Fraunhofer FOKUS
Institut für Offene Kommunikationssysteme

AWT PJ
Video Streaming Mixer Library

Poonam Kumari Roy, Mohamed Mesto, Yuni Quintero | AWT PJ | Workshop 2

Agenda

1. Wrap up: Problem Statement
2. Proposed Solution
3. Tools
4. Data Structures
5. Functions
6. Details on the implementation
7. Algorithm A and B
8. Challenges
9. Demo
10. Schedule and Next Steps

Poonam Kumari Roy, Mohamed Mesto, Yuni Quintero | AWT PJ | Workshop 2

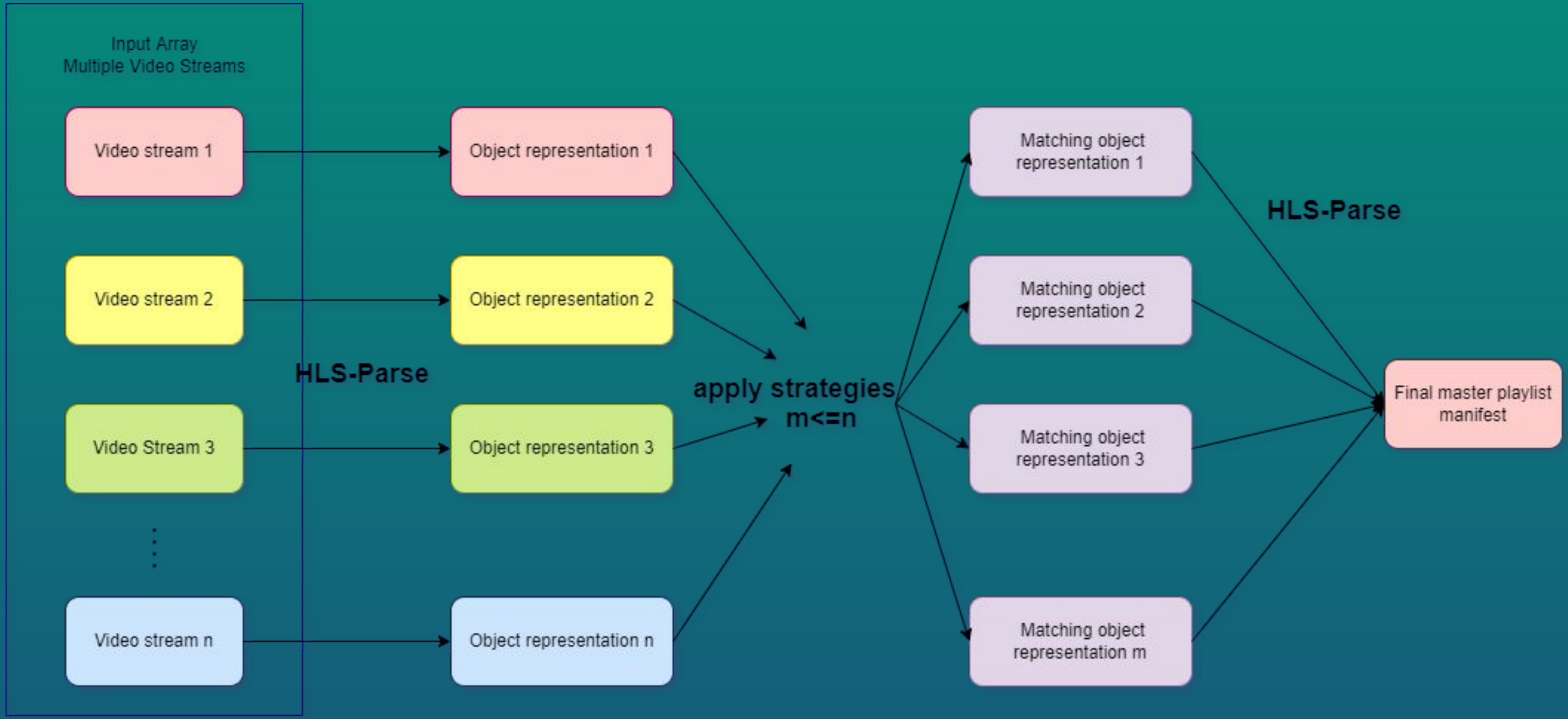
Wrap up: Problem Statement

Adaptive bitrate streaming standards, like HLS, provide a solution to avoid buffering while consuming video streams over the internet.

To enjoy multiple streams, these streams should be compatible with each other in terms of resolution.

We need to implement a strategy that finds and selects which streams are a match.

Proposed Solution



Strategy A: Match representations against first element of input array

Strategy B: Find intersection of representations

Tools

IDE: Visual Studio Code

Version Control: Git

Programming language: Javascript

hls-parser library

Free HLS m3u8 URLs:
https://ottverse.com/free-hls-m3u8-test-urls/

Data Structures

URLs array: <String>[]

Variants Dictionary:

{key<Int>: value<Variant>[]}

The keys are the corresponding indexes of the URLs array and have as value an array of the different variants of a stream as an object representation

Objects array: <Playlist>[]

Filled with the object representation Playlist of the multiple streams

Functions

run()

parseStreamData(Response) -> <Variant>[]

removeDuplicates(array) -> array

getResolutions(array) -> array

algorithmA()

algorithmB()

Details on the Implementation

1. Define an array of video streams URLs.
2. Fetch the manifest text data from the urls.
3. Parse the manifest text into object representation with hls-parser.
4. Create the needed Data Structures: Object array and Variants Dictionary.
5. We execute algorithm A and B.

Algorithm A

1. We need as a final output a master playlist that contains the exact same representations available from the first video stream of the input array.
2. Extract resolutions of the first element of the array from the Variants dictionary, these are our needed resolutions.
3. We iterate over the other streams and for each we will obtain its variants.
4. Then we compare each needed resolution against all the representations available for every other stream, checking if it is present.
5. We save the object representation of the matching streams into an array for later use.

Algorithm B

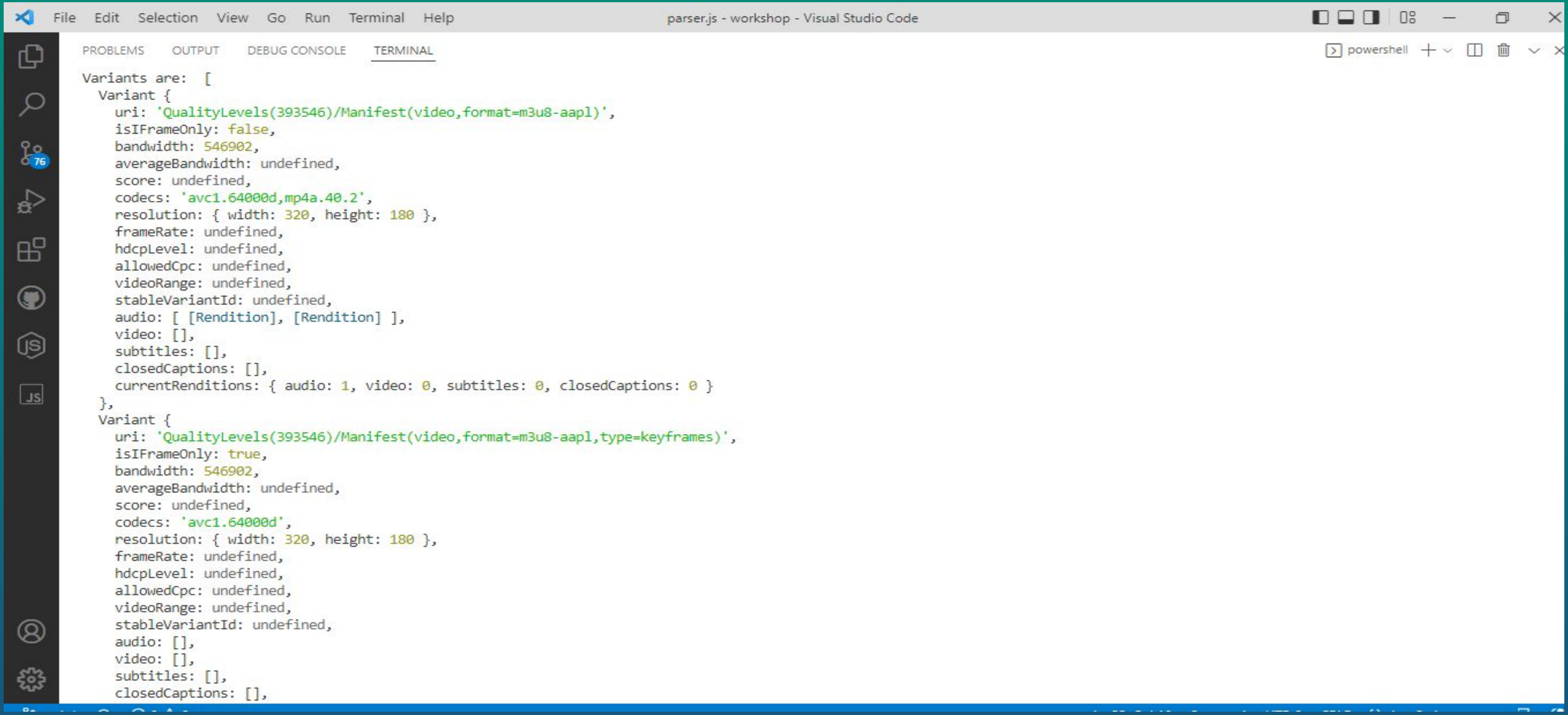
1. We need as a final output a master playlist that contains the representations resulting from an intersection, if one exists, of all the streams’ variants.
2. From the variants dictionary we get the representations for each stream, making an array of arrays.
3. Then, for every stream representation we check if it exists in every other streams variants.
4. The number of times a representation should appear in the other streams in order to be in the intersection should equal to the amount of video streams minus 1.
5. If no representation is present in every stream, the intersection is empty.

Challenges

1. Duplicated resolutions with different bitrate.

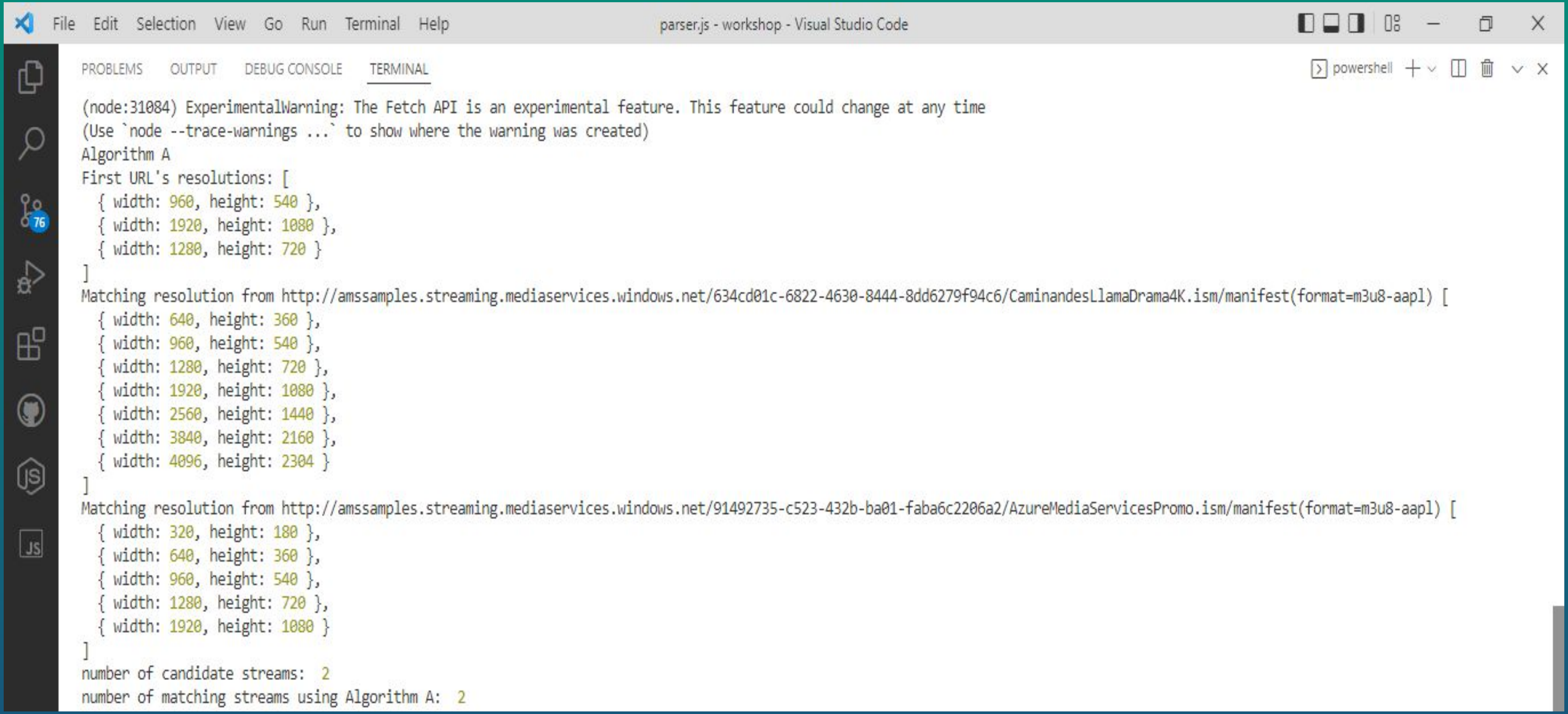
2. As the elements of the variants array are objects of type {width: Int, height: Int}, finding an algorithm to search for elements in the array and the intersection were not trivial. For loops were used to reach a working solution.

Demo



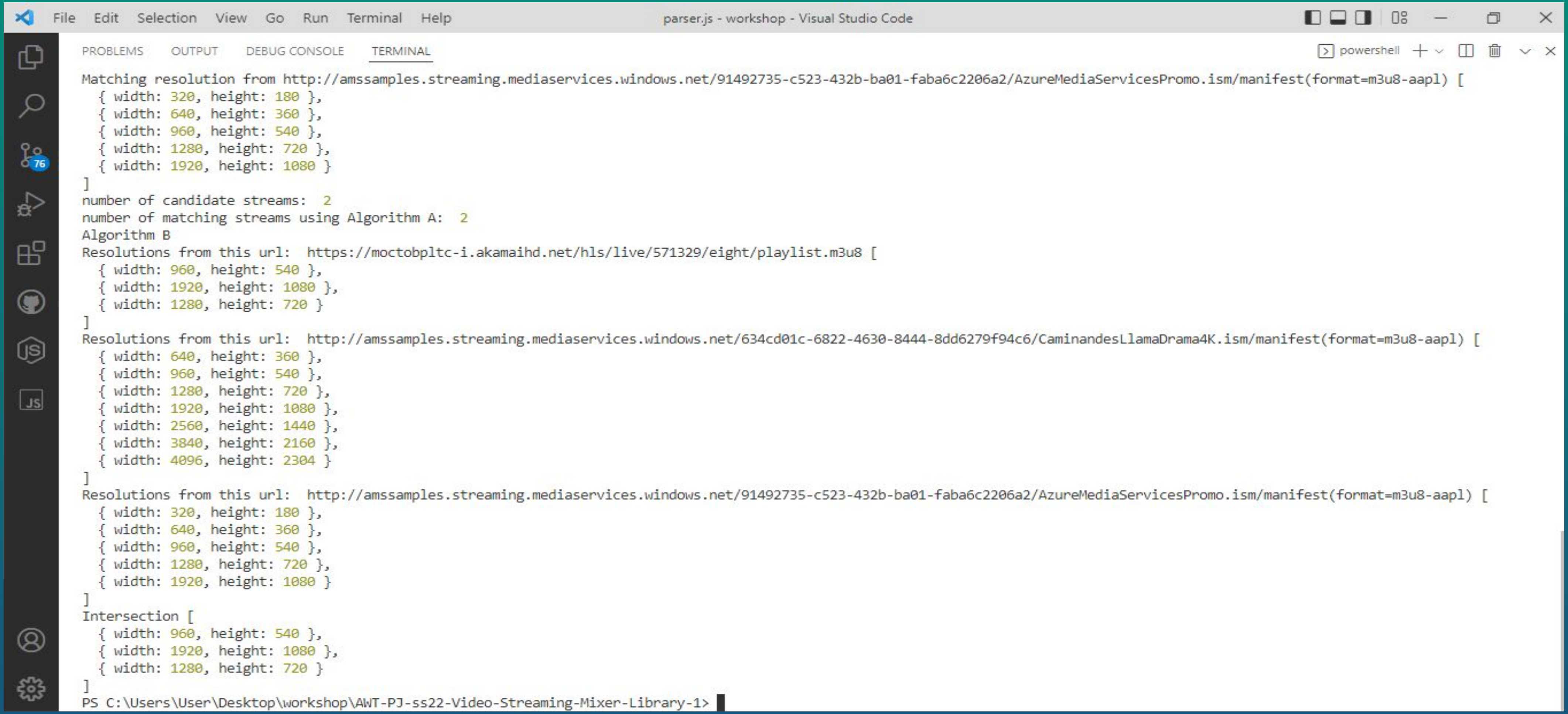
```
Variants are:  [
  Variant {
    uri: 'QualityLevels(393546)/Manifest(video,format=m3u8-aapl)',
    isIFrameOnly: false,
    bandwidth: 546902,
    averageBandwidth: undefined,
    score: undefined,
    codecs: 'avc1.64000d,mp4a.40.2',
    resolution: { width: 320, height: 180 },
    frameRate: undefined,
    hdcpLevel: undefined,
    allowedCpc: undefined,
    videoRange: undefined,
    stableVariantId: undefined,
    audio: [ [Rendition], [Rendition] ],
    video: [],
    subtitles: [],
    closedCaptions: [],
    currentRenditions: { audio: 1, video: 0, subtitles: 0, closedCaptions: 0 }
  },
  Variant {
    uri: 'QualityLevels(393546)/Manifest(video,format=m3u8-aapl,type=keyframes)',
    isIFrameOnly: true,
    bandwidth: 546902,
    averageBandwidth: undefined,
    score: undefined,
    codecs: 'avc1.64000d',
    resolution: { width: 320, height: 180 },
    frameRate: undefined,
    hdcpLevel: undefined,
    allowedCpc: undefined,
    videoRange: undefined,
    stableVariantId: undefined,
    audio: [],
    video: [],
    subtitles: [],
    closedCaptions: [],
```

Demo


File Edit Selection View Go Run Terminal Help
parser.js - workshop - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell

```
(node:31084) ExperimentalWarning: The Fetch API is an experimental feature. This feature could change at any time
(Use `node --trace-warnings ...` to show where the warning was created)
Algorithm A
First URL's resolutions: [
  { width: 960, height: 540 },
  { width: 1920, height: 1080 },
  { width: 1280, height: 720 }
]
Matching resolution from http://amssamples.streaming.mediaservices.windows.net/634cd01c-6822-4630-8444-8dd6279f94c6/CaminandesLlamaDrama4K.ism/manifest(format=m3u8-aapl) [
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 },
  { width: 2560, height: 1440 },
  { width: 3840, height: 2160 },
  { width: 4096, height: 2304 }
]
Matching resolution from http://amssamples.streaming.mediaservices.windows.net/91492735-c523-432b-ba01-faba6c2206a2/AzureMediaServicesPromo.ism/manifest(format=m3u8-aapl) [
  { width: 320, height: 180 },
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 }
]
number of candidate streams:  2
number of matching streams using Algorithm A:  2
```

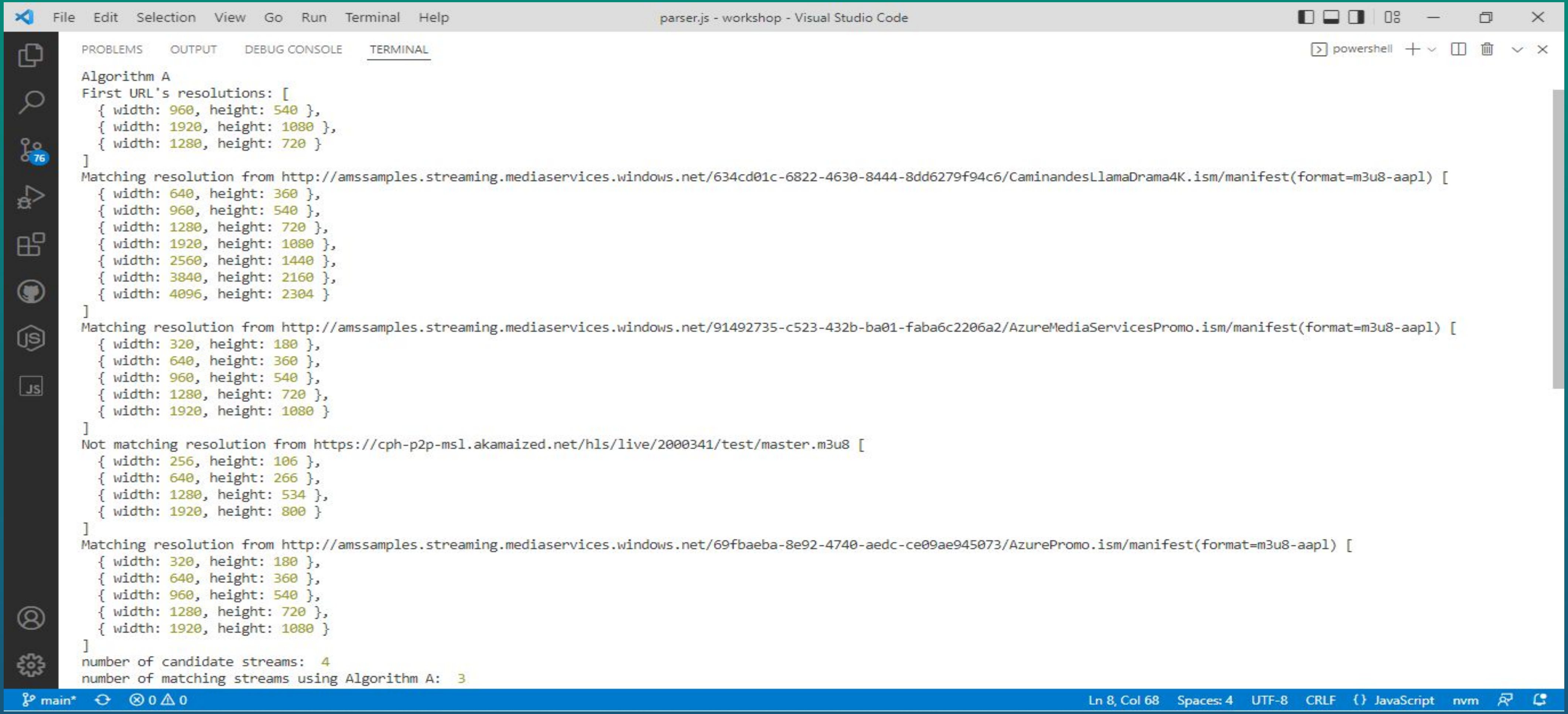
Demo


File Edit Selection View Go Run Terminal Help
parser.js - workshop - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell

```
Matching resolution from http://amssamples.streaming.mediaservices.windows.net/91492735-c523-432b-ba01-faba6c2206a2/AzureMediaServicesPromo.ism/manifest(format=m3u8-aapl) [
  { width: 320, height: 180 },
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 }
]
number of candidate streams:  2
number of matching streams using Algorithm A:  2
Algorithm B
Resolutions from this url:  https://moctobpltc-i.akamaihd.net/hls/live/571329/eight/playlist.m3u8 [
  { width: 960, height: 540 },
  { width: 1920, height: 1080 },
  { width: 1280, height: 720 }
]
Resolutions from this url:  http://amssamples.streaming.mediaservices.windows.net/634cd01c-6822-4630-8444-8dd6279f94c6/CaminandesLlamaDrama4K.ism/manifest(format=m3u8-aapl) [
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 },
  { width: 2560, height: 1440 },
  { width: 3840, height: 2160 },
  { width: 4096, height: 2304 }
]
Resolutions from this url:  http://amssamples.streaming.mediaservices.windows.net/91492735-c523-432b-ba01-faba6c2206a2/AzureMediaServicesPromo.ism/manifest(format=m3u8-aapl) [
  { width: 320, height: 180 },
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 }
]
Intersection [
  { width: 960, height: 540 },
  { width: 1920, height: 1080 },
  { width: 1280, height: 720 }
]
PS C:\Users\User\Desktop\workshop\AWT-PJ-ss22-Video-Streaming-Mixer-Library-1>
```

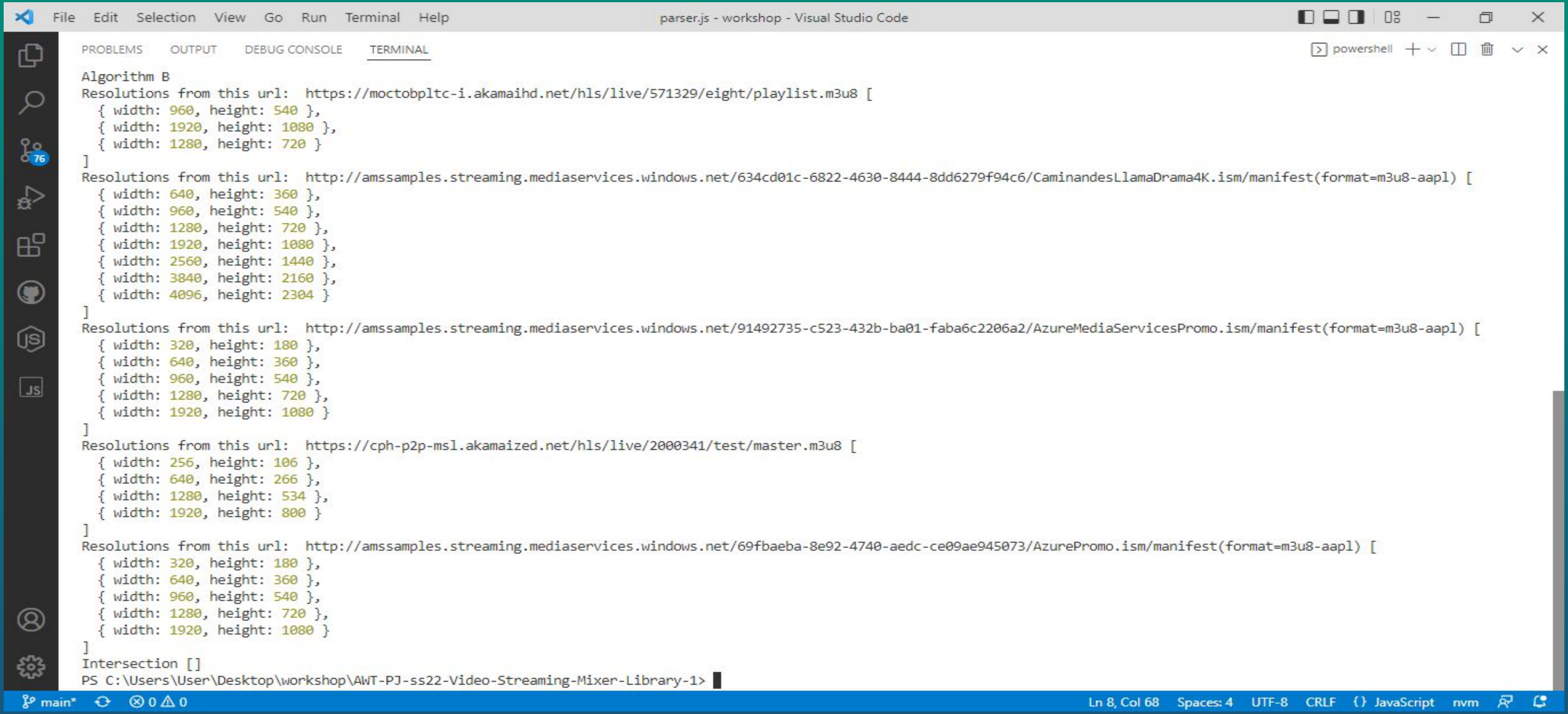
Poonam Kumari Roy, Mohamed Mesto, Yuni Quintero | AWT PJ | Workshop 2

Demo


File Edit Selection View Go Run Terminal Help
parser.js - workshop - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell
main* 0 0
Ln 8, Col 68 Spaces: 4 UTF-8 CRLF JavaScript nvm

```
Algorithm A
First URL's resolutions: [
  { width: 960, height: 540 },
  { width: 1920, height: 1080 },
  { width: 1280, height: 720 }
]
Matching resolution from http://amssamples.streaming.mediaservices.windows.net/634cd01c-6822-4630-8444-8dd6279f94c6/CaminandesLlamaDrama4K.ism/manifest(format=m3u8-aapl) [
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 },
  { width: 2560, height: 1440 },
  { width: 3840, height: 2160 },
  { width: 4096, height: 2304 }
]
Matching resolution from http://amssamples.streaming.mediaservices.windows.net/91492735-c523-432b-ba01-faba6c2206a2/AzureMediaServicesPromo.ism/manifest(format=m3u8-aapl) [
  { width: 320, height: 180 },
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 }
]
Not matching resolution from https://cph-p2p-msl.akamaized.net/hls/live/2000341/test/master.m3u8 [
  { width: 256, height: 106 },
  { width: 640, height: 266 },
  { width: 1280, height: 534 },
  { width: 1920, height: 800 }
]
Matching resolution from http://amssamples.streaming.mediaservices.windows.net/69fbaeba-8e92-4740-aedc-ce09ae945073/AzurePromo.ism/manifest(format=m3u8-aapl) [
  { width: 320, height: 180 },
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 }
]
number of candidate streams:  4
number of matching streams using Algorithm A:  3
```

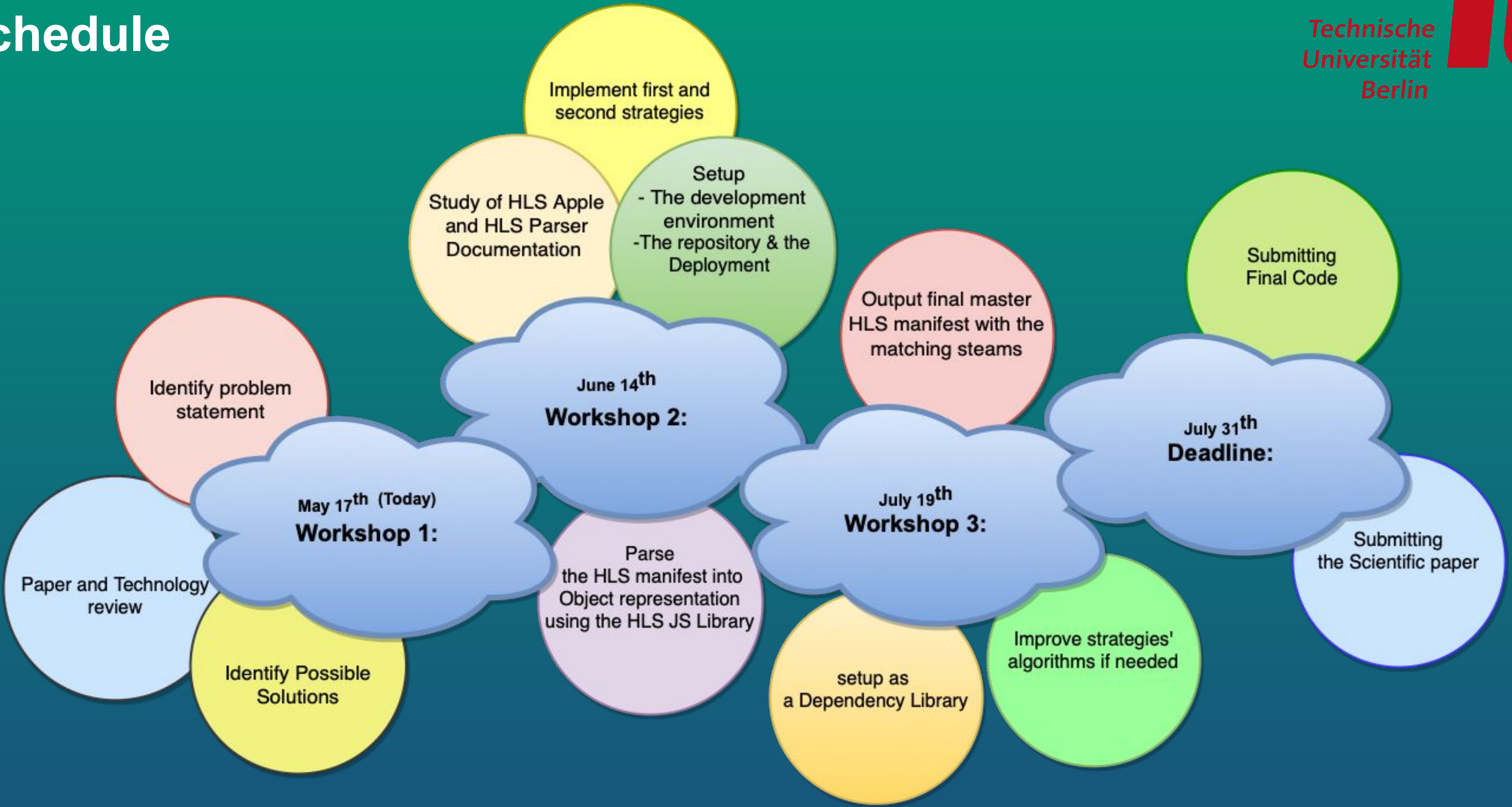
Demo


File Edit Selection View Go Run Terminal Help
parser.js - workshop - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell
main* ⊗ 0 ⚠ 0
Ln 8, Col 68 Spaces: 4 UTF-8 CRLF {} JavaScript nvm

```
Algorithm B
Resolutions from this url:  https://moctobpltc-i.akamaihd.net/hls/live/571329/eight/playlist.m3u8 [
  { width: 960, height: 540 },
  { width: 1920, height: 1080 },
  { width: 1280, height: 720 }
]
Resolutions from this url:  http://amssamples.streaming.mediaservices.windows.net/634cd01c-6822-4630-8444-8dd6279f94c6/CaminandesLlamaDrama4K.ism/manifest(format=m3u8-aapl) [
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 },
  { width: 2560, height: 1440 },
  { width: 3840, height: 2160 },
  { width: 4096, height: 2304 }
]
Resolutions from this url:  http://amssamples.streaming.mediaservices.windows.net/91492735-c523-432b-ba01-faba6c2206a2/AzureMediaServicesPromo.ism/manifest(format=m3u8-aapl) [
  { width: 320, height: 180 },
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 }
]
Resolutions from this url:  https://cph-p2p-msl.akamaized.net/hls/live/2000341/test/master.m3u8 [
  { width: 256, height: 106 },
  { width: 640, height: 266 },
  { width: 1280, height: 534 },
  { width: 1920, height: 800 }
]
Resolutions from this url:  http://amssamples.streaming.mediaservices.windows.net/69fbaeba-8e92-4740-aedc-ce09ae945073/AzurePromo.ism/manifest(format=m3u8-aapl) [
  { width: 320, height: 180 },
  { width: 640, height: 360 },
  { width: 960, height: 540 },
  { width: 1280, height: 720 },
  { width: 1920, height: 1080 }
]
Intersection []
PS C:\Users\User\Desktop\workshop\AWT-PJ-ss22-Video-Streaming-Mixer-Library-1>
```

Poonam Kumari Roy, Mohamed Mesto, Yuni Quintero | AWT PJ | Workshop 2

Schedule


Implement first and second strategies
Study of HLS Apple and HLS Parser Documentation
Setup - The development environment -The repository & the Deployment
Identify problem statement
June 14th Workshop 2:
Output final master HLS manifest with the matching steams
Submitting Final Code
July 31th Deadline:
May 17th (Today) Workshop 1:
Paper and Technology review
Parse the HLS manifest into Object representation using the HLS JS Library
July 19th Workshop 3:
Submitting the Scientific paper
Identify Possible Solutions
setup as a Dependency Library
Improve strategies' algorithms if needed

Next Steps

1. Refine and improve the algorithms if needed
2. Create and output final master playlist manifest using hls-parser with the matching streams from each strategy
3. Refactor code to deploy as a dependency. Algorithms A and B should receive URLs as parameters

Resources

- https://bitmovin.com/adaptive-streaming/
- https://www.wowza.com/blog/adaptive-bitrate-streaming
- HLS documentation https://datatracker.ietf.org/doc/html/rfc8216
- hls-parse js library https://www.npmjs.com/package/hls-playlist-parser
- R. Seeliger, D. Silhavy, Dr. S. Arbanowski "Dynamic ad-insertion and content orchestration workflows through manifest manipulation in HLS and MPEG-DASH" https://ieeexplore.ieee.org/document/8228708
- https://developer.apple.com/documentation/http_live_streaming/about_apple_s_http_live_streaming_tools