AlphCons

Alphcons React One Page Template

Alphcons documentation version 1.0

- Working with links

Basic information

- **Template Name :** Alphcons React One Page Responsive Template
- **Template Version :** v 1.0
- **Author :** UI2Fly
- **Template URL :** https://alphcons.netlify.app

Local Development #back to top

FOLLOW THE NEXT STEPS TO START USING THIS TEMPLATE:

- Install NodeJs from NodeJs Official Page
- Unzip the downloaded file to a folder in your computer
- Open Terminal
- Go to your file project
- Run in terminal:

  ```
  npm install
  ```

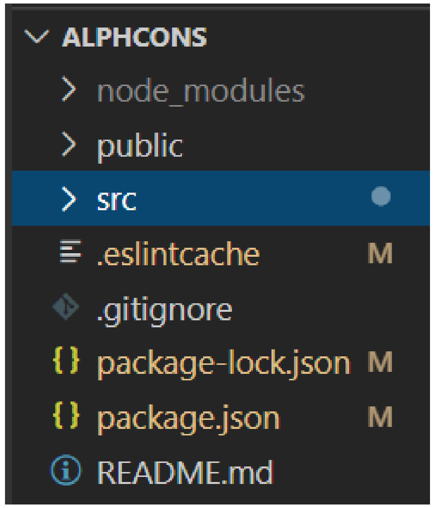
- Then run:

  ```
  npm start
  ```

  This will install the node_modules and start the project.

Structure of Template #back to top

Folder Structure



NODE_MODULES

Contains all the dependencies that are needed for an initial working react app

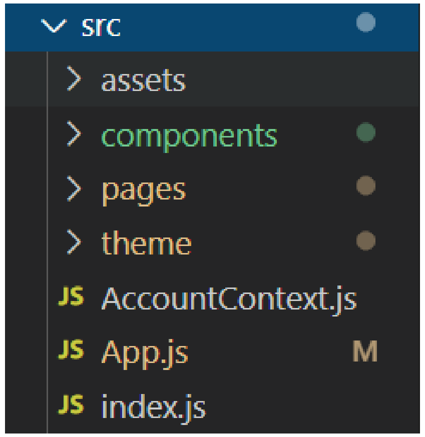
PACKAGE.JSON

This file contains various metadata that is relevant to the project. It specifies the dependencies being used in the project which helps node package manager setup the same environment for our project on different machine.

PUBLIC FOLDER

It is the root folder that gets served up as the react app.

SOURCE FOLDER (SRC)



It is the brain of our application, where everything happens. Basically, this folder contains the components, pages, and assets of the project.

- 🗁 **assets contains the images and videos used in the templates. Here you can add your custom assets.**
- 🗁 **components contains all the editable components of our project.**
- 🗁 **pages contains the pages of the project**
- 🗁 **theme contains the global styles of the project**
- 🗋 **AccountContext.js contains the context object**
- 🗋 **App.js renders the template pages**

- 🗋 **index.js provide entry points for components. Its code says what to render and where to render.**

Fonts #back to top

In the **public/index.html**, within the head section we have the fonts used in the template

```
<link href="https://fonts.googleapis.com/css2?
family=IBM+Plex+Sans:ital,wght@0,200;0,300;0,400;0,500;0,600;0,700;1,200;1,300;1,400;1,500;1,600;1,700&display=swap"
rel="stylesheet">
```

The font-family is called in **src/theme/GlobalStyle.js**

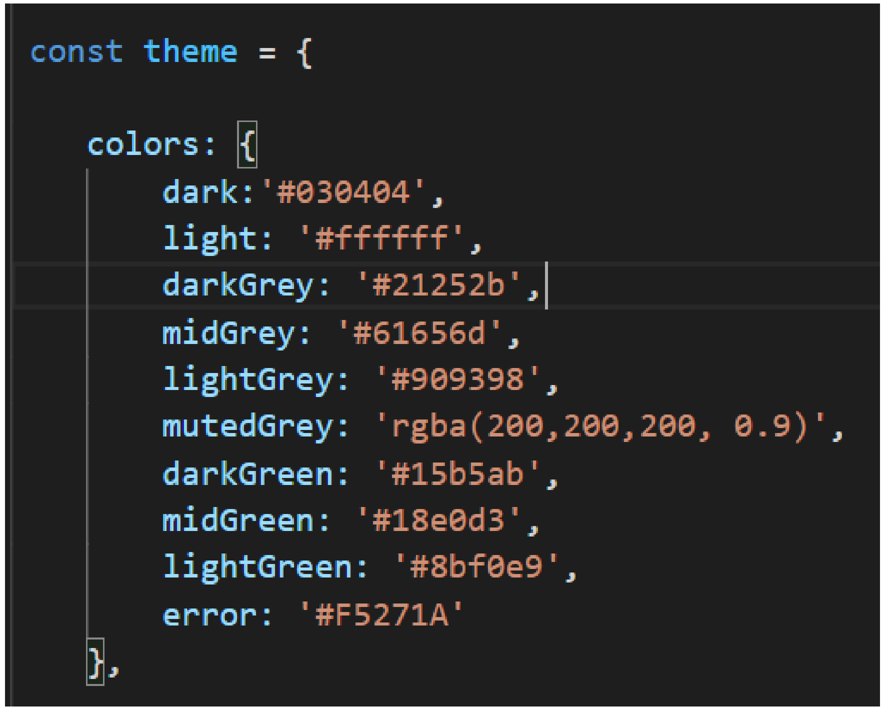
```
*{ font-family: "IBM Plex Sans",sans-serif; }
```

Template Customization #back to top

THEME GENERAL STYLES - #BACK TO TOP

You customize the global styles in the **src/theme/GlobalStyle.js**

You can customize the theme general styles such as colors, fontSizes, transition, border radius, in the **src/theme/theme.js.** Feel free to add your own properties and values.
Here is an usage example: e.g. for *colors* property, in theme.js we have the values for each color:



```
const theme = {

    colors: {
        dark:'#030404',
        light: '#ffffff',
        darkGrey: '#21252b',
        midGrey: '#61656d',
        lightGrey: '#909398',
        mutedGrey: 'rgba(200,200,200, 0.9)',
        darkGreen: '#15b5ab',
        midGreen: '#18e0d3',
        lightGreen: '#8bf0e9',
        error: '#F5271A'
    },
```

And here is how you call it in styled components: `color: ${(props) => props.theme.colors.dark};`

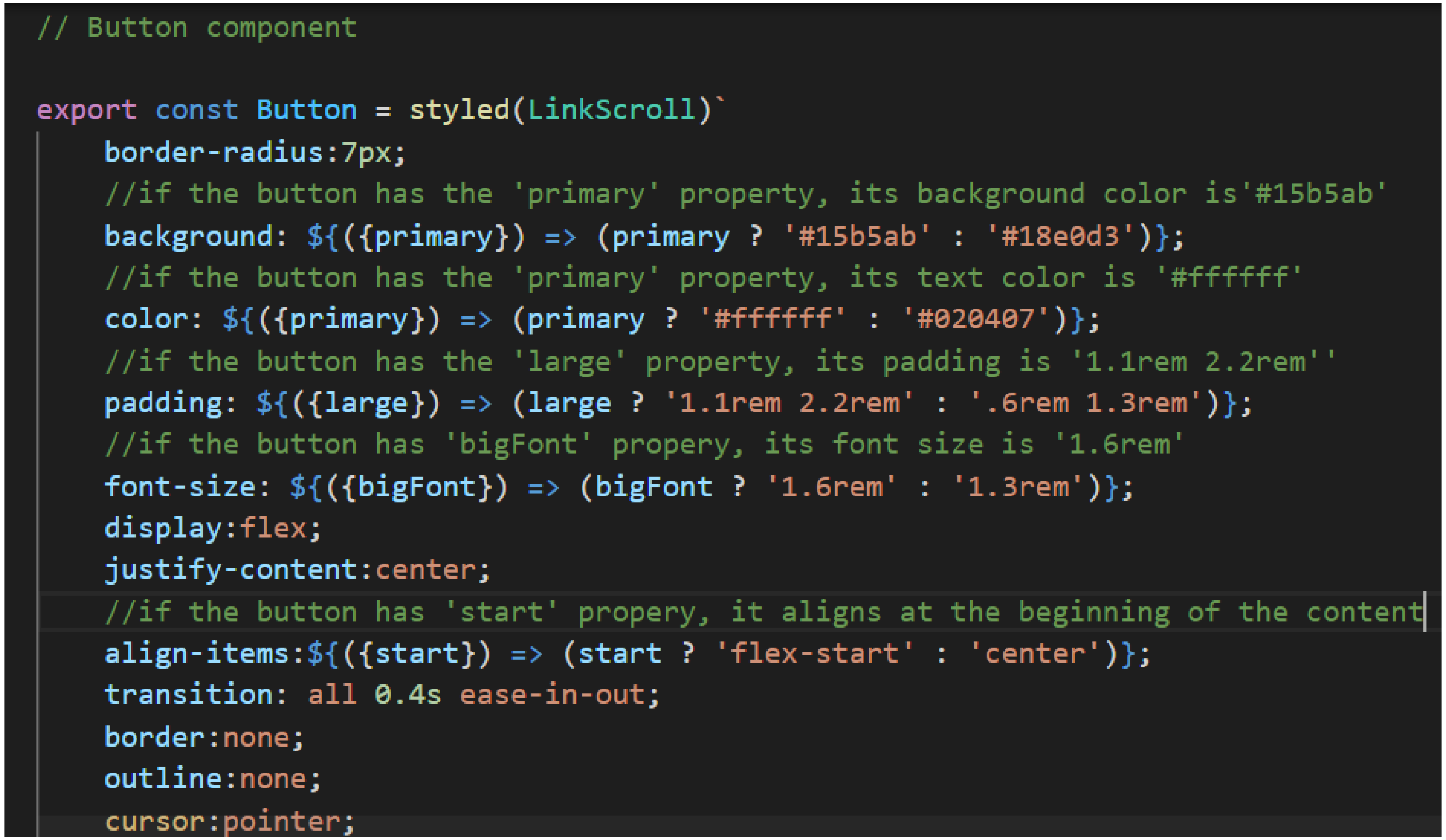
HOW TO CHANGE THE LOGO - #BACK TO TOP

You can change the Logo in the **src/components/Navbar/index.js** in the lines:

```
import siteLogo from './../../assets/logo.png'
```

BUTTON COMPONENT - #BACK TO TOP

There is a default Button component, which we call in few components. You find it in **src/components/ButtonComponent.js**. Feel free to customize its style values:

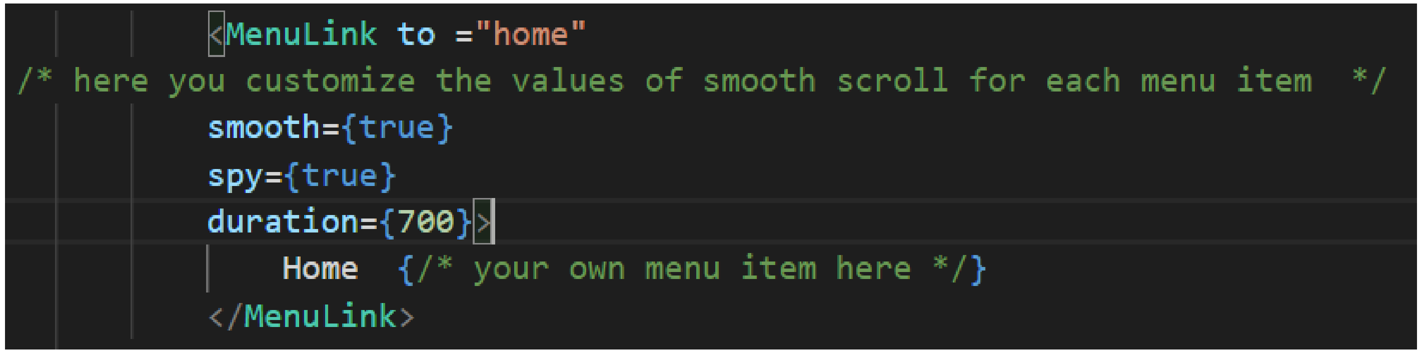


```
// Button component

export const Button = styled(LinkScroll)`
    border-radius:7px;
    //if the button has the 'primary' property, its background color is'#15b5ab'
    background: ${({primary}) => (primary ? '#15b5ab' : '#18e0d3')};
    //if the button has the 'primary' property, its text color is '#ffffff'
    color: ${({primary}) => (primary ? '#ffffff' : '#020407')};
    //if the button has the 'large' property, its padding is '1.1rem 2.2rem''
    padding: ${({large}) => (large ? '1.1rem 2.2rem' : '.6rem 1.3rem')};
    //if the button has 'bigFont' propery, its font size is '1.6rem'
    font-size: ${({bigFont}) => (bigFont ? '1.6rem' : '1.3rem')};
    display:flex;
    justify-content:center;
    //if the button has 'start' propery, it aligns at the beginning of the content
    align-items:${({start}) => (start ? 'flex-start' : 'center')};
    transition: all 0.4s ease-in-out;
    border:none;
    outline:none;
    cursor:pointer;
```

NAVIGATION COMPONENT- #BACK TO TOP

You customize the Navigation in the **src/components/Navbar**. There are 2 files: **index.js** , which handles the logic of the component, and **NavbarComponents.js** (styled components), which handles the style.

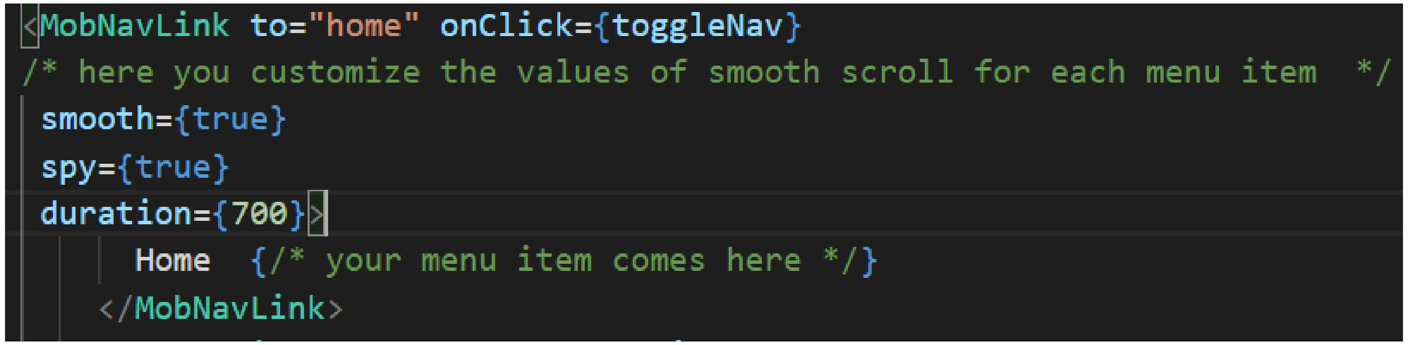
In **src/components/Navbar/index.js** you add your own menu item:



```
        <MenuLink to ="home"
/* here you customize the values of smooth scroll for each menu item  */
        smooth={true}
        spy={true}
        duration={700}>
            Home  {/* your own menu item here */}
        </MenuLink>
```

You customize the MobileNavigation in the **src/components/MobNavbar**. There are 2 files: **index.js**, which handles the logic of the component, and **MobNavbarComponents.js** (styled components), which handles the style.

In **src/components/MobNavbar/index.js** you add your own menu item:



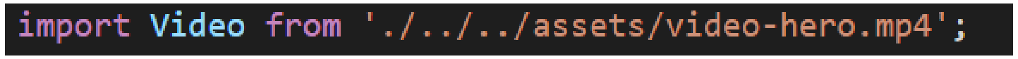
```
<MobNavLink to="home" onClick={toggleNav}
/* here you customize the values of smooth scroll for each menu item  */
 smooth={true}
 spy={true}
 duration={700}>
      Home  {/* your menu item comes here */}
    </MobNavLink>
```

Each styled component is well commented, so you can change the style according to your needs.

HERO COMPONENT - #BACK TO TOP

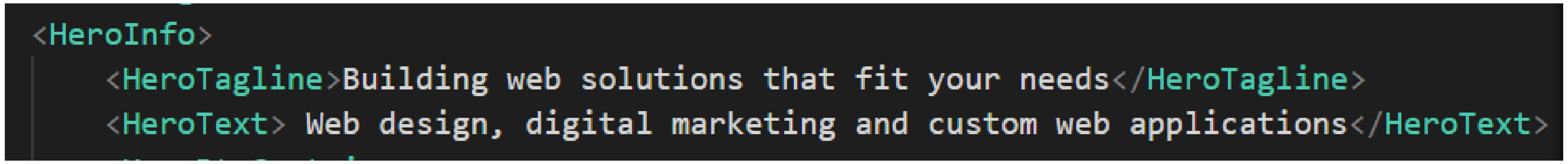
You customize the Hero in the **src/components/Hero**. There are 2 files: **index.js** , which handles the logic of the component, and **HeroComponents.js** (styled components), which handles the style.

In **src/components/Hero/index.js** you add your own video:



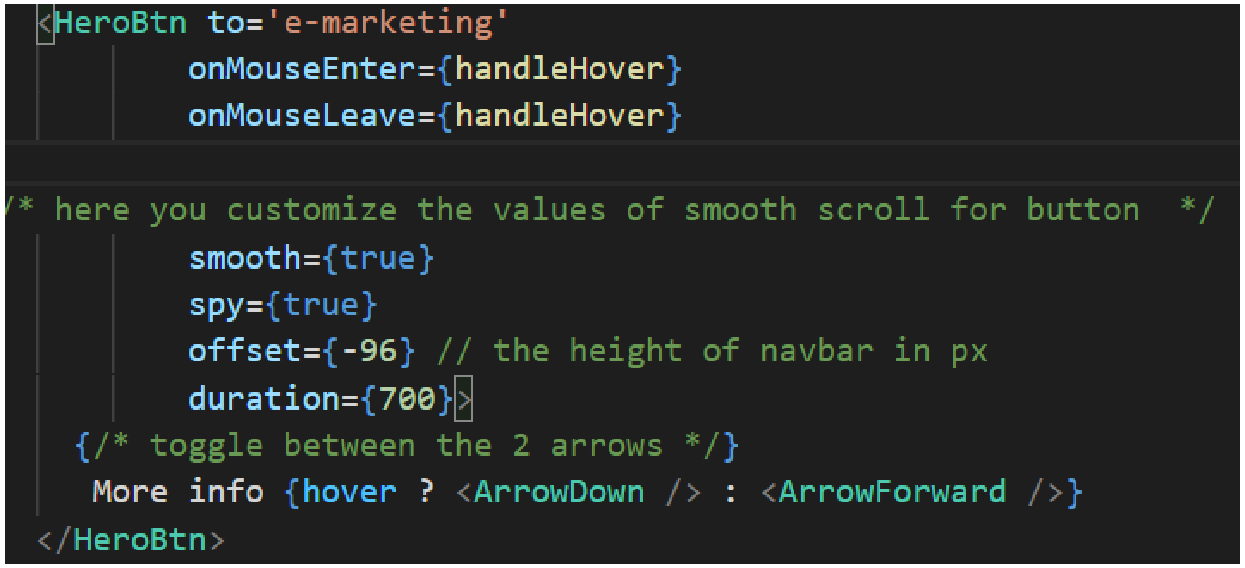
```
import Video from './../../assets/video-hero.mp4';
```

Here you can change the hero tagline and description:



```
  <HeroInfo>
      <HeroTagline>Building web solutions that fit your needs</HeroTagline>
      <HeroText> Web design, digital marketing and custom web applications</HeroText>
```

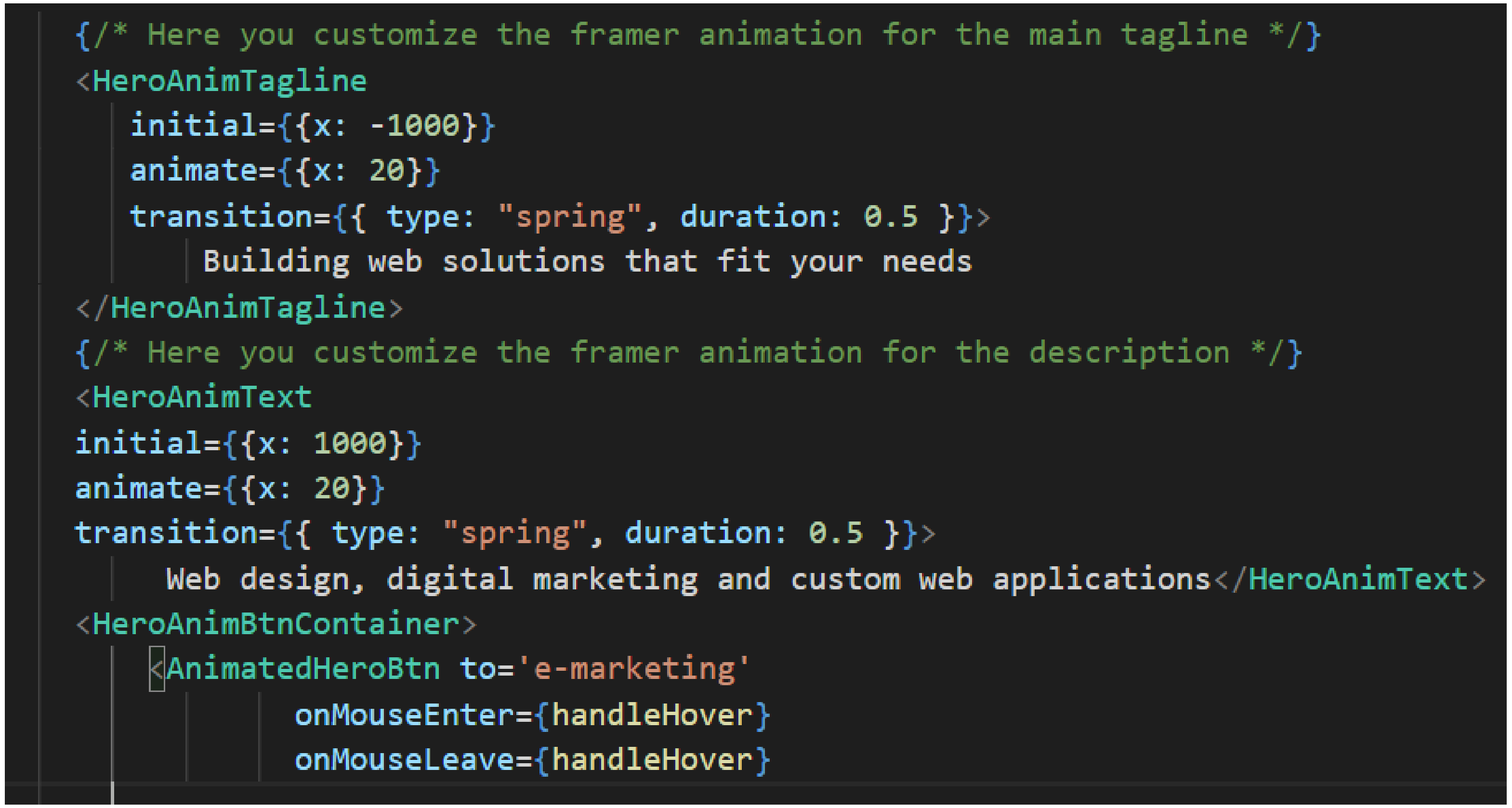
And here you customize the hero button:



```
  <HeroBtn to='e-marketing'
          onMouseEnter={handleHover}
          onMouseLeave={handleHover}

/* here you customize the values of smooth scroll for button  */
          smooth={true}
          spy={true}
          offset={-96} // the height of navbar in px
          duration={700}>
    {/* toggle between the 2 arrows */}
     More info {hover ? <ArrowDown /> : <ArrowForward />}
  </HeroBtn>
```

Animated Hero

In **src/components/HeroAnimated/index.js** you customize the animated hero text:

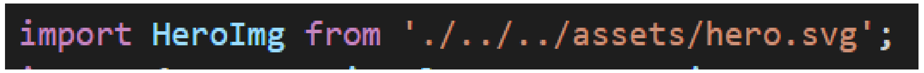


```
{/* Here you customize the framer animation for the main tagline */}
<HeroAnimTagline
   initial={{x: -1000}}
   animate={{x: 20}}
   transition={{ type: "spring", duration: 0.5 }}>
       Building web solutions that fit your needs
</HeroAnimTagline>
{/* Here you customize the framer animation for the description */}
<HeroAnimText
initial={{x: 1000}}
animate={{x: 20}}
transition={{ type: "spring", duration: 0.5 }}>
     Web design, digital marketing and custom web applications</HeroAnimText>
<HeroAnimBtnContainer>
    <AnimatedHeroBtn to='e-marketing'
           onMouseEnter={handleHover}
           onMouseLeave={handleHover}
```

To customize the styled components, you go to **src/components/HeroAnimated/HeroAnimatedComponents**

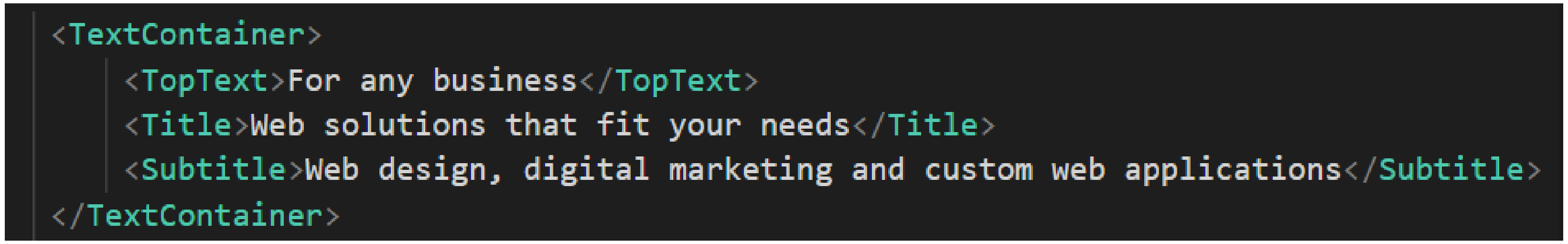
Static Hero

In **src/components/HeroStatic/index.js** you can add your own image:



```
import HeroImg from './../../assets/hero.svg';
```

In **src/components/HeroStatic/index.js** you customize the static hero top text, title and subtitle:



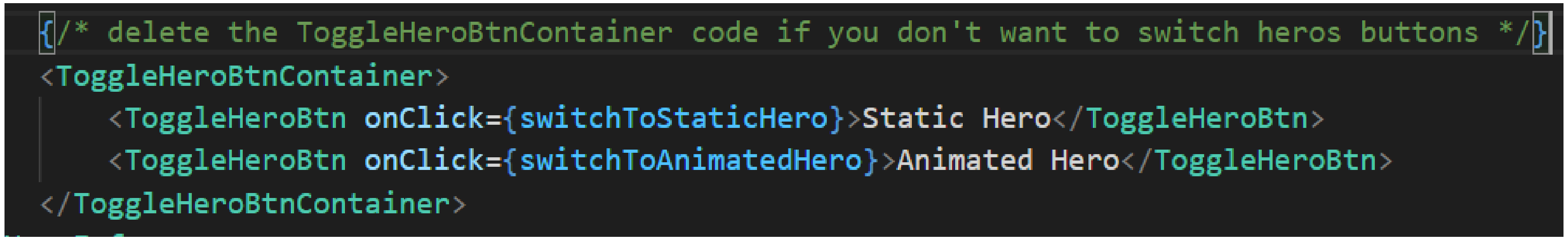
```
<TextContainer>
    <TopText>For any business</TopText>
    <Title>Web solutions that fit your needs</Title>
    <Subtitle>Web design, digital marketing and custom web applications</Subtitle>
</TextContainer>
```

To customize the styled components, you go to **src/components/HeroStatic/HeroStaticComponents**

Each styled component is well commented, so you can change the style according to your needs.

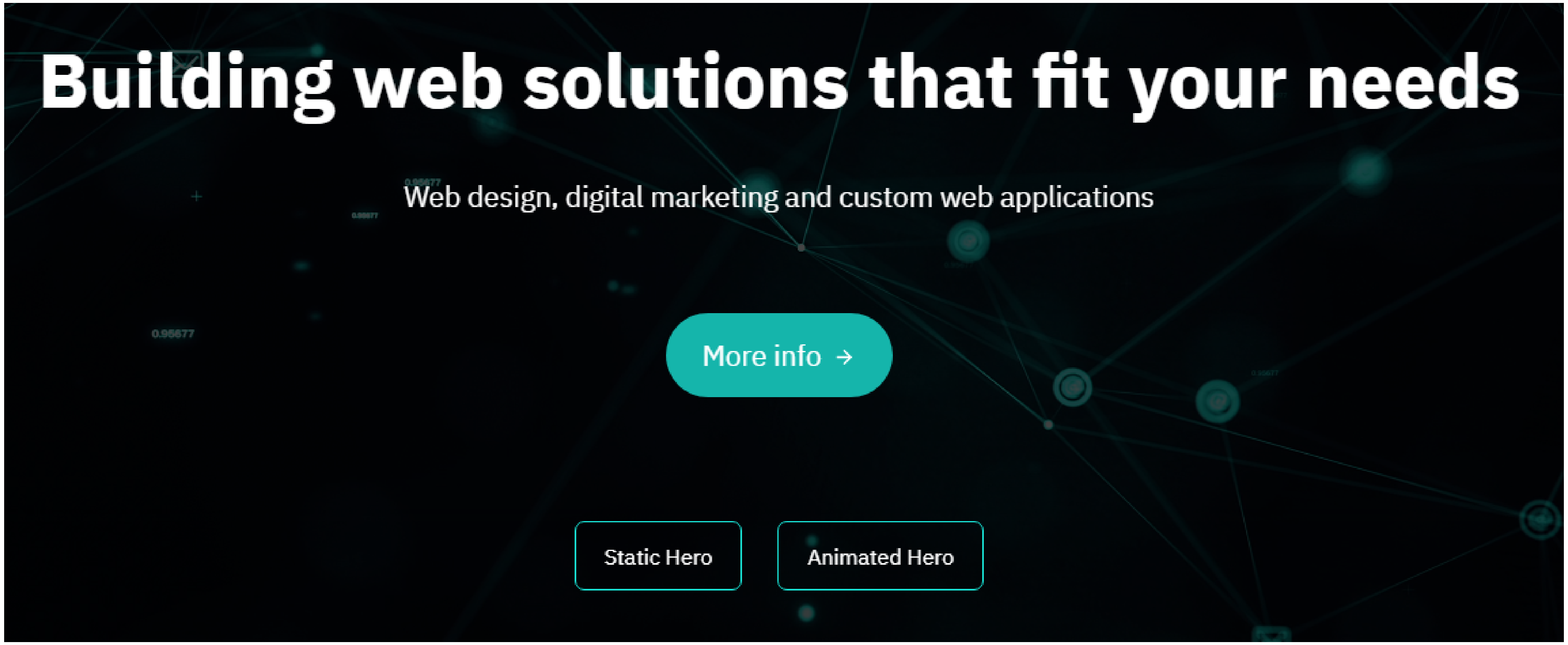
Switch between the three heros

Here are the lines of code that display the buttons which give you the possibility to choose another header:



```
{/* delete the ToggleHeroBtnContainer code if you don't want to switch heros buttons */}
<ToggleHeroBtnContainer>
    <ToggleHeroBtn onClick={switchToStaticHero}>Static Hero</ToggleHeroBtn>
    <ToggleHeroBtn onClick={switchToAnimatedHero}>Animated Hero</ToggleHeroBtn>
</ToggleHeroBtnContainer>
```

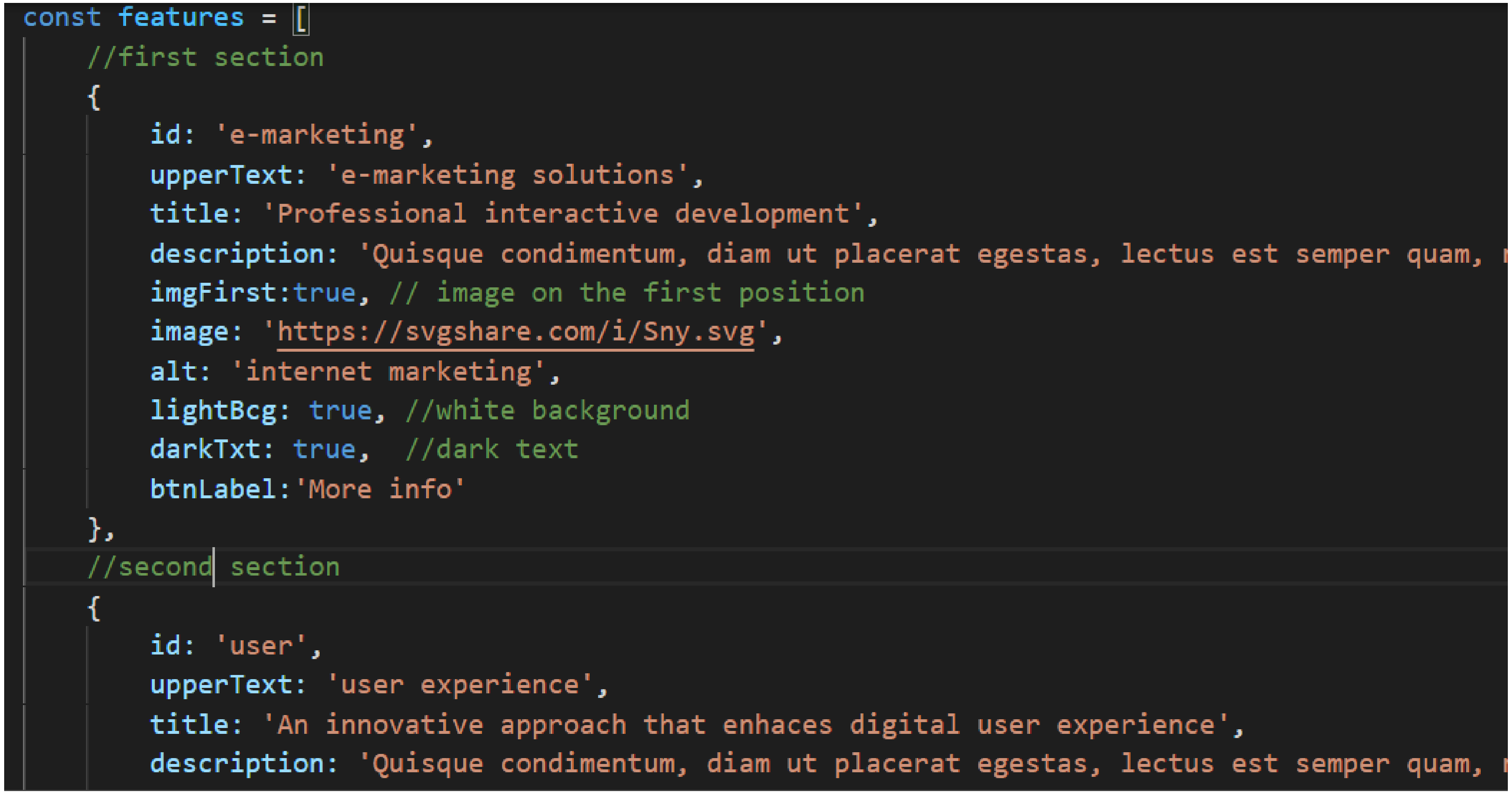
here is the UI display:


Building web solutions that fit your needs
Web design, digital marketing and custom web applications
More info →
Static Hero
Animated Hero

FEATURES COMPONENT - #BACK TO TOP

You customize the Features in the **src/components/Features**. There are 3 files: **data.js**, which holds the data of the three features sections, **index.js** , which handles the logic of the component, and **HeroComponents.js** (styled components), which handles the style.

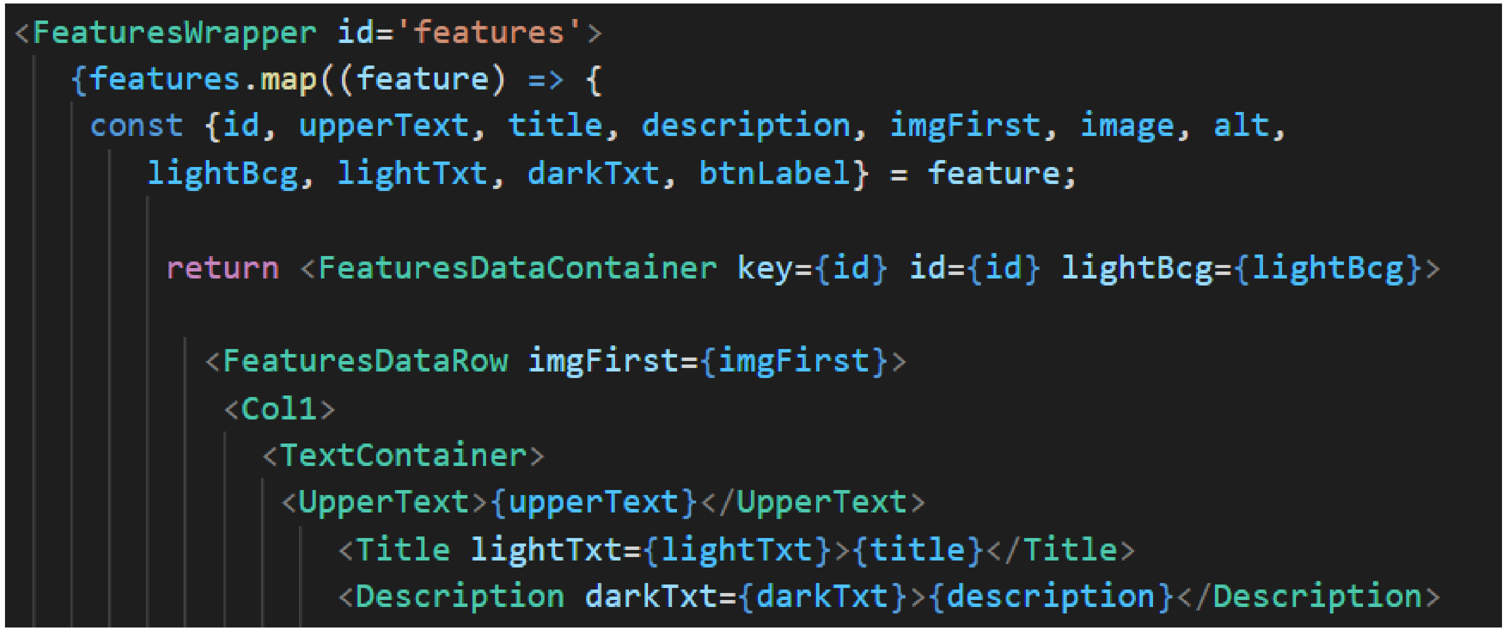
Here is the structure of the **data.js**. This is the place where you can add your own data. Also, feel free to add more properties or items:



```
const features = [
    //first section
    {
        id: 'e-marketing',
        upperText: 'e-marketing solutions',
        title: 'Professional interactive development',
        description: 'Quisque condimentum, diam ut placerat egestas, lectus est semper quam,
        imgFirst:true, // image on the first position
        image: 'https://svgshare.com/i/Sny.svg',
        alt: 'internet marketing',
        lightBcg: true, //white background
        darkTxt: true,  //dark text
        btnLabel:'More info'
    },
    //second section
    {
        id: 'user',
        upperText: 'user experience',
        title: 'An innovative approach that enhaces digital user experience',
        description: 'Quisque condimentum, diam ut placerat egestas, lectus est semper quam,
```

Basically, there are 3 features sections, each with its own properties and values. You find more explanations in the file comments.

In **src/components/Features/index.js** , we map through the features array and display the three features sections. The properties from data.js are assigned to each component:


```
<FeaturesWrapper id='features'>
  {features.map((feature) => {
   const {id, upperText, title, description, imgFirst, image, alt,
      lightBcg, lightTxt, darkTxt, btnLabel} = feature;

       return <FeaturesDataContainer key={id} id={id} lightBcg={lightBcg}>

         <FeaturesDataRow imgFirst={imgFirst}>
          <Col1>
            <TextContainer>
             <UpperText>{upperText}</UpperText>
                <Title lightTxt={lightTxt}>{title}</Title>
                <Description darkTxt={darkTxt}>{description}</Description>
```

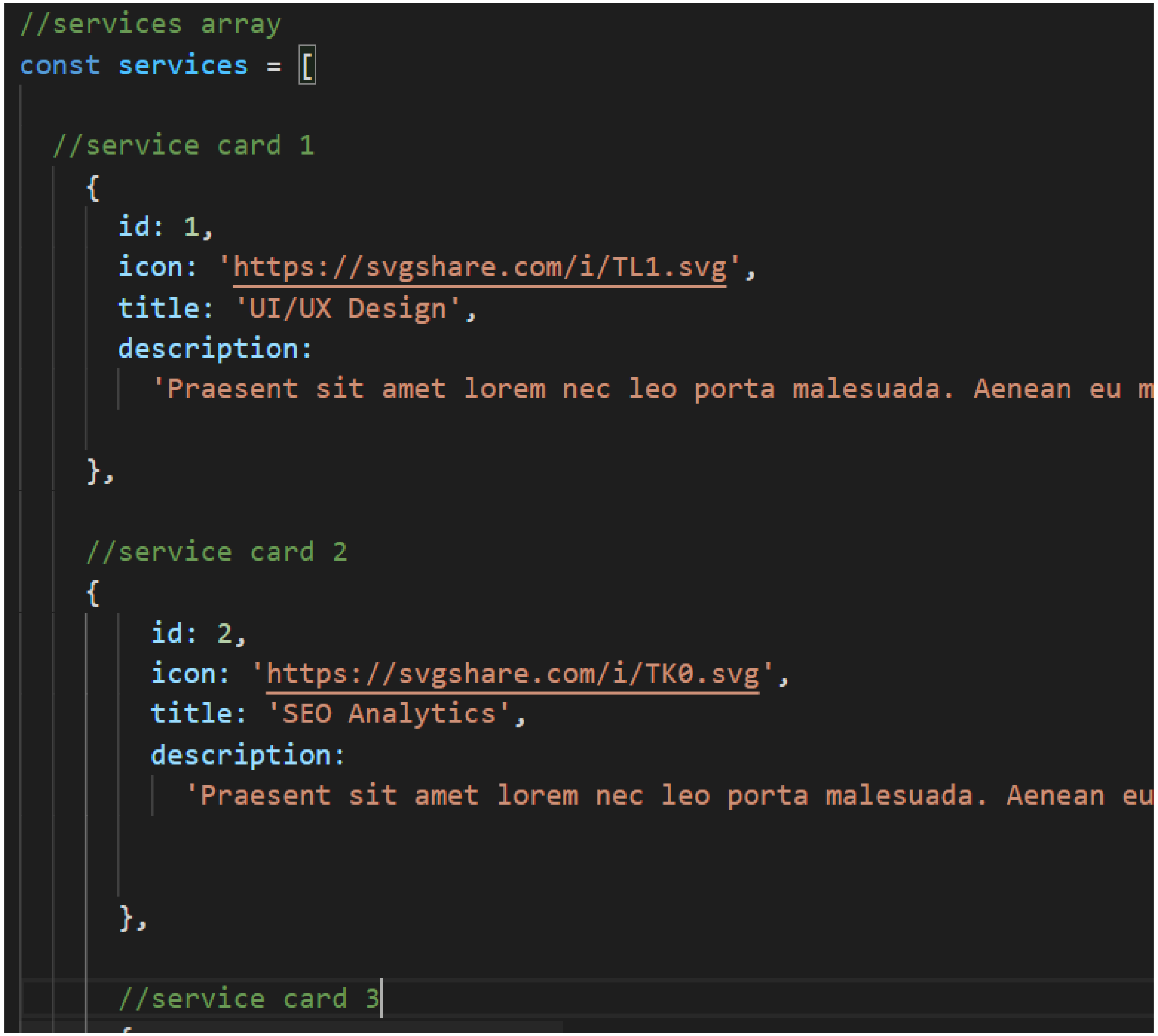
To customize the styled components, you go to **src/components/Features/FeaturesComponents**

Each styled component is well commented, so you can change the style according to your needs.

SERVICES COMPONENT - #BACK TO TOP

You customize the Services section in the **src/components/Services**. There are 3 files: **data.js**, which holds the data of the three features sections, **index.js** , which handles the logic of the component, and **ServicesComponents.js** (styled components), which handles the style.

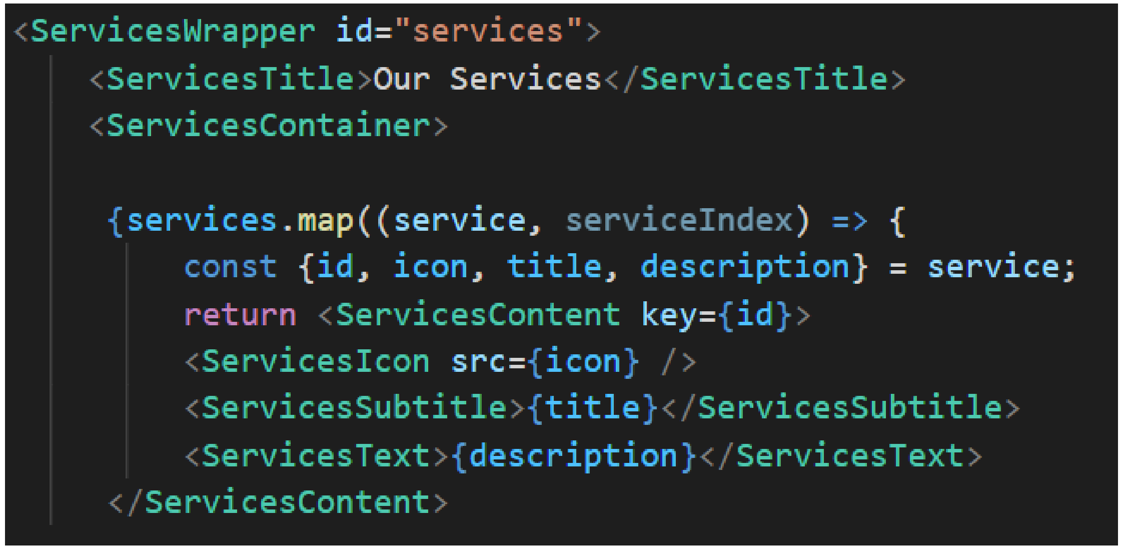
Here is the structure of the **data.js**. This is the place where you can add your own data. Also, feel free to add more properties or items:



```
//services array
const services = [

  //service card 1
    {
      id: 1,
      icon: 'https://svgshare.com/i/TL1.svg',
      title: 'UI/UX Design',
      description:
        'Praesent sit amet lorem nec leo porta malesuada. Aenean eu m

    },

    //service card 2
    {
        id: 2,
        icon: 'https://svgshare.com/i/TK0.svg',
        title: 'SEO Analytics',
        description:
          'Praesent sit amet lorem nec leo porta malesuada. Aenean eu


      },

      //service card 3
```

Each array item represents one service card, each with its own properties and values.

In **src/components/Services/index.js** , we map through the items array and display the four projects items.
Here you customize the portfolio navigation filters: The properties from data.js are assigned to each component:



```
<ServicesWrapper id="services">
    <ServicesTitle>Our Services</ServicesTitle>
    <ServicesContainer>

     {services.map((service, serviceIndex) => {
         const {id, icon, title, description} = service;
         return <ServicesContent key={id}>
         <ServicesIcon src={icon} />
         <ServicesSubtitle>{title}</ServicesSubtitle>
         <ServicesText>{description}</ServicesText>
     </ServicesContent>
```

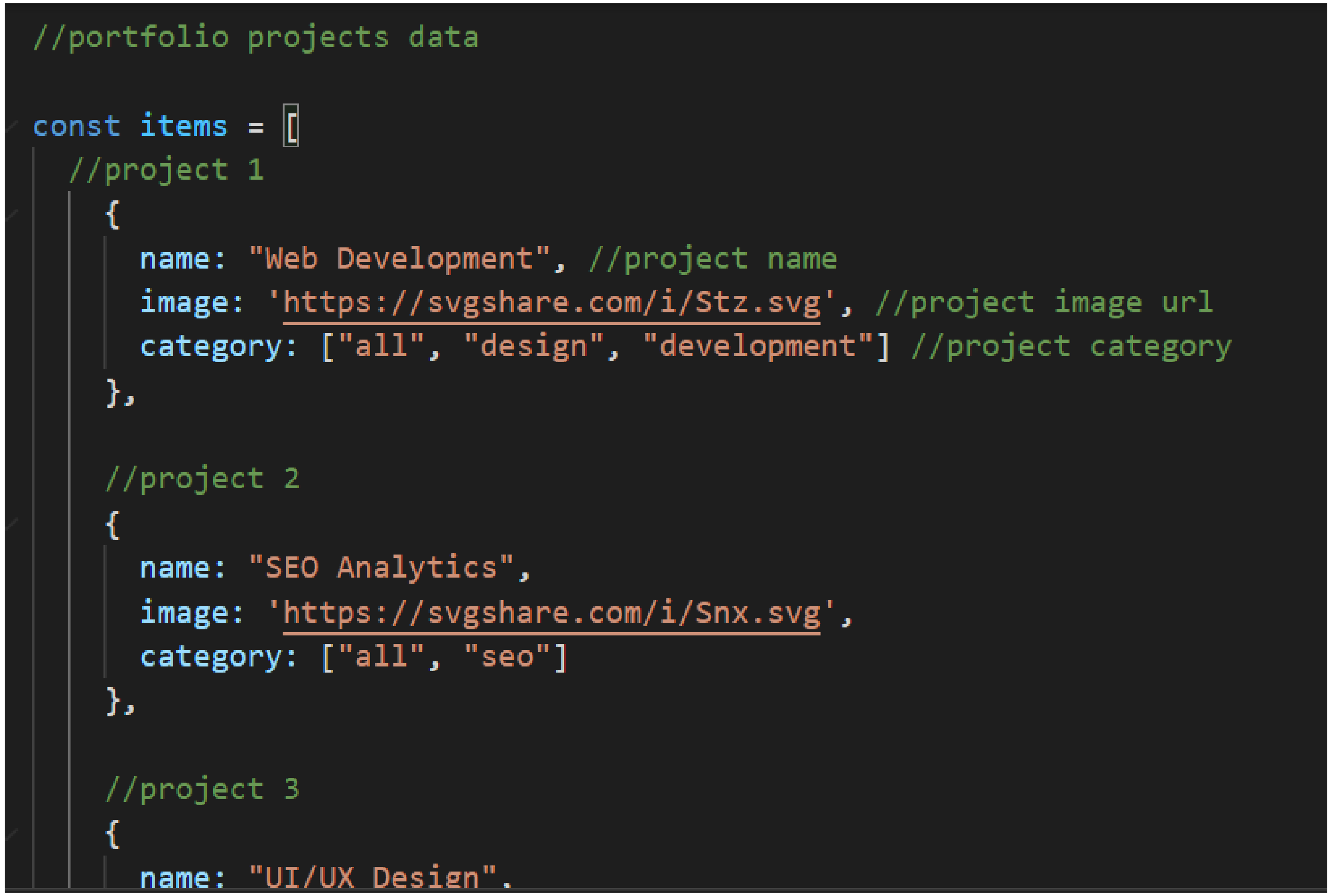
To customize the styled components, you go to **src/components/Services/ServicesComponents**

Each styled component is well commented, so you can change the style according to your needs.

PROJECTS COMPONENT - #BACK TO TOP

You customize the Projects section in the **src/components/Projects**. There are 3 files: **data.js**, which holds the data of the three features sections, **index.js** , which handles the logic of the component, and **ProjectsComponents.js** (styled components), which handles the style.

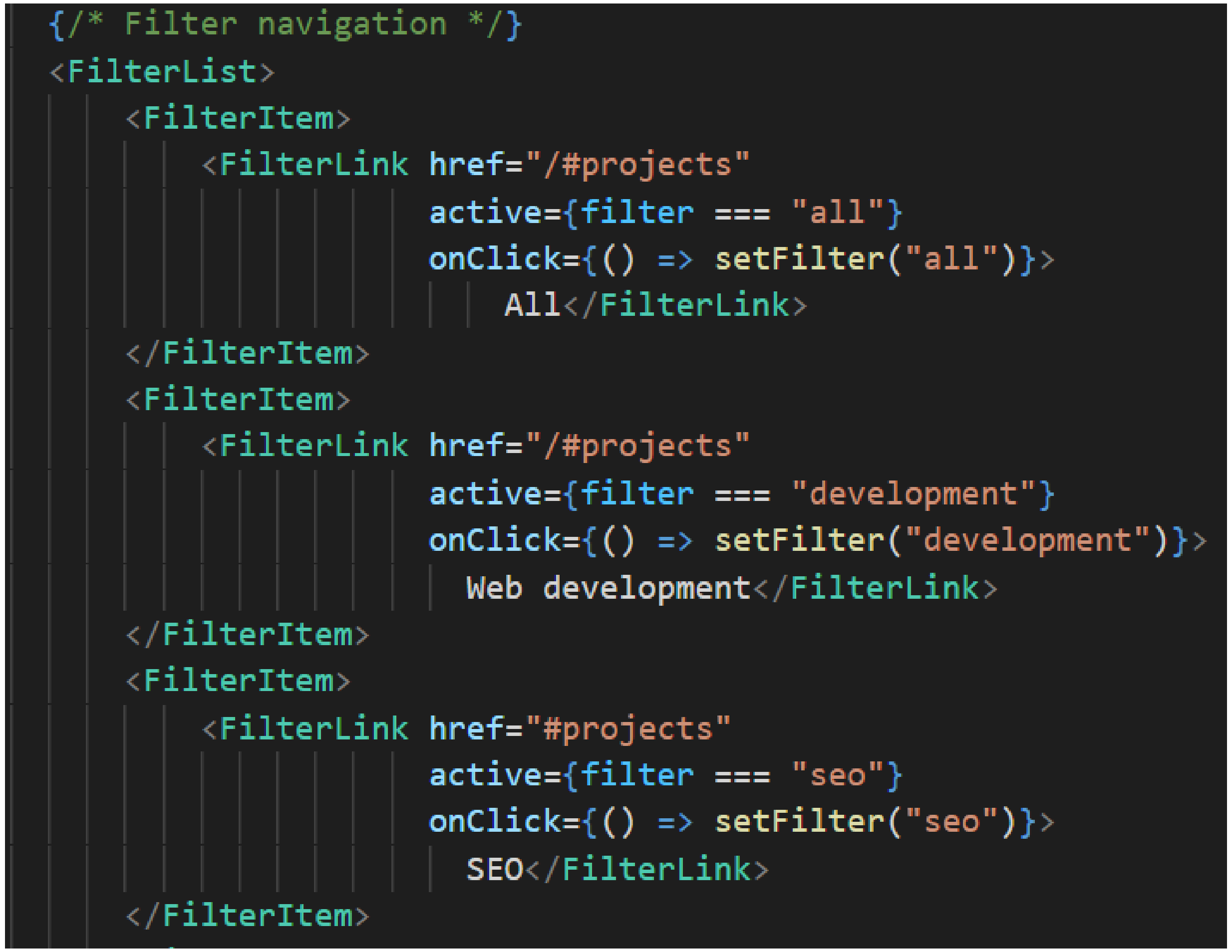
Here is the structure of the **data.js**. This is the place where you can add your own data. Also, feel free to add more properties or items:



```
//portfolio projects data

const items = [
  //project 1
    {
      name: "Web Development", //project name
      image: 'https://svgshare.com/i/Stz.svg', //project image url
      category: ["all", "design", "development"] //project category
    },

    //project 2
    {
      name: "SEO Analytics",
      image: 'https://svgshare.com/i/Snx.svg',
      category: ["all", "seo"]
    },

    //project 3
    {
      name: "UI/UX Design",
```

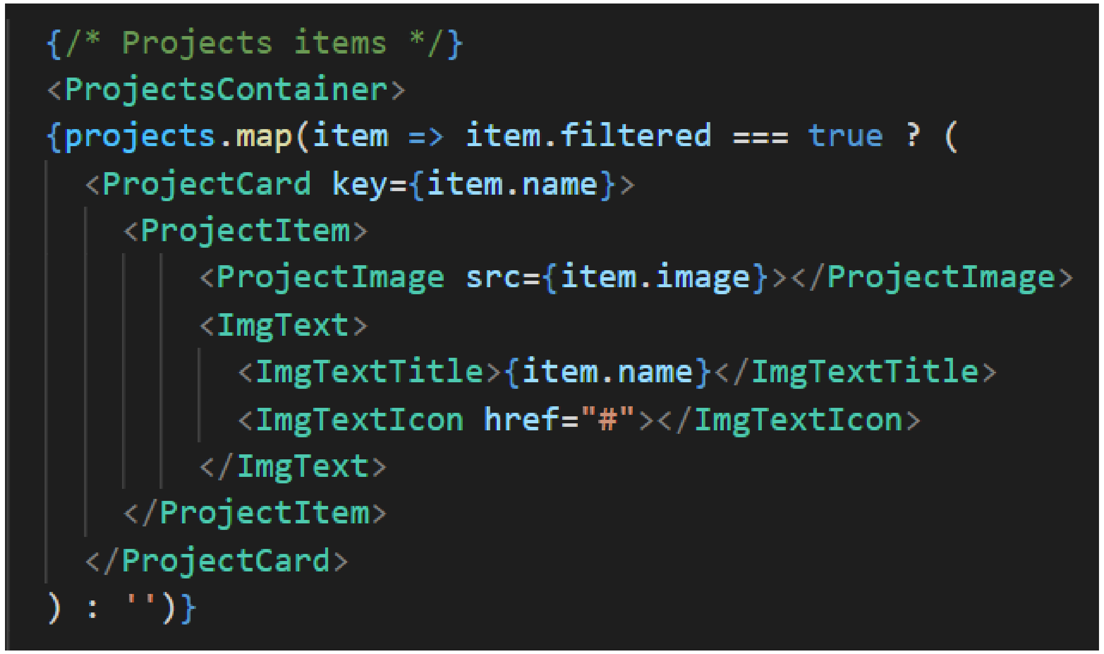
Each array item represents one portfolio project, each with its own properties and values.

In **src/components/Projects/index.js** , you customize the portfolio navigation filters:



```
{/* Filter navigation */}
<FilterList>
    <FilterItem>
        <FilterLink href="/#projects"
                    active={filter === "all"}
                    onClick={() => setFilter("all")}>
                        All</FilterLink>
    </FilterItem>
    <FilterItem>
        <FilterLink href="/#projects"
                    active={filter === "development"}
                    onClick={() => setFilter("development")}>
                    Web development</FilterLink>
    </FilterItem>
    <FilterItem>
        <FilterLink href="#projects"
                    active={filter === "seo"}
                    onClick={() => setFilter("seo")}>
                    SEO</FilterLink>
    </FilterItem>
```

We map through the items array and display the four portfolio items. The properties from data.js are assigned to each component:



```
{/* Projects items */}
<ProjectsContainer>
{projects.map(item => item.filtered === true ? (
  <ProjectCard key={item.name}>
    <ProjectItem>
        <ProjectImage src={item.image}></ProjectImage>
        <ImgText>
          <ImgTextTitle>{item.name}</ImgTextTitle>
          <ImgTextIcon href="#"></ImgTextIcon>
        </ImgText>
    </ProjectItem>
  </ProjectCard>
) : '')}
```

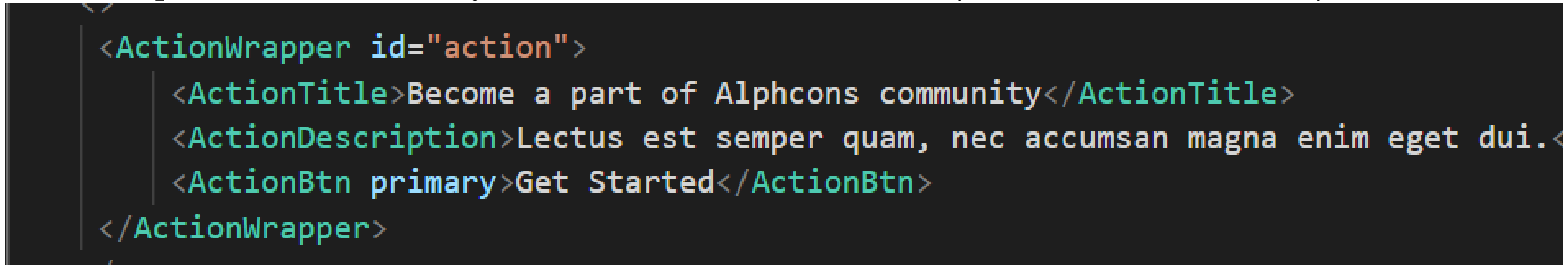
To customize the styled components, you go to **src/components/Projects/ProjectsComponents**

Each styled component is well commented, so you can change the style according to your needs.

CALL TO ACTION COMPONENT #BACK TO TOP

You customize the Contact section in the **src/components/Action**. There are 2 files: **index.js** , which handles the logic of the component, and **ActionComponents.js** (styled components), which handles the style.

In **src/components/Action/index.js** , we render the action data. Here you can customize it with your own content.



```
<ActionWrapper id="action">
    <ActionTitle>Become a part of Alphcons community</ActionTitle>
    <ActionDescription>Lectus est semper quam, nec accumsan magna enim eget dui.<
    <ActionBtn primary>Get Started</ActionBtn>
</ActionWrapper>
```

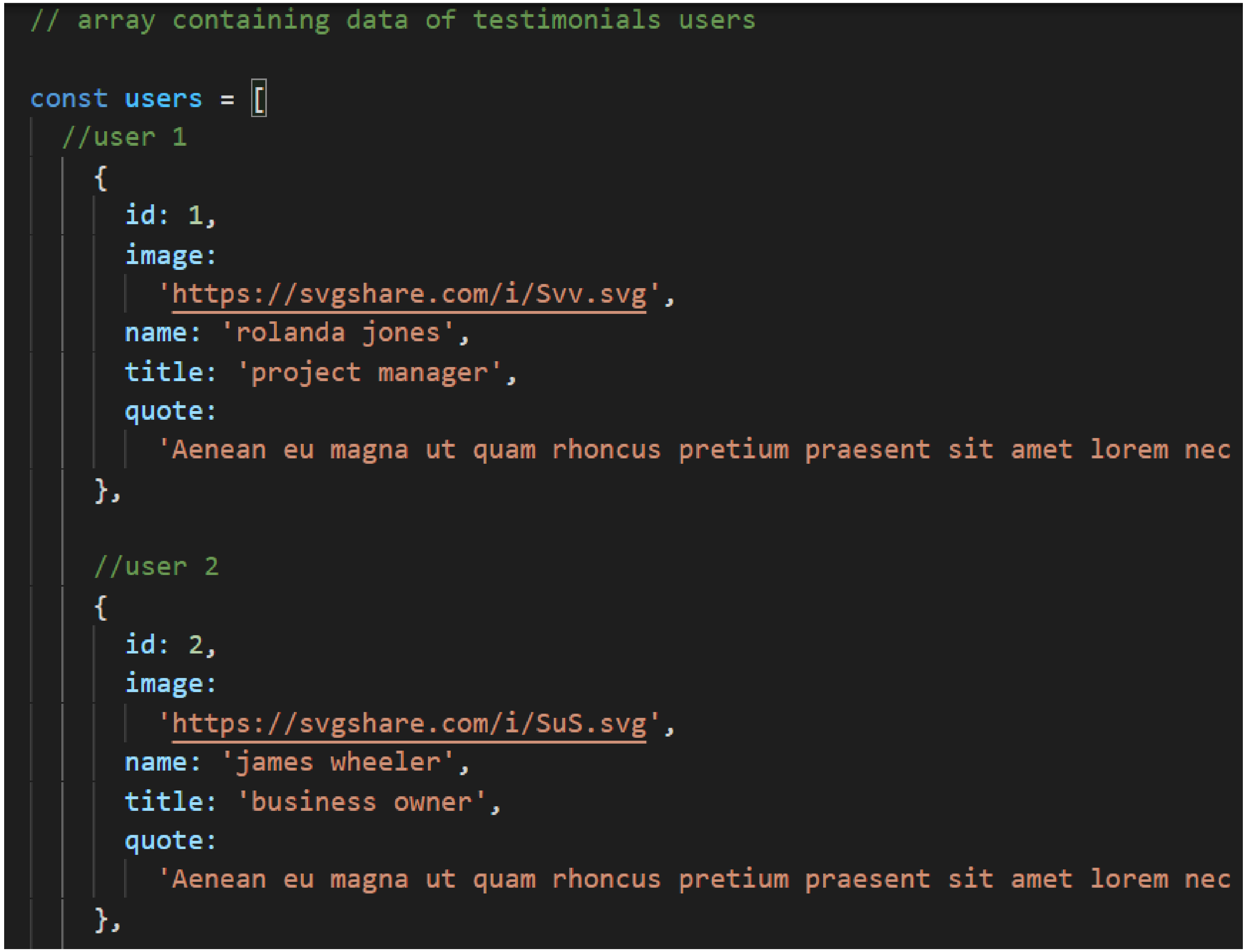
To customize the styled components, you go to **src/components/Action/ActionComponents**

Each styled component is well commented, so you can change the style according to your needs.

TESTIMONIALS COMPONENT - #BACK TO TOP

You customize the Testimonials section in the **src/components/Testimonials**. There are 3 files: **data.js**, which holds the data of the reviewers, **index.js** , which handles the logic of the component, and **TestimonialsComponents.js** (styled components), which handles the style.

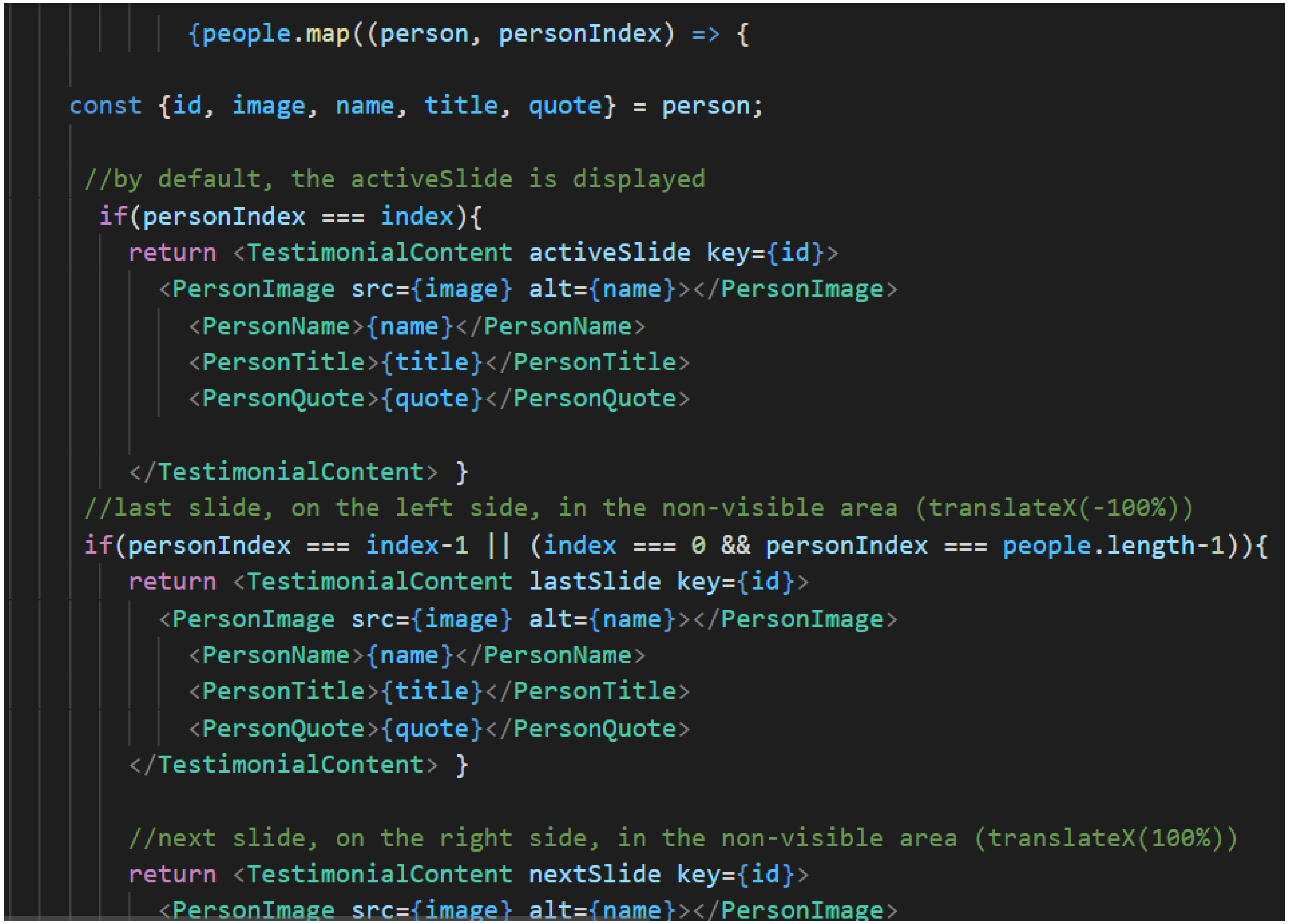
Here is the structure of the **data.js**. This is the place where you can add your own data. Also, feel free to add more properties or items:



```
// array containing data of testimonials users

const users = [
  //user 1
    {
      id: 1,
      image:
        'https://svgshare.com/i/Svv.svg',
      name: 'rolanda jones',
      title: 'project manager',
      quote:
        'Aenean eu magna ut quam rhoncus pretium praesent sit amet lorem nec
    },

    //user 2
    {
      id: 2,
      image:
        'https://svgshare.com/i/SuS.svg',
      name: 'james wheeler',
      title: 'business owner',
      quote:
        'Aenean eu magna ut quam rhoncus pretium praesent sit amet lorem nec
    },
```

Each array item represents one user, each with its own properties and values.

In **src/components/Testimonials/index.js** , we map through the users array and display them in a carousel. The properties from data.js are assigned to each component:



```
        {people.map((person, personIndex) => {

  const {id, image, name, title, quote} = person;

   //by default, the activeSlide is displayed
    if(personIndex === index){
      return <TestimonialContent activeSlide key={id}>
        <PersonImage src={image} alt={name}></PersonImage>
          <PersonName>{name}</PersonName>
          <PersonTitle>{title}</PersonTitle>
          <PersonQuote>{quote}</PersonQuote>

      </TestimonialContent> }
   //last slide, on the left side, in the non-visible area (translateX(-100%))
   if(personIndex === index-1 || (index === 0 && personIndex === people.length-1)){
      return <TestimonialContent lastSlide key={id}>
        <PersonImage src={image} alt={name}></PersonImage>
          <PersonName>{name}</PersonName>
          <PersonTitle>{title}</PersonTitle>
          <PersonQuote>{quote}</PersonQuote>
      </TestimonialContent> }

      //next slide, on the right side, in the non-visible area (translateX(100%))
      return <TestimonialContent nextSlide key={id}>
        <PersonImage src={image} alt={name}></PersonImage>
```

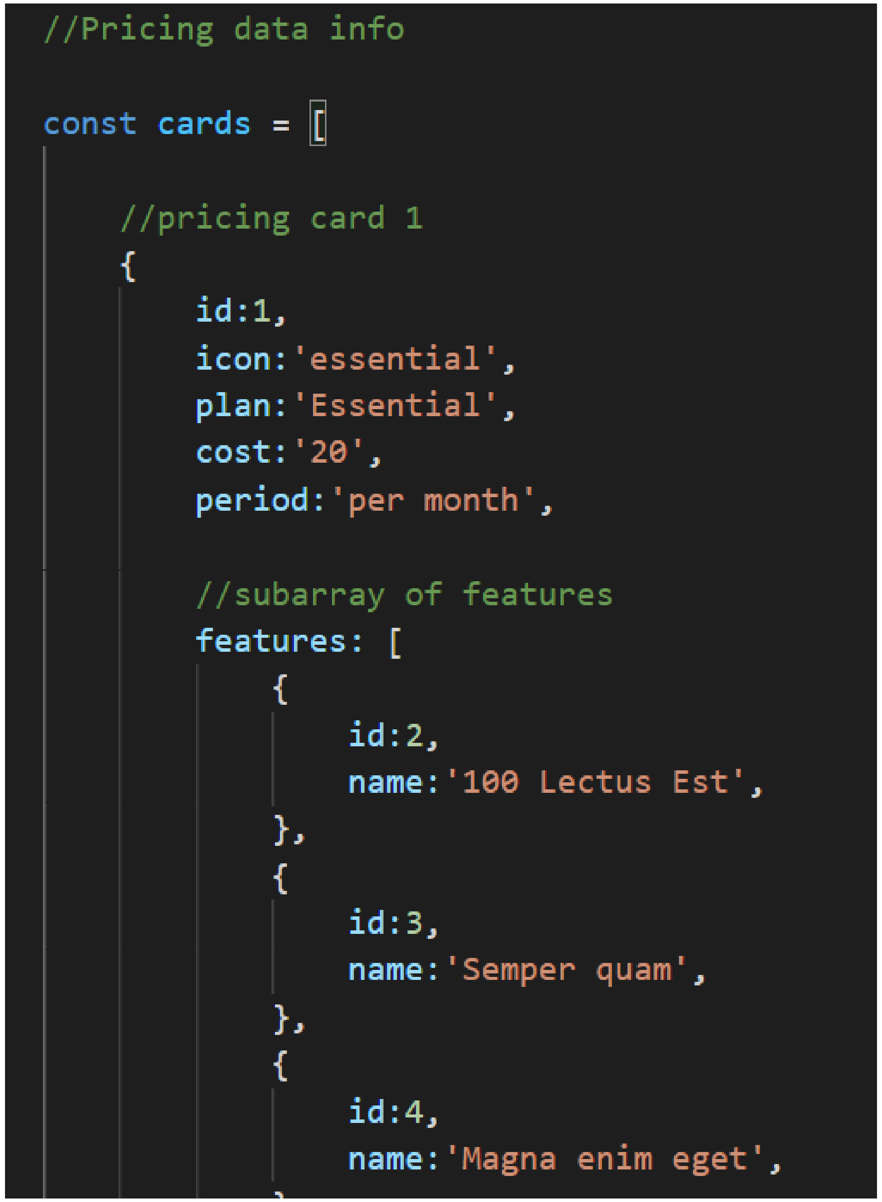
To customize the styled components, you go to **src/components/Testimonials/TestimonialsComponents**

Each styled component is well commented, so you can change the style according to your needs.

PRICING COMPONENT - #BACK TO TOP

You customize the Pricing section in the **src/components/Pricing**. There are 3 files: **data.js**, which holds the data of the three pricing cards, **index.js** , which handles the logic of the component, and **PricingComponents.js** (styled components), which handles the style.

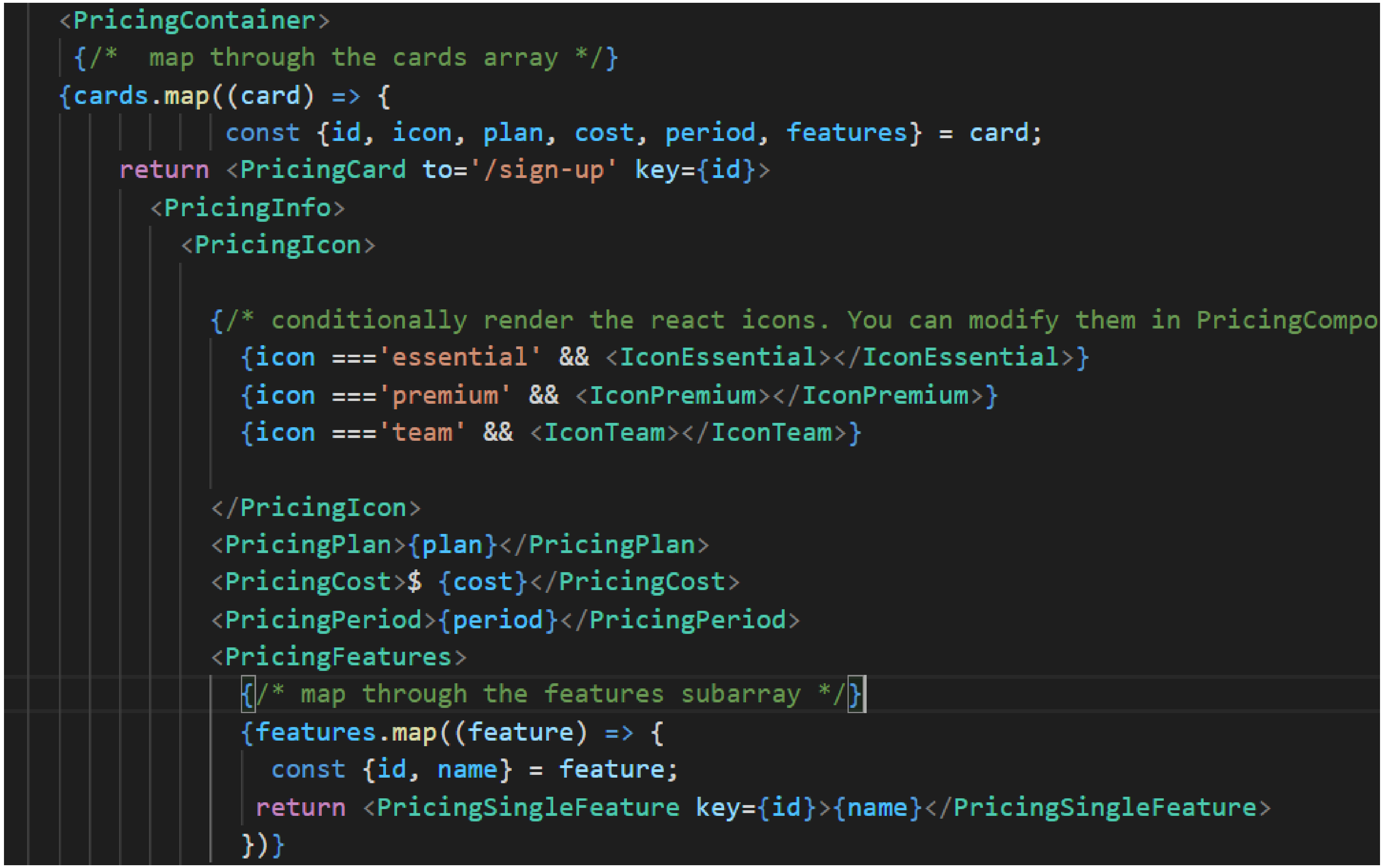
Here is the structure of the **data.js**. This is the place where you can add your own data. Also, feel free to add more properties or items:



```
//Pricing data info

const cards = [

    //pricing card 1
    {
        id:1,
        icon:'essential',
        plan:'Essential',
        cost:'20',
        period:'per month',

        //subarray of features
        features: [
            {
                id:2,
                name:'100 Lectus Est',
            },
            {
                id:3,
                name:'Semper quam',
            },
            {
                id:4,
                name:'Magna enim eget',
```

Each array item represents one pricing card, each with its own properties and values.

In **src/components/Pricing/index.js** , we map through the items array and display the three pricing cards. The properties from data.js are assigned to each component:



```
    <PricingContainer>
     {/*  map through the cards array */}
    {cards.map((card) => {
                const {id, icon, plan, cost, period, features} = card;
        return <PricingCard to='/sign-up' key={id}>
          <PricingInfo>
            <PricingIcon>

              {/* conditionally render the react icons. You can modify them in PricingCompo
                {icon ==='essential' && <IconEssential></IconEssential>}
                {icon ==='premium' && <IconPremium></IconPremium>}
                {icon ==='team' && <IconTeam></IconTeam>}

              </PricingIcon>
              <PricingPlan>{plan}</PricingPlan>
              <PricingCost>$ {cost}</PricingCost>
              <PricingPeriod>{period}</PricingPeriod>
              <PricingFeatures>
                {/* map through the features subarray */}
                {features.map((feature) => {
                  const {id, name} = feature;
                 return <PricingSingleFeature key={id}>{name}</PricingSingleFeature>
                })}
```

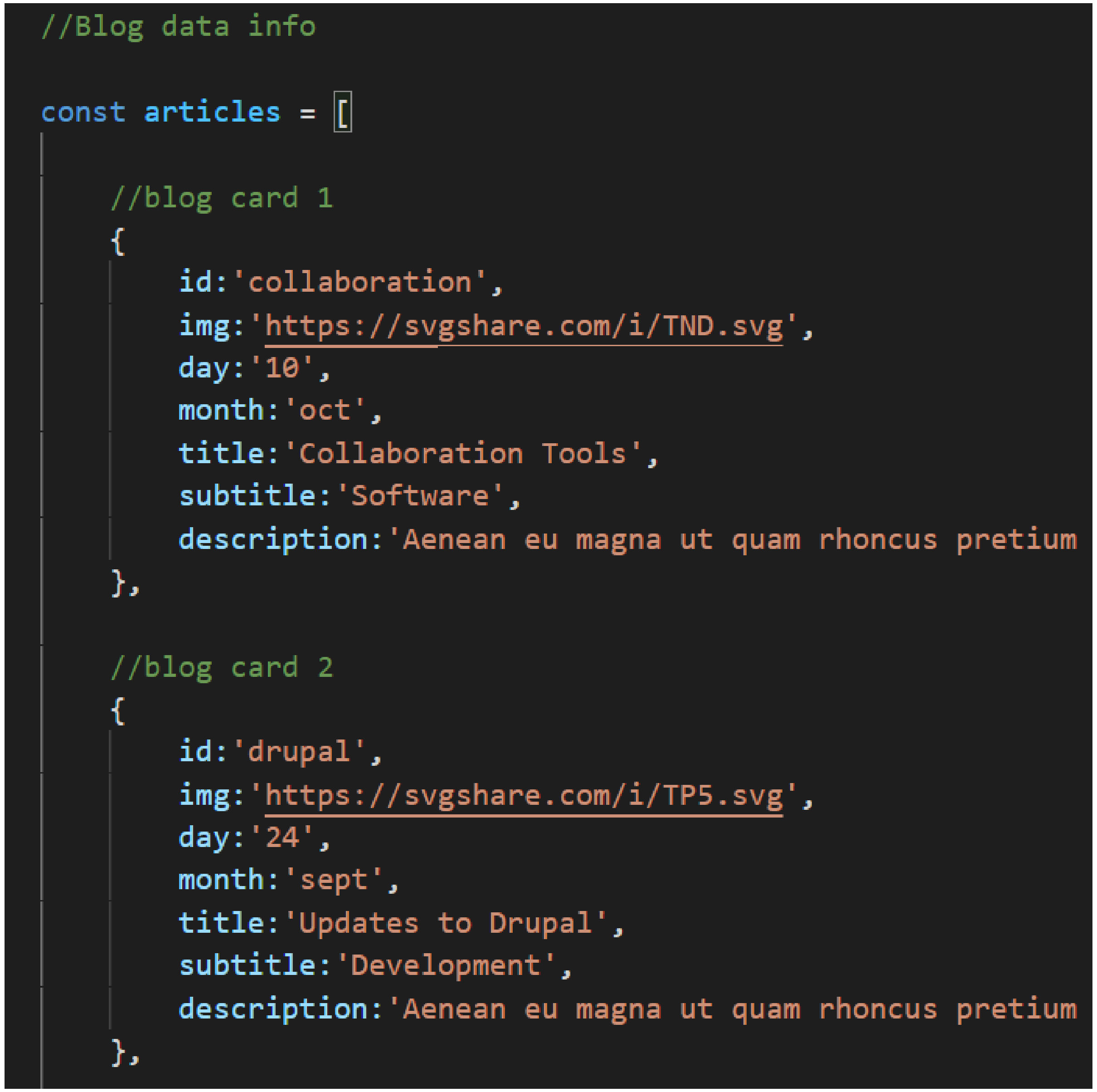
To customize the styled components, you go to **src/components/Pricing/PricingComponents**

Each styled component is well commented, so you can change the style according to your needs.

BLOG COMPONENT #BACK TO TOP

You customize the Blog section in the **src/components/Blog**. There are 3 files: **data.js**, which holds the data of the three blog cards, **index.js** , which handles the logic of the component, and **BlogComponents.js** (styled components), which handles the style.

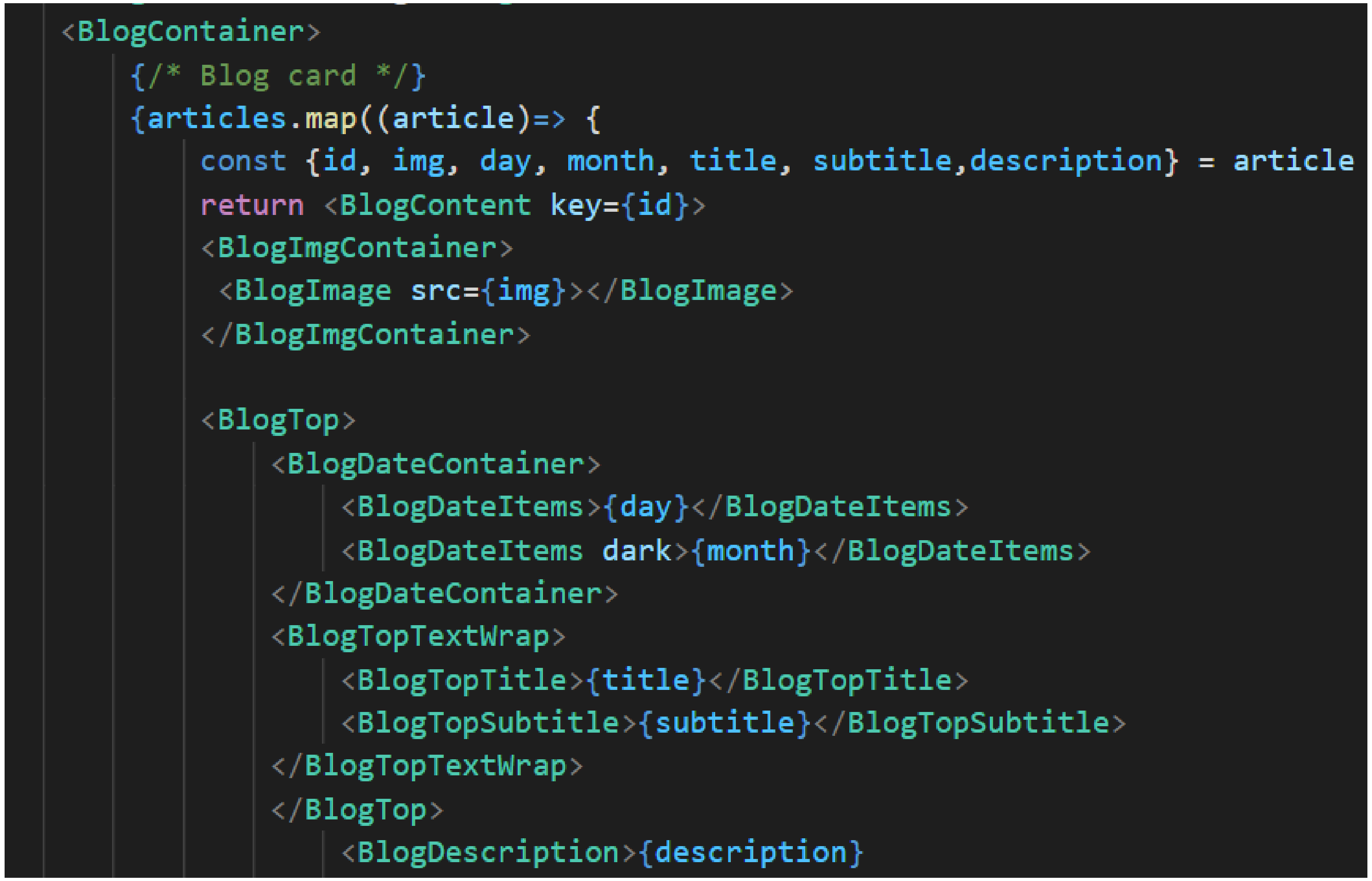
Here is the structure of the **data.js**. This is the place where you can add your own data. Also, feel free to add more properties or items:



```
//Blog data info

const articles = [

    //blog card 1
    {
        id:'collaboration',
        img:'https://svgshare.com/i/TND.svg',
        day:'10',
        month:'oct',
        title:'Collaboration Tools',
        subtitle:'Software',
        description:'Aenean eu magna ut quam rhoncus pretium
    },

    //blog card 2
    {
        id:'drupal',
        img:'https://svgshare.com/i/TP5.svg',
        day:'24',
        month:'sept',
        title:'Updates to Drupal',
        subtitle:'Development',
        description:'Aenean eu magna ut quam rhoncus pretium
    },
```

Each array item represents one blog card, each with its own properties and values.

In **src/components/Blog/index.js** , we map through the items array and display the three pricing cards. The properties from data.js are assigned to each component:



```
<BlogContainer>
    {/* Blog card */}
    {articles.map((article)=> {
        const {id, img, day, month, title, subtitle,description} = article
        return <BlogContent key={id}>
        <BlogImgContainer>
         <BlogImage src={img}></BlogImage>
        </BlogImgContainer>

        <BlogTop>
            <BlogDateContainer>
                <BlogDateItems>{day}</BlogDateItems>
                <BlogDateItems dark>{month}</BlogDateItems>
            </BlogDateContainer>
            <BlogTopTextWrap>
                <BlogTopTitle>{title}</BlogTopTitle>
                <BlogTopSubtitle>{subtitle}</BlogTopSubtitle>
            </BlogTopTextWrap>
            </BlogTop>
                <BlogDescription>{description}
```

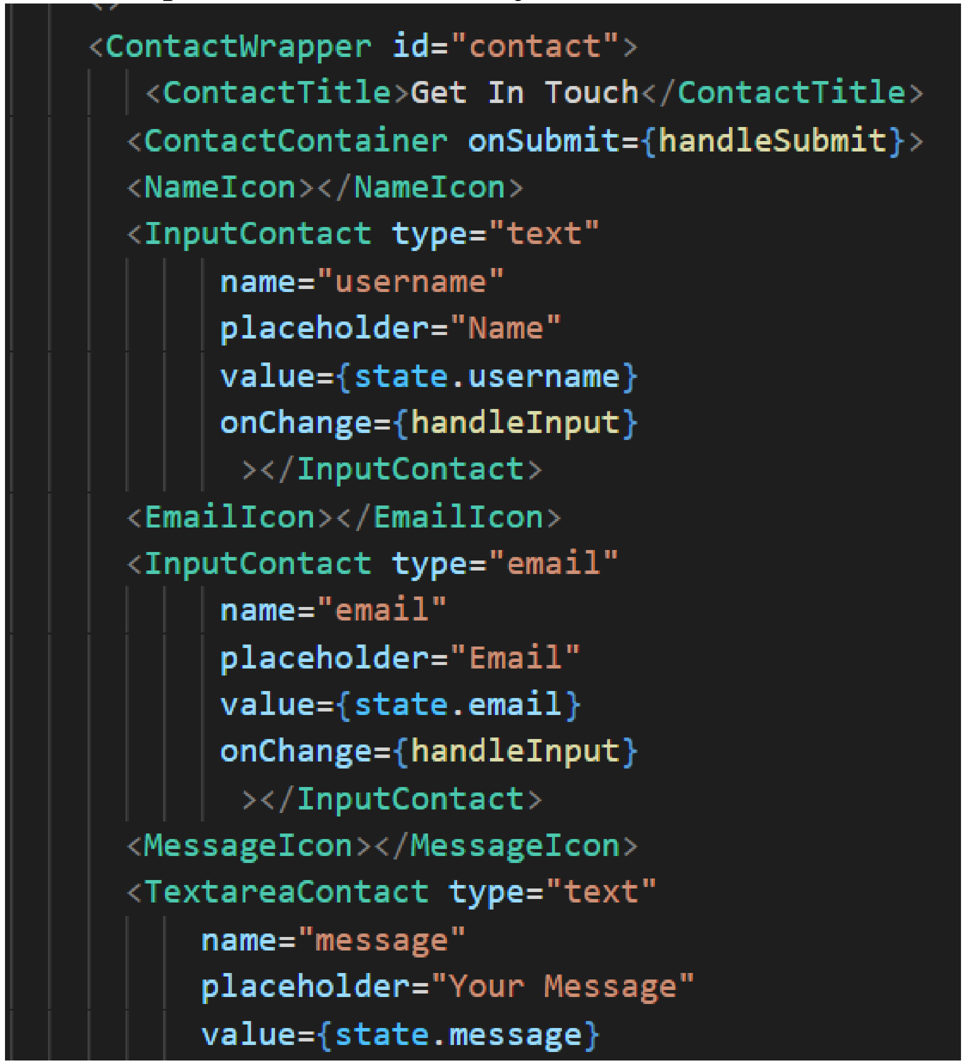
To customize the styled components, you go to **src/components/Blog/BlogComponents**

Each styled component is well commented, so you can change the style according to your needs.

CONTACT COMPONENT #BACK TO TOP

You customize the Contact section in the **src/components/Contact**. There are 2 files: **index.js** , which handles the logic of the component, and **ContactComponents.js** (styled components), which handles the style.

In **src/components/Contact/index.js** , we render the contact form. Here you can customize it with your own content.



```
    <ContactWrapper id="contact">
      <ContactTitle>Get In Touch</ContactTitle>
    <ContactContainer onSubmit={handleSubmit}>
    <NameIcon></NameIcon>
    <InputContact type="text"
          name="username"
          placeholder="Name"
          value={state.username}
          onChange={handleInput}
           ></InputContact>
    <EmailIcon></EmailIcon>
    <InputContact type="email"
          name="email"
          placeholder="Email"
          value={state.email}
          onChange={handleInput}
           ></InputContact>
    <MessageIcon></MessageIcon>
    <TextareaContact type="text"
         name="message"
         placeholder="Your Message"
         value={state.message}
```

You can read more about the form functions at here

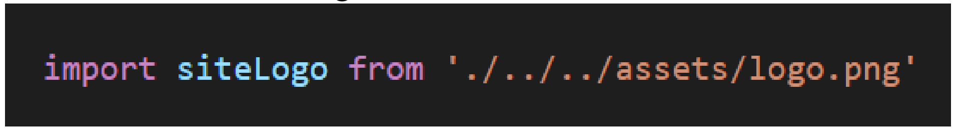
To customize the styled components, you go to **src/components/Contact/ContactComponents**

Each styled component is well commented, so you can change the style according to your needs.

FOOTER COMPONENT #BACK TO TOP

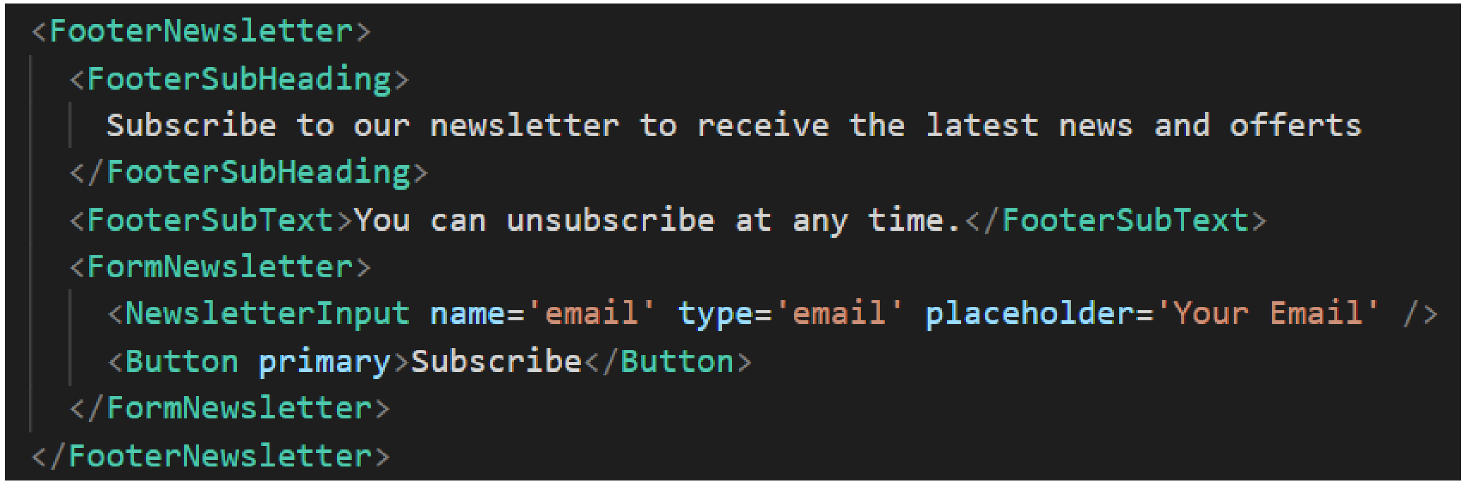
You customize the Footer section in the **src/components/Footer**. There are 2 files: **index.js** , which handles the logic of the component, and **FooterComponents.js** (styled components), which handles the style.

In **src/components/Footer/index.js** , we render the footer information: the newsletter, links, and copyright sections. Here you can customize the footer logo:



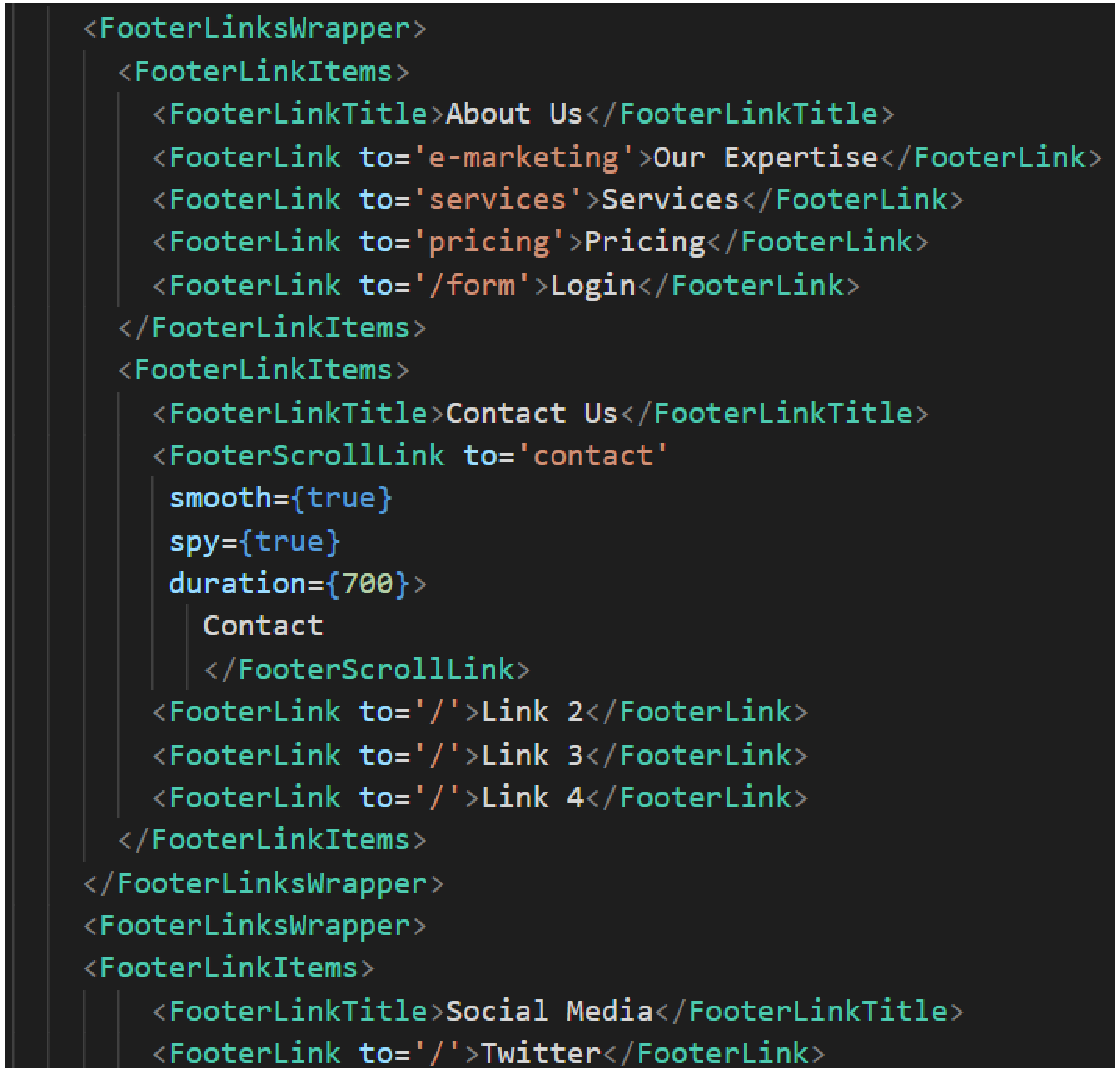
```
import siteLogo from './../../assets/logo.png'
```

You customize the newsletter section here:



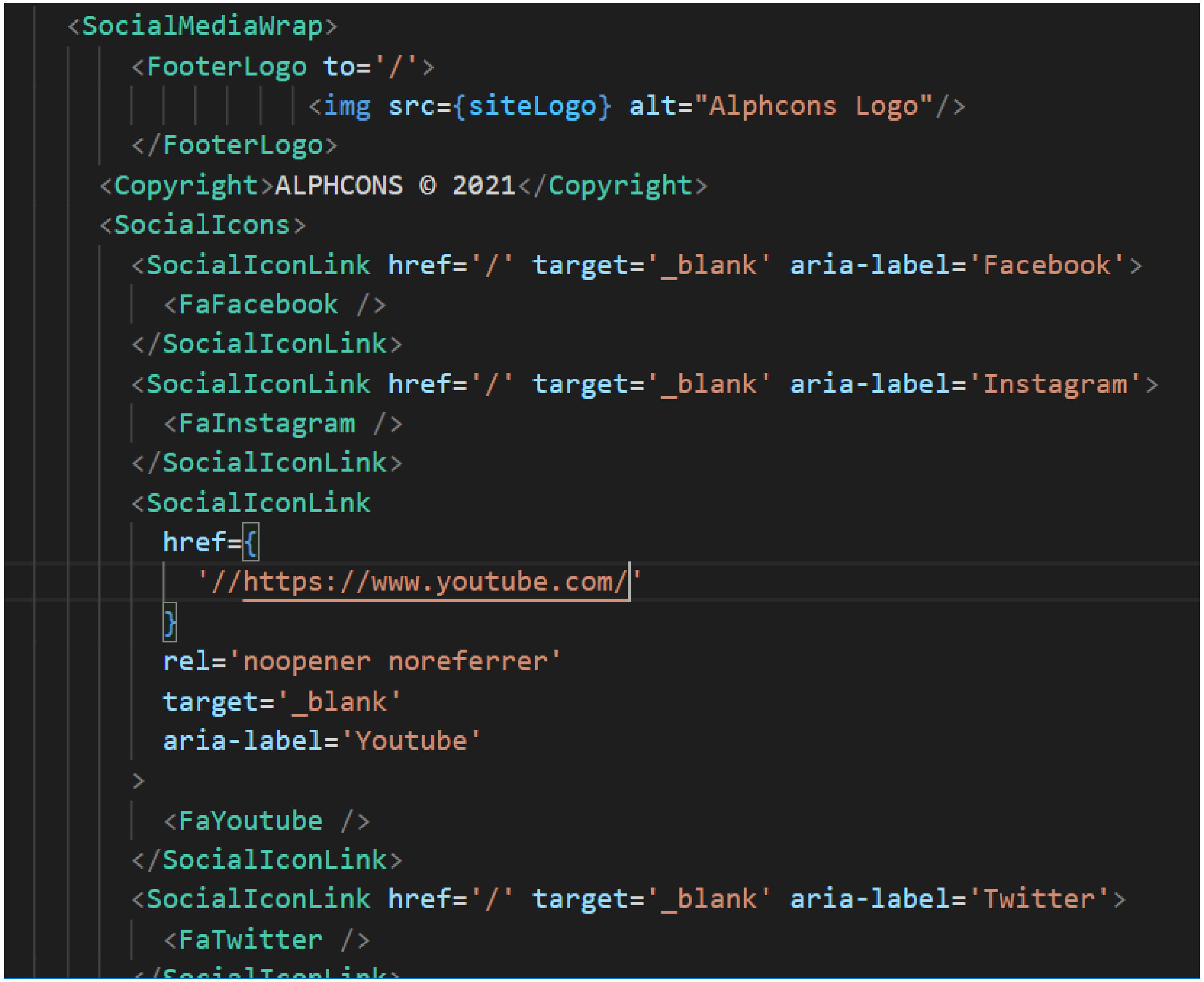
```
<FooterNewsletter>
  <FooterSubHeading>
    Subscribe to our newsletter to receive the latest news and offerts
  </FooterSubHeading>
  <FooterSubText>You can unsubscribe at any time.</FooterSubText>
  <FormNewsletter>
    <NewsletterInput name='email' type='email' placeholder='Your Email' />
    <Button primary>Subscribe</Button>
  </FormNewsletter>
</FooterNewsletter>
```

You customize the footer links here:



```
    <FooterLinksWrapper>
      <FooterLinkItems>
        <FooterLinkTitle>About Us</FooterLinkTitle>
        <FooterLink to='e-marketing'>Our Expertise</FooterLink>
        <FooterLink to='services'>Services</FooterLink>
        <FooterLink to='pricing'>Pricing</FooterLink>
        <FooterLink to='/form'>Login</FooterLink>
      </FooterLinkItems>
      <FooterLinkItems>
        <FooterLinkTitle>Contact Us</FooterLinkTitle>
        <FooterScrollLink to='contact'
         smooth={true}
         spy={true}
         duration={700}>
          Contact
          </FooterScrollLink>
        <FooterLink to='/'>Link 2</FooterLink>
        <FooterLink to='/'>Link 3</FooterLink>
        <FooterLink to='/'>Link 4</FooterLink>
      </FooterLinkItems>
    </FooterLinksWrapper>
    <FooterLinksWrapper>
    <FooterLinkItems>
        <FooterLinkTitle>Social Media</FooterLinkTitle>
        <FooterLink to='/'>Twitter</FooterLink>
```

You customize the copyright section here:



```
<SocialMediaWrap>
    <FooterLogo to='/'>
                <img src={siteLogo} alt="Alphcons Logo"/>
    </FooterLogo>
  <Copyright>ALPHCONS © 2021</Copyright>
  <SocialIcons>
    <SocialIconLink href='/' target='_blank' aria-label='Facebook'>
      <FaFacebook />
    </SocialIconLink>
    <SocialIconLink href='/' target='_blank' aria-label='Instagram'>
      <FaInstagram />
    </SocialIconLink>
    <SocialIconLink
      href={
        '//https://www.youtube.com/'
      }
      rel='noopener noreferrer'
      target='_blank'
      aria-label='Youtube'
    >
      <FaYoutube />
    </SocialIconLink>
    <SocialIconLink href='/' target='_blank' aria-label='Twitter'>
      <FaTwitter />
```

To customize the styled components, you go to **src/components/Footer/FooterComponents**

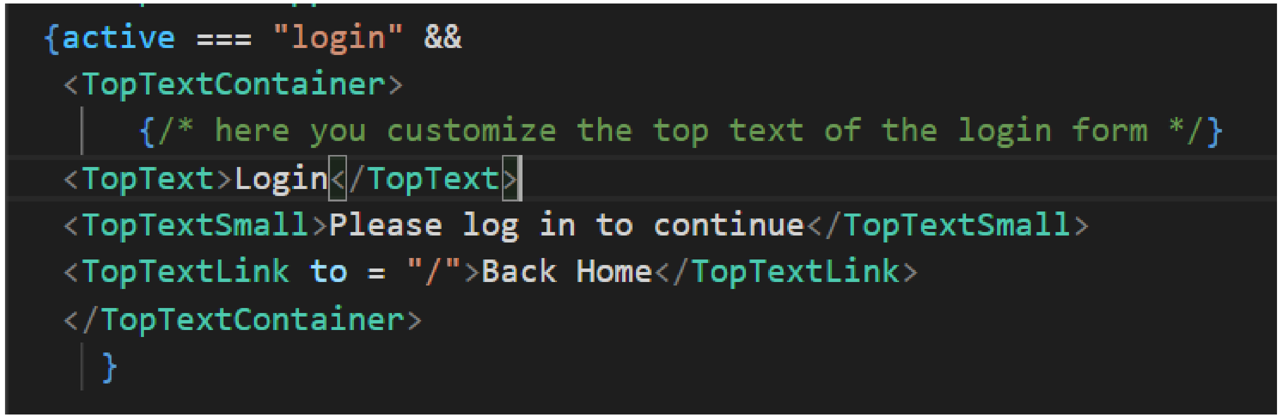
Each styled component is well commented, so you can change the style according to your needs.

FORMS CUSTOMIZATION AND VALIDATION #BACK TO TOP

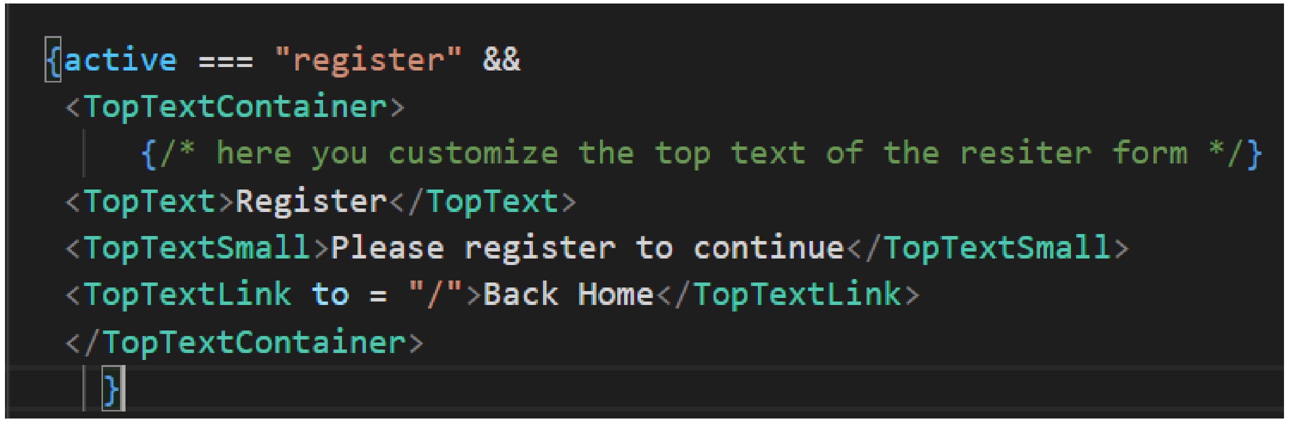
In our project, we have 3 forms: login, register and contact form.

Form Customization

To customize the top text in login and register form, go to **src/pages/formPage.js:**

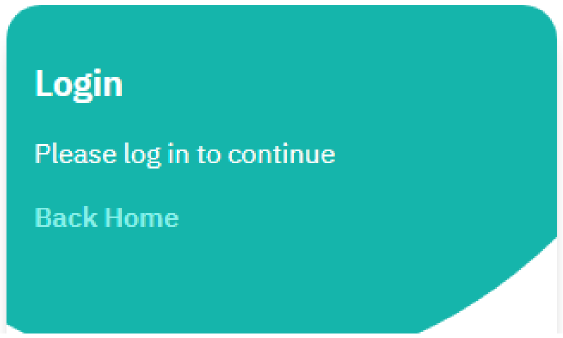


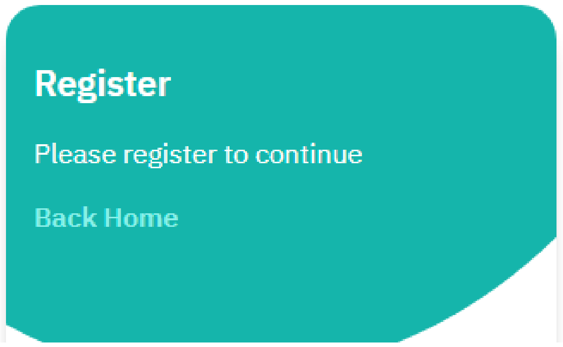
```
{active === "login" &&
 <TopTextContainer>
   {/* here you customize the top text of the login form */}
 <TopText>Login</TopText>
 <TopTextSmall>Please log in to continue</TopTextSmall>
 <TopTextLink to = "/">Back Home</TopTextLink>
 </TopTextContainer>
  }
```



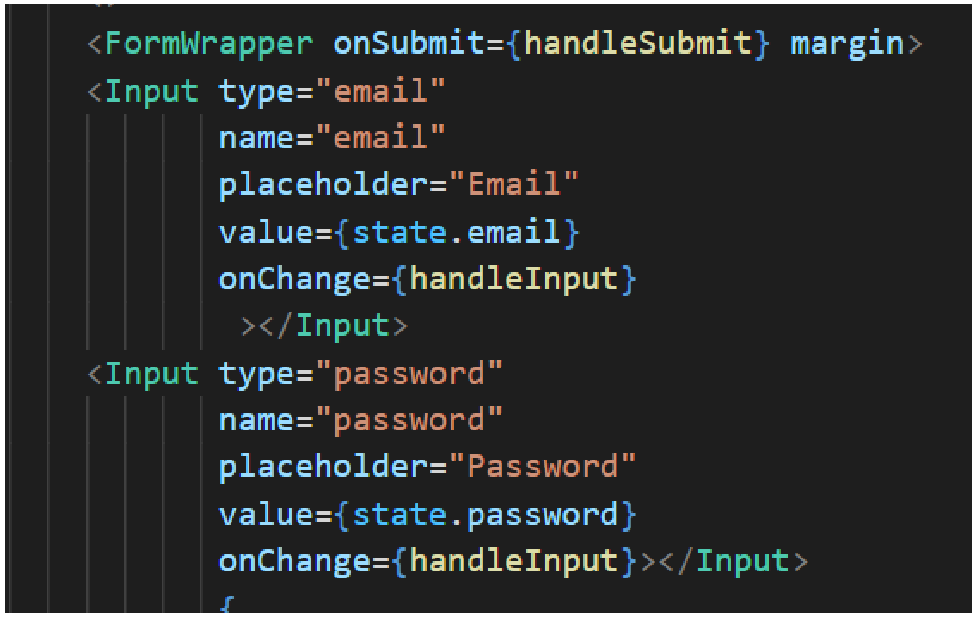
```
{active === "register" &&
 <TopTextContainer>
   {/* here you customize the top text of the resiter form */}
 <TopText>Register</TopText>
 <TopTextSmall>Please register to continue</TopTextSmall>
 <TopTextLink to = "/">Back Home</TopTextLink>
 </TopTextContainer>
  }
```

Here is the output of the top text:



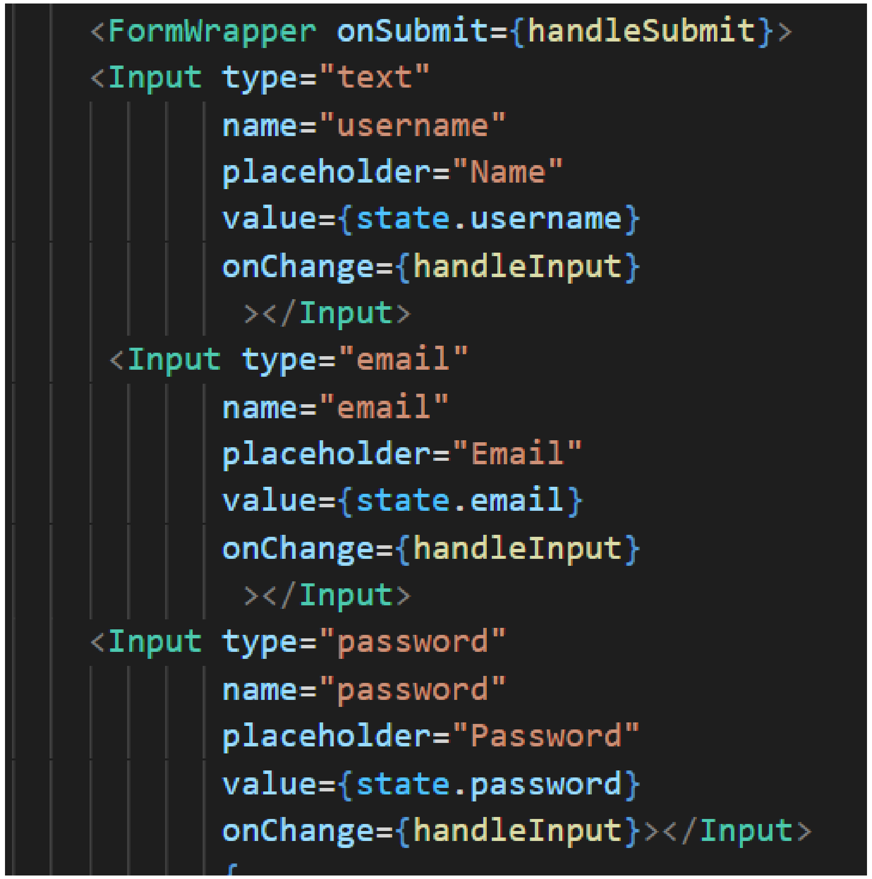


To customize the fields in login form, go to **src/components/Form/login.js:**



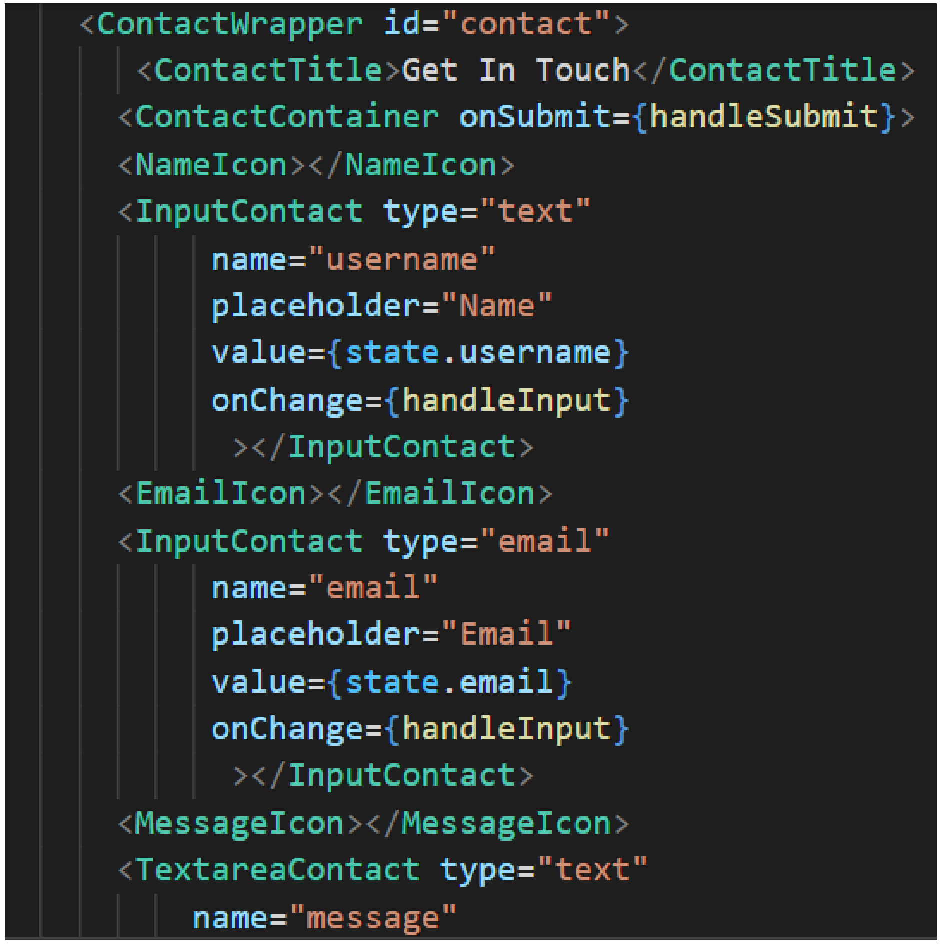
```
<FormWrapper onSubmit={handleSubmit} margin>
<Input type="email"
       name="email"
       placeholder="Email"
       value={state.email}
       onChange={handleInput}
        ></Input>
<Input type="password"
       name="password"
       placeholder="Password"
       value={state.password}
       onChange={handleInput}></Input>
       {
```

For register form fields, go to **src/components/Form/register.js:**



```
<FormWrapper onSubmit={handleSubmit}>
<Input type="text"
      name="username"
      placeholder="Name"
      value={state.username}
      onChange={handleInput}
       ></Input>
 <Input type="email"
      name="email"
      placeholder="Email"
      value={state.email}
      onChange={handleInput}
       ></Input>
<Input type="password"
      name="password"
      placeholder="Password"
      value={state.password}
      onChange={handleInput}></Input>
```

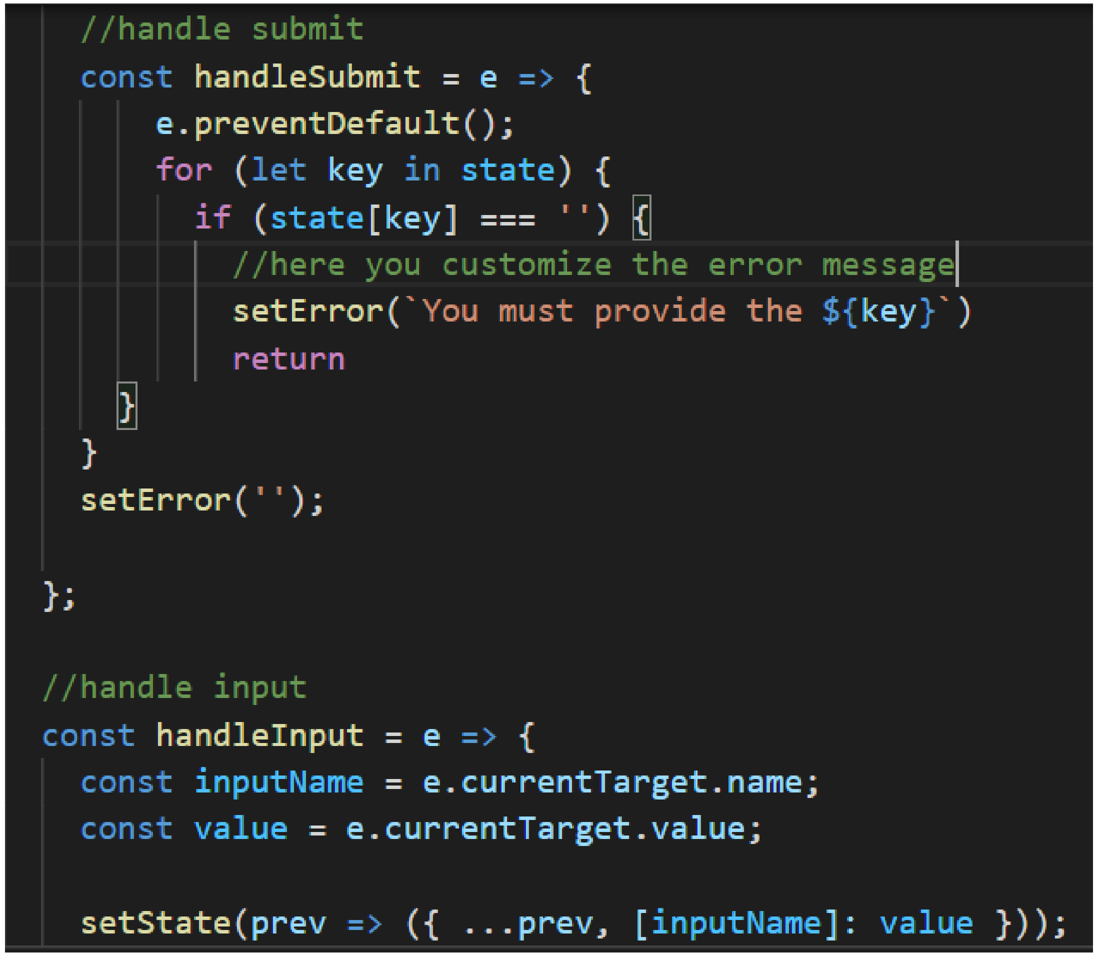
To customize the Contact form fields go to **src/components/Contact/index.js:**



```
<ContactWrapper id="contact">
  <ContactTitle>Get In Touch</ContactTitle>
  <ContactContainer onSubmit={handleSubmit}>
  <NameIcon></NameIcon>
  <InputContact type="text"
      name="username"
      placeholder="Name"
      value={state.username}
      onChange={handleInput}
      ></InputContact>
  <EmailIcon></EmailIcon>
  <InputContact type="email"
      name="email"
      placeholder="Email"
      value={state.email}
      onChange={handleInput}
      ></InputContact>
  <MessageIcon></MessageIcon>
  <TextareaContact type="text"
      name="message"
```

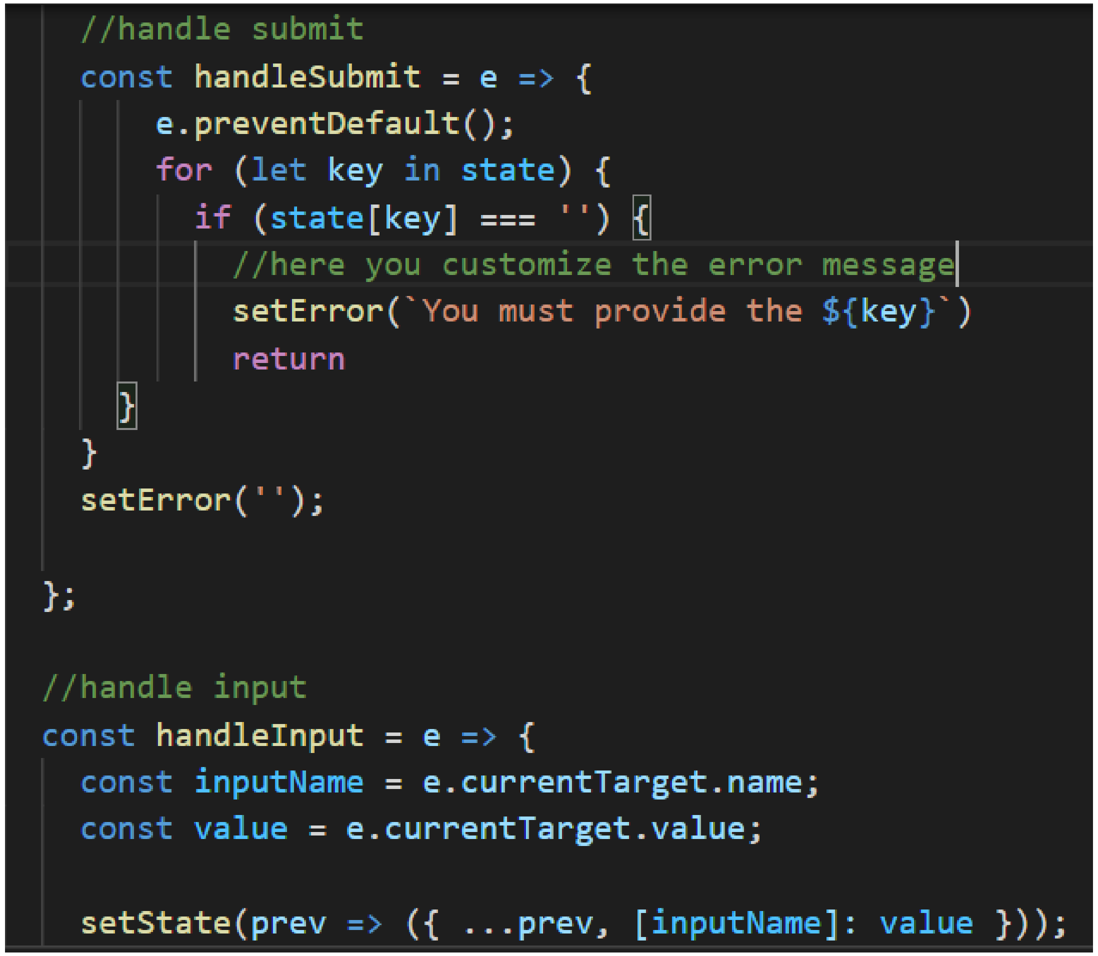
Form Validation

The login form validation is done in **src/components/Form/Login.js**:

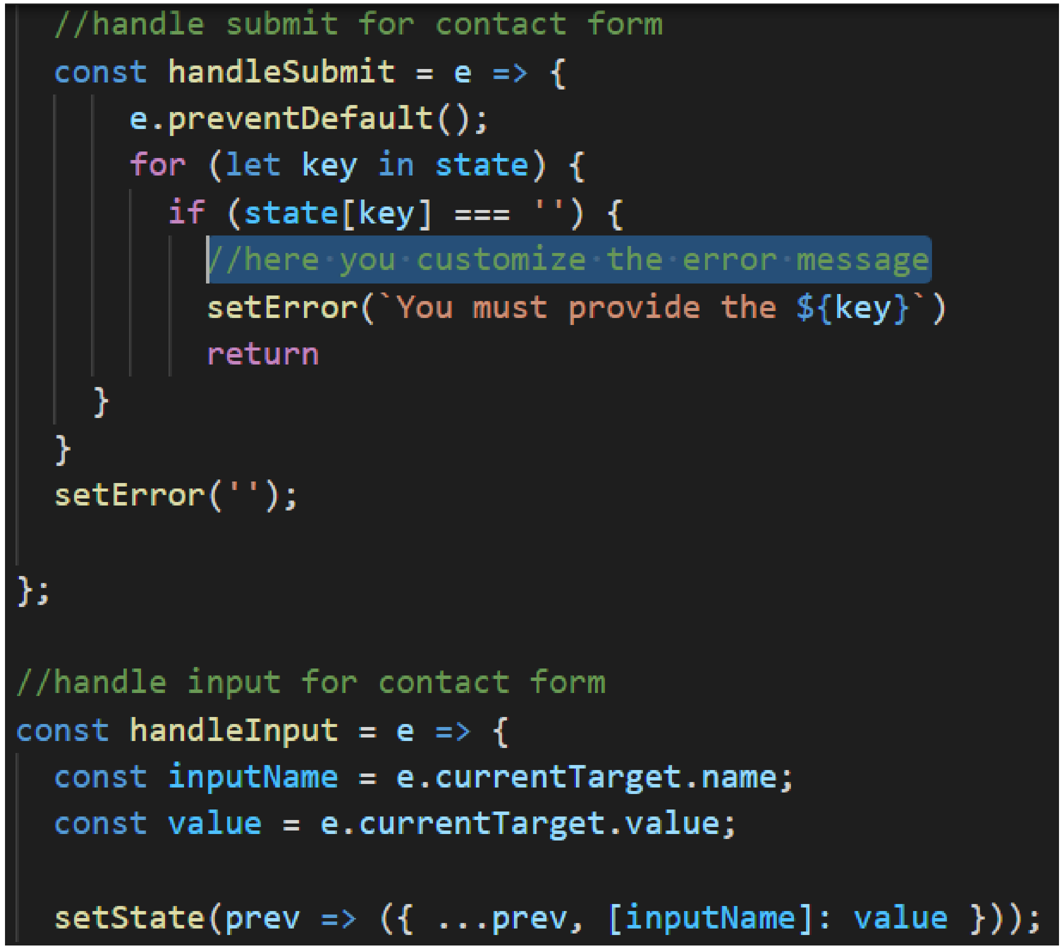


```
  //handle submit
  const handleSubmit = e => {
      e.preventDefault();
      for (let key in state) {
        if (state[key] === '') {
          //here you customize the error message
          setError(`You must provide the ${key}`)
          return
        }
    }
  setError('');

};

//handle input
const handleInput = e => {
  const inputName = e.currentTarget.name;
  const value = e.currentTarget.value;

  setState(prev => ({ ...prev, [inputName]: value }));
```

The register form validation is done in **src/components/Form/Register.js**:


```
  //handle submit
  const handleSubmit = e => {
      e.preventDefault();
      for (let key in state) {
        if (state[key] === '') {
          //here you customize the error message
          setError(`You must provide the ${key}`)
          return
        }
    }
  setError('');

};

//handle input
const handleInput = e => {
  const inputName = e.currentTarget.name;
  const value = e.currentTarget.value;

  setState(prev => ({ ...prev, [inputName]: value }));
```

The contact form validation is done in **src/components/Contact/index.js**:



```
  //handle submit for contact form
  const handleSubmit = e => {
      e.preventDefault();
      for (let key in state) {
        if (state[key] === '') {
          //here·you·customize·the·error·message
          setError(`You must provide the ${key}`)
          return
        }
    }
  setError('');

};

//handle input for contact form
const handleInput = e => {
  const inputName = e.currentTarget.name;
  const value = e.currentTarget.value;

  setState(prev => ({ ...prev, [inputName]: value }));
```

To customize the forms styled components, you go to **src/components/Form/FormComponents**

Each styled component is well commented, so you can change the style according to your needs.

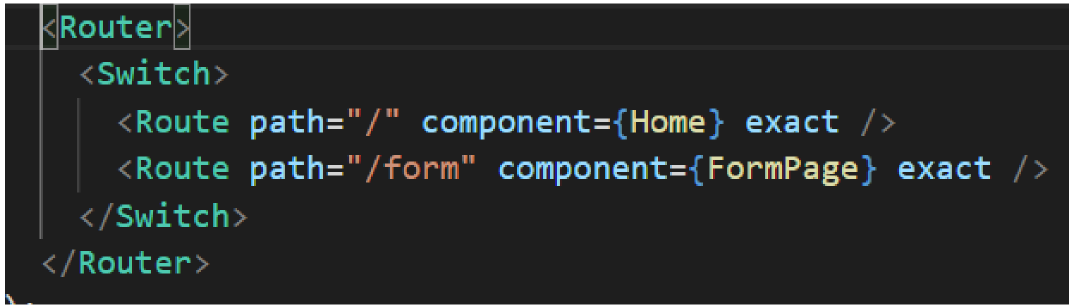
WORKING WITH LINKS #BACK TO TOP

We use 2 types of links in the project: router links and scroll links

Router Links

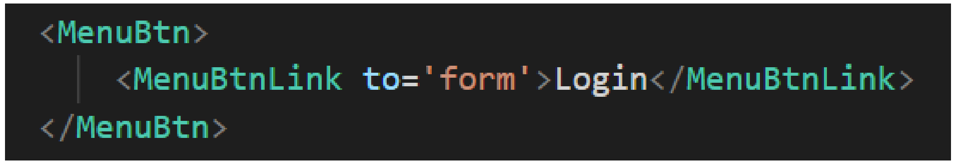
Router links are used to navigate to a different page. You can read more about it here.

In our template, we have 2 pages handled by the router: the home page, and the form page. You can customize the routing structure in **src/App.js:**



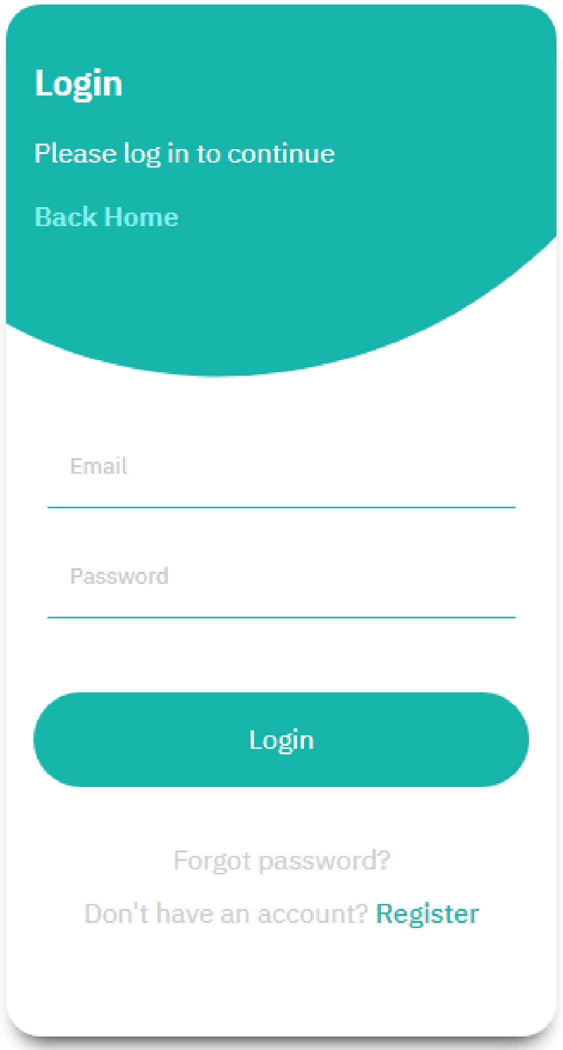
```
<Router>
  <Switch>
    <Route path="/" component={Home} exact />
    <Route path="/form" component={FormPage} exact />
  </Switch>
</Router>
```

The login/register forms open in a new page. see the Login button in the Navbar:

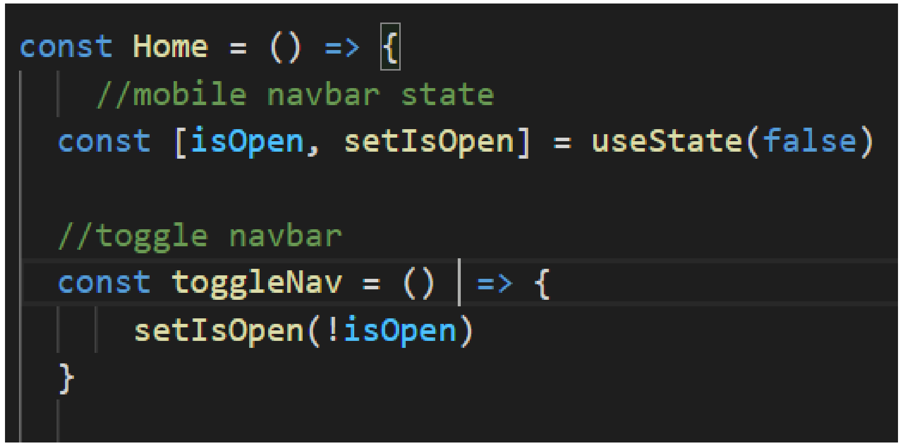


```
<MenuBtn>
    <MenuBtnLink to='form'>Login</MenuBtnLink>
</MenuBtn>
```

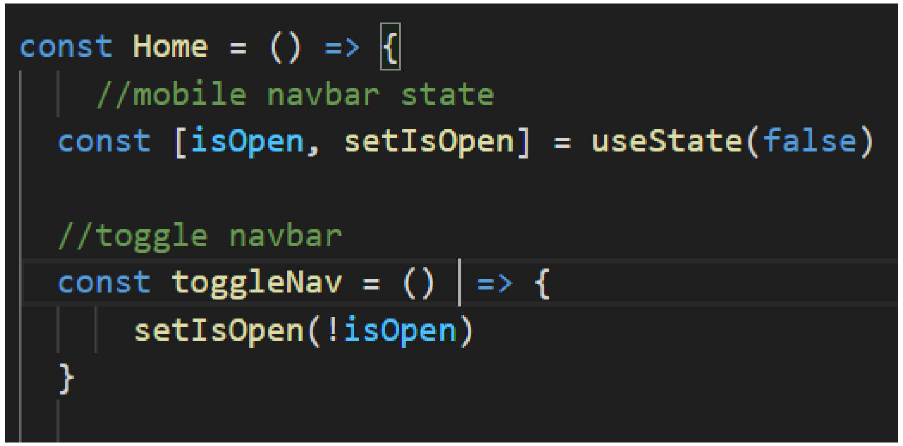
To switch from login to registration form, we need to click on register link in the bottom. We use framer-motion to change the forms:



The home page, **src/pages/index.js** is where all the components are rendered. Also, here we handle the toggle navbar states:


```
const Home = () => {
    //mobile navbar state
  const [isOpen, setIsOpen] = useState(false)

  //toggle navbar
  const toggleNav = () => {
    setIsOpen(!isOpen)
  }
```

The formPage page, **src/pages/formPage.js** handles the login and register forms

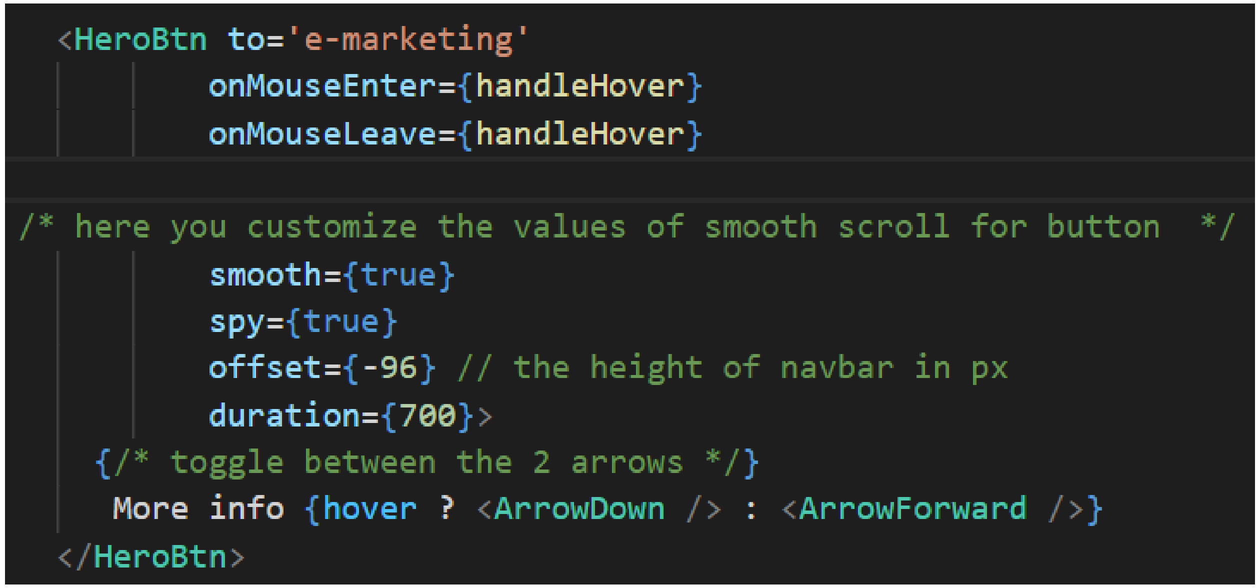


```
const Home = () => {
    //mobile navbar state
  const [isOpen, setIsOpen] = useState(false)

  //toggle navbar
  const toggleNav = () => {
    setIsOpen(!isOpen)
  }
```

Scroll Links

You can read more about scroll links here.

We use scroll links to scroll to a certain link on the same page. E.g.



```
  <HeroBtn to='e-marketing'
          onMouseEnter={handleHover}
          onMouseLeave={handleHover}

/* here you customize the values of smooth scroll for button  */
          smooth={true}
          spy={true}
          offset={-96} // the height of navbar in px
          duration={700}>
    {/* toggle between 2 arrows */}
     More info {hover ? <ArrowDown /> : <ArrowForward />}
  </HeroBtn>
```

Thank you for purchasing Alphcons template.
Best Regards!