Project Structure:

```
root
    static
    templates
        index.html
        movie.html
    app.py
    data.json
    db.sqlite
external libraries
    DASH
        dash_script.js
    HLS
        hls_script.js
```

This project is hosted using Flask and as the scale is small, both the back-end and the front-end are served via a single application. Flask serves views in the **Templates** folder via the **render_template** in the **Flask** library.

```
from flask import Flask
from flask import render_template
```

The back-end is also API-based and served by the methods available in the **app.py** file.

How DASH script works:
In order to use DASH we first need to input the media file into the ffmpeg library and then generate chunks accordingly.

```
from ffmpeg_streaming import Formats

video = ffmpeg_streaming.input('the-godfather.mp4')
dash = video.dash(Formats.h264())
dash.auto_generate_representations()
dash.output('./dash.mpd')
```

The above command generates the chunks as below:

```
-rw-rw-r-- 1 thatssoheil thatssoheil  80K Jun 30 12:43  chunk-stream0-00007.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  80K Jun 30 12:44  chunk-stream0-00008.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  80K Jun 30 12:44  chunk-stream0-00009.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  80K Jun 30 12:44  chunk-stream0-00010.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  80K Jun 30 12:44  chunk-stream0-00011.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  80K Jun 30 12:44  chunk-stream0-00012.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  80K Jun 30 12:44  chunk-stream0-00013.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  79K Jun 30 12:44  chunk-stream0-00014.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  20K Jun 30 12:44  chunk-stream0-00015.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  316 Jun 30 12:44  chunk-stream0-00016.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 393K Jun 30 12:43  chunk-stream1-00007.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 822K Jun 30 12:44  chunk-stream1-00008.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 664K Jun 30 12:44  chunk-stream1-00009.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 395K Jun 30 12:44  chunk-stream1-00010.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 459K Jun 30 12:44  chunk-stream1-00011.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 132K Jun 30 12:44  chunk-stream1-00012.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 153K Jun 30 12:44  chunk-stream1-00013.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  36K Jun 30 12:44  chunk-stream1-00014.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  67K Jun 30 12:44  chunk-stream1-00015.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  733 Jun 30 12:44  chunk-stream1-00016.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  78K Jun 30 12:43  chunk-stream2-00007.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  78K Jun 30 12:44  chunk-stream2-00008.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  77K Jun 30 12:44  chunk-stream2-00009.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  76K Jun 30 12:44  chunk-stream2-00010.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  78K Jun 30 12:44  chunk-stream2-00011.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  78K Jun 30 12:44  chunk-stream2-00012.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  79K Jun 30 12:44  chunk-stream2-00013.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  76K Jun 30 12:44  chunk-stream2-00014.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  13K Jun 30 12:44  chunk-stream2-00015.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  250 Jun 30 12:44  chunk-stream2-00016.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 210K Jun 30 12:43  chunk-stream3-00007.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 270K Jun 30 12:44  chunk-stream3-00008.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 191K Jun 30 12:44  chunk-stream3-00009.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 142K Jun 30 12:44  chunk-stream3-00010.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil 162K Jun 30 12:44  chunk-stream3-00011.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  80K Jun 30 12:44  chunk-stream3-00012.m4s
-rw-rw-r-- 1 thatssoheil thatssoheil  51K Jun 30 12:44  chunk-stream3-00013.m4s
```

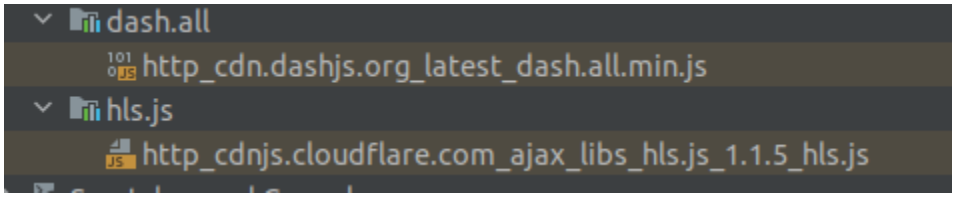
As well as the **.mpd** file:

```
-rw-rw-r-- 1 thatssoheil thatssoheil 3.0K Jun 30 12:44  out.mpd
```

Now in order to stream the above chunks we need to serve these static files on Nginx web server.

```
rtmp {
    server {
        listen 1935;
        application live {
            live on;
            dash on;
            dash_path /tmp/dash;
            dash_fragment 15s;
        }
    }
}

http {
    server {
        listen 80;
        location /movie/{:int} {
            root /tmp/dash;
        }
    }

    types {
        text/html html;
        application/dash+xml mpd;
    }
}
```

The above configuration serves the chunks on **localhost/movie/{:_id}**, and passing this value as a URL to HTML should be done via JS libraries.




```
<script src="http://cdn.dashjs.org/latest/dash.all.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/hls.js/1.1.5/hls.min.js"></script>
<script>
```

```
            (function () {
                let url = '{{ content[7] }}';
                const ext = url.split(/[#?]/)[0].split('.').pop().trim();

                if (ext === 'mpd') {
                    console.log('DASH')
                    let player = dashjs.MediaPlayer().create();
                         player.initialize(document.querySelector("#trailer"), url,
true);
                }

                if (ext === 'm3u8') {
                    console.log('HLS')
                    let video = document.getElementById('trailer');
                    if (Hls.isSupported()) {
                        let hls = new Hls();
                        hls.loadSource(url);
                        hls.attachMedia(video);
                     } else if (video.canPlayType('application/vnd.apple.mpegurl'))
{
                        video.src = url;
                    }
                }
            })();
        </script>
```

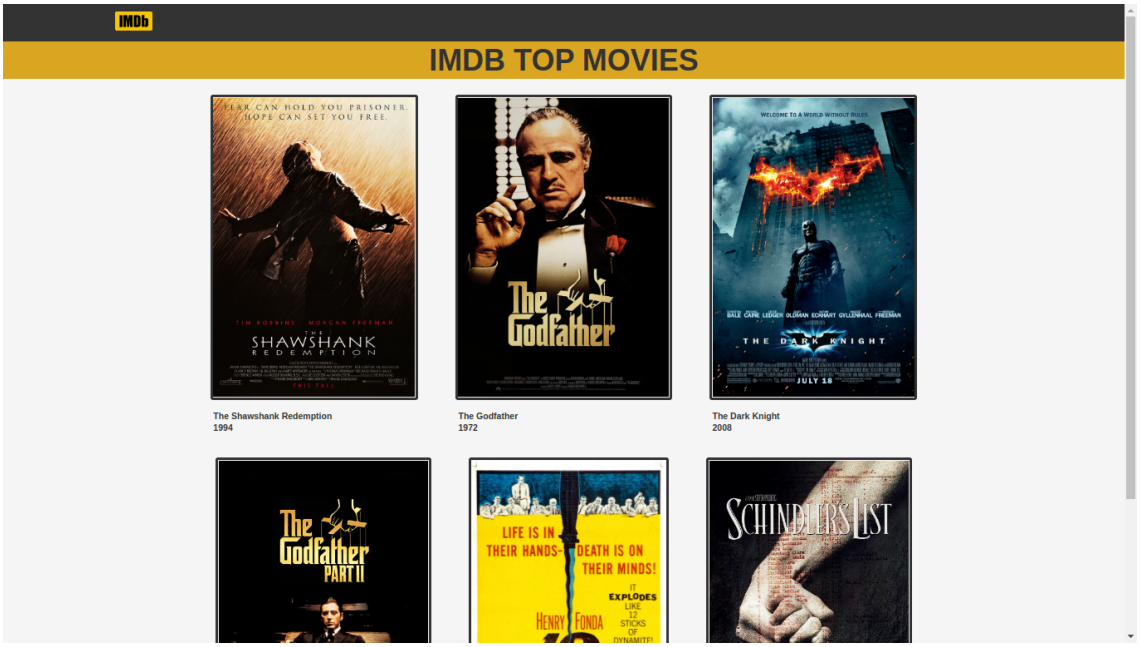
The url for each movie is set in the db and gets passed to the HTML file using flask content property, then the url is fetched for each movie and both the HLS and DASH scripts for browsers are also imported. We then use **MediaPlayer()** - for DASH - and **Hls** - for HLS - to load the relevant stream files accordingly.

Back-end end-points:

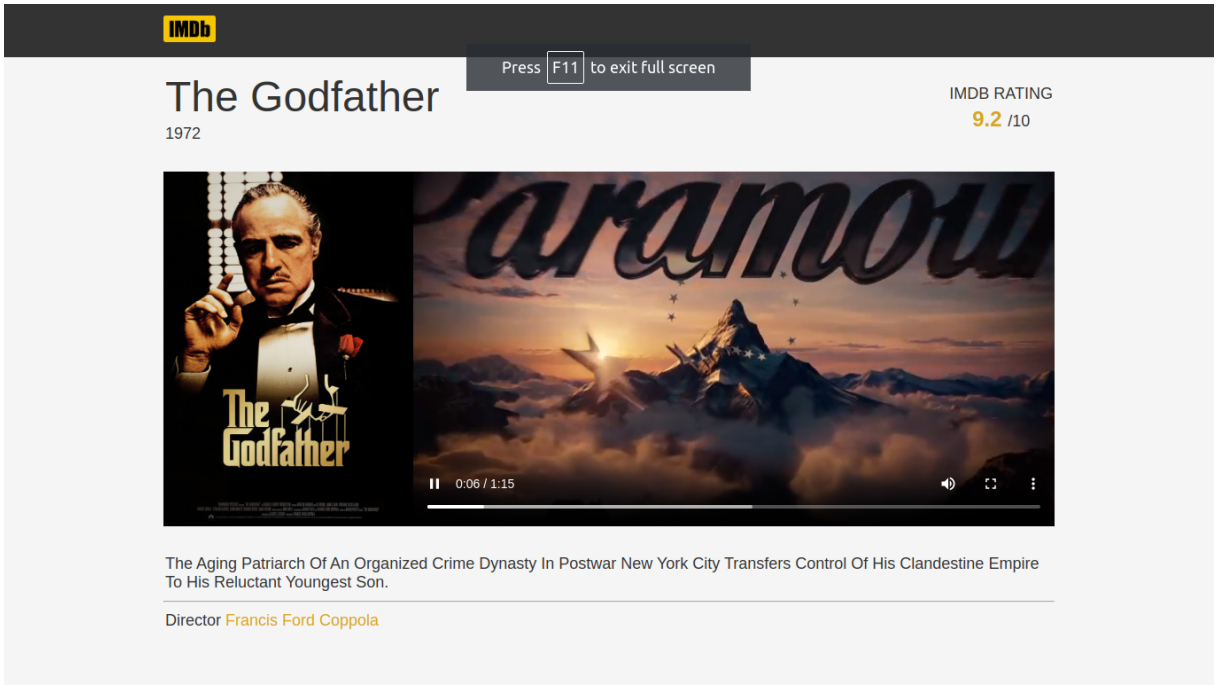
```
@app.route('/')
@app.route('/index')
def home():
    repo = fetch_all()
    return render_template('index.html', content=repo)

@app.route('/movies/<int:_id>')
def movie(_id):
    repo = fetch(_id)[0]
    return render_template('movie.html', content=repo)
```

The above endpoints are responsible for serving the requests and sending back the relevant HTML file accordingly. **Repo** is also fetched from **db.sqlite** by **fetch_all()** and **fetch(_id)** methods.

Main Page:

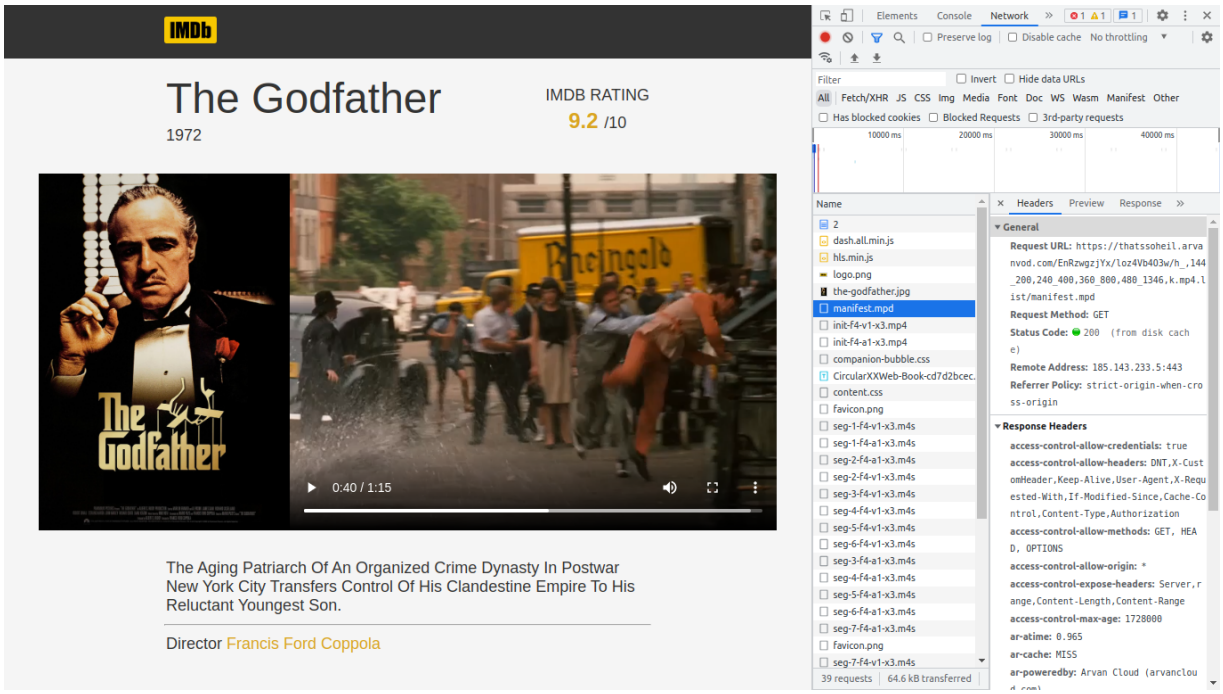


Movie Page:

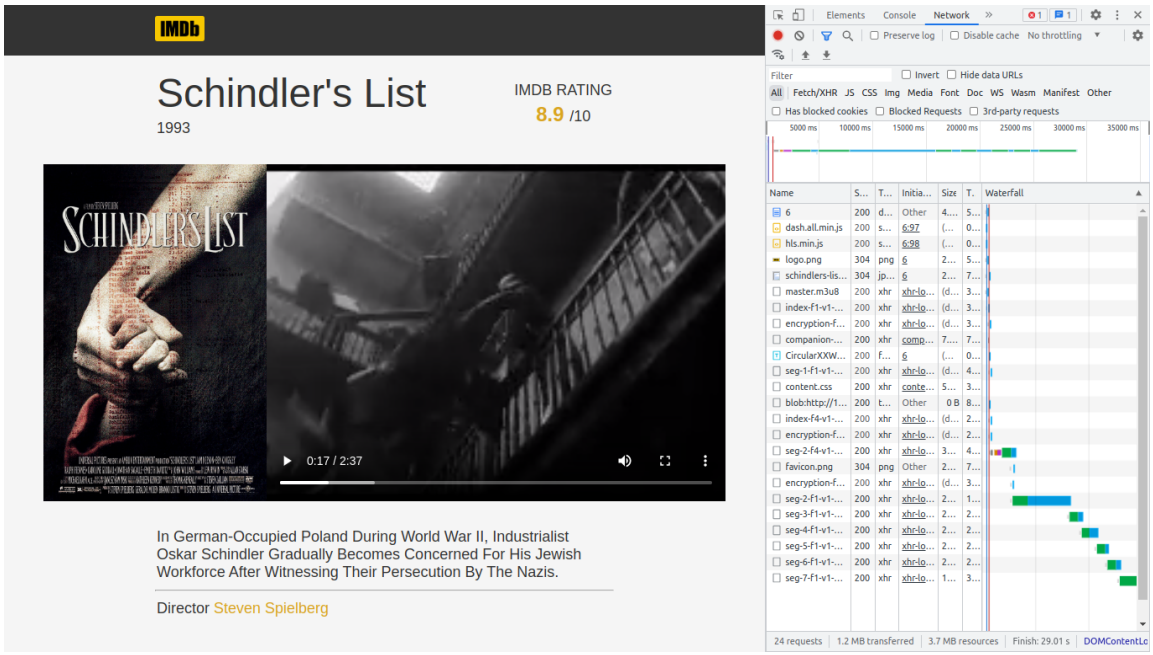


Analyzing the Stream:

DASH Streaming -



As seen above the browser initially loaded the **.mpd** as well as the js libraries for both the **DASH** and **HLS**, then proceeded to load the segments as the play button was clicked. The segments are the chunks loaded relative to the user's bandwidth.



The same thing applies to the HLS protocol as the **.m3u8** first loaded using the js library and then as the play button is clicked it starts getting media segments one by one. The waterfall diagram on the right also indicates the timing of each segment delivery.

Team Management:
We worked closely together in the same environment in order to accomplish what is presented in this report, thus each of the contributors in this project have worked on all the components mentioned in the previous sections.
Front-end mostly developed by **Soheil Fakour** as dividing such matter into micro-tasks was not an option in the limited time we had for this project.
The rest have been worked on in a pair-programming setup helping both the programmers debug each other's flaws and faults.

IN ORDER TO WATCH A DEMO OF THE PROJECT WE PREPARED A LOOM FOOTAGE PLACED IN THE ROOT OF THE PROJECT:

Soheil's Demo: soheil-fakour_demo.mp4
Mahdiyeh's Demo: mahdiyeh-sadat_demo.mp4

Contributors:
Soheil Fakour
Faculty of Computer Engineering, Amirkabir University of Technology
2016 Entry
soheil.fakour@gmail.com

Mahdieh Sadat Benis
Faculty of Computer Engineering, Amirkabir University of Technology
2019 Entry
Mahdiehsadat20@gmail.com