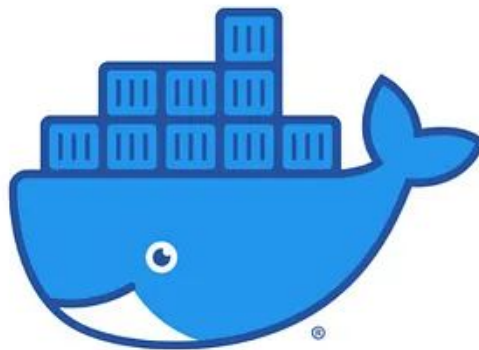


# TMDB Poster Download Project

TEAM 7



Dockerized Flask + MongoDB Application



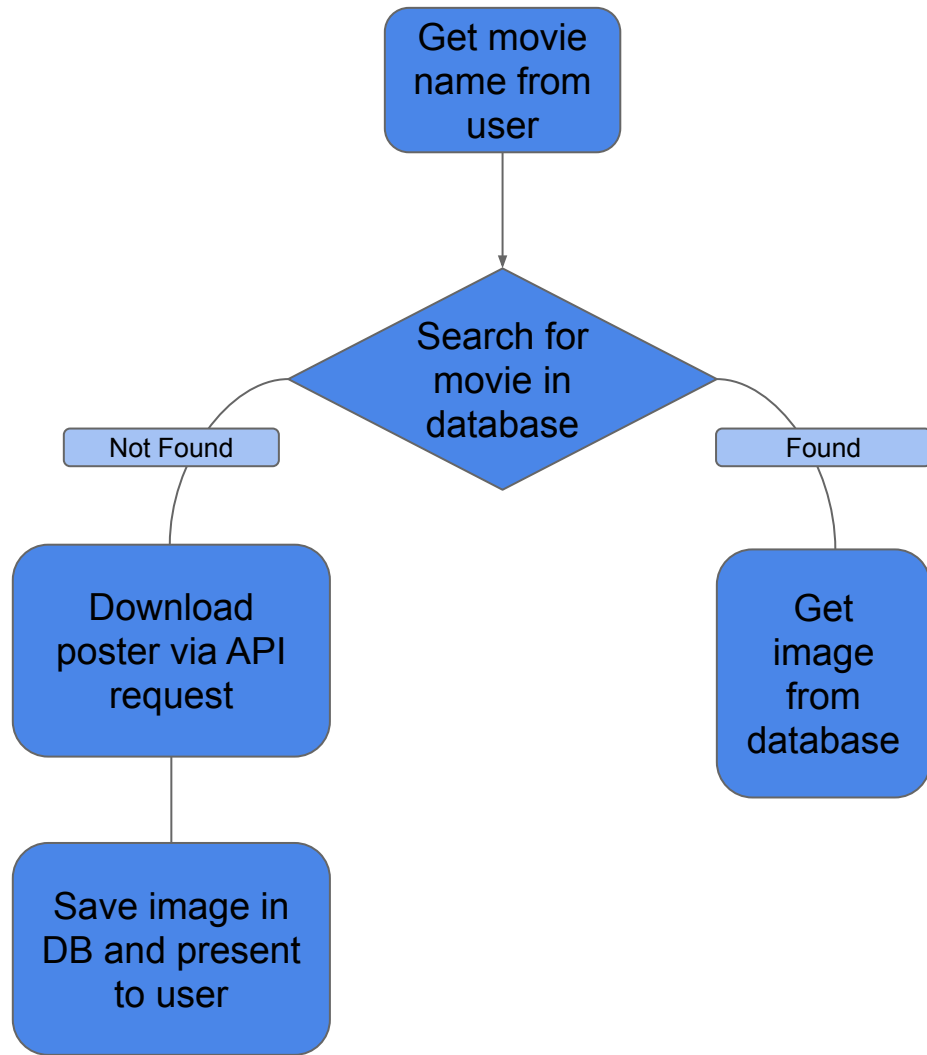
# Team Members

?

# Project Objectives

- Get Movie Name
- Search Movie poster in TMDB Database
- Download poster and poster meta data and save it in MongoDB
- Construct a full Database API
- Write a simple web app to present the posters
- Dockerize the application
- Deploy application on AWS EC2 instance

# Algorithm Design



# Teamwork and responsibilities

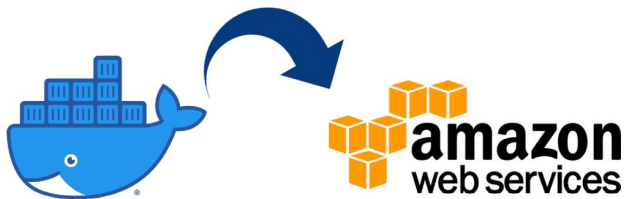


# Software Design



- OOP approach - Allows scalable code and services autonomy.
- **TMDB\_Downloader.py** module - holds **TMDBDownloader** class and handles all the logic of downloading posters
- **MongoDBAPI.py** module - holds **MongoAPI** class and handles all the API management logic including all CRUD operations: reading records, creating records, updating records and deleting records.
- **mongo\_tmdb\_logic.py** module - holds the "Business logic", integration between the two classes - if movie poster is not found in DB then download it from TMDB website and insert to DB.
- **api.py** module - hold web application logic using Flask, currently handles only presenting posters feature.

# Dockerize the App and deploy on AWS



- Write Dockerfile
- Write docker-compose file
- Write user\_data script to deploy the App on an EC2 instance

```
#!/bin/bash
sudo yum update -y
sudo amazon-linux-extras install docker -y
sudo systemctl start docker
sudo usermod -a -G docker ec2-user
sudo systemctl enable docker
sudo curl -SL https://github.com/docker/compose/releases/download/v2.4.1/docker-compose-linux-x86_64 -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
sudo yum install git -y
sudo git clone https://github.com/AmirKatorza/AWS_reStart_Docker-Flask-Mongo_App.git
# wget https://github.com/AmirKatorza/AWS_reStart_Docker-Flask-Mongo_App/archive/refs/heads/master.zip
# sudo yum install unzip -y
# sudo unzip master.zip
sudo cd ./AWS_reStart_Docker-Flask-Mongo_App/
sudo docker-compose build
sudo docker-compose up
```

Reference:

<https://gist.github.com/npearce/6f3c7826c7499587f00957fee62f8ee9>

# GET requests screenshots – Postman



POSTMAN

The screenshot displays the Postman application interface. The top navigation bar includes 'Home', 'Workspaces', 'API Network', and 'Explore'. The main workspace is titled 'My Workspace' and shows a 'New' button and an 'Import' button. A sidebar on the left contains links to 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', 'Flows', and 'History'. The central area shows a 'Your collection' dialog with a 'Type' dropdown set to 'API key'. Below this, a message says 'Create a collection for your requests' with a 'Create Collection' button and a 'Use a Template' link. The right panel shows a GET request to 'http://127.0.0.1:5001/search'. The 'Body' tab is selected, showing a table with 'movie\_name' set to 'superman'. The 'Send' button is visible. Below the request, the 'Body' tab shows the response in 'Pretty' format, displaying a JSON object: 

```
{  "_id": {    "id": "643a7f25e877e3085794baba"  },  "Status": "Found in DB",  "file_name": "superman.jpeg"}
```

. The status bar at the bottom indicates '200 OK', '19 ms', and '265 B'.

Home Workspaces API Network Explore Search Postman Invite Upgrade

My Workspace New Import Overview GET http://127.0.0.1:5001/search No Environment

Collections + Your collection + Your collection Authorization Type API key

Create a collection for your requests

A collection lets you group related requests and easily set common authorization, tests, scripts, and variables for all requests in it.

Create Collection or Use a Template

GET http://127.0.0.1:5001/search Save Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

Key	Value	Description
<input checked="" type="checkbox"/> movie_name	superman	
Key	Value	Description

Body Cookies Headers (5) Test Results 200 OK 19 ms 265 B Save as Example

Pretty Raw Preview Visualize JSON

```
1 {
2   "_id": {
3     "id": "643a7f25e877e3085794baba"
4   },
5   "Status": "Found in DB",
6   "file_name": "superman.jpeg"
7 }
```

Start working with APIs 67% Next: Save requests. Show me

Online Find and Replace Console Cookies Capture requests Runner Trash



# GET requests screenshots – Postman



POSTMAN

The screenshot displays the Postman application interface. The top navigation bar includes 'Home', 'Workspaces', 'API Network', and 'Explore'. The main workspace is titled 'My Workspace' and shows a collection named 'My first collection' with two folders: 'First folder inside collection' and 'Second folder inside collection'. A sidebar on the left lists various features like Collections, APIs, Environments, Mock Servers, Monitors, Flows, and History.

The central panel shows a GET request to 'http://127.0.0.1:5001/search'. The request is configured with the following parameters:

Key	Value	Description
<input checked="" type="checkbox"/> movie_name	black panther	

The response is displayed in the bottom panel, showing a 200 OK status with a response time of 2.15s and a body size of 270 B. The response body is formatted as JSON:

```
1 {
2   "_id": {
3     "$oid": "643a903e02bdade075f4d93e"
4   },
5   "Status": "Added to DB",
6   "file_name": "black panther.jpeg"
7 }
```

At the bottom of the interface, there is a progress bar for 'Start working with APIs' at 67% and a 'Show me' link.

# What can be improved?

- Improved UI (User Interface) Design - currently a simple HTML form with only search feature.
- Full API coverage by the web application including updating records and deleting records.
- Scale UP - currently downloading only the first movie from search results and only the first poster image.

We want to save all movies from search results and all the posters related.

# Challenges

- Learning new technologies in a very short amount of time.
- Learn how to integrate and orchestrate all the technologies to work together.
- Not enough ManPower to handle a project in that scale.

Thank You  
For Listening!

A dark blue diagonal gradient bar that starts from the bottom left and extends towards the top right, covering the lower half of the slide.