Dependencies:

Yarn add scss

Yarn add @material-ui/core

Yarn add @material-ui/icons

SCSS:

Creating variable in app.scss to use in whole app where needed

//accessible in all files

:root{

  --main-color: #0b0b0b

}

Now using in home.scss

.home{

    background-color: var(--main-color);

}

Navbar with changing color on scrolling:

If no scroll then transparent color and when scroll then navbar color changes to black or any other color, done using useState and passing className based on condition

import { ArrowDropDown, Notifications, Search } from '@material-ui/icons';

import React from 'react';

import { useState } from 'react';

import './Navbar.scss'

function Navbar() {

    const [isScroll, setIsScroll] = useState(false);

    // console.log(window.scrollY)

    window.onscroll =  () => {

        setIsScroll(window.scrollY === 0 ? false : true)

        return () => (window.onscroll = null)

    }

    // console.log(isScroll)

    return (

        <div className={isScroll ? 'navbar scrolled' : 'navbar'}>

            <div className="container">

                <div className="left">

                    <img

                        src="https://upload.wikimedia.org/wikipedia/commons/thumb/0/08/Netflix\_2015\_logo.svg/2560px-Netflix\_2015\_logo.svg.png"

                        alt=""

                    />

                    <span>Homepage</span>

                    <span>Series</span>

                    <span>Movies</span>

                    <span>New and Popular</span>

                    <span>My list</span>

                </div>

                <div className="right">

                    <Search className="icon" />

                    <span>KID</span>

                    <Notifications className="icon" />

                    <img

                    src="https://images.pexels.com/photos/6899260/pexels-photo-6899260.jpeg?auto=compress&cs=tinysrgb&dpr=2&w=500"

                    alt=""

                    />

                    <div className="profile">

                        <ArrowDropDown className="icon" />

                        <div className="options">

                           <span>Settings</span>

                            <span>Logout</span>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    )

}

export default Navbar

SCSS:

.navbar{

    color: white;

    font-size: 14px;

    position: fixed;

    top: 0;

    width: 100%;

    z-index: 999;

    background: linear-gradient(to top, transparent 0%, rgb(0, 0, 0, 0.3) 50%);

    &.scrolled{

        background-color: var(--main-color);

    }

    .container{

        padding: 0px 50px;

        display: flex;

        align-items: center;

        justify-content: space-between;

        height: 70px;

        img{

            height: 25px;

        }

        .left{

            display: flex;

            align-items: center;

            img{

                height: 25px;

                margin-right: 40px;

            }

            span{

                margin-right: 20px;

                cursor: pointer;

            }

        }

        .right{

            display: flex;

            align-items: center;

            img{

                width: 30px;

                height: 30px;

                border-radius: 5px;

                object-fit: cover;

                cursor: pointer;

            }

            .icon{

                margin: 0 15px;

                cursor: pointer;

            }

            .profile{

                .options{

                    display: none;

                    background-color: var(--main-color);

                    border-radius: 5px;

                    span{

                        padding: 10px;

                        cursor: pointer;

                    }

                }

                &:hover{

                    .options{

                        display: flex;

                        flex-direction: column;

                        position: absolute;

                    }

                }

            }

        }

    }

}

Icon and text inside button:

<button className="play">

                        <PlayArrow />

                        <span>Play</span>

                    </button>

Call same level classes or same level element and class in scss:

Same level Classes:

<div className={isScroll ? 'navbar scrolled' : 'navbar'}>

.navbar{

    color: white;

    font-size: 14px;

    position: fixed;

    top: 0;

    width: 100%;

    z-index: 999;

    background: linear-gradient(to top, transparent 0%, rgb(0, 0, 0, 0.3) 50%);

    &.scrolled{

        background-color: var(--main-color);

    }

Same level Element and class:

<button className="play">

                        <PlayArrow />

                        <span>Play</span>

                    </button>

                    <button className="more">

                        <InfoOutlined />

                        <span>Info</span>

                    </button>

button{

                padding: 10px 20px;

                border: none;

                border-radius: 5px;

                display: flex;

                align-items: center;

                justify-content: center;

                font-size: 18px;

                font-weight: 500;

                margin-right: 10px;

                cursor: pointer;

                &.play{

                    background-color: white;

                    color: var(--main-color)

                }

                &.more{

                    background-color: gray;

                    color: white;

                }

                span{

                    margin-left: 5px;

                }

Position:

Top bottom left right used with position absolute. One thing to remember is that parent should always be position relative of position absolute component

To make things in Center in position absolute:

top: 0;

left: 0;

margin: auto (optional)

useRef() hook:

Taking input using useRef():

const [email, setEmail] = useState('')

    const [password, setPassword] = useState('')

    const emailRef = useRef();

    const passwordRef = useRef();

    const handleStart = () => {

        setEmail(emailRef.current.value)

    }

    const handleFinish = () => {

        setEmail(passwordRef.current.value)

    }

{!email ? (

                        <div className="input">

                            <input type="email" placeholder="email address" ref={emailRef}/>

                            <button onClick={handleStart} className="register\_\_button">Get started</button>

                         </div>

                    ): (

                        <form className="input">

                            <input type=”password" placeholder="password" ref={passwordRef}/>

                            <button onClick={handleFinish} className="register\_\_button">Start</button>

                        </form>

                    )

                }

To select any container or html element, we use useRef hook and pass ref={ref} in that element. Later we can call that ref and do whatever we want

const listRef = useRef()

 <div className="container" ref={listRef}>

Now using listRef to do work with container

const handleClick = (direction) => {

        let distance = listRef.current.getBoundingClientRect().x - 50;

        if(direction === "left"){

            listRef.current.style.transform = `translateX(${230 + distance}px)`

        }

        if(direction === "right"){

            listRef.current.style.transform = `translateX(${-230 + distance}px)`

        }

        console.log(distance)

    }

.current is very important

Slider Code:

import { ArrowBackIosOutlined, ArrowForwardIosOutlined } from '@material-ui/icons'

import React from 'react'

import { useState } from 'react'

import { useRef } from 'react'

import ListItem from '../listItem/ListItem'

import './List.scss'

function List() {

    const [slideNumber, setSlideNumber] = useState(0)

    const [isMoved, setIsMoved] = useState(false)

    const listRef = useRef()

    const handleClick = (direction) => {

        setIsMoved(true)

        let distance = listRef.current.getBoundingClientRect().x - 50;

        if(direction === "left" && slideNumber > 0){

            setSlideNumber(slideNumber - 1)

            listRef.current.style.transform = `translateX(${230 + distance}px)`

        }

        if(direction === "right" && slideNumber < 5){

            setSlideNumber(slideNumber + 1)

            listRef.current.style.transform = `translateX(${-230 + distance}px)`

        }

        console.log(distance)

    }

    return (

        <div className="list">

            <span className="list\_\_title">Continue to watch</span>

            <div className="wrapper">

                <ArrowBackIosOutlined

                    className="sliderArrow left"

                    onClick={()=>handleClick("left")}

                    style={{display: !isMoved && 'none'}}

                />

                <div className="container" ref={listRef}>

                    <ListItem/>

                    <ListItem/>

                    <ListItem/>

                    <ListItem/>

                    <ListItem/>

                    <ListItem/>

                    <ListItem/>

                    <ListItem/>

                    <ListItem/>

                    <ListItem/>

                </div>

                <ArrowForwardIosOutlined

                    className="sliderArrow right"

                    onClick={()=>handleClick("right")}

                />

            </div>

        </div>

    )

}

export default List

Scss:

.list{

    width: 100%;

    margin-top: 10px;

    .list\_\_title{

        color: white;

        font-size: 20px;

        font-weight: 500;

        margin-left: 50px;

    }

    .wrapper{

        position: relative;

        .container{

            margin-left: 50px;

            display: flex;

            width: max-content;

            margin-top: 10px;

            transform: translateX(0px);

            transition: all 1s ease;

        }

        .sliderArrow{

            width: 50px;

            height: 100%;

            background-color: rgb(22, 22, 22, 0.5);

            color: white;

            position: absolute;

            z-index: 99;

            top: 0;

            bottom: 0;

            margin: auto;

            cursor: pointer;

            &.left{

                left: 0;

            }

            &.right{

                right: 0;

            }

        }

    }

}

ListItem:

import React from 'react'

import './ListItem.scss'

function ListItem() {

    return (

        <div className="list\_\_item">

            Item

        </div>

    )

}

export default ListItem

Scss:

.list\_\_item{

    width: 225px;

    height: 120px;

    background-color: white;

    margin-right: 5px;

}

Video in React:

const trailer = "https://player.vimeo.com/external/371433846.sd.mp4?s=236da2f3c0fd273d2c6d9a064f3ae35579b2bbdf&profile\_id=139&oauth2\_token\_id=57447761"

<video src={trailer} autoPlay={true} loop/>

Second way:

<video

                className="video"

                autoPlay

                progress

                controls

                src="https://vod-progressive.akamaized.net/exp=1624452918~acl=%2Fvimeo-prod-skyfire-std-us%2F01%2F2400%2F14%2F362003850%2F1486625955.mp4~hmac=d6f829e7bb83f1ee6a28047d00aa2c1083c8fe5036c8084a4adf1c3903085856/vimeo-prod-skyfire-std-us/01/2400/14/362003850/1486625955.mp4"

             />

It includes loading progress component with pause play and volume controls

Hover Container or element:

function ListItem({index}) {

    const [isHovered, setIsHovered] = useState(false);

    const trailer = "https://player.vimeo.com/external/371433846.sd.mp4?s=236da2f3c0fd273d2c6d9a064f3ae35579b2bbdf&profile\_id=139&oauth2\_token\_id=57447761"

    return (

        <div className="list\_\_item"

            style={{left: isHovered && index \* 225 - 50 + index \* 2.5}}

            onMouseEnter={()=>setIsHovered(true)}

            onMouseLeave={()=>setIsHovered(false)}

        >

            <img src="https://occ-0-1723-92.1.nflxso.net/dnm/api/v6/X194eJsgWBDE2aQbaNdmCXGUP-Y/AAAABU7D36jL6KiLG1xI8Xg\_cZK-hYQj1L8yRxbQuB0rcLCnAk8AhEK5EM83QI71bRHUm0qOYxonD88gaThgDaPu7NuUfRg.jpg?r=4ee" alt="" />

            {isHovered && (

            <>

            <video src={trailer} autoPlay={true} loop/>

            <div className="item\_\_info">

                <div className="icons">

                    <PlayArrow className="icon" />

                    <Add className="icon" />

                    <ThumbUpAltOutlined className="icon" />

                    <ThumbDownOutlined className="icon" />

                </div>

                <div className="itemInfo\_\_top">

                    <span>1 hour 14 min</span>

                    <span className="limit">+16</span>

                    <span>1999</span>

                </div>

                <div className="desc">

                    Lorem ipsum dolor, sit amet consectetur adipisicing elit.

                    Praesentium hic rem eveniet error possimus, neque ex doloribus.

                </div>

                <div className="gener">Action</div>

            </div>

            </>

            )}

        </div>

    )

}