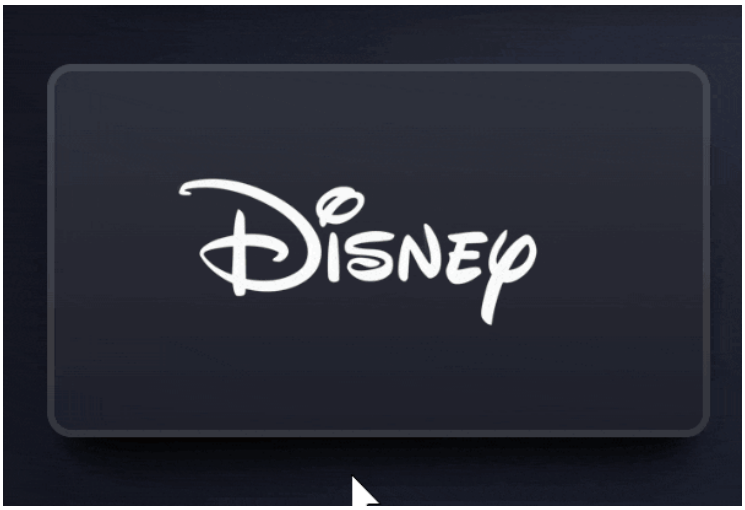
5. Animating with video



Let's break it down ...

Filter.js is a simply component that takes in filter data (see example right) and builds a component for item in the list.

Filters.css does a couple of things:

- Modifies the filter component on hover
- Switches out the image and video presentation on hover

Modify shape on hover:

This is a fairly simple one. This component is designed to have a slight 3D feel to it, created by a gradient **background**, emphasised **border** and **box-shadow.** When we hover here we use the transition property to lerp between our current values and the values on (filter.css) lines 15-16. This gives simple but elegant user feedback on hover and is similar to the user selection.

Image to Video:

Setting up the image, the key entry is the **obje ct-fit** (line 23), this dictates how the image is presented in the container and scales accordingly. We use the **cover** option to expand our image to fill the box.

Because we want the image and the video to take up the same space we need to set the **po sition** element to be **absolute**, this way they sit on top of each other and not side by side. We additionally set the **z-index** of the **image** 1 higher than the **video** so that it stays in place and the video plays beneath.

For the hover (line 41), we simply adjust the video opacity back up to 1.

Data

```
 "filters": {
        "disney":{
            "image": "/images
/disney.png",
            "video": "/videos
/disney.mp4"
        },
        "pixar":{
            "image": "/images
/pixar.png",
            "video": "/videos
/pixar.mp4"
        }
}
```

Filter.js

```
import { routes } from 'Router';
import { useNavigate, useParams
} from 'react-router-dom';
import styles from './filters.
module.css';

const Filters = ({ filters })
=> {
  const navigate =
useNavigate();
  const goTo = (category) => {
    navigate(`${routes.category}
/${category}`);
  };
  return (
    <div className={styles.
container}>
      {Object.keys(filters).map
((key, index) => (
        <div className={styles.
filter} key={index} onClick={()
=> goTo(key)}>
          <img src={filters
[key].image} />
          <video src={filters
[key].video} autoPlay={true}
loop={true} playsInline={true}
type='video/mp4' />
        </div>
      ))}
    </div>
  );
};
export default Filters;
```

filters.css

```
.filter {
    padding-top: 56.25%;
    border-radius: 10px;
    background: linear-gradient
(rgb(48, 50, 62), rgb(30, 31,
42));
    box-shadow: rgb(0 0 0 /
69%) 0px 26px 30px -10px, rgb(0
0 0 / 73%) 0px 16px 10px -10px;
    cursor: pointer;
    overflow: hidden;
    position: relative;
    transition: all 250ms cubic-
bezier(0.25, 0.46, 0.45, 0.94)
0s;
    border: 3px solid rgba(249,
249, 249, 0.1);
  }

  .filter:hover {
    box-shadow: rgb(0 0 0 /
69%) 0px 26px 30px -10px, rgb(0
0 0 / 73%) 0px 16px 10px -10px;
    transform: scale(1.05);
    border-color: rgba(249,
249, 249, 0.8);
  }

  .filter > img {
      inset: 0px;
      display: block;
      height: 100%;
      object-fit: cover;
      opacity: 1;
      position: absolute;
      transition: opacity 500ms
ease-in-out 0s;
      width: 100%;
      z-index: 1;
      top: 0;
  }

  .filter > video {
      width: 100%;
      height: 100%;
      position: absolute;
      top: 0px;
      opacity: 0;
      z-index: 0;
  }

  .filter:hover > video{
      opacity: 1;
  }
```