

MoviesHttpServiceRxJS (overview)

export interface Movie {

    id: number;

    poster\_path: string;

    adult: boolean;

    overview: string;

    title: string;

    release\_date: string;

    genres: string;

    vote\_average: string;

}

…

      <nav class="nav navbar-nav mr-auto movieservice">

        <a class="nav-item nav-link" **(click)="popularMovies()">**Popular </a>

….  
      </nav>

        <form class="nav-item justify-content-end" role="search">

            <div class=" input-group">

                <input type="text" …. >

                <button …. **(click)="onClick(searchName.value)"** >…</button>

            </div>

        </form>

      </nav>

    ….

   <li **\*ngFor**="let **movie** of **movies**; let first = first"  class="movielist">

     <div \*ngIf="movie.poster\_path">

…               
   <img class="img-responsive img-rounded" src={{**movie.poster\_path**}} >

         <…  {{**movie.title**}} ….>

         <…  {{**movie.vote\_average**}} ….>

….

         <…  {{**movie.overview**}} ….>

          Play Trailer <button … (**click)="playMovie(movie)"** ></button>

    </li>

Map the json to our Movie type

RxJS pipe is a composer operator, each operator in the pipe returns a new observables

remove all movies without posters

RxJS Observable, push the array of Movie to the subscriber

import { **Observable** } from 'rxjs';  
import **{ HttpClient** } from '@angular/common/http';  
import { **map** } from 'rxjs/operators';

....  
interface IMovieData {results: Movie[]; }

export interface IVideoData {results: IVideo[]; }

export interface IVideo {'id': string; 'key': string; 'name': string; 'site': string; 'type': string; }

@Injectable()

export class **HttpService** {

  private movies: any;

  private key: string;

  constructor(private **http**: **HttpClient**) { }

  public **getMovie(title: string):** **Observable**<Movie[]> {

    return this.http.get<IMovieData>('http://api.themoviedb.org/3/search/movie?query=' + title + '&api\_key=…')

    .pipe(map(movies => {

       return movies.results.filter(movie => movie.poster\_path !== null).map(movie => {

        return { id: movie.id,

                 title: movie.title,

                 poster\_path: 'https://image.tmdb.org/t/p/w185\_and\_h278\_bestv2' + movie.poster\_path,

                 adult: movie.adult,

                 … });

    }));

  }

  public **getMostPopularMovies():** **Observable**<Movie[]> {

    return this.**http.get**<IMovieData>('http://api.themoviedb.org/3/discover/movie?sort\_by=popularity.desc&api\_key=…')

**.pipe**(**map**(movies => {

        return movies.results.**filter**(movie => movie.poster\_path !== null).**map**(movie => {

        return { id : movie.id,

                 title : movie.title,

                 poster\_path : 'https://image.tmdb.org/t/p/w185\_and\_h278\_bestv2' + movie.poster\_path,

                 adult : movie.adult,

                 overview : movie.overview,

**release\_date : movie.release\_date,**

**genres : movie.genres,**

**vote\_average : movie.vote\_average } as Movie; });**

    }));

  }

public **async getVideoPath(id: number):** Promise<string> {

    const videos = **await this.http.get**<IVideoData>('https://api.themoviedb.org/3/movie/' + id + '/videos?api\_key=…').toPromise();

    return videos.results[0].key;

}

RxJS subscribe

4

6

7

7

7

7

5

3

2

1

export class **MoviesComponent**  {

private **movies**: Movie[];

constructor(private **service: HttpService**,…) { }

public **async playMovie(movie: Movie)** {

      const path: string = await this.service.**getVideoPath(movie.id);  
    this.router.navigate(['/video-play', path]);**

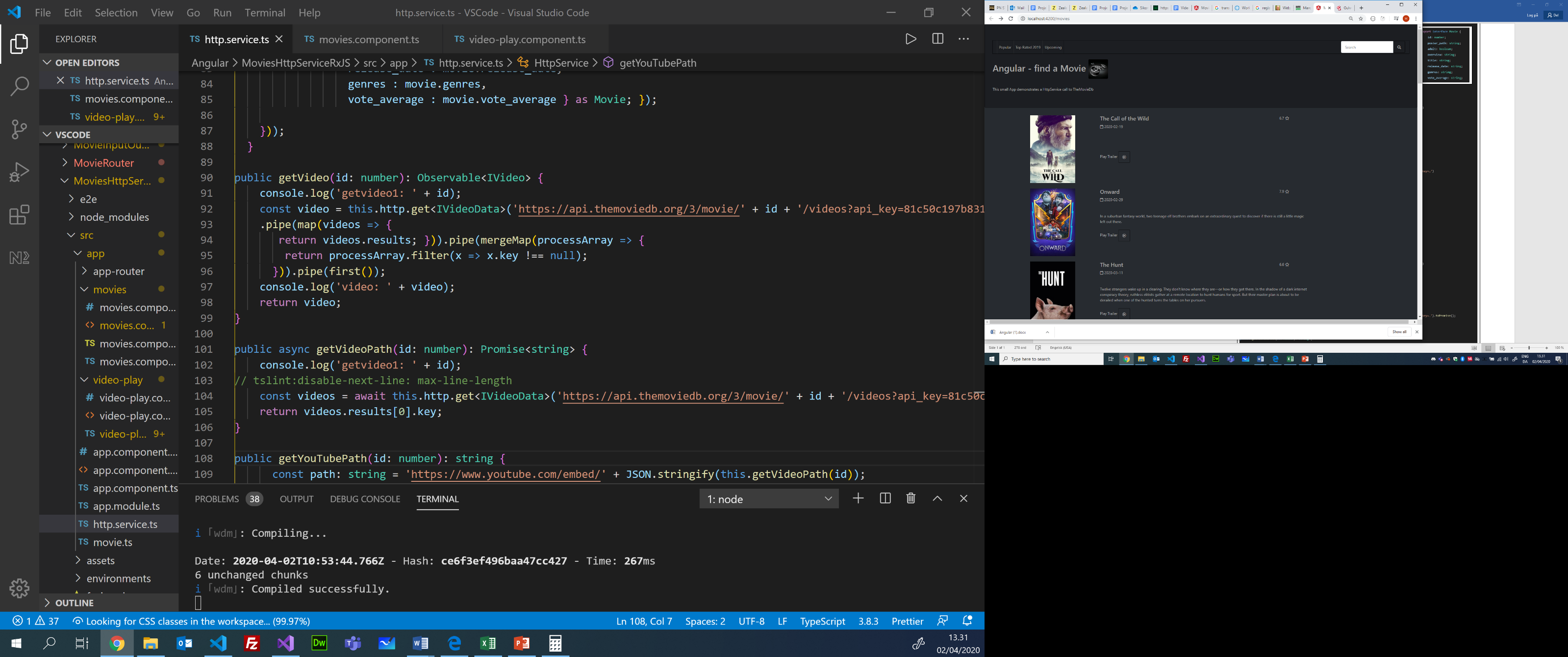
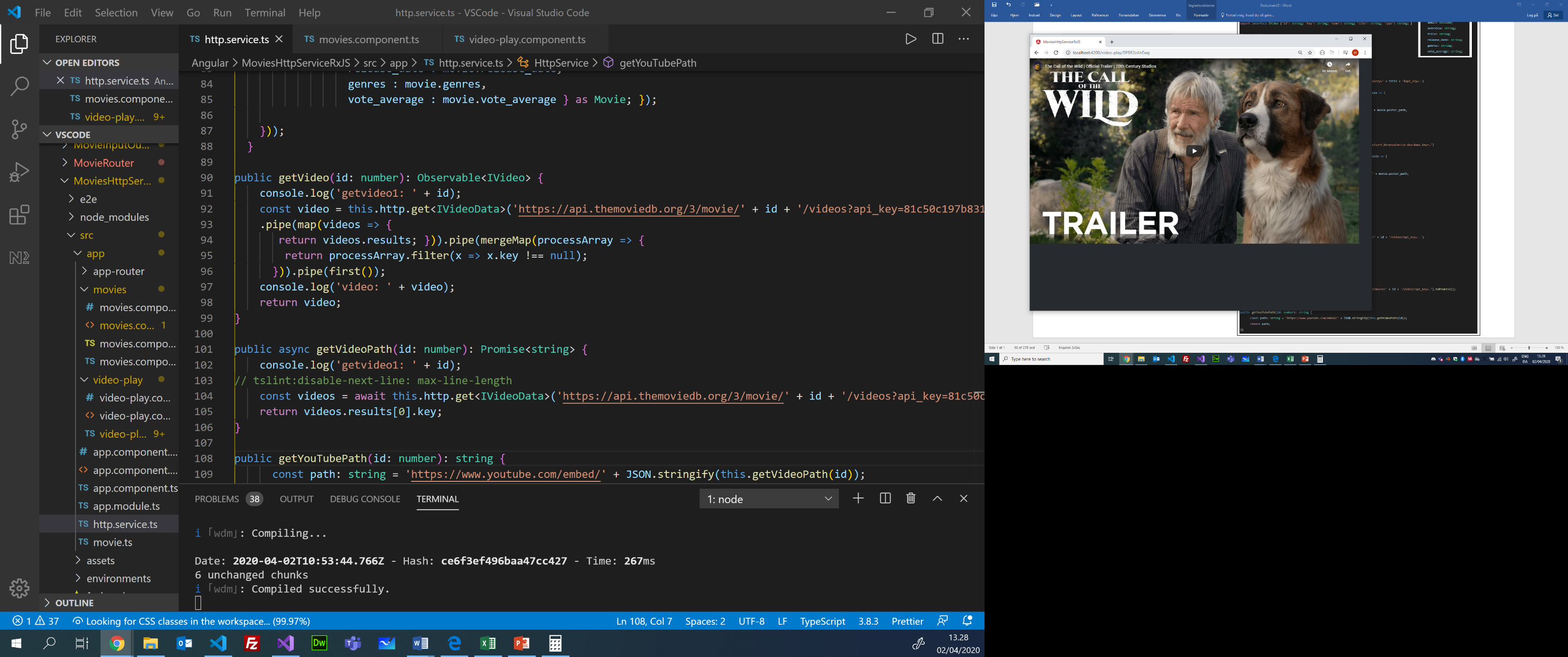
  }

 public **onClick(searchName: string)** { this.service.**getMovie(searchName**).**subscribe**(response => this.movies = response); }

 public **popularMovies()** { this.service.**getMostPopularMovies().subscribe(response => this.movies = response); }**

**….**

}



export interface Movie {

    id: number;

    poster\_path: string;

    adult: boolean;

    overview: string;

    title: string;

    release\_date: string;

    genres: string;

    vote\_average: string;

}

…

      <nav class="nav navbar-nav mr-auto movieservice">

        <a class="nav-item nav-link" **(click)="popularMovies()">**Popular </a>

….  
      </nav>

        <form class="nav-item justify-content-end" role="search">

            <div class=" input-group">

                <input type="text" …. >

                <button …. **(click)="onClick(searchName.value)"** >…</button>

            </div>

        </form>

      </nav>

    ….

   <li \*ngFor="let **movie** of **movies**; let first = first"  class="movielist">

     <div \*ngIf="movie.poster\_path">

…               
   <img class="img-responsive img-rounded" src={{**movie.poster\_path**}} >

         <…  {{**movie.title**}} ….>

         <…  {{**movie.vote\_average**}} ….>

….

         <…  {{**movie.overview**}} ….>

          Play Trailer <button … (**click)="playMovie(movie)"** ></button>

    </li>

import { **Observable** } from 'rxjs';  
import **{ HttpClient** } from '@angular/common/http';  
import { **map** } from 'rxjs/operators';

....  
interface IMovieData {results: Movie[]; }

export interface IVideoData {results: IVideo[]; }

export interface IVideo {'id': string; 'key': string; 'name': string; 'site': string; 'type': string; }

@Injectable()

export class **HttpService** {

  private movies: any;

  private key: string;

  constructor(private http: **HttpClient**) { }

  public **getMovie(title: string):** Observable<Movie[]> {

    return this.http.get<IMovieData>('http://api.themoviedb.org/3/search/movie?query=' + title + '&api\_key=…')

    .pipe(map(movies => {

       return movies.results.filter(movie => movie.poster\_path !== null).map(movie => {

        return { id: movie.id,

                 title: movie.title,

                 poster\_path: 'https://image.tmdb.org/t/p/w185\_and\_h278\_bestv2' + movie.poster\_path,

                 adult: movie.adult,

                 … });

    }));

  }

  public **getMostPopularMovies():** Observable<Movie[]> {

    return this.**http.get**<IMovieData>('http://api.themoviedb.org/3/discover/movie?sort\_by=popularity.desc&api\_key=…')

**.pipe**(**map**(movies => {

        return movies.results.**filter**(movie => movie.poster\_path !== null).**map**(movie => {

        return { id : movie.id,

                 title : movie.title,

                 poster\_path : 'https://image.tmdb.org/t/p/w185\_and\_h278\_bestv2' + movie.poster\_path,

                 adult : movie.adult,

                 overview : movie.overview,

…. });

    }));

  }

public **async getVideoPath(id: number):** Promise<string> {

    const videos = **await this.http.get**<IVideoData>('https://api.themoviedb.org/3/movie/' + id + '/videos?api\_key=…').toPromise();

    return videos.results[0].key;

}

6

5

4

3

2

1

export class **MoviesComponent**  {

private **movies**: Movie[];

constructor(private **service: HttpService**,…) { }

public **async playMovie(movie: Movie)** {

      const path: string = await this.service.**getVideoPath(movie.id);  
    this.router.navigate(['/video-play', path]);**

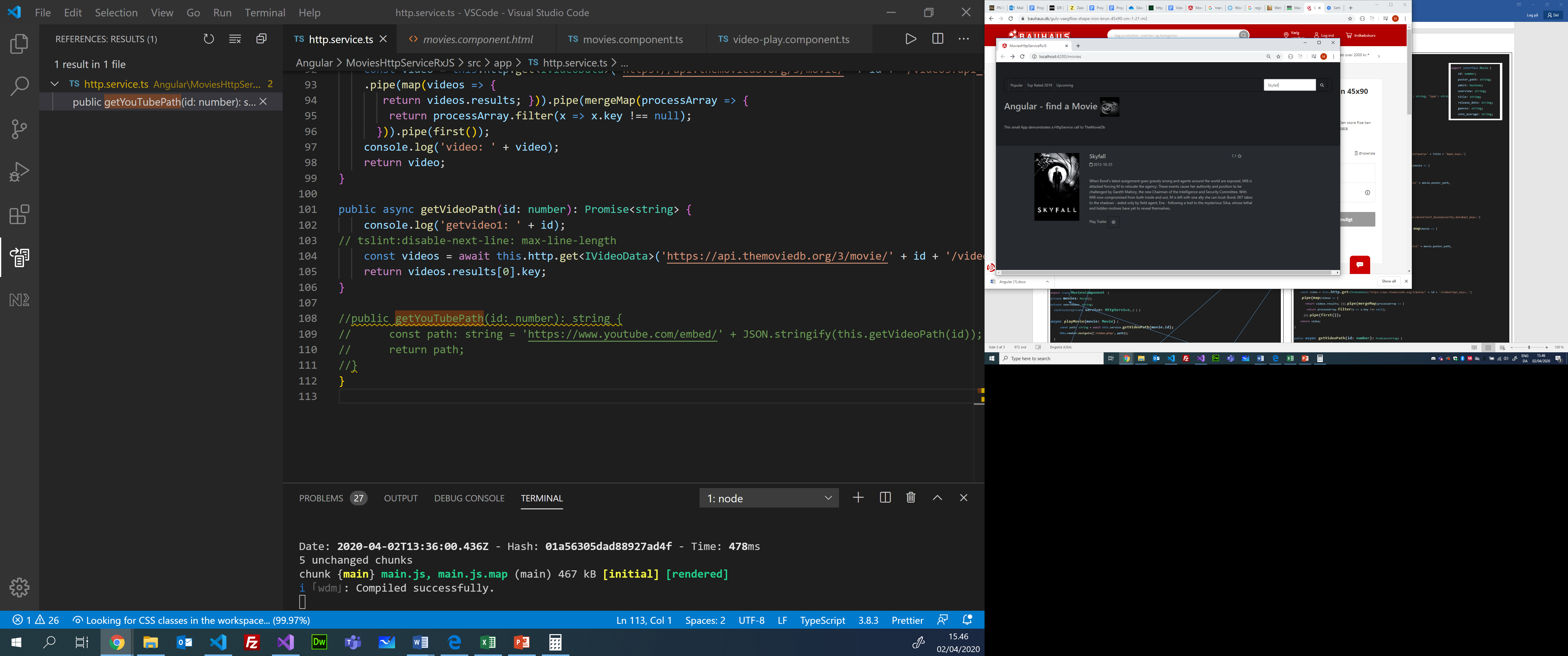
  }

 public **onClick(searchName: string)** { this.service.**getMovie(searchName**).**subscribe**(response => this.movies = response); }

 public **popularMovies()** { this.service.**getMostPopularMovies().subscribe**(response => this.**movies** = response); }

….

}



export interface Movie {

    id: number;

    poster\_path: string;

    adult: boolean;

    overview: string;

    title: string;

    release\_date: string;

    genres: string;

    vote\_average: string;

}

C/Henrik Høltzer, Zealand April 14. 2020

5

3

4

8

8

8

8

7

6

2

1

import**{ Observable**} from 'rxjs';  
import **{ HttpClient** } from '@angular/common/http';  
import { **map** } from 'rxjs/operators';

....  
interface IMovieData {results: Movie[]; }

export interface IVideoData {results: IVideo[]; }

export interface IVideo {'id': string; 'key': string; 'name': string; 'site': string; 'type': string; }

@Injectable()

export class **HttpService** {

  private movies: any;

  private key: string;

  constructor(private http: **HttpClient**) { }

  public **getMovie(title: string):** Observable<Movie[]> {

    return this.http.get<IMovieData>('http://api.themoviedb.org/3/search/movie?query=' + title + '&api\_key=…')

    .pipe(map(movies => {

       return movies.results.filter(movie => movie.poster\_path !== null).map(movie => {

        return { id: movie.id,

                 title: movie.title,

                 poster\_path: 'https://image.tmdb.org/t/p/w185\_and\_h278\_bestv2' + movie.poster\_path,

                 adult: movie.adult,

                 … });

    }));

  }

  public **getMostPopularMovies():** Observable<Movie[]> {

    return this.**http.get**<IMovieData>('http://api.themoviedb.org/3/discover/movie?sort\_by=popularity.desc&api\_key=…')

**.pipe**(**map**(movies => {

        return movies.results.**filter**(movie => movie.poster\_path !== null).**map**(movie => {

        return { id : movie.id,

                 title : movie.title,

                 poster\_path : 'https://image.tmdb.org/t/p/w185\_and\_h278\_bestv2' + movie.poster\_path,

                 adult : movie.adult,

                 overview : movie.overview,

…. });

    }));

  }

public **async getVideoPath(id: number):** Promise<string> {

    const videos = **await this.http.get**<IVideoData>('https://api.themoviedb.org/3/movie/' + id + '/videos?api\_key=…').toPromise();

    return videos.results[0].key;

}

…

      <nav class="nav navbar-nav mr-auto movieservice">

        <a class="nav-item nav-link" **(click)="popularMovies()">**Popular </a>

….  
      </nav>

        <form class="nav-item justify-content-end" role="search">

            <div class=" input-group">

                <input type="text" …. >

                <button …. **(click)="onClick(searchName.value)"** >…</button>

            </div>

        </form>

      </nav>

    ….

   <li \*ngFor="let **movie** of **movies**; let first = first"  class="movielist">

     <div \*ngIf="movie.poster\_path">

…               
   <img class="img-responsive img-rounded" src={{**movie.poster\_path**}} >

         <…  {{**movie.title**}} ….>

         <…  {{**movie.vote\_average**}} ….>

….

         <…  {{**movie.overview**}} ….>

          Play Trailer <button … (**click)="playMovie(movie)"** ></button>

    </li>

export class **MoviesComponent**  {

private **movies**: Movie[];

constructor(private **service: HttpService**,…) { }

public **async playMovie(movie: Movie)** {

      const path: string = await this.service.**getVideoPath(movie.id);  
    this.router.navigate(['/video-play', path]);**

  }

 public **onClick(searchName: string)** { this.service.**getMovie(searchName**).**subscribe**(response => this.movies = response); }

 public **popularMovies()** { this.service.**getMostPopularMovies().subscribe**(response => this.**movies** = response); }

….

}

