**CODE:-**

**LOGIN:-**

import React, { useState } from 'react'

import styles from './Login.module.css'

import InputControl from "../InputControl/InputControl"

import { Link, useNavigate } from 'react-router-dom'

import { auth } from "../../firebase";

import { signInWithEmailAndPassword } from "firebase/auth";

const Login = () => {

const navigate = useNavigate();

const [values, setValues] = useState({

email: "",

pass: "",

});

const [errorMsg, setErrorMsg] = useState("");

const [submitButtonDisabled, setSubmitButtonDisabled] = useState(false);

const handleSubmission = () => {

if (!values.email || !values.pass) {

setErrorMsg("Fill all fields");

return;

}

setErrorMsg("");

setSubmitButtonDisabled(true);

signInWithEmailAndPassword(auth, values.email, values.pass)

.then(async (res) => {

setSubmitButtonDisabled(false);

navigate("/home");

})

.catch((err) => {

setSubmitButtonDisabled(false);

setErrorMsg(err.message);

});

};

return (

<div className={styles.container}>

<div className={styles.innerbox}>

<h1 className={styles.headeing}>LOGIN</h1>

<InputControl label="Email" placeholder='Enter Email Address'

onChange={(event) =>

setValues((prev) => ({ ...prev, email: event.target.value }))

}/>

<InputControl label="Password" placeholder='Enter Password'

onChange={(event) =>

setValues((prev) => ({ ...prev, pass: event.target.value }))

}/>

<div className={styles.footers}>

<b className={styles.error}>{errorMsg}</b>

<button disabled={submitButtonDisabled} onClick={handleSubmission}>Login</button>

<p>

Already have an account? {""}

<span>

<Link to="/Signup">Sign Up</Link>

</span>

</p>

</div>

</div>

</div>

)

}

export default Login

**LOGIN-STYLES:-**

.container{

height: 100%;

min-height: 100vh;

width: 100%;

background: linear-gradient(to right, #239ce2, #1369b9);

display: flex;

justify-content: center;

align-items: center;

}

.innerbox{

min-width: 480px;

height: fit-content;

width: fit-content;

background-color: #fff;

box-shadow: 1px 1px 4px rgba(0, 0, 0, 0.2);

padding: 30px;

border-radius: 10px;

display: flex;

flex-direction: column;

gap: 30px;

}

.footers{

display: flex;

flex-direction: column;

gap: 10px;

}

.footers .error {

font-weight: bold;

font-size: 0.875rem;

color: #ff3300;

text-align: center;

}

.footers button{

outline: none;

border: none;

background-color: #1369b9;

color: #fff;

border-radius: 5px;

font-weight: bold;

font-size: 1rem;

padding: 10px 16px;

width: 100%;

transition: 100ms;

cursor: pointer;

}

.footers button:disabled {

background-color: gray !important;

}

.footers button:hover{

background-color: #239ce2;

}

.footers p{

font-weight: 700;

color: #000;

}

.footers p span a{

font-weight: bold;

color: #1369b9;

letter-spacing: 1px;

font-size: 1rem;

text-decoration: none;

}

**SIGNUP:-**

import React, { useState } from 'react'

import InputControl from '../InputControl/InputControl'

import styles from './Signup.module.css'

import { Link, useNavigate } from 'react-router-dom'

import { createUserWithEmailAndPassword, updateProfile } from "firebase/auth";

import { auth } from "../../firebase";

const Signup = () => {

const navigate = useNavigate();

const [values, setValues] = useState({

name: "",

email: "",

pass: "",

});

const [errorMsg, setErrorMsg] = useState("");

const [submitButtonDisabled, setSubmitButtonDisabled] = useState(false);

const handleSubmission = () => {

if (!values.name || !values.email || !values.pass) {

setErrorMsg("Fill all fields");

return;

}

setErrorMsg("");

setSubmitButtonDisabled(true);

createUserWithEmailAndPassword(auth, values.email, values.pass)

.then(async (res) => {

setSubmitButtonDisabled(false);

const user = res.user;

await updateProfile(user, {

displayName: values.name,

});

navigate("/");

})

.catch((err) => {

setSubmitButtonDisabled(false);

setErrorMsg(err.message);

});

};

return (

<div className={styles.container}>

<div className={styles.innerbox}>

<h1 className={styles.headeing}>SIGN UP</h1>

<InputControl label="Name" placeholder='Enter Your Name'

onChange={(event) =>

setValues((prev) => ({ ...prev, name: event.target.value }))} />

<InputControl label="Email" placeholder='Enter Email Address'

onChange={(event) =>

setValues((prev) => ({ ...prev, email: event.target.value }))} />

<InputControl label="Password" placeholder='Enter Password'

onChange={(event) =>

setValues((prev) => ({ ...prev, pass: event.target.value }))} />

<div className={styles.footers}>

<b className={styles.error}>{errorMsg}</b>

<button onClick={handleSubmission} disabled={submitButtonDisabled}>Signup</button>

<p>

Already have an account? {" "}

<span>

<Link to="/">Login</Link>

</span>

</p>

</div>

</div>

</div>

)

}

export default Signup

**SIGNUP-STYLES:-**

.container{

height: 100%;

min-height: 100vh;

width: 100%;

background: linear-gradient(to right, #239ce2, #1369b9);

display: flex;

justify-content: center;

align-items: center;

}

.innerbox{

min-width: 480px;

height: fit-content;

width: fit-content;

background-color: #fff;

box-shadow: 1px 1px 4px rgba(0, 0, 0, 0.2);

padding: 30px;

border-radius: 10px;

display: flex;

flex-direction: column;

gap: 30px;

}

.footers{

display: flex;

flex-direction: column;

gap: 20px;

}

.footers .error {

font-weight: bold;

font-size: 0.875rem;

color: #ff3300;

text-align: center;

}

.footers button{

outline: none;

border: none;

background-color: #1369b9;

color: #fff;

border-radius: 5px;

font-weight: bold;

font-size: 1rem;

padding: 10px 16px;

width: 100%;

transition: 100ms;

cursor: pointer;

}

.footers button:disabled {

background-color: gray !important;

}

.footers button:hover{

background-color: #239ce2;

}

.footers p{

font-weight: 700;

color: #000;

}

.footers p span a{

font-weight: bold;

color: #1369b9;

letter-spacing: 1px;

font-size: 1rem;

text-decoration: none;

}

**NAV-BAR:-**

import React from 'react'

import './Nav.css'

import IMG1 from '../../assets/resumelogo.png'

import { Link } from "react-router-dom";

const Nav = () => {

return (

<header className="navbar1">

<img src={IMG1} alt="logo\_img" className='logo\_img1' />

<div className="navbar\_options">

<ul className='navbar\_menu'>

<li><Link to="/" className='btn' target="\_blank" rel="noopener noreferrer">LOGIN</Link></li>

<li><Link to="/Signup" className='btn'target="\_blank" rel="noopener noreferrer">SIGNUP</Link></li>

</ul>

</div>

</header>

)

}

export default Nav;

**NAV-BAR—STYLES:-**

.btn {

width: max-content;

display: inline-block;

color: #000;

padding: 0.75rem 1.2rem;

border-radius: 0.4rem;

cursor: pointer;

border: 1px solid #000;

transition: all 400ms ease;

text-decoration: none;

}

.btn:hover {

background: #000;

color: #fff;

border-color: transparent;

}

.navbar1 {

display: flex;

justify-content: space-between;

padding: 0 4rem;

background-color: #72bcf1;

padding: 0.8rem;

}

.navbar2 {

background: #000;

backdrop-filter: blur(15px);

z-index: 1;

}

.navbar\_menu li {

list-style: none;

}

.logo\_img1 {

width: 90px;

height: 80px;

border-radius: 20px;

}

.navbar\_options {

margin: 0 2rem;

}

.navbar\_menu {

display: flex;

flex-wrap: wrap;

gap: 0.6rem;

}

**BODY:-**

import React, { useRef, useState } from "react";

import ReactToPrint from "react-to-print";

import { ArrowDown } from "react-feather";

import Editor from "../Editor/Editor";

import Resume from "../Resume/Resume";

import Resume1 from "../Resume/Resume1"

import Resume2 from "../Resume/Resume2"

import Footer from "../Footer/Footer"

import styles from "./Body.module.css";

import { Tab, Tabs, TabList, TabPanel } from 'react-tabs';

import 'react-tabs/style/react-tabs.css';

function Body() {

const colors = ["#239ce2", "#48bb78", "#0bc5ea", "#a0aec0", "#ed8936"];

const sections = {

basicInfo: "Basic Info",

workExp: "Work Experience",

project: "Projects",

education: "Education",

achievement: "Achievements",

summary: "Summary",

other: "Other",

};

const resumeRef = useRef();

const [activeColor, setActiveColor] = useState(colors[0]);

const [resumeInformation, setResumeInformation] = useState({

[sections.basicInfo]: {

id: sections.basicInfo,

sectionTitle: sections.basicInfo,

detail: {},

},

[sections.workExp]: {

id: sections.workExp,

sectionTitle: sections.workExp,

details: [],

},

[sections.project]: {

id: sections.project,

sectionTitle: sections.project,

details: [],

},

[sections.education]: {

id: sections.education,

sectionTitle: sections.education,

details: [],

},

[sections.achievement]: {

id: sections.achievement,

sectionTitle: sections.achievement,

points: [],

},

[sections.summary]: {

id: sections.summary,

sectionTitle: sections.summary,

detail: "",

},

[sections.other]: {

id: sections.other,

sectionTitle: sections.other,

detail: "",

},

});

return (

<div>

<div className={styles.container}>

<p className={styles.heading}>Resume Builder</p>

<div className={styles.main}>

<Editor

sections={sections}

information={resumeInformation}

setInformation={setResumeInformation}

/>

</div>

<div className={styles.toolbar}>

<div className={styles.colors}>

{colors.map((item) => (

<span

key={item}

style={{ backgroundColor: item }}

className={`${styles.color} ${activeColor === item ? styles.active : ""

}`}

onClick={() => setActiveColor(item)}

/>

))}

</div>

<ReactToPrint

trigger={() => {

return (

<button>

Download <ArrowDown />

</button>

);

}}

content={() => resumeRef.current}

/>

</div>

<Tabs>

<TabList>

<Tab>Template 1</Tab>

<Tab>Template 2</Tab>

<Tab>Template 2</Tab>

</TabList>

<TabPanel>

<Resume

ref={resumeRef}

sections={sections}

information={resumeInformation}

activeColor={activeColor}

/>

</TabPanel>

<TabPanel>

<Resume1

ref={resumeRef}

sections={sections}

information={resumeInformation}

activeColor={activeColor}

/>

</TabPanel>

<TabPanel>

<Resume2

ref={resumeRef}

sections={sections}

information={resumeInformation}

activeColor={activeColor}

/>

</TabPanel>

</Tabs>

</div>

<Footer />

</div>

);

}

export default Body;

**BODY-STYLES:-**

.container {

padding: 30px;

display: flex;

flex-direction: column;

align-items: center;

gap: 30px;

padding-top: 0;

}

.heading {

font-weight: 500;

font-size: 2.1rem;

}

.toolbar {

width: 100%;

display: flex;

gap: 40px;

justify-content: space-between;

align-items: center;

}

.colors {

display: flex;

gap: 20px;

padding: 0 30px;

}

.colors .color {

height: 36px;

width: 36px;

border-radius: 50%;

background-color: #239ce2;

}

.colors .active {

border: 2px solid #000;

}

.toolbar button {

padding: 8px 16px;

border-radius: 5px;

background-color: #239ce2;

color: #fff;

outline: none;

border: none;

font-weight: 500;

font-size: 1rem;

letter-spacing: 1px;

display: flex;

gap: 5px;

align-items: center;

cursor: pointer;

}

.toolbar button svg {

height: 20px;

width: 20px;

}

.main {

display: flex;

flex-direction: column;

gap: 30px;

width: 100%;

}

**RESUME:-**

import React, { forwardRef, useEffect, useRef, useState } from "react";

import {

AtSign,

Calendar,

GitHub,

Linkedin,

MapPin,

Paperclip,

Phone,

} from "react-feather";

import styles from "./Resume.module.css";

const Resume = forwardRef((props, ref) => {

const information = props.information;

const sections = props.sections;

const containerRef = useRef();

const [columns, setColumns] = useState([[], []]);

const [source, setSource] = useState("");

const [target, seTarget] = useState("");

const info = {

workExp: information[sections.workExp],

project: information[sections.project],

achievement: information[sections.achievement],

education: information[sections.education],

basicInfo: information[sections.basicInfo],

summary: information[sections.summary],

other: information[sections.other],

};

const getFormattedDate = (value) => {

if (!value) return "";

const date = new Date(value);

return `${date.getDate()}/${date.getMonth() + 1}/${date.getFullYear()}`;

};

const sectionDiv = {

[sections.workExp]: (

<div

key={"workexp"}

draggable

onDragOver={() => seTarget(info.workExp?.id)}

onDragEnd={() => setSource(info.workExp?.id)}

className={`${styles.section} ${info.workExp?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.workExp.sectionTitle}</div>

<div className={styles.content}>

{info.workExp?.details?.map((item) => (

<div className={styles.item} key={item.title}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.companyName ? (

<p className={styles.subTitle}>{item.companyName}</p>

) : (

<span />

)}

{item.certificationLink ? (

<a className={styles.link} href={item.certificationLink}>

<Paperclip />

{item.certificationLink}

</a>

) : (

<span />

)}

{item.startDate && item.endDate ? (

<div className={styles.date}>

<Calendar /> {getFormattedDate(item.startDate)}-

{getFormattedDate(item.endDate)}

</div>

) : (

<div />

)}

{item.location ? (

<p className={styles.date}>

<MapPin /> {item.location}

</p>

) : (

<span />

)}

{item.points?.length > 0 ? (

<ul className={styles.points}>

{item.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

))}

</div>

</div>

),

[sections.project]: (

<div

key={"project"}

draggable

onDragOver={() => seTarget(info.project?.id)}

onDragEnd={() => setSource(info.project?.id)}

className={`${styles.section} ${info.project?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.project.sectionTitle}</div>

<div className={styles.content}>

{info.project?.details?.map((item) => (

<div className={styles.item}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.link ? (

<a className={styles.link} href={item.link}>

<Paperclip />

{item.link}

</a>

) : (

<span />

)}

{item.github ? (

<a className={styles.link} href={item.github}>

<GitHub />

{item.github}

</a>

) : (

<span />

)}

{item.overview ? (

<p className={styles.overview}>{item.overview} </p>

) : (

<span />

)}

{item.points?.length > 0 ? (

<ul className={styles.points}>

{item.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

))}

</div>

</div>

),

[sections.education]: (

<div

key={"education"}

draggable

onDragOver={() => seTarget(info.education?.id)}

onDragEnd={() => setSource(info.education?.id)}

className={`${styles.section} ${info.education?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>

{info.education?.sectionTitle}

</div>

<div className={styles.content}>

{info.education?.details?.map((item) => (

<div className={styles.item}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.college ? (

<p className={styles.subTitle}>{item.college}</p>

) : (

<span />

)}

{item.startDate && item.endDate ? (

<div className={styles.date}>

<Calendar /> {getFormattedDate(item.startDate)} -

{getFormattedDate(item.endDate)}

</div>

) : (

""

)}

</div>

))}

</div>

</div>

),

[sections.achievement]: (

<div

key={"achievement"}

draggable

onDragOver={() => seTarget(info.achievement?.id)}

onDragEnd={() => setSource(info.achievement?.id)}

className={`${styles.section} ${info.achievement?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>

{info.achievement?.sectionTitle}

</div>

<div className={styles.content}>

{info.achievement?.points?.length > 0 ? (

<ul className={styles.numbered}>

{info.achievement?.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

</div>

),

[sections.summary]: (

<div

key={"summary"}

draggable

onDragOver={() => seTarget(info.summary?.id)}

onDragEnd={() => setSource(info.summary?.id)}

className={`${styles.section} ${info.summary?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.summary?.sectionTitle}</div>

<div className={styles.content}>

<p className={styles.overview}>{info.summary?.detail}</p>

</div>

</div>

),

[sections.other]: (

<div

key={"other"}

draggable

onDragOver={() => seTarget(info.other?.id)}

onDragEnd={() => setSource(info.other?.id)}

className={`${styles.section} ${info.other?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.other?.sectionTitle}</div>

<div className={styles.content}>

<p className={styles.overview}>{info?.other?.detail}</p>

</div>

</div>

),

};

const swapSourceTarget = (source, target) => {

if (!source || !target) return;

const tempColumns = [[...columns[0]], [...columns[1]]];

let sourceRowIndex = tempColumns[0].findIndex((item) => item === source);

let sourceColumnIndex = 0;

if (sourceRowIndex < 0) {

sourceColumnIndex = 1;

sourceRowIndex = tempColumns[1].findIndex((item) => item === source);

}

let targetRowIndex = tempColumns[0].findIndex((item) => item === target);

let targetColumnIndex = 0;

if (targetRowIndex < 0) {

targetColumnIndex = 1;

targetRowIndex = tempColumns[1].findIndex((item) => item === target);

}

const tempSource = tempColumns[sourceColumnIndex][sourceRowIndex];

tempColumns[sourceColumnIndex][sourceRowIndex] =

tempColumns[targetColumnIndex][targetRowIndex];

tempColumns[targetColumnIndex][targetRowIndex] = tempSource;

setColumns(tempColumns);

};

useEffect(() => {

setColumns([

[sections.project, sections.education, sections.summary],

[sections.workExp, sections.achievement, sections.other],

]);

}, []);

useEffect(() => {

swapSourceTarget(source, target);

}, [source]);

useEffect(() => {

const container = containerRef.current;

if (!props.activeColor || !container) return;

container.style.setProperty("--color", props.activeColor);

}, [props.activeColor]);

return (

<div ref={ref}>

<div ref={containerRef} className={styles.container}>

<div className={styles.header}>

<p className={styles.heading}>{info.basicInfo?.detail?.name}</p>

<p className={styles.subHeading}>{info.basicInfo?.detail?.title}</p>

<div className={styles.links}>

{info.basicInfo?.detail?.email ? (

<a className={styles.link} type="email">

<AtSign /> {info.basicInfo?.detail?.email}

</a>

) : (

<span />

)}

{info.basicInfo?.detail?.phone ? (

<a className={styles.link}>

<Phone /> {info.basicInfo?.detail?.phone}

</a>

) : (

<span />

)}

{info.basicInfo?.detail?.linkedin ? (

<a className={styles.link}>

<Linkedin /> {info.basicInfo?.detail?.linkedin}

</a>

) : (

<span />

)}

{info.basicInfo?.detail?.github ? (

<a className={styles.link}>

<GitHub /> {info.basicInfo?.detail?.github}

</a>

) : (

<span />

)}

</div>

</div>

<div className={styles.main}>

<div className={styles.col1}>

{columns[0].map((item) => sectionDiv[item])}

</div>

<div className={styles.col2}>

{columns[1].map((item) => sectionDiv[item])}

</div>

</div>

</div>

</div>

);

});

export default Resume;

**RESUME-STYLES:-**

.container {

--color: #239ce2;

min-width: 700px;

max-width: 850px;

margin: 0 auto;

flex: 1.2;

height: fit-content;

min-height: 900px;

box-shadow: 1px 1px 3px 2px rgba(0, 0, 0, 0.1);

display: flex;

flex-direction: column;

gap: 30px;

padding: 30px;

}

.header {

display: flex;

flex-direction: column;

}

.header .heading {

font-size: 2.7rem;

font-weight: 500;

text-transform: capitalize;

}

.header .subHeading {

color: var(--color);

font-weight: 500;

font-size: 1.1rem;

}

.header .links {

margin-top: 15px;

display: flex;

gap: 30px;

flex-wrap: wrap;

row-gap: 10px;

}

.header .link {

font-size: 0.875rem;

display: flex;

align-items: center;

gap: 5px;

cursor: pointer;

}

.header .link svg {

color: var(--color);

height: 16px;

width: 16px;

}

.main {

display: flex;

gap: 30px;

}

.col1 {

flex: 1.3;

display: flex;

flex-direction: column;

gap: 20px;

}

.col2 {

flex: 1;

display: flex;

flex-direction: column;

gap: 20px;

}

.section .sectionTitle {

font-size: 1.4rem;

font-weight: bold;

width: 100%;

border-bottom: 2px solid #000;

}

.section .content {

display: flex;

flex-direction: column;

gap: 10px;

padding: 10px 0;

}

.section .content .item {

border-bottom: 1px dotted lightgray;

display: flex;

flex-direction: column;

gap: 4px;

padding-bottom: 10px;

}

.section .content .item:last-child {

border-bottom: none;

}

.content .title {

font-weight: 500;

font-size: 1rem;

line-height: 1.3rem;

}

.content .subTitle {

font-weight: bold;

color: var(--color);

font-size: 1rem;

}

.content .overview {

font-size: 0.875rem;

}

.content .link {

display: flex;

gap: 5px;

font-size: 0.75rem;

cursor: pointer;

color: var(--color);

}

.content .link svg {

height: 14px;

width: 14px;

color: var(--color);

}

.content .date {

display: flex;

gap: 5px;

align-items: center;

font-size: 0.875rem;

}

.content .date svg {

height: 14px;

width: 14px;

}

.content .points,

.numbered {

padding-left: 18px;

font-size: 0.875rem;

}

.numbered li {

list-style-type: decimal;

text-align: start;

}

.hidden {

display: none;

}

.points {

text-align: start;

}

**RESUME 1:-**

import React, { forwardRef, useEffect, useRef, useState } from "react";

import {

AtSign,

Calendar,

GitHub,

Linkedin,

MapPin,

Paperclip,

Phone,

} from "react-feather";

import styles from "./Resume1.module.css";

const Resume1 = forwardRef((props, ref) => {

const information = props.information;

const sections = props.sections;

const containerRef = useRef();

const [columns, setColumns] = useState([[], []]);

const [source, setSource] = useState("");

const [target, seTarget] = useState("");

const info = {

workExp: information[sections.workExp],

project: information[sections.project],

achievement: information[sections.achievement],

education: information[sections.education],

basicInfo: information[sections.basicInfo],

summary: information[sections.summary],

other: information[sections.other],

};

const getFormattedDate = (value) => {

if (!value) return "";

const date = new Date(value);

return `${date.getDate()}/${date.getMonth() + 1}/${date.getFullYear()}`;

};

const sectionDiv = {

[sections.workExp]: (

<div

key={"workexp"}

draggable

onDragOver={() => seTarget(info.workExp?.id)}

onDragEnd={() => setSource(info.workExp?.id)}

className={`${styles.section} ${info.workExp?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.workExp.sectionTitle}</div>

<div className={styles.content}>

{info.workExp?.details?.map((item) => (

<div className={styles.item} key={item.title}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.companyName ? (

<p className={styles.subTitle}>{item.companyName}</p>

) : (

<span />

)}

{item.certificationLink ? (

<a className={styles.link} href={item.certificationLink}>

<Paperclip />

{item.certificationLink}

</a>

) : (

<span />

)}

{item.startDate && item.endDate ? (

<div className={styles.date}>

<Calendar /> {getFormattedDate(item.startDate)}-

{getFormattedDate(item.endDate)}

</div>

) : (

<div />

)}

{item.location ? (

<p className={styles.date}>

<MapPin /> {item.location}

</p>

) : (

<span />

)}

{item.points?.length > 0 ? (

<ul className={styles.points}>

{item.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

))}

</div>

</div>

),

[sections.project]: (

<div

key={"project"}

draggable

onDragOver={() => seTarget(info.project?.id)}

onDragEnd={() => setSource(info.project?.id)}

className={`${styles.section} ${info.project?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.project.sectionTitle}</div>

<div className={styles.content}>

{info.project?.details?.map((item) => (

<div className={styles.item}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.link ? (

<a className={styles.link} href={item.link}>

<Paperclip />

{item.link}

</a>

) : (

<span />

)}

{item.github ? (

<a className={styles.link} href={item.github}>

<GitHub />

{item.github}

</a>

) : (

<span />

)}

{item.overview ? (

<p className={styles.overview}>{item.overview} </p>

) : (

<span />

)}

{item.points?.length > 0 ? (

<ul className={styles.points}>

{item.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

))}

</div>

</div>

),

[sections.education]: (

<div

key={"education"}

draggable

onDragOver={() => seTarget(info.education?.id)}

onDragEnd={() => setSource(info.education?.id)}

className={`${styles.section} ${info.education?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>

{info.education?.sectionTitle}

</div>

<div className={styles.content}>

{info.education?.details?.map((item) => (

<div className={styles.item}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.college ? (

<p className={styles.subTitle}>{item.college}</p>

) : (

<span />

)}

{item.startDate && item.endDate ? (

<div className={styles.date}>

<Calendar /> {getFormattedDate(item.startDate)} -

{getFormattedDate(item.endDate)}

</div>

) : (

""

)}

</div>

))}

</div>

</div>

),

[sections.achievement]: (

<div

key={"achievement"}

draggable

onDragOver={() => seTarget(info.achievement?.id)}

onDragEnd={() => setSource(info.achievement?.id)}

className={`${styles.section} ${info.achievement?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>

{info.achievement?.sectionTitle}

</div>

<div className={styles.content}>

{info.achievement?.points?.length > 0 ? (

<ul className={styles.numbered}>

{info.achievement?.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

</div>

),

[sections.summary]: (

<div

key={"summary"}

draggable

onDragOver={() => seTarget(info.summary?.id)}

onDragEnd={() => setSource(info.summary?.id)}

className={`${styles.section} ${info.summary?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.summary?.sectionTitle}</div>

<div className={styles.content}>

<p className={styles.overview}>{info.summary?.detail}</p>

</div>

</div>

),

[sections.other]: (

<div

key={"other"}

draggable

onDragOver={() => seTarget(info.other?.id)}

onDragEnd={() => setSource(info.other?.id)}

className={`${styles.section} ${info.other?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.other?.sectionTitle}</div>

<div className={styles.content}>

<p className={styles.overview}>{info?.other?.detail}</p>

</div>

</div>

),

};

const swapSourceTarget = (source, target) => {

if (!source || !target) return;

const tempColumns = [[...columns[0]], [...columns[1]]];

let sourceRowIndex = tempColumns[0].findIndex((item) => item === source);

let sourceColumnIndex = 0;

if (sourceRowIndex < 0) {

sourceColumnIndex = 1;

sourceRowIndex = tempColumns[1].findIndex((item) => item === source);

}

let targetRowIndex = tempColumns[0].findIndex((item) => item === target);

let targetColumnIndex = 0;

if (targetRowIndex < 0) {

targetColumnIndex = 1;

targetRowIndex = tempColumns[1].findIndex((item) => item === target);

}

const tempSource = tempColumns[sourceColumnIndex][sourceRowIndex];

tempColumns[sourceColumnIndex][sourceRowIndex] =

tempColumns[targetColumnIndex][targetRowIndex];

tempColumns[targetColumnIndex][targetRowIndex] = tempSource;

setColumns(tempColumns);

};

useEffect(() => {

setColumns([

[sections.project, sections.education, sections.summary],

[sections.workExp, sections.achievement, sections.other],

]);

}, []);

useEffect(() => {

swapSourceTarget(source, target);

}, [source]);

useEffect(() => {

const container = containerRef.current;

if (!props.activeColor || !container) return;

container.style.setProperty("--color", props.activeColor);

}, [props.activeColor]);

return (

<div ref={ref}>

<div ref={containerRef} className={styles.container}>

<div className={styles.header}>

<p className={styles.heading}>{info.basicInfo?.detail?.name}</p>

<p className={styles.subHeading}>{info.basicInfo?.detail?.title}</p>

<div className={styles.links}>

{info.basicInfo?.detail?.email ? (

<a className={styles.link} type="email">

<AtSign /> {info.basicInfo?.detail?.email}

</a>

) : (

<span />

)}

{info.basicInfo?.detail?.phone ? (

<a className={styles.link}>

<Phone /> {info.basicInfo?.detail?.phone}

</a>

) : (

<span />

)}

{info.basicInfo?.detail?.linkedin ? (

<a className={styles.link}>

<Linkedin /> {info.basicInfo?.detail?.linkedin}

</a>

) : (

<span />

)}

{info.basicInfo?.detail?.github ? (

<a className={styles.link}>

<GitHub /> {info.basicInfo?.detail?.github}

</a>

) : (

<span />

)}

</div>

</div>

<div className={styles.main}>

<div className={styles.col1}>

{columns[0].map((item) => sectionDiv[item])}

</div>

<div className={styles.col2}>

{columns[1].map((item) => sectionDiv[item])}

</div>

</div>

</div>

</div>

);

});

export default Resume1;

**RESUME 1-STYLES:-**

.container {

--color: #239ce2;

min-width: 700px;

max-width: 850px;

margin: 0 auto;

flex: 1.2;

height: fit-content;

min-height: 900px;

box-shadow: 1px 1px 3px 2px rgba(0, 0, 0, 0.1);

display: flex;

flex-direction: column;

gap: 30px;

padding: 30px;

}

.header {

display: flex;

flex-direction: column;

}

.header .heading {

font-size: 3.7rem;

font-weight: 600;

text-transform: capitalize;

text-align: start;

font-family: serif;

letter-spacing: 2px;

margin-bottom: 4px;

margin-top: 10px;

}

.header .subHeading {

color: var(--color);

font-weight: 500;

font-size: 1.3rem;

text-align: start;

padding-left: 2px;

margin-bottom: 2rem;

}

.header .links {

margin-top: 15px;

display: flex;

gap: 30px;

flex-wrap: wrap;

row-gap: 10px;

}

.header .link {

font-size: 0.875rem;

display: flex;

align-items: center;

gap: 5px;

cursor: pointer;

}

.header .link svg {

color: var(--color);

height: 16px;

width: 16px;

}

.main {

display: flex;

gap: 30px;

}

.col1 {

flex: 1.3;

display: flex;

flex-direction: column;

gap: 20px;

}

.col2 {

flex: 1.3;

display: flex;

flex-direction: column;

gap: 20px;

}

.section .sectionTitle {

font-size: 1.7rem;

font-weight: bold;

width: 100%;

/\* border-bottom: 2px solid #000; \*/

text-align: start;

}

.section .content {

display: flex;

flex-direction: column;

gap: 10px;

padding: 10px 0;

}

.section .content .item {

border-bottom: 1px dotted lightgray;

display: flex;

flex-direction: column;

gap: 4px;

padding-bottom: 10px;

}

.section .content .item:last-child {

border-bottom: none;

}

.content .title {

font-weight: 500;

font-size: 1rem;

line-height: 1.3rem;

text-align: start;

margin-bottom: 2px;

}

.content .subTitle {

font-weight: bold;

color: var(--color);

font-size: 1rem;

text-align: start;

margin-top: 0.8rem;

}

.content .overview {

font-size: 0.875rem;

text-align: start;

}

.content .link {

display: flex;

gap: 5px;

font-size: 0.75rem;

cursor: pointer;

color: var(--color);

}

.content .link svg {

height: 14px;

width: 14px;

color: var(--color);

}

.content .date {

display: flex;

gap: 5px;

align-items: center;

font-size: 0.875rem;

}

.content .date svg {

height: 14px;

width: 14px;

}

.content .points,

.numbered {

padding-left: 18px;

font-size: 0.875rem;

margin-top: 3px;

}

.numbered li {

list-style-type: decimal;

text-align: start;

}

.hidden {

display: none;

}

.points {

text-align: start;

}

**RESUME 2:-**

import React, { forwardRef, useEffect, useRef, useState } from "react";

import {

AtSign,

Calendar,

GitHub,

Linkedin,

MapPin,

Paperclip,

Phone,

} from "react-feather";

import styles from "./Resume2.module.css";

const Resume2 = forwardRef((props, ref) => {

const information = props.information;

const sections = props.sections;

const containerRef = useRef();

const [columns, setColumns] = useState([[], []]);

const [source, setSource] = useState("");

const [target, seTarget] = useState("");

const info = {

workExp: information[sections.workExp],

project: information[sections.project],

achievement: information[sections.achievement],

education: information[sections.education],

basicInfo: information[sections.basicInfo],

summary: information[sections.summary],

other: information[sections.other],

};

const getFormattedDate = (value) => {

if (!value) return "";

const date = new Date(value);

return `${date.getDate()}/${date.getMonth() + 1}/${date.getFullYear()}`;

};

const sectionDiv = {

[sections.workExp]: (

<div

key={"workexp"}

draggable

onDragOver={() => seTarget(info.workExp?.id)}

onDragEnd={() => setSource(info.workExp?.id)}

className={`${styles.section} ${info.workExp?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.workExp.sectionTitle}</div>

<div className={styles.content}>

{info.workExp?.details?.map((item) => (

<div className={styles.item} key={item.title}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.companyName ? (

<p className={styles.subTitle}>{item.companyName}</p>

) : (

<span />

)}

{item.certificationLink ? (

<a className={styles.link} href={item.certificationLink}>

<Paperclip />

{item.certificationLink}

</a>

) : (

<span />

)}

{item.startDate && item.endDate ? (

<div className={styles.date}>

<Calendar /> {getFormattedDate(item.startDate)}-

{getFormattedDate(item.endDate)}

</div>

) : (

<div />

)}

{item.location ? (

<p className={styles.date}>

<MapPin /> {item.location}

</p>

) : (

<span />

)}

{item.points?.length > 0 ? (

<ul className={styles.points}>

{item.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

))}

</div>

</div>

),

[sections.project]: (

<div

key={"project"}

draggable

onDragOver={() => seTarget(info.project?.id)}

onDragEnd={() => setSource(info.project?.id)}

className={`${styles.section} ${info.project?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.project.sectionTitle}</div>

<div className={styles.content}>

{info.project?.details?.map((item) => (

<div className={styles.item}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.link ? (

<a className={styles.link} href={item.link}>

<Paperclip />

{item.link}

</a>

) : (

<span />

)}

{item.github ? (

<a className={styles.link} href={item.github}>

<GitHub />

{item.github}

</a>

) : (

<span />

)}

{item.overview ? (

<p className={styles.overview}>{item.overview} </p>

) : (

<span />

)}

{item.points?.length > 0 ? (

<ul className={styles.points}>

{item.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

))}

</div>

</div>

),

[sections.education]: (

<div

key={"education"}

draggable

onDragOver={() => seTarget(info.education?.id)}

onDragEnd={() => setSource(info.education?.id)}

className={`${styles.section} ${info.education?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>

{info.education?.sectionTitle}

</div>

<div className={styles.content}>

{info.education?.details?.map((item) => (

<div className={styles.item}>

{item.title ? (

<p className={styles.title}>{item.title}</p>

) : (

<span />

)}

{item.college ? (

<p className={styles.subTitle}>Some college name</p>

) : (

<span />

)}

{item.startDate && item.endDate ? (

<div className={styles.date}>

<Calendar /> {getFormattedDate(item.startDate)} -

{getFormattedDate(item.endDate)}

</div>

) : (

""

)}

</div>

))}

</div>

</div>

),

[sections.achievement]: (

<div

key={"achievement"}

draggable

onDragOver={() => seTarget(info.achievement?.id)}

onDragEnd={() => setSource(info.achievement?.id)}

className={`${styles.section} ${info.achievement?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>

{info.achievement?.sectionTitle}

</div>

<div className={styles.content}>

{info.achievement?.points?.length > 0 ? (

<ul className={styles.numbered}>

{info.achievement?.points?.map((elem, index) => (

<li className={styles.point} key={elem + index}>

{elem}

</li>

))}

</ul>

) : (

<span />

)}

</div>

</div>

),

[sections.summary]: (

<div

key={"summary"}

draggable

onDragOver={() => seTarget(info.summary?.id)}

onDragEnd={() => setSource(info.summary?.id)}

className={`${styles.section} ${info.summary?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.summary?.sectionTitle}</div>

<div className={styles.content}>

<p className={styles.overview}>{info.summary?.detail}</p>

</div>

</div>

),

[sections.other]: (

<div

key={"other"}

draggable

onDragOver={() => seTarget(info.other?.id)}

onDragEnd={() => setSource(info.other?.id)}

className={`${styles.section} ${info.other?.sectionTitle ? "" : styles.hidden

}`}

>

<div className={styles.sectionTitle}>{info.other?.sectionTitle}</div>

<div className={styles.content}>

<p className={styles.overview}>{info?.other?.detail}</p>

</div>

</div>

),

};

const swapSourceTarget = (source, target) => {

if (!source || !target) return;

const tempColumns = [[...columns[0]], [...columns[1]]];

let sourceRowIndex = tempColumns[0].findIndex((item) => item === source);

let sourceColumnIndex = 0;

if (sourceRowIndex < 0) {

sourceColumnIndex = 1;

sourceRowIndex = tempColumns[1].findIndex((item) => item === source);

}

let targetRowIndex = tempColumns[0].findIndex((item) => item === target);

let targetColumnIndex = 0;

if (targetRowIndex < 0) {

targetColumnIndex = 1;

targetRowIndex = tempColumns[1].findIndex((item) => item === target);

}

const tempSource = tempColumns[sourceColumnIndex][sourceRowIndex];

tempColumns[sourceColumnIndex][sourceRowIndex] =

tempColumns[targetColumnIndex][targetRowIndex];

tempColumns[targetColumnIndex][targetRowIndex] = tempSource;

setColumns(tempColumns);

};

useEffect(() => {

setColumns([

[sections.project, sections.education, sections.summary],

[sections.workExp, sections.achievement, sections.other],

]);

}, []);

useEffect(() => {

swapSourceTarget(source, target);

}, [source]);

useEffect(() => {

const container = containerRef.current;

if (!props.activeColor || !container) return;

container.style.setProperty("--color", props.activeColor);

}, [props.activeColor]);

return (

<div ref={ref}>

<div ref={containerRef} className={styles.container}>

<div className={styles.header}>

<div className={styles.header2}>

<p className={styles.heading}>{info.basicInfo?.detail?.name}</p>

<p className={styles.subHeading}>{info.basicInfo?.detail?.title}</p>

</div>

<div className={styles.links}>

<ul>

<li>{info.basicInfo?.detail?.email ? (

<a className={styles.link} type="email">

<AtSign /> {info.basicInfo?.detail?.email}

</a>

) : (

<span />

)}</li>

<li> {info.basicInfo?.detail?.phone ? (

<a className={styles.link}>

<Phone /> {info.basicInfo?.detail?.phone}

</a>

) : (

<span />

)}</li>

<li>{info.basicInfo?.detail?.linkedin ? (

<a className={styles.link}>

<Linkedin /> {info.basicInfo?.detail?.linkedin}

</a>

) : (

<span />

)}</li>

<li>{info.basicInfo?.detail?.github ? (

<a className={styles.link}>

<GitHub /> {info.basicInfo?.detail?.github}

</a>

) : (

<span />

)}</li>

</ul>

</div>

</div>

<div className={styles.main}>

<div className={styles.col1}>

{columns[0].map((item) => sectionDiv[item])}

</div>

<div className={styles.col2}>

{columns[1].map((item) => sectionDiv[item])}

</div>

</div>

</div>

</div>

);

});

export default Resume2;

**RESUME 2-STYLES:-**

.container {

--color: #239ce2;

min-width: 700px;

max-width: 850px;

margin: 0 auto;

flex: 1.2;

height: fit-content;

min-height: 900px;

box-shadow: 1px 1px 3px 2px rgba(0, 0, 0, 0.1);

display: flex;

flex-direction: column;

gap: 30px;

padding: 30px;

}

.header {

display: flex;

/\* flex-direction: column; \*/

justify-content: space-between;

}

.header .heading {

font-size: 2.7rem;

font-weight: 500;

text-transform: capitalize;

margin-bottom: 5px;

}

.header .subHeading {

color: var(--color);

font-weight: 500;

font-size: 1.1rem;

}

.header .links {

margin-top: 15px;

display: flex;

gap: 30px;

flex-wrap: wrap;

row-gap: 10px;

}

.header .link {

font-size: 0.875rem;

display: flex;

align-items: center;

gap: 5px;

cursor: pointer;

padding: 5px;

}

.header .link svg {

color: var(--color);

height: 16px;

width: 16px;

}

.main {

display: flex;

gap: 30px;

}

.col1 {

flex: 1.3;

display: flex;

flex-direction: column;

gap: 20px;

}

.col2 {

flex: 1;

display: flex;

flex-direction: column;

gap: 20px;

}

.section .sectionTitle {

font-size: 1.4rem;

font-weight: bold;

width: 100%;

border-bottom: 2px solid #000;

}

.section .content {

display: flex;

flex-direction: column;

gap: 10px;

padding: 10px 0;

}

.section .content .item {

border-bottom: 1px dotted lightgray;

display: flex;

flex-direction: column;

gap: 4px;

padding-bottom: 10px;

}

.section .content .item:last-child {

border-bottom: none;

}

.content .title {

font-weight: 500;

font-size: 1rem;

line-height: 1.3rem;

}

.content .subTitle {

font-weight: bold;

color: var(--color);

font-size: 1rem;

}

.content .overview {

font-size: 0.875rem;

}

.content .link {

display: flex;

gap: 5px;

font-size: 0.75rem;

cursor: pointer;

color: var(--color);

}

.content .link svg {

height: 14px;

width: 14px;

color: var(--color);

}

.content .date {

display: flex;

gap: 5px;

align-items: center;

font-size: 0.875rem;

}

.content .date svg {

height: 14px;

width: 14px;

}

.content .points,

.numbered {

padding-left: 18px;

font-size: 0.875rem;

}

.numbered li {

list-style-type: decimal;

text-align: start;

}

.hidden {

display: none;

}

.points {

text-align: start;

}

.links li {

padding: 2px;

list-style: none;

}

ul {

padding-left: 10px;

}

**Header:-**

import React from "react";

import resumeSvg from "../../assets/undraw\_resume.svg";

import styles from "./Header.module.css";

import Nav from "../Nav/Nav";

import Body from "../Body/Body"

function Header() {

return (

<div>

<Nav/>

<div className={styles.container}>

<div className={styles.left}>

<p className={styles.heading}>

A <span>Resume</span> that stands out!

</p>

<p className={styles.heading}>

Make your own resume. <span>It's free</span>

</p>

</div>

<div className={styles.right}>

<img src={resumeSvg} alt="Resume" />

</div>

</div>

<Body />

</div>

);

}

export default Header;

**Footer:-**

import React from 'react'

// import BsYoutube from 'react-icons/bs'

import './Footer.css'

const Footer = () => {

return (

<section id="footer">

<div className="footer\_container">

<div className="row">

<div className="copy-r">

<p>&copy; 2022 - all rights reserved, Resume-Builder.</p>

</div>

</div>

</div>

</section>

)

}

export default Footer