Compose file



```
authentication-service:
  container_name: authentication-service
  build: ./authentication-service
  ports :
    - 8083:8083
  restart: always
  depends_on:
    mysql:
      condition: service_started
```

- The Docker file



```
FROM openjdk:18
EXPOSE 8083
ADD target/authentication-service.jar authentication-service.jar
CMD ["java","-jar","authentication-service.jar"]
```

Description

This image is built using Spring Boot (back-end framework), it communicate with the database image to save the users and generate JWT to authenticate the users and it has three endpoints.

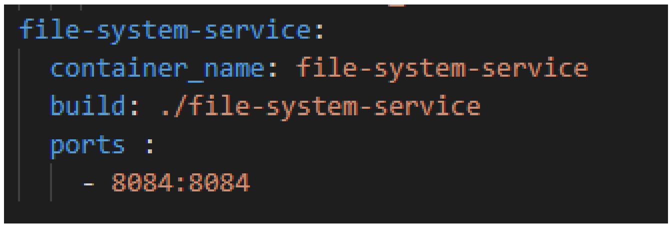
-Login endpoint that authenticate the user and generate the token

-Register endpoint that create the user and save it in the database

-Verify endpoint that check if the token is not expired and valid

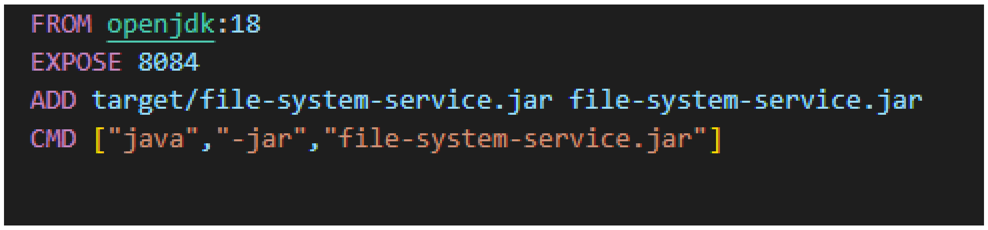
File System Service

- The Implementation for this image in the Docker Compose file



```
file-system-service:
  container_name: file-system-service
  build: ./file-system-service
  ports :
    - 8084:8084
```

- The Docker file



```
FROM openjdk:18
EXPOSE 8084
ADD target/file-system-service.jar file-system-service.jar
CMD ["java","-jar","file-system-service.jar"]
```

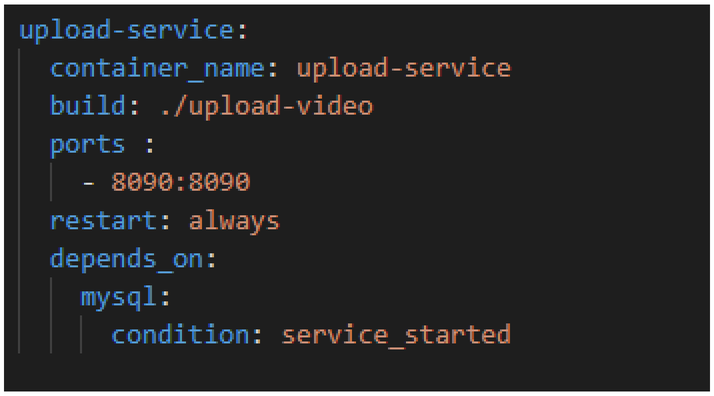
Description

This image is built using Spring Boot (back-end framework) to upload files, I used S3 Bucket service from AWS and it has one endpoint.

-Upload endpoint that upload files to S3 and return link for the uploaded file.

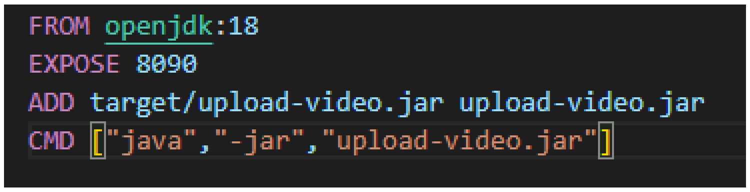
Upload Service

- The Implementation for this image in the Docker Compose file



```
upload-service:
  container_name: upload-service
  build: ./upload-video
  ports :
    - 8090:8090
  restart: always
  depends_on:
    mysql:
      condition: service_started
```

- The Docker file



```
FROM openjdk:18
EXPOSE 8090
ADD target/upload-video.jar upload-video.jar
CMD ["java","-jar","upload-video.jar"]
```

Description

This image is built using Spring Boot (back-end framework) to upload videos by save the information of the videos in the database, it has five endpoints

-Upload end point that take video information then communicate with the File System. Service to upload the video and thumbnail then communicate with the database to save the information.

-Get Video endpoint that communicate with the database to take video information by It's Id.

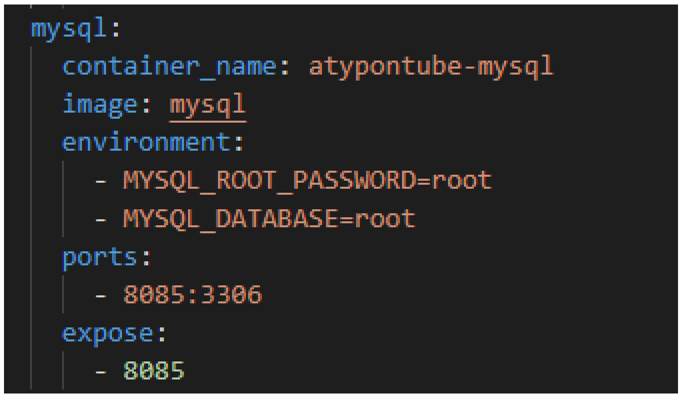
-Get All Videos endpoint that get all the videos details from the database

-View Video endpoint that increase the video views after giving the endpoint video id.

-Get User Videos endpoint that get the user videos from the database by user id.

MySql Database

- The Implementation for this image in the Docker Compose file



```
mysql:
  container_name: atypontube-mysql
  image: mysql
  environment:
    - MYSQL_ROOT_PASSWORD=root
    - MYSQL_DATABASE=root
  ports:
    - 8085:3306
  expose:
    - 8085
```

Description

This image is built using MySql (Structured Query Language Database) that saves the users and videos information and it has two tables.

-User Table that contains (userId,email,password,name).

-Video Table that contains (videoId,videoLink,thumbnailLink,name,views,author,userId).