Birkbeck University of London

Film, Media and Cultural Studies

WEB PROJECT ASSIGNMENT

FINAL REPORT

Nikolina Marinova Marinova

Pg Cert Web Design and Development

Student's number: 13182119, Mobile: 07801082832,

E-mail: nikolina_marinova@yahoo.com

https://giacomosorbi.github.io/nikolina-marinova3-module-ii/

Academic Advisers:

Giacomo Sorbi, Bhavik Sheth

2020

Table of Contents

Project Details

Project

Design and development of a compelling, functional, responsible and interactive website suitable for a company for flower bulbs and corms.

Technical details

The website development is implemented in **HTML**, **CSS**, **Java Script**, **and React**.

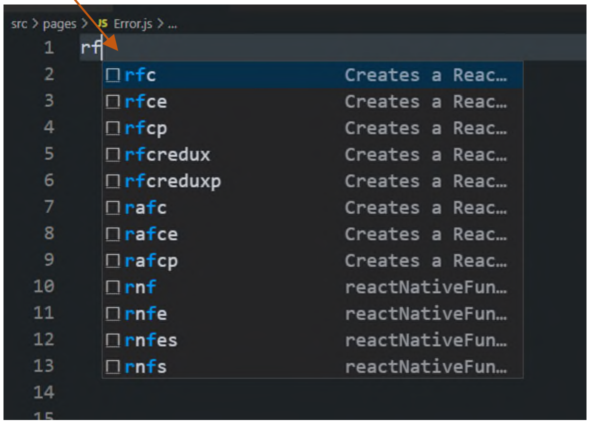
General Notes

Editor's settings and Resources

1. **Visual Studio Code extensions:**
   1.1. **Prettier**
   1.2. **AutoClose Tag**
   1.3. **Auto Rename Tag**
   1.4. **Path Intelisense**
   1.5. **Paste and Indent**
   1.6. **Color Highlight**
   1.7. **ES7 React/Redux/Graph snippets** – for quick creation of components**.** For example:

   rafc and **rfc** for functional components and **rcc** for class components.

   Most of the components in the project are functional components.



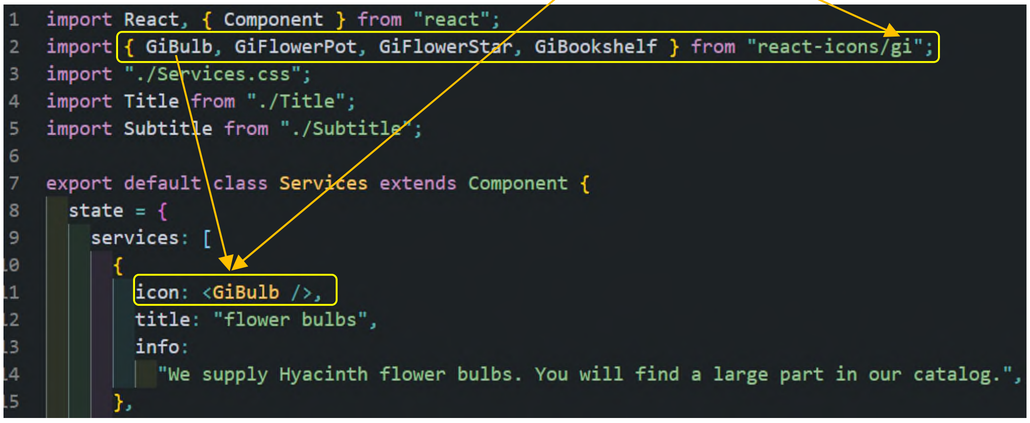
```
src > pages > Error.js > ...
rf
  rfc          Creates a Reac…
  rfce         Creates a Reac…
  rfcp         Creates a Reac…
  rfcredux     Creates a Reac…
  rfcreduxp    Creates a Reac…
  rafc         Creates a Reac…
  rafce        Creates a Reac…
  rafcp        Creates a Reac…
  rnf          reactNativeFun…
  rnfe         reactNativeFun…
  rnfes        reactNativeFun…
  rnfs         reactNativeFun…
```

2. **Images** are downloaded from **The Flower Council of Holland - Image Bank**. (https://www.flowercouncil.co.uk/image-bank). On requirement of the database a link is included in the footer.
3. The project uses **local data** from **data.js**, however the data inside follows the **structure of a external data based generator**. Data can be changed any time.
4. **React icons** are included from the URL below following the instructions.
   4.1. https://react-icons.github.io/react-icons/
   4.2. https://react-icons.github.io/icons?name=gi – **an icon of a flower bulb**.

   Installation

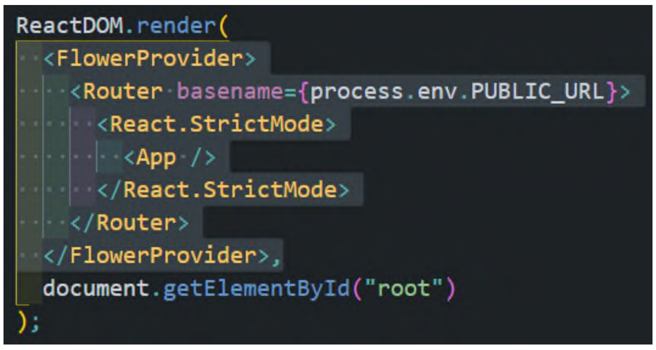
```
yarn add react-icons --save
```

   Usage in the project



```
import React, { Component } from "react";
import { GiBulb, GiFlowerPot, GiFlowerStar, GiBookshelf } from "react-icons/gi";
import "./Services.css";
import Title from "./Title";
import Subtitle from "./Subtitle";

export default class Services extends Component {
  state = {
    services: [
      {
        icon: <GiBulb />,
        title: "flower bulbs",
        info:
          "We supply Hyacinth flower bulbs. You will find a large part in our catalog.",
      },
```

5. Router setup for **routing** and navigation - **react-router-dom –** is used following the documentation on its creators' website below:
   https://reacttraining.com/react-router/web/guides/quick-start
6. The browser router is wrapping the whole React component tree.
7.

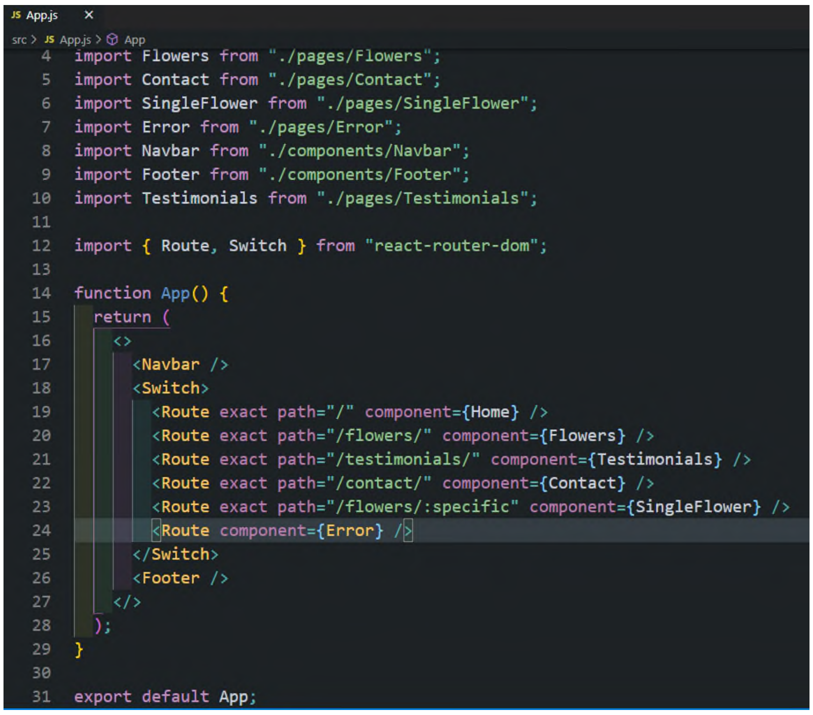


```
ReactDOM.render(
  <FlowerProvider>
    <Router basename={process.env.PUBLIC_URL}>
      <React.StrictMode>
        <App />
      </React.StrictMode>
    </Router>
  </FlowerProvider>,
  document.getElementById("root")
);
```

Route, **Switch**, **Link** from "react-router-dom" are also used in this project.

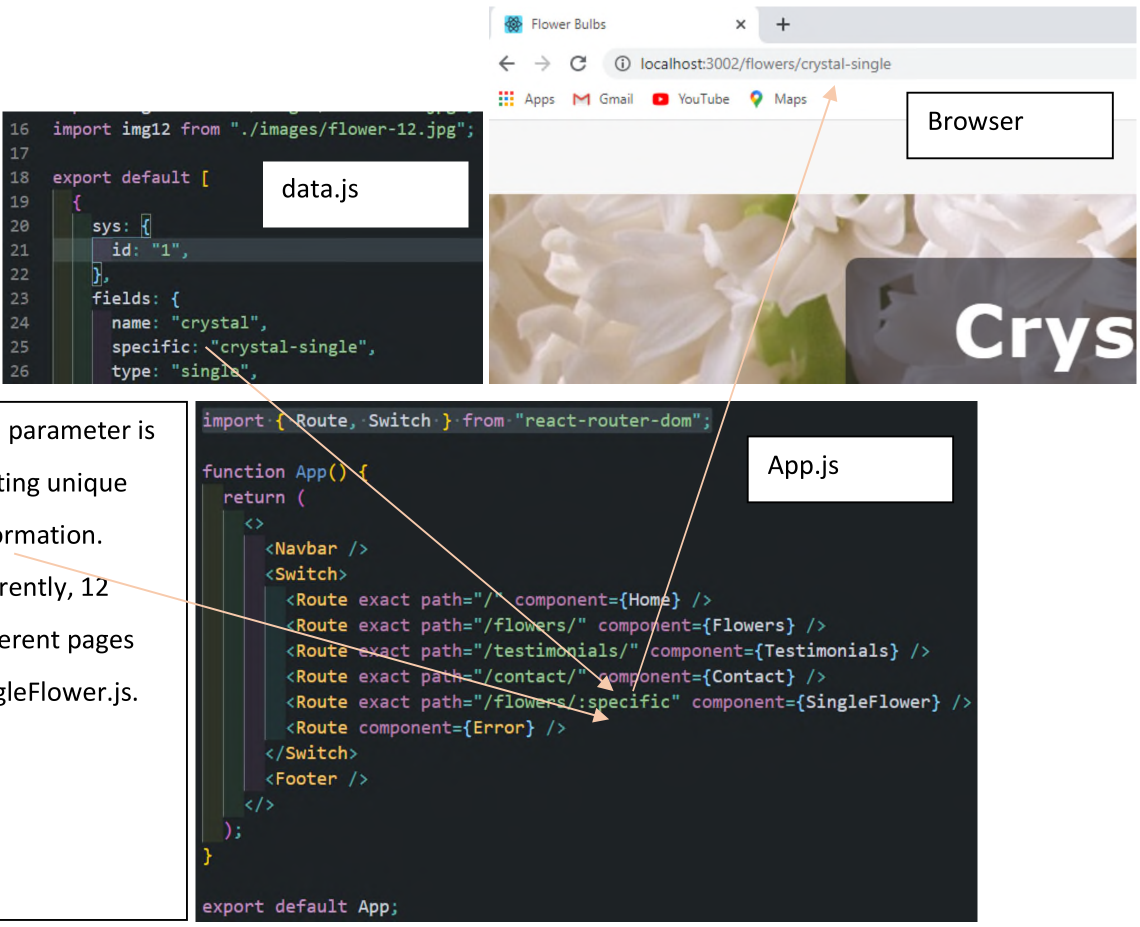
The project has **5 pages** including an **Error 404 page** and **12 different Single flower pages (SingleFlower.js)** generated from **data.js** with the help of **context.js.**

<Navbar /> has a special position on the top and around **<Switch>** since it appears on every single page and is used for navigation. The parameter **:specific** is used to access the 12 different pages and exists in the **data.js file.**



```
import Flowers from "./pages/Flowers";
import Contact from "./pages/Contact";
import SingleFlower from "./pages/SingleFlower";
import Error from "./pages/Error";
import Navbar from "./components/Navbar";
import Footer from "./components/Footer";
import Testimonials from "./pages/Testimonials";

import { Route, Switch } from "react-router-dom";

function App() {
  return (
    <>
      <Navbar />
      <Switch>
        <Route exact path="/" component={Home} />
        <Route exact path="/flowers/" component={Flowers} />
        <Route exact path="/testimonials/" component={Testimonials} />
        <Route exact path="/contact/" component={Contact} />
        <Route exact path="/flowers/:specific" component={SingleFlower} />
        <Route component={Error} />
      </Switch>
      <Footer />
    </>
  );
}

export default App;
```

Notes on Error 404 page. The method works locally, hoever is commented till further research is done on https://create-react-app.dev/docs/deployment/, when there is time.



```
import img12 from "./images/flower-12.jpg";

export default [
  {
    sys: {
      id: "1",
    },
    fields: {
      name: "crystal",
      specific: "crystal-single",
      type: "single",
```

The parameter is getting unique information. Currently, 12 different pages SingleFlower.js.

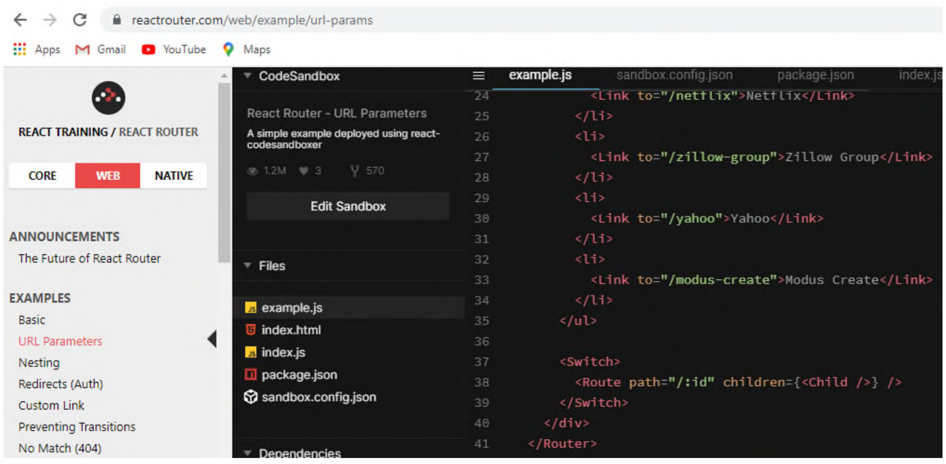
```
import { Route, Switch } from "react-router-dom";

function App() {
  return (
    <>
      <Navbar />
      <Switch>
        <Route exact path="/" component={Home} />
        <Route exact path="/flowers/" component={Flowers} />
        <Route exact path="/testimonials/" component={Testimonials} />
        <Route exact path="/contact/" component={Contact} />
        <Route exact path="/flowers/:specific" component={SingleFlower} />
        <Route component={Error} />
      </Switch>
      <Footer />
    </>
  );
}

export default App;
```

A **<Switch>** renders the first child **<Route>** that matches and a **<Route>** with no path always matches. In this case **Error 404** page will be loaded if ther is no match above.

https://reactrouter.com/web/guides/quick-start

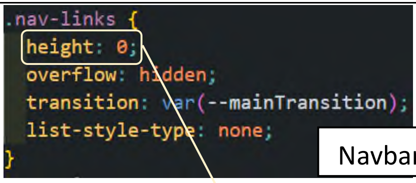
https://reactrouter.com/web/example/no-match

https://reactrouter.com/web/example/url-params

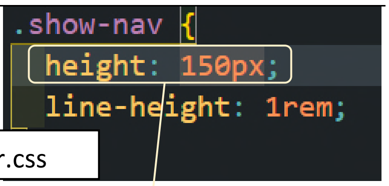


```
            <Link to="/netflix">Netflix</Link>
          </li>
          <li>
            <Link to="/zillow-group">Zillow Group</Link>
          </li>
          <li>
            <Link to="/yahoo">Yahoo</Link>
          </li>
          <li>
            <Link to="/modus-create">Modus Create</Link>
          </li>
        </ul>

        <Switch>
          <Route path="/:id" children={<Child />} />
        </Switch>
      </div>
    </Router>
```

Components – Navbar – Navbar.js

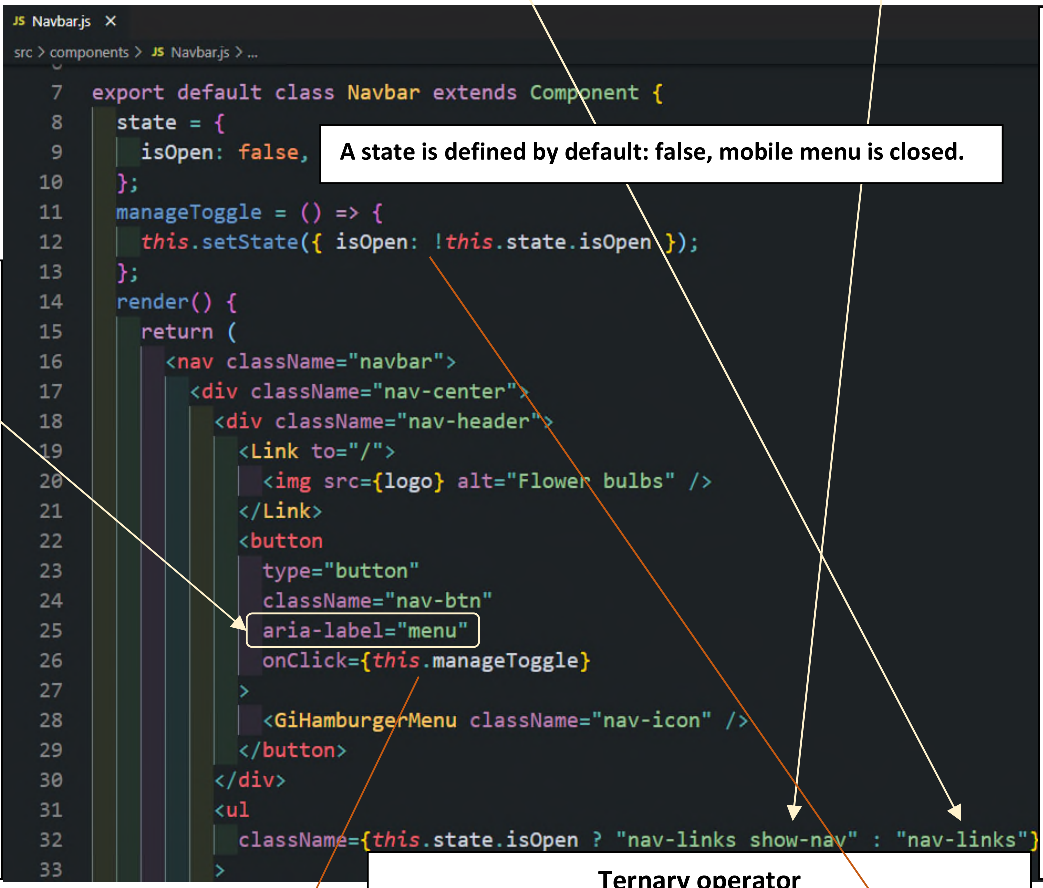


```css
.nav-links {
  height: 0;
  overflow: hidden;
  transition: var(--mainTransition);
  list-style-type: none;
}
```



```css
.show-nav {
  height: 150px;
  line-height: 1rem;
```

Navbar.css



```jsx
export default class Navbar extends Component {
  state = {
    isOpen: false,
  };
  manageToggle = () => {
    this.setState({ isOpen: !this.state.isOpen });
  };
  render() {
    return (
      <nav className="navbar">
        <div className="nav-center">
          <div className="nav-header">
            <Link to="/">
              <img src={logo} alt="Flower bulbs" />
            </Link>
            <button
              type="button"
              className="nav-btn"
              aria-label="menu"
              onClick={this.manageToggle}
            >
              <GiHamburgerMenu className="nav-icon" />
            </button>
          </div>
          <ul
            className={this.state.isOpen ? "nav-links show-nav" : "nav-links"}
          >
```

A state is defined by default: false, mobile menu is closed.

aria-label=""

aria-label was included after a requirement from **Audits** for **Accessibility purposes.**

.nav-links

The mobile menu is hidden, since **height: 0;** (in my previous module project is used **width: 0;** which is all the same for hiding a container).

.show-nav

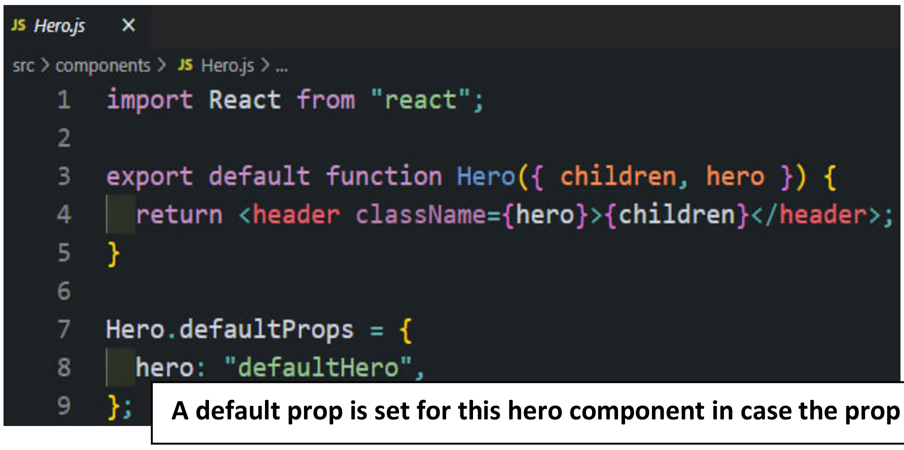
The mobile menu is visible since **height: 150px;**

Ternary operator

The function manageToggle is called onClick and is toggling between true and false.

Components – Hero – Hero.js

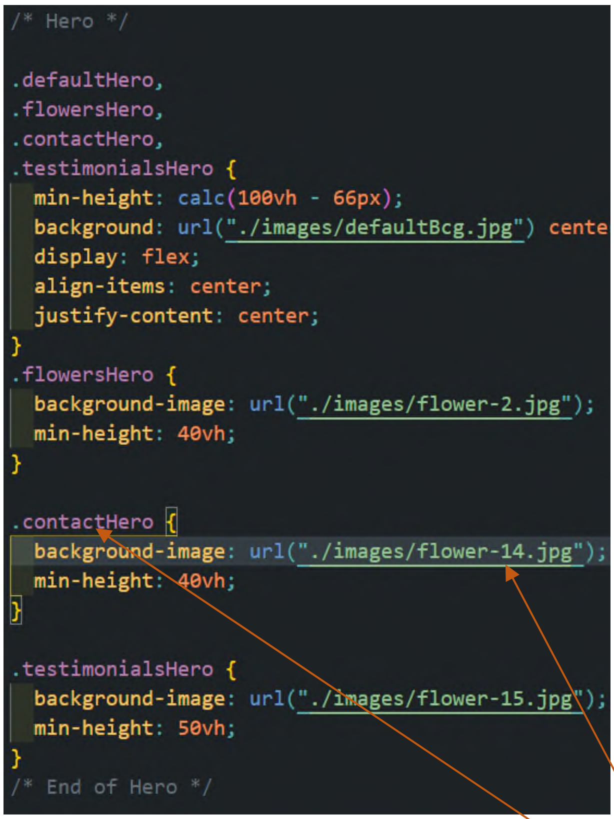
Hero component is defined and is reused for each and every single page. The functional Hero component is created by typing **rfc**. There are two **props children** for rendering a **banner** and a **button** inside and the **second** is for the **hero**.



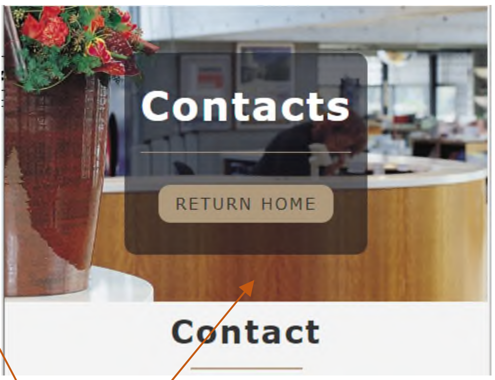
```
Hero.js
src > components > Hero.js > ...
import React from "react";

export default function Hero({ children, hero }) {
  return <header className={hero}>{children}</header>;
}

Hero.defaultProps = {
  hero: "defaultHero",
};
```

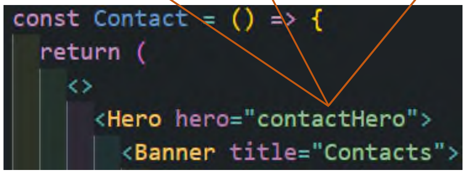
A default prop is set for this hero component in case the prop is forgot.

So each and every time while rendering this hero component there will be an option of changing the class name. **Examples** for the changes in the **CSS** code below. **First example** in **Contact Page**.



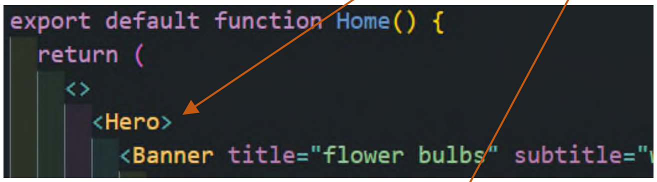
```
/* Hero */

.defaultHero,
.flowersHero,
.contactHero,
.testimonialsHero {
  min-height: calc(100vh - 66px);
  background: url("./images/defaultBcg.jpg") cente
  display: flex;
  align-items: center;
  justify-content: center;
}
.flowersHero {
  background-image: url("./images/flower-2.jpg");
  min-height: 40vh;
}

.contactHero {
  background-image: url("./images/flower-14.jpg");
  min-height: 40vh;
}

.testimonialsHero {
  background-image: url("./images/flower-15.jpg");
  min-height: 50vh;
}
/* End of Hero */
```



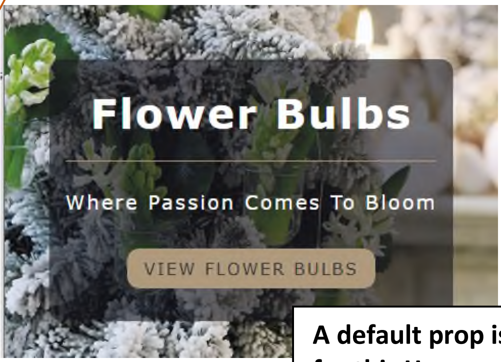


```
const Contact = () => {
  return (
    <>
      <Hero hero="contactHero">
        <Banner title="Contacts">
```

Second example in **Home** and **Error 404 pages**. Prop is **not defined** in Home and Error Pages that is why the **default** prop is called.



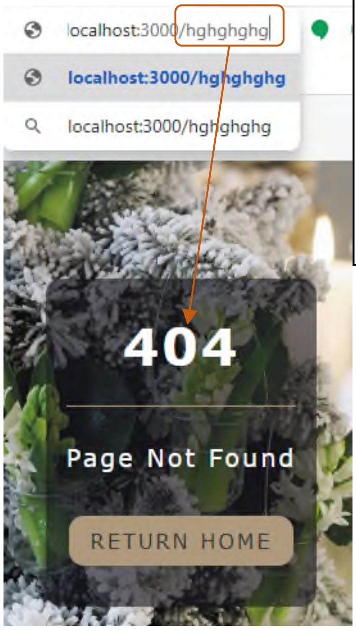
```
export default function Home() {
  return (
    <>
      <Hero>
        <Banner title="flower bulbs" subtitle="
```

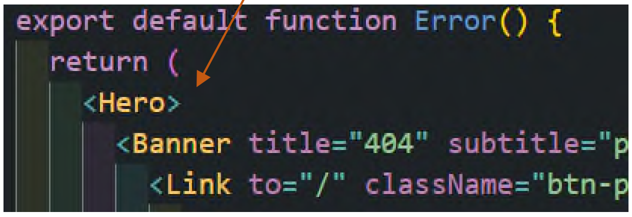


A default prop is set for this Hero component in case the prop is forgot.

The image is called

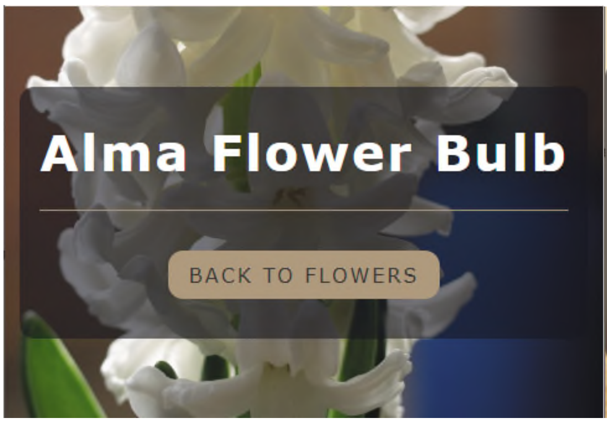
defaultBcg.jpg and also appear in **Home** and **Error 404 pages.**

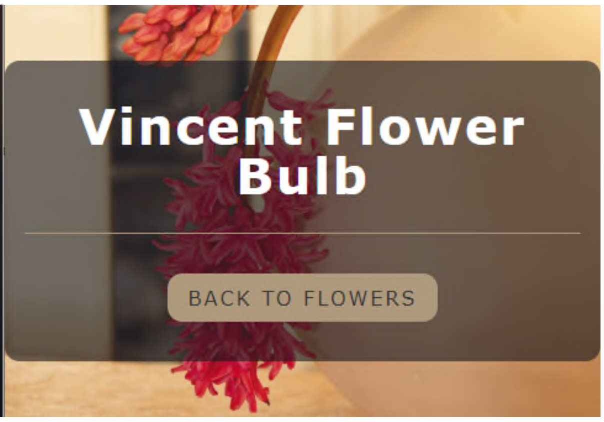




```
export default function Error() {
  return (
    <Hero>
      <Banner title="404" subtitle="p
        <Link to="/" className="btn-p
```

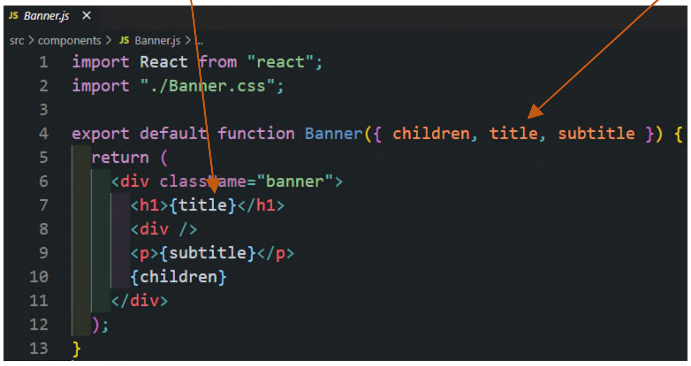
Every single page of the **12 flower pages** has its own **specific** hero images with the **unique** flower names.





Components – Banner – Banner.js

The **Banner.js** is a functional component with three props – **title**, **subtitle** and **children** for everything that is defined inside the component. They are listed **alphabetically**: children, title, subtitle, but can be used in a **different way**. In this case – title, subtitle and children.

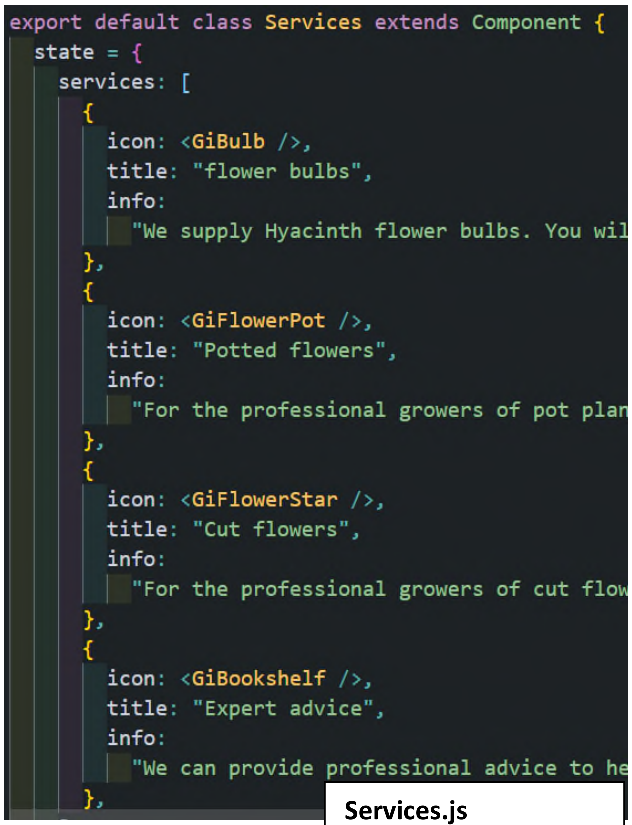


```
import React from "react";
import "./Banner.css";

export default function Banner({ children, title, subtitle }) {
  return (
    <div className="banner">
      <h1>{title}</h1>
      <div />
      <p>{subtitle}</p>
      {children}
    </div>
  );
}
```

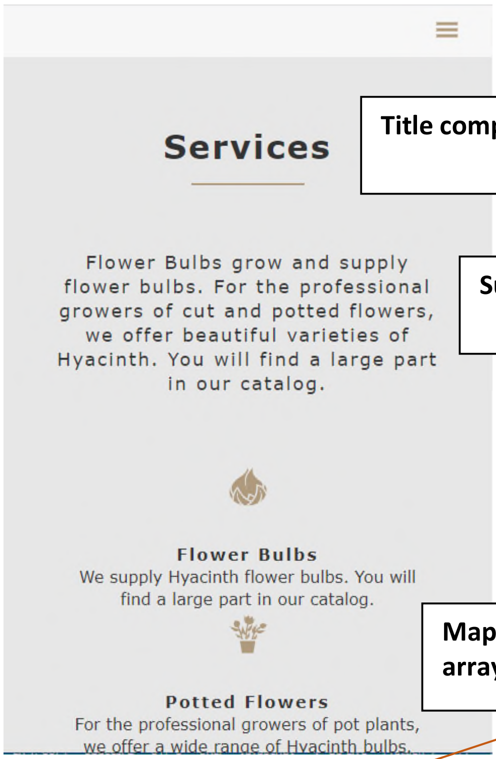
Components – Services – Services.js

Component Services uses **Title** and **Subtitle** components that are reused in the **Contact** and **Testimonials pages**. Array of objects is used to take the data for this Services component.

Testimonial and Contact pages work on the same principal as Services.



```
export default class Services extends Component {
  state = {
    services: [
      {
        icon: <GiBulb />,
        title: "flower bulbs",
        info:
          "We supply Hyacinth flower bulbs. You wil
      },
      {
        icon: <GiFlowerPot />,
        title: "Potted flowers",
        info:
          "For the professional growers of pot plan
      },
      {
        icon: <GiFlowerStar />,
        title: "Cut flowers",
        info:
          "For the professional growers of cut flow
      },
      {
        icon: <GiBookshelf />,
        title: "Expert advice",
        info:
          "We can provide professional advice to he
      },
```



Services

Flower Bulbs grow and supply flower bulbs. For the professional growers of cut and potted flowers, we offer beautiful varieties of Hyacinth. You will find a large part in our catalog.

Flower Bulbs
We supply Hyacinth flower bulbs. You will find a large part in our catalog.

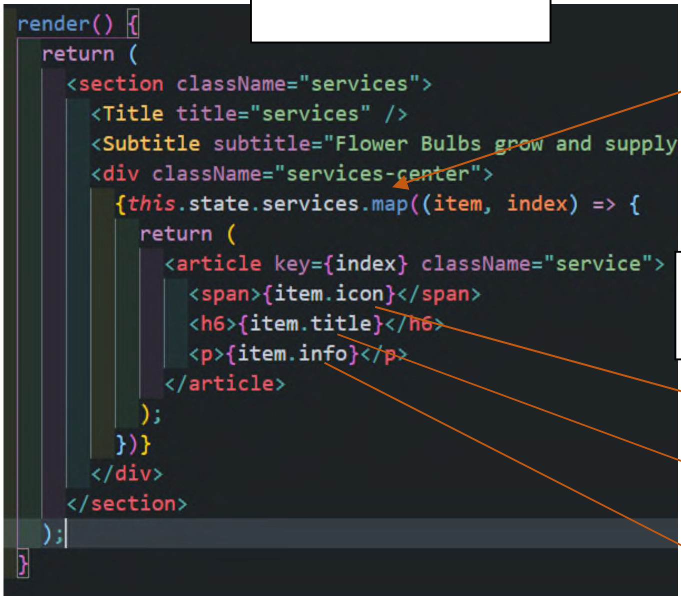
Potted Flowers
For the professional growers of pot plants, we offer a wide range of Hyacinth bulbs.

Title component.

Subtitle component.

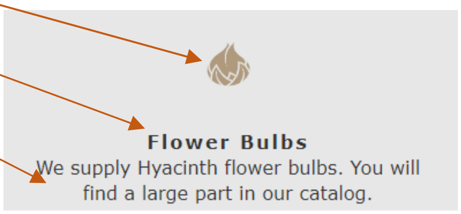
Mapping over the array of objects.

Services.js



```
render() {
  return (
    <section className="services">
      <Title title="services" />
      <Subtitle subtitle="Flower Bulbs grow and supply
      <div className="services-center">
        {this.state.services.map((item, index) => {
          return (
            <article key={index} className="service">
              <span>{item.icon}</span>
              <h6>{item.title}</h6>
              <p>{item.info}</p>
            </article>
          );
        })}
      </div>
    </section>
  );
}
```

An article for each and every item in the array is returned.

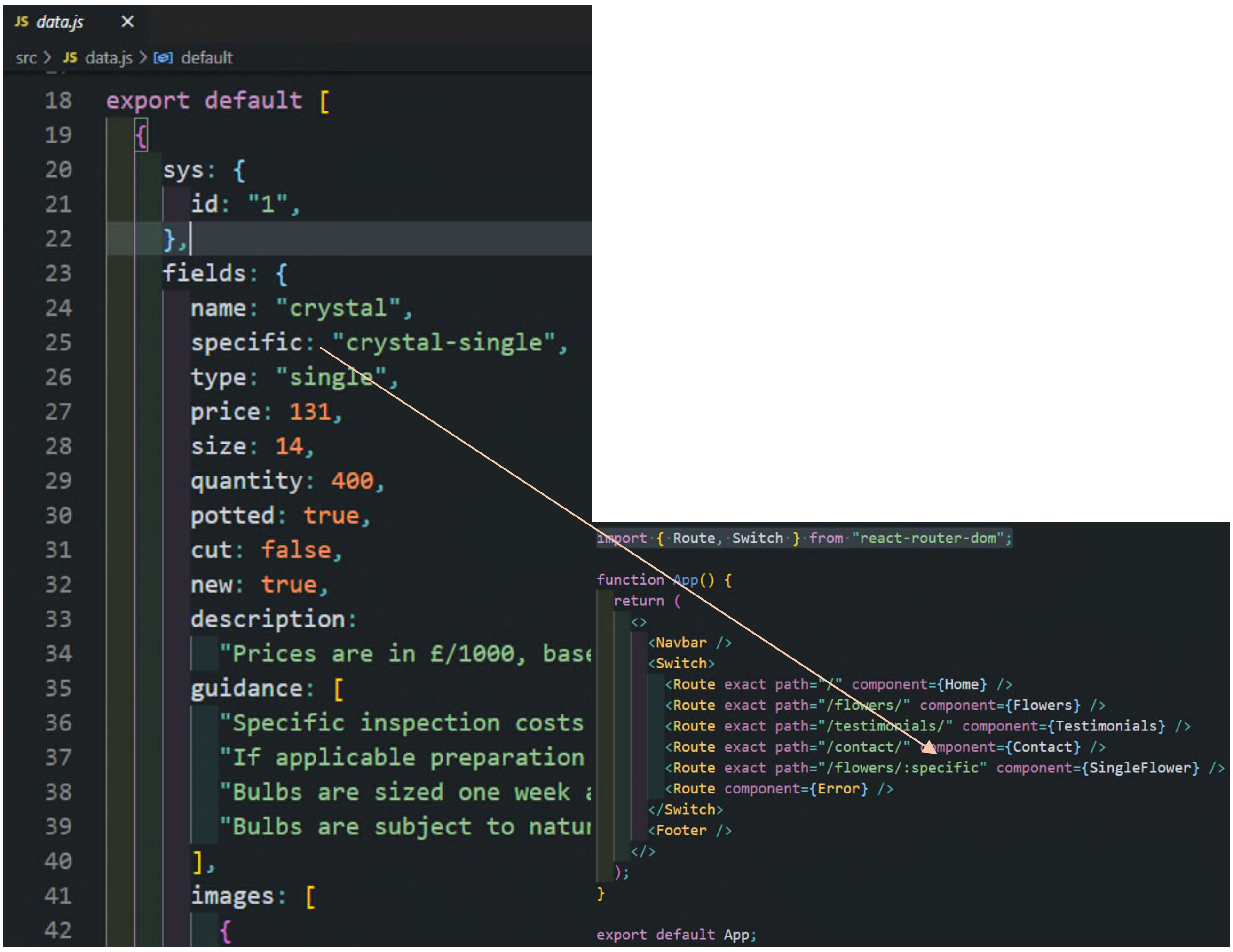


Flower Bulbs
We supply Hyacinth flower bulbs. You will find a large part in our catalog.

Local Data – data.js

Component Services uses **Title** and **Subtitle** components that are reused in the **Contact** and **Testimonials pages**. Array of objects is used to take the data for this Services component.

data.js is a array of 12 objects**.** Every object contains different information for 12 different pages.



```
export default [
  {
    sys: {
      id: "1",
    },
    fields: {
      name: "crystal",
      specific: "crystal-single",
      type: "single",
      price: 131,
      size: 14,
      quantity: 400,
      potted: true,
      cut: false,
      new: true,
      description:
        "Prices are in £/1000, base
      guidance: [
        "Specific inspection costs
        "If applicable preparation
        "Bulbs are sized one week a
        "Bulbs are subject to natur
      ],
      images: [
        {
```

```
import { Route, Switch } from "react-router-dom";

function App() {
  return (
    <>
      <Navbar />
      <Switch>
        <Route exact path="/" component={Home} />
        <Route exact path="/flowers/" component={Flowers} />
        <Route exact path="/testimonials/" component={Testimonials} />
        <Route exact path="/contact/" component={Contact} />
        <Route exact path="/flowers/:specific" component={SingleFlower} />
        <Route component={Error} />
      </Switch>
      <Footer />
    </>
  );
}

export default App;
```

Context API Data – context.js

The data is structured in **data.js,** however **context.js** is responsible for distributing the data. Distribution of the data will be implemented following the instructions of React Context API on:

https://reactjs.org/docs/context.html

"In a typical React application, data is passed top-down (parent to child) via props, but this can be cumbersome for certain types of props (e.g. locale preference, UI theme) that are required by many components within an application. Context provides a way to share values like these between components without having to explicitly pass a prop through every level of the tree." (Facebook open source, 2020) https://reactjs.org/docs/context.html

"Context is designed to share data that can be considered "global" for a tree of React components…." (Facebook open source, 2020) https://reactjs.org/docs/context.html

API - Theory

1. `React.createContext`

```
const MyContext = React.createContext(defaultValue);
```

We have a method available for creating the context – createContext().

In the Project

```
const FlowerContext = React.createContext();
```

"Creates a Context object. When React renders a component that subscribes to this Context object it will read the current context value from the closest matching Provider above it in the tree." (Facebook open source, 2020) https://reactjs.org/docs/context.html

After creating the Context we have access to two components – a Provider and a Consumer:

Theory

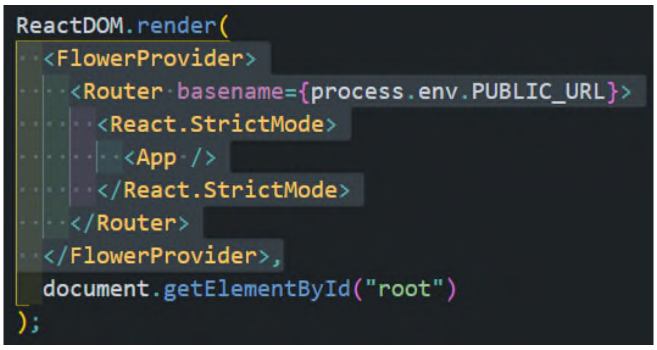
2. `Context.Provider`

```
<MyContext.Provider value={/* some value */}>
```

"Every Context object comes with a Provider React component that allows consuming components to subscribe to context changes.

"Accepts a value prop to be passed to consuming components that are descendants of this Provider. One Provider can be connected to many consumers. Providers can be nested to override values deeper within the tree." (Facebook open source, 2020) https://reactjs.org/docs/context.html

Provider is used to wrap the component tree and to deliver in order all the components to access to the information and Consumer is used to access this information. Basename is added to the Router.

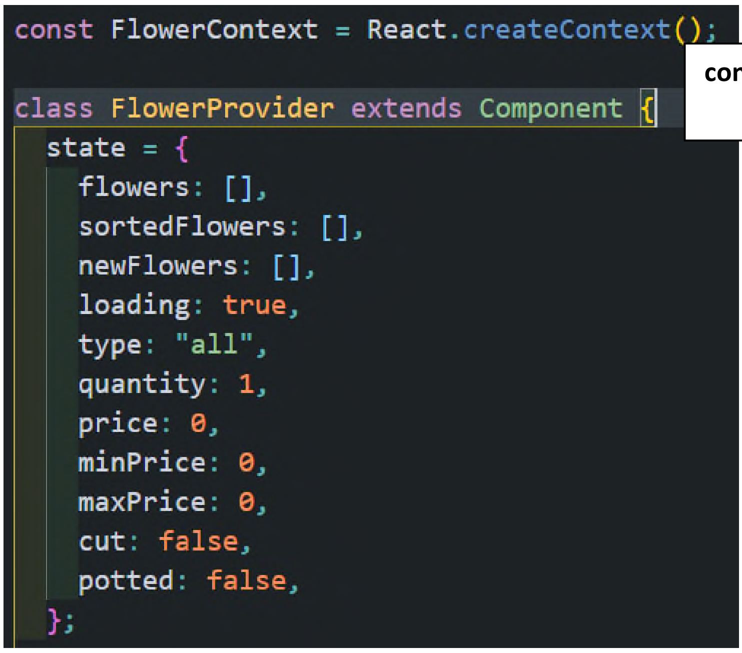


```
ReactDOM.render(
  <FlowerProvider>
    <Router basename={process.env.PUBLIC_URL}>
      <React.StrictMode>
        <App />
      </React.StrictMode>
    </Router>
  </FlowerProvider>,
  document.getElementById("root")
);
```

In the Project

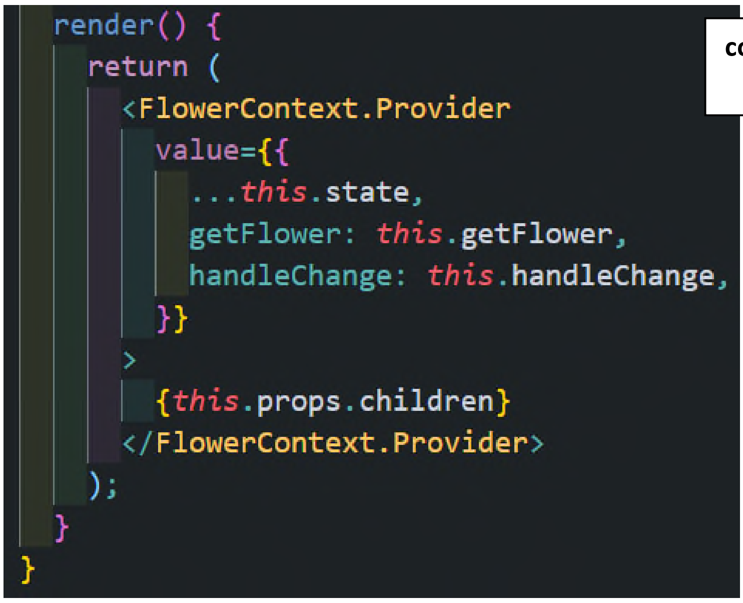
The Provider is called FlowerProvider and is a class-based component, because a state inside is defined it is been passed inside the value. An object is passed in the {value}

```
<MyContext.Provider value={/* some value */}>
```



```
const FlowerContext = React.createContext();

class FlowerProvider extends Component {
  state = {
    flowers: [],
    sortedFlowers: [],
    newFlowers: [],
    loading: true,
    type: "all",
    quantity: 1,
    price: 0,
    minPrice: 0,
    maxPrice: 0,
    cut: false,
    potted: false,
  };
```

context.js



```
  render() {
    return (
      <FlowerContext.Provider
        value={{
          ...this.state,
          getFlower: this.getFlower,
          handleChange: this.handleChange,
        }}
      >
        {this.props.children}
      </FlowerContext.Provider>
    );
  }
}
```

context.js

Second component is a Consumer called FlowerConsumer:

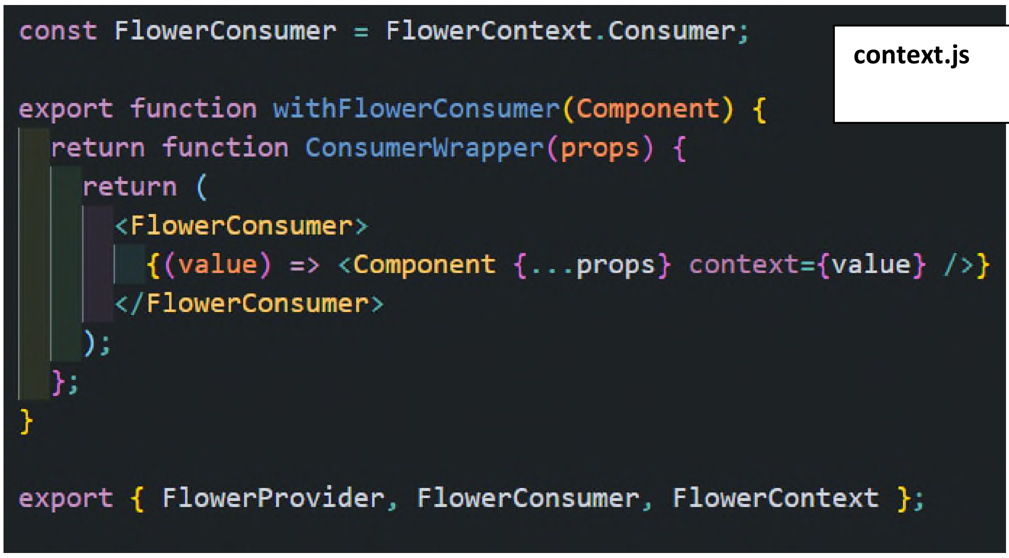
Theory

3. `Context.Consumer`

```
<MyContext.Consumer>
  {value => /* render something based on the context value */}
</MyContext.Consumer>
```

A React component that subscribes to context changes. This lets you subscribe to a context within a function component.

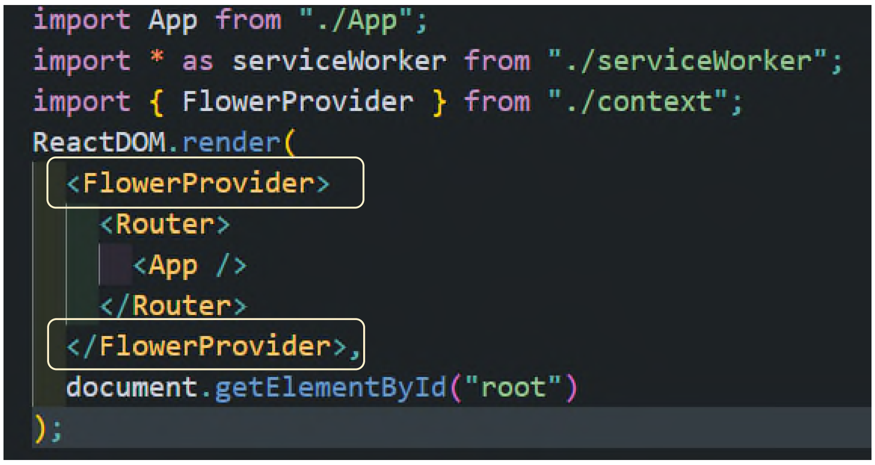
In the Project



```
const FlowerConsumer = FlowerContext.Consumer;

export function withFlowerConsumer(Component) {
  return function ConsumerWrapper(props) {
    return (
      <FlowerConsumer>
        {(value) => <Component {...props} context={value} />}
      </FlowerConsumer>
    );
  };
}

export { FlowerProvider, FlowerConsumer, FlowerContext };
```

context.js

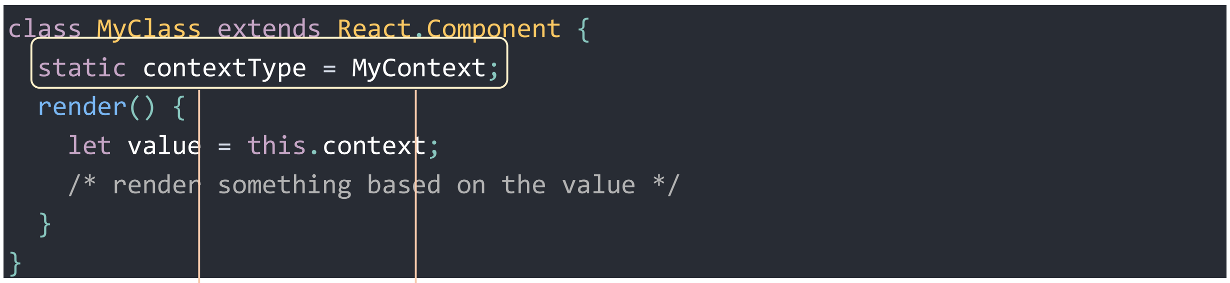
<FlowerProvider> wraps the whole **<App/>**



```
import App from "./App";
import * as serviceWorker from "./serviceWorker";
import { FlowerProvider } from "./context";
ReactDOM.render(
  <FlowerProvider>
    <Router>
      <App />
    </Router>
  </FlowerProvider>,
  document.getElementById("root")
);
```

The first example in using the context is in **NewFlowers** component. New Flowers component is placed in the end of Home page and the context is helping to access the props without drilling and passing the props that the component tree unnessesarily.

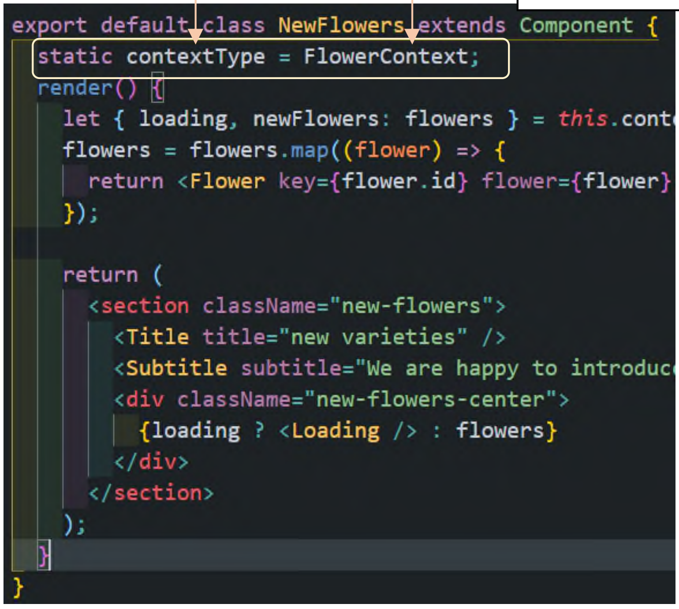
Theory



```
class MyClass extends React.Component {
  static contextType = MyContext;
  render() {
    let value = this.context;
    /* render something based on the value */
  }
}
```

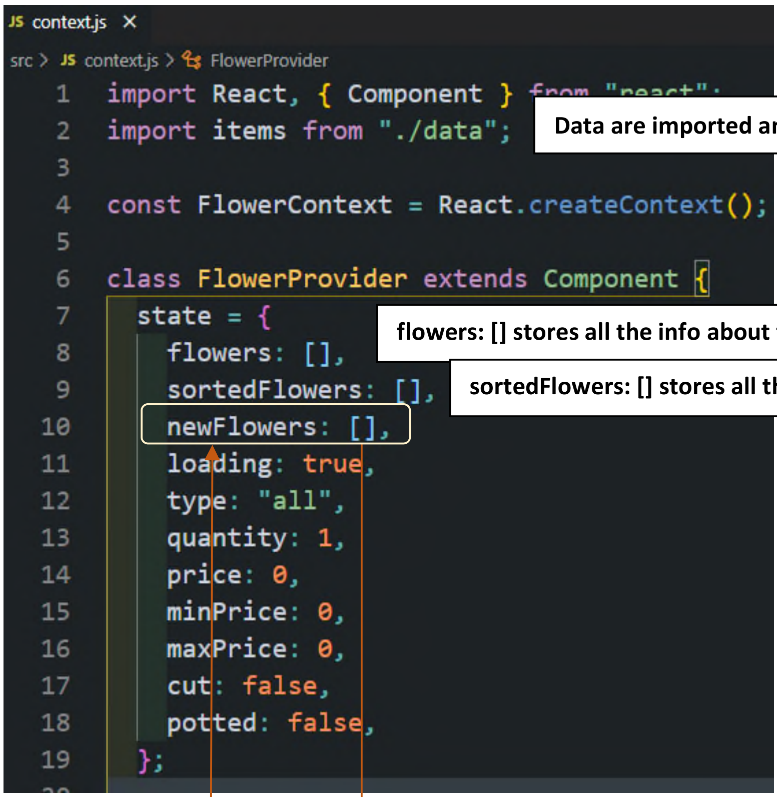
In the Project

NewFlowers.js



```
export default class NewFlowers extends Component {
  static contextType = FlowerContext;
  render() {
    let { loading, newFlowers: flowers } = this.cont
    flowers = flowers.map((flower) => {
      return <Flower key={flower.id} flower={flower}
    });

    return (
      <section className="new-flowers">
        <Title title="new varieties" />
        <Subtitle subtitle="We are happy to introduc
        <div className="new-flowers-center">
          {loading ? <Loading /> : flowers}
        </div>
      </section>
    );
  }
}
```

Format Data – context.js

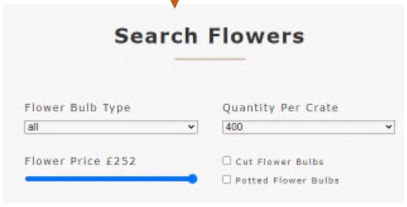


```
import React, { Component } from "react";
import items from "./data";

const FlowerContext = React.createContext();

class FlowerProvider extends Component {
  state = {
    flowers: [],
    sortedFlowers: [],
    newFlowers: [],
    loading: true,
    type: "all",
    quantity: 1,
    price: 0,
    minPrice: 0,
    maxPrice: 0,
    cut: false,
    potted: false,
  };
```

Data are imported and named ***items*.**

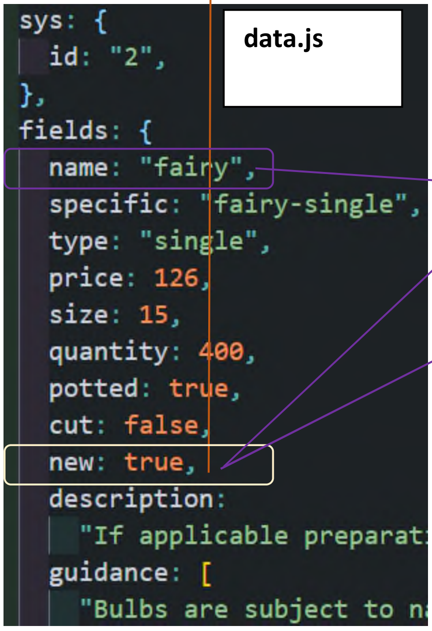
flowers: [] stores all the info about flowers

sortedFlowers: [] stores all the info after filtering



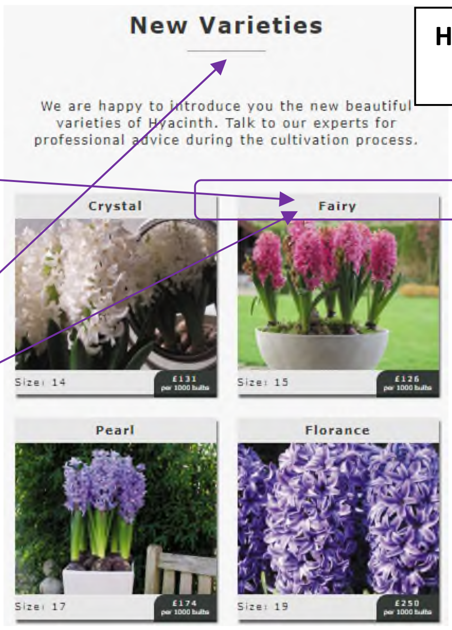
newFlowers: [] stores all the info in New Varieties Section

data.js

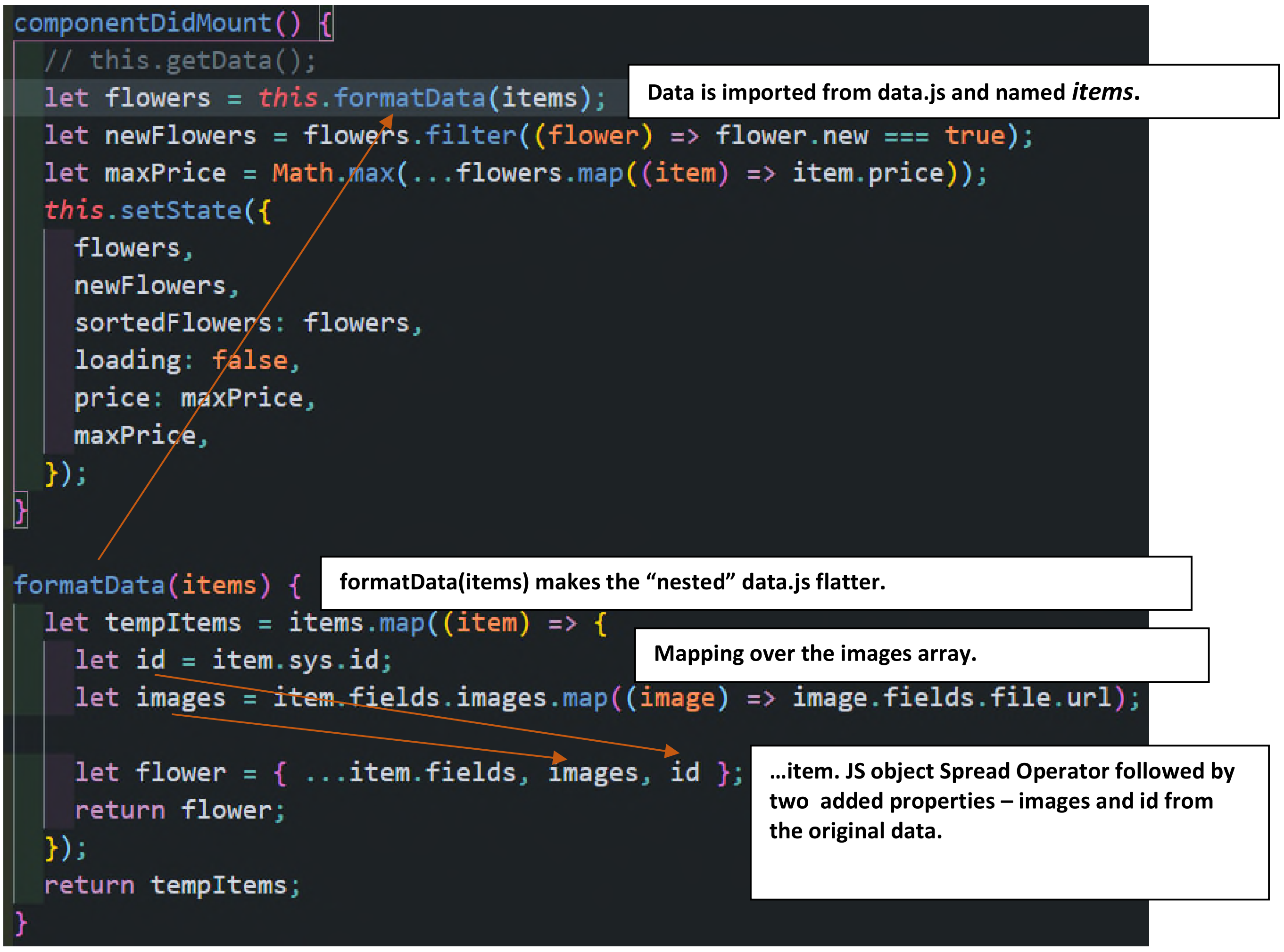


```
sys: {
  id: "2",
},
fields: {
  name: "fairy",
  specific: "fairy-single",
  type: "single",
  price: 126,
  size: 15,
  quantity: 400,
  potted: true,
  cut: false,
  new: true,
  description:
    "If applicable preparat
  guidance: [
    "Bulbs are subject to n
```

Home page


New Varieties
We are happy to introduce you the new beautiful varieties of Hyacinth. Talk to our experts for professional advice during the cultivation process.
Crystal
Size: 14
£131 per 1000 bulbs
Fairy
Size: 15
£126 per 1000 bulbs
Pearl
Size: 17
£174 per 1000 bulbs
Florance
Size: 19
£250 per 1000 bulbs

A lifecycle method is set up, when this component mount then the values in the state will be updated.



```
componentDidMount() {
  // this.getData();
  let flowers = this.formatData(items);
  let newFlowers = flowers.filter((flower) => flower.new === true);
  let maxPrice = Math.max(...flowers.map((item) => item.price));
  this.setState({
    flowers,
    newFlowers,
    sortedFlowers: flowers,
    loading: false,
    price: maxPrice,
    maxPrice,
  });
}

formatData(items) {
  let tempItems = items.map((item) => {
    let id = item.sys.id;
    let images = item.fields.images.map((image) => image.fields.file.url);

    let flower = { ...item.fields, images, id };
    return flower;
  });
  return tempItems;
}
```

Data is imported from data.js and named *items*.

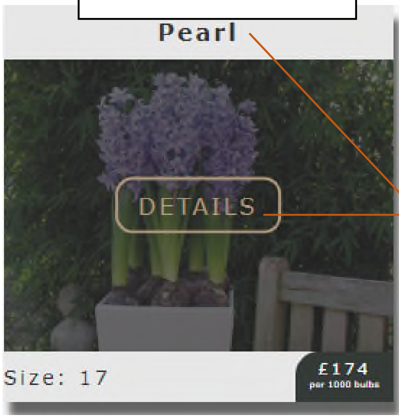
formatData(items) makes the “nested” data.js flatter.

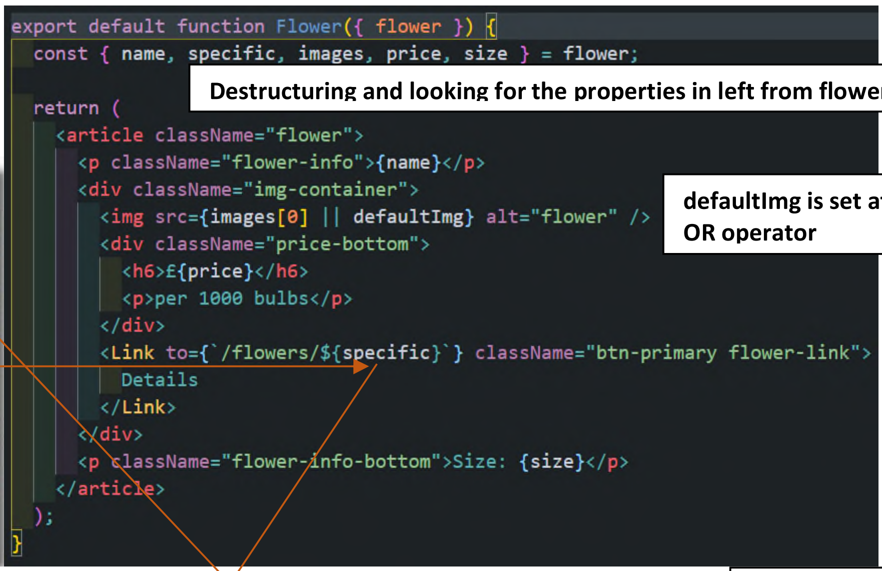
Mapping over the images array.

…item. JS object Spread Operator followed by two added properties – images and id from the original data.

Flower Component – Flower.js

Flower.js



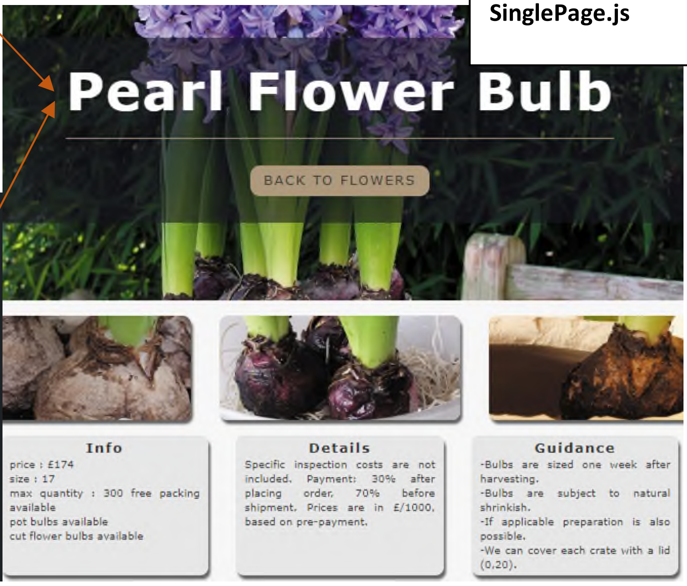


```
export default function Flower({ flower }) {
  const { name, specific, images, price, size } = flower;

  return (
    <article className="flower">
      <p className="flower-info">{name}</p>
      <div className="img-container">
        <img src={images[0] || defaultImg} alt="flower" />
        <div className="price-bottom">
          <h6>£{price}</h6>
          <p>per 1000 bulbs</p>
        </div>
        <Link to={`/flowers/${specific}`} className="btn-primary flower-link">
          Details
        </Link>
      </div>
      <p className="flower-info-bottom">Size: {size}</p>
    </article>
  );
}
```

Destructuring and looking for the properties in left from flower

defaultImg is set after OR operator

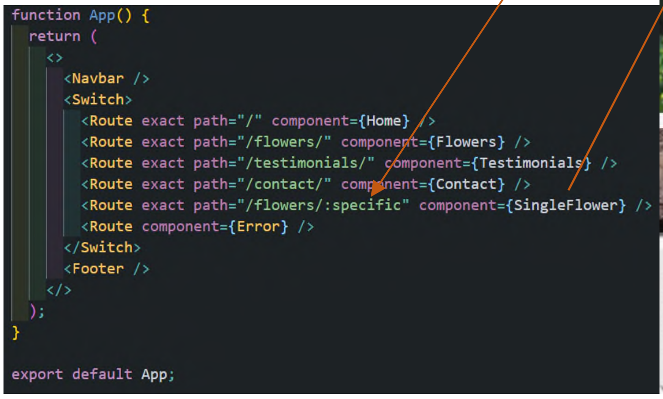
SinglePage.js



Info
price : £174
size : 17
max quantity : 300 free packing available
pot bulbs available
cut flower bulbs available

Details
Specific inspection costs are not included. Payment: 30% after placing order, 70% before shipment. Prices are in £/1000, based on pre-payment.

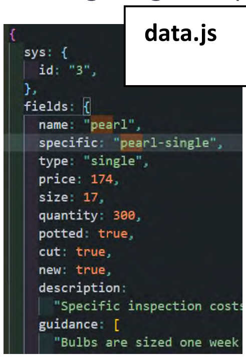
Guidance
-Bulbs are sized one week after harvesting.
-Bulbs are subject to natural shrinkish.
-If applicable preparation is also possible.
-We can cover each crate with a lid (0,20).



```
function App() {
  return (
    <>
      <Navbar />
      <Switch>
        <Route exact path="/" component={Home} />
        <Route exact path="/flowers/" component={Flowers} />
        <Route exact path="/testimonials/" component={Testimonials} />
        <Route exact path="/contact/" component={Contact} />
        <Route exact path="/flowers/:specific" component={SingleFlower} />
        <Route component={Error} />
      </Switch>
      <Footer />
    </>
  );
}

export default App;
```

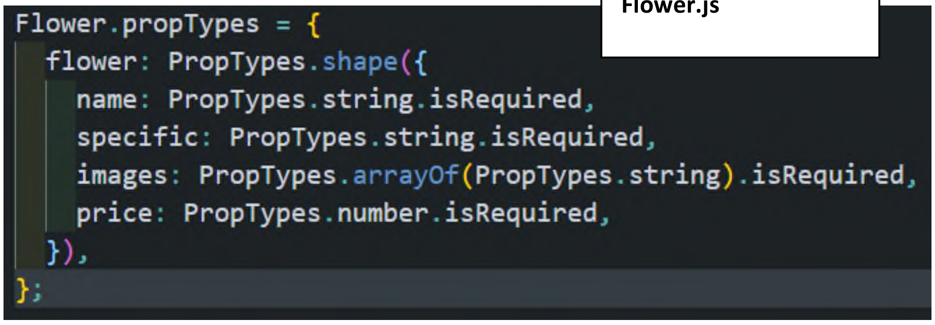
The specific URL parameter points to **SingleFlower** component, where specific page for this Flower Variety appears in the **SinglePage** component. A mechanism for checking if the the propps are passed is defined in **Flower.propTypes**

data.js



```
{
  sys: {
    id: "3",
  },
  fields: {
    name: "pearl",
    specific: "pearl-single",
    type: "single",
    price: 174,
    size: 17,
    quantity: 300,
    potted: true,
    cut: true,
    new: true,
    description:
      "Specific inspection costs
    guidance: [
      "Bulbs are sized one week
```

Flower.js

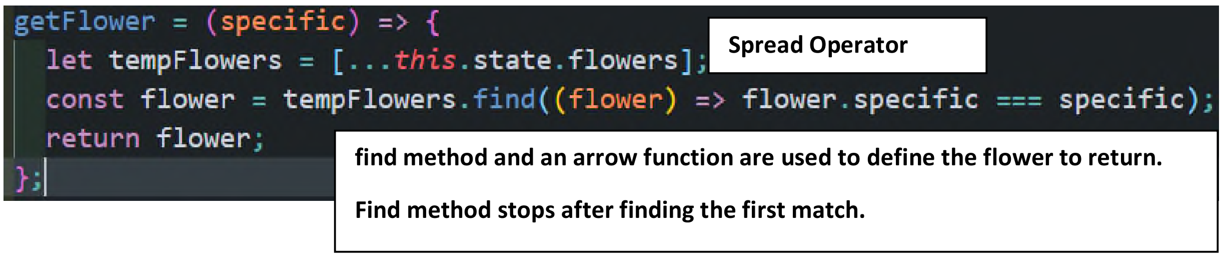


```
Flower.propTypes = {
  flower: PropTypes.shape({
    name: PropTypes.string.isRequired,
    specific: PropTypes.string.isRequired,
    images: PropTypes.arrayOf(PropTypes.string).isRequired,
    price: PropTypes.number.isRequired,
  }),
};
```

getFlower function in context.js

The **getFlower** function accepts one argument (**specific**)

tempFlowers is defined and a Spread Operator is used.



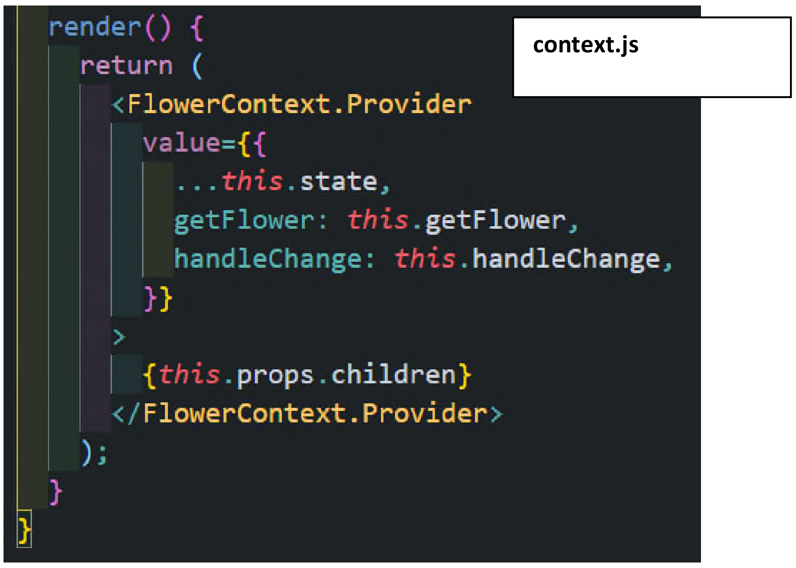
```
getFlower = (specific) => {
  let tempFlowers = [...this.state.flowers];
  const flower = tempFlowers.find((flower) => flower.specific === specific);
  return flower;
};
```

Spread Operator

find method and an arrow function are used to define the flower to return.

Find method stops after finding the first match.

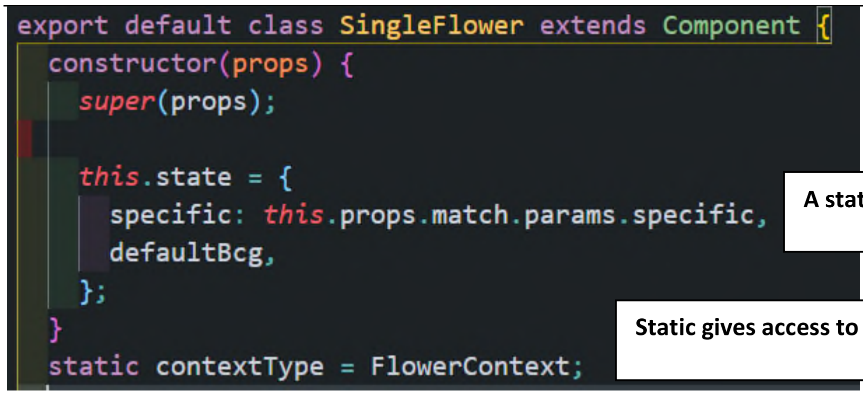
getFlower function is available in the Context below.



context.js

```
  render() {
    return (
      <FlowerContext.Provider
        value={{
          ...this.state,
          getFlower: this.getFlower,
          handleChange: this.handleChange,
        }}
      >
        {this.props.children}
      </FlowerContext.Provider>
    );
  }
}
```

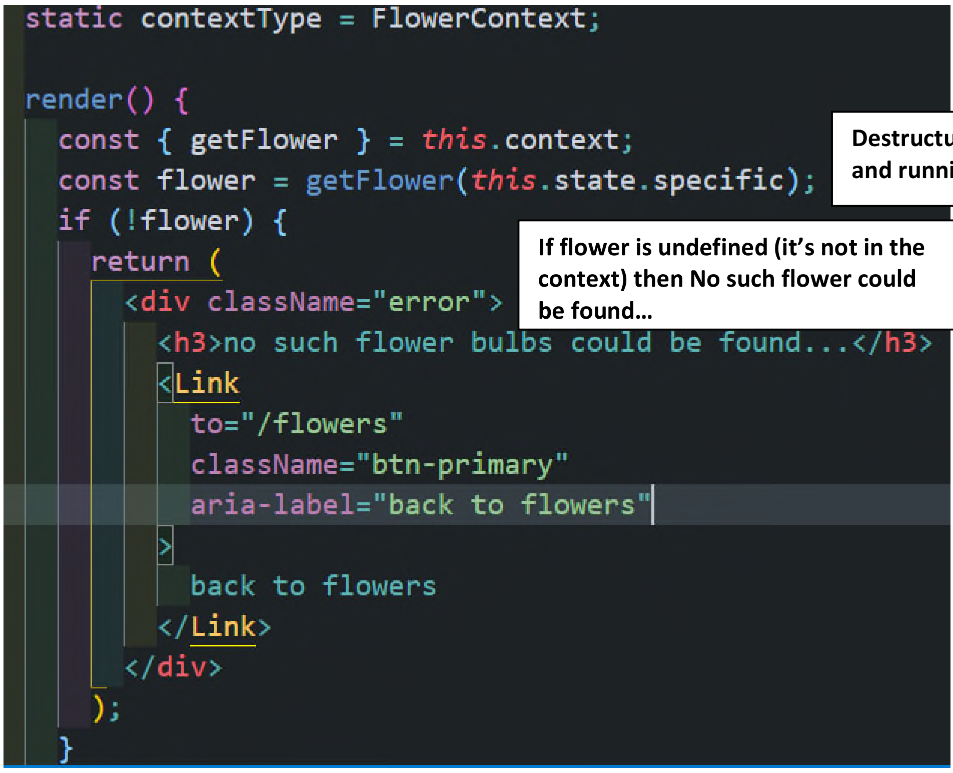
SingleFlower Setup – SingleFlower.js



```
export default class SingleFlower extends Component {
  constructor(props) {
    super(props);

    this.state = {
      specific: this.props.match.params.specific,
      defaultBcg,
    };
  }
  static contextType = FlowerContext;
```

A state is defined

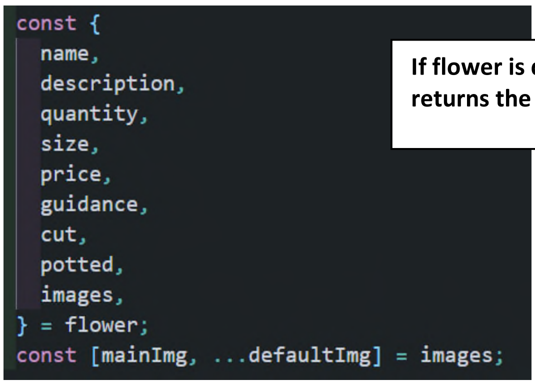
Static gives access to the context



```
  static contextType = FlowerContext;

  render() {
    const { getFlower } = this.context;
    const flower = getFlower(this.state.specific);
    if (!flower) {
      return (
        <div className="error">
          <h3>no such flower bulbs could be found...</h3>
          <Link
            to="/flowers"
            className="btn-primary"
            aria-label="back to flowers"
          >
            back to flowers
          </Link>
        </div>
      );
    }
```

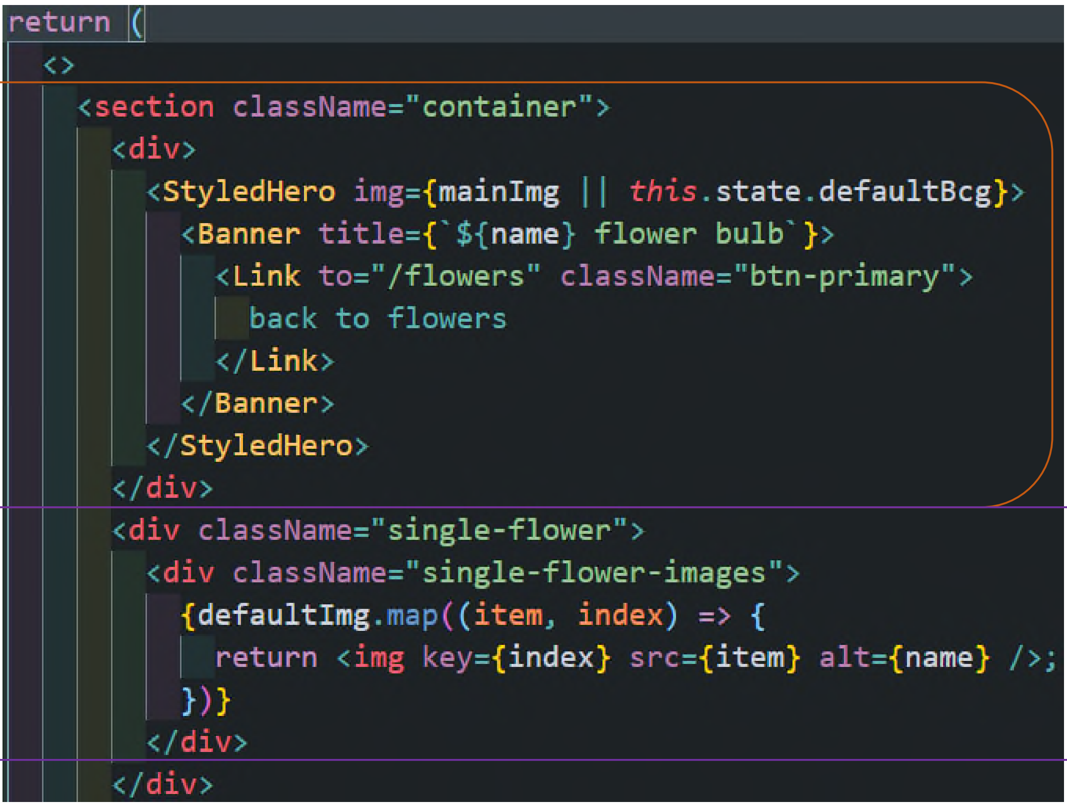
Destructuring getFlower function and running the function.

If flower is undefined (it's not in the context) then No such flower could be found...

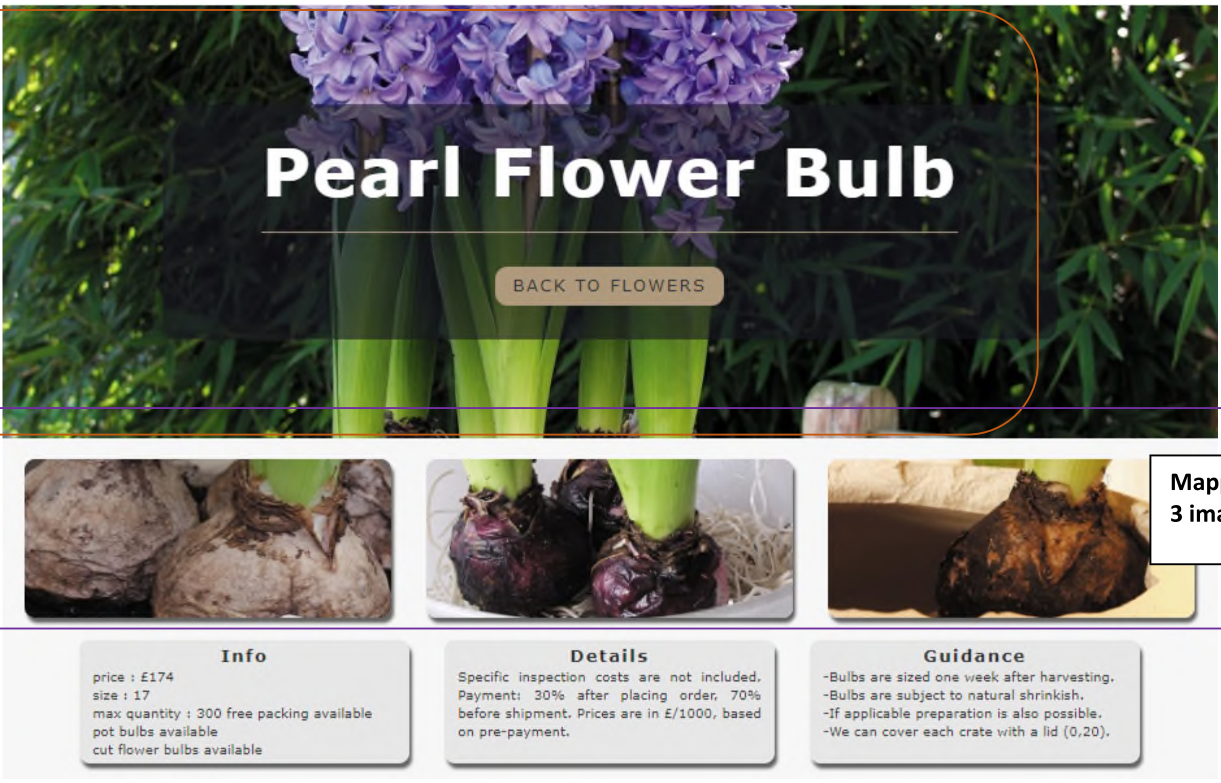


```
const {
  name,
  description,
  quantity,
  size,
  price,
  guidance,
  cut,
  potted,
  images,
} = flower;
const [mainImg, ...defaultImg] = images;
```

If flower is defined destructuring process is defined and returns the section below:



```jsx
return (
  <>
    <section className="container">
      <div>
        <StyledHero img={mainImg || this.state.defaultBcg}>
          <Banner title={`${name} flower bulb`}>
            <Link to="/flowers" className="btn-primary">
              back to flowers
            </Link>
          </Banner>
        </StyledHero>
      </div>
      <div className="single-flower">
        <div className="single-flower-images">
          {defaultImg.map((item, index) => {
            return <img key={index} src={item} alt={name} />;
          })}
        </div>
      </div>
```

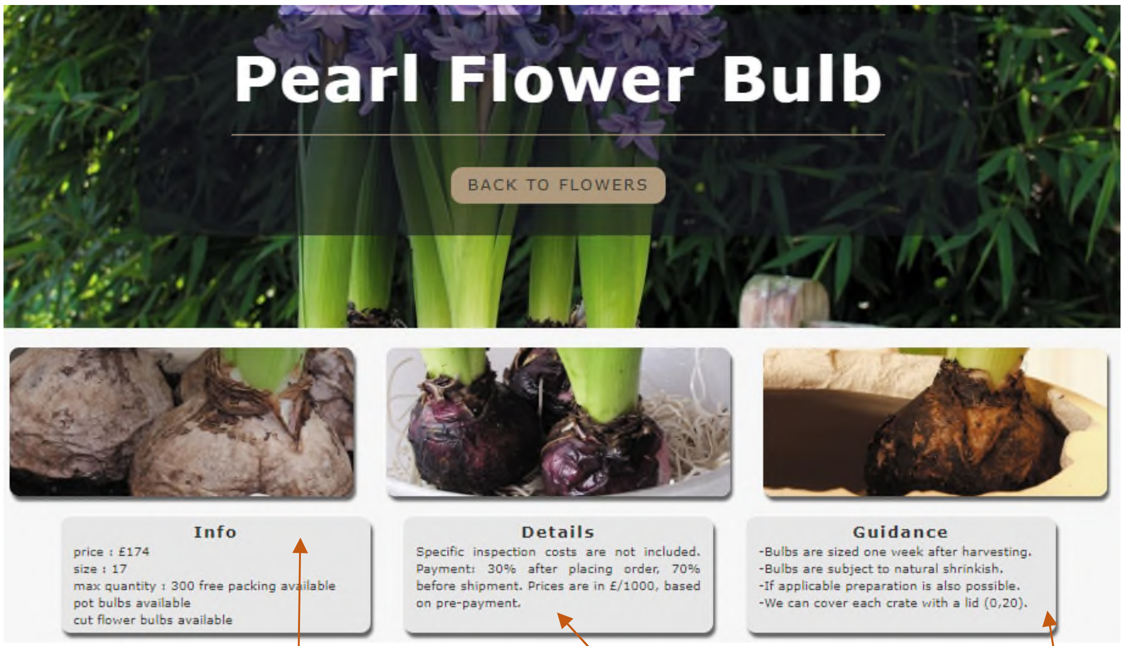


Info
price : £174
size : 17
max quantity : 300 free packing available
pot bulbs available
cut flower bulbs available

Details
Specific inspection costs are not included. Payment: 30% after placing order, 70% before shipment. Prices are in £/1000, based on pre-payment.

Guidance
-Bulbs are sized one week after harvesting.
-Bulbs are subject to natural shrinkish.
-If applicable preparation is also possible.
-We can cover each crate with a lid (0,20).

Mapping over the 3 images



Info

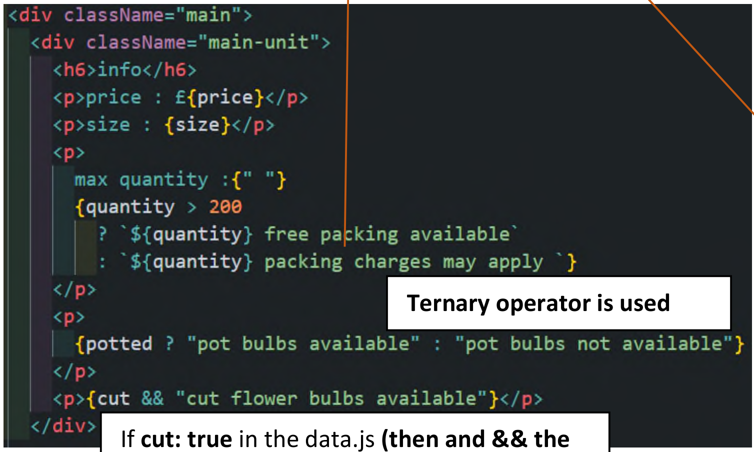
price : £174
size : 17
max quantity : 300 free packing available
pot bulbs available
cut flower bulbs available

Details

Specific inspection costs are not included. Payment: 30% after placing order, 70% before shipment. Prices are in £/1000, based on pre-payment.

Guidance

-Bulbs are sized one week after harvesting.
-Bulbs are subject to natural shrinkish.
-If applicable preparation is also possible.
-We can cover each crate with a lid (0,20).



```
<div className="main">
  <div className="main-unit">
    <h6>info</h6>
    <p>price : £{price}</p>
    <p>size : {size}</p>
    <p>
      max quantity :{" "}
      {quantity > 200
        ? `${quantity} free packing available`
        : `${quantity} packing charges may apply `}
    </p>
    <p>
      {potted ? "pot bulbs available" : "pot bulbs not available"}
    </p>
    <p>{cut && "cut flower bulbs available"}</p>
  </div>
```

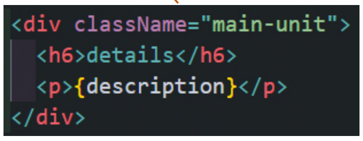
Ternary operator is used

If **cut: true** in the data.js **(then and && the message appears – "cut flower bulbs available".**

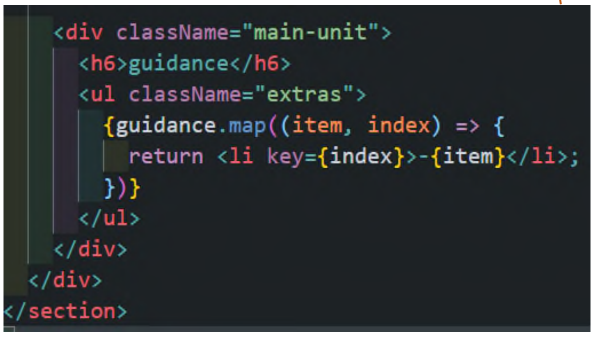
This syntax is used in other files in the project as an alternative to the ternary operator above, however there is a slight difference.

For ex. If cut: false then the message will not appear.

In the ternary operator one of the two messages will appear dependd on the conditional phrase before question mark



```
<div className="main-unit">
  <h6>details</h6>
  <p>{description}</p>
</div>
```

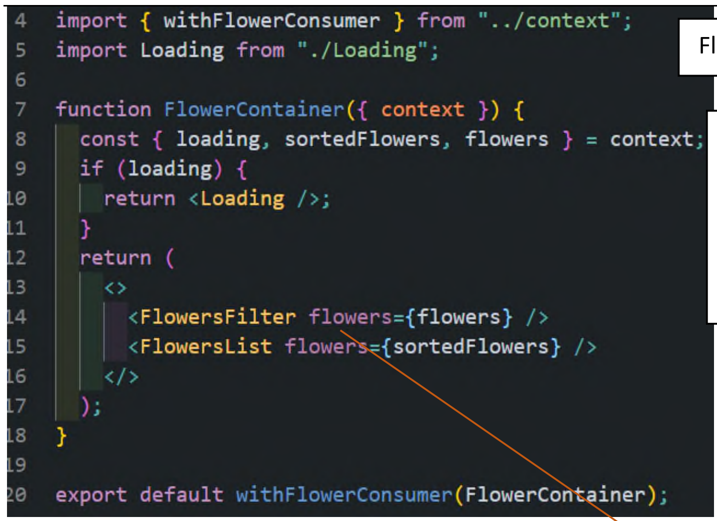


```
    <div className="main-unit">
      <h6>guidance</h6>
      <ul className="extras">
        {guidance.map((item, index) => {
          return <li key={index}>-{item}</li>;
        })}
      </ul>
    </div>
  </div>
</section>
```

All the 12 flower pages contain defferent specific information for the flower bulbs.

All the sections and divs are wrapped in a single React fragment <> </>, in order to be returned successfully.

FlowerContainer - FlowerContainer.js in Flowers.js



```
import { withFlowerConsumer } from "../context";
import Loading from "./Loading";

function FlowerContainer({ context }) {
  const { loading, sortedFlowers, flowers } = context;
  if (loading) {
    return <Loading />;
  }
  return (
    <>
      <FlowersFilter flowers={flowers} />
      <FlowersList flowers={sortedFlowers} />
    </>
  );
}

export default withFlowerConsumer(FlowerContainer);
```

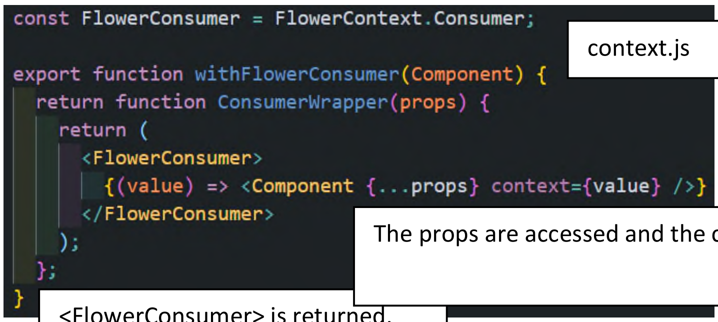
FlowerContainer.js

Destructuring and taking the values,

Passing flowers down to flowers and **FlowersFilter** down to **FlowersList.**

Defining a function in which a Component is passed. In this case is **FlowerContainer.**

The function returns another function **ConsumerWrapper** (props).



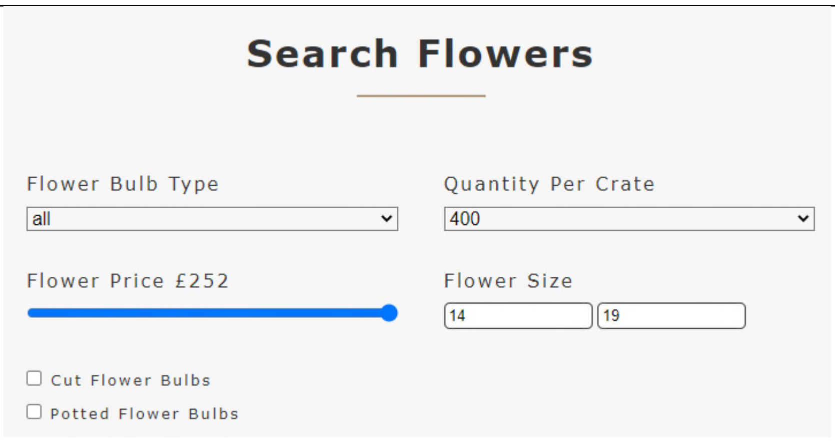
```
const FlowerConsumer = FlowerContext.Consumer;

export function withFlowerConsumer(Component) {
  return function ConsumerWrapper(props) {
    return (
      <FlowerConsumer>
        {(value) => <Component {...props} context={value} />}
      </FlowerConsumer>
    );
  };
}
```

context.js

The props are accessed and the context is set.

<FlowerConsumer> is returned. The value is wrapped in {} following the rules.

FlowerFilter Setup – FlowerFilter.js in Flowers.js



Search Flowers

Flower Bulb Type: all

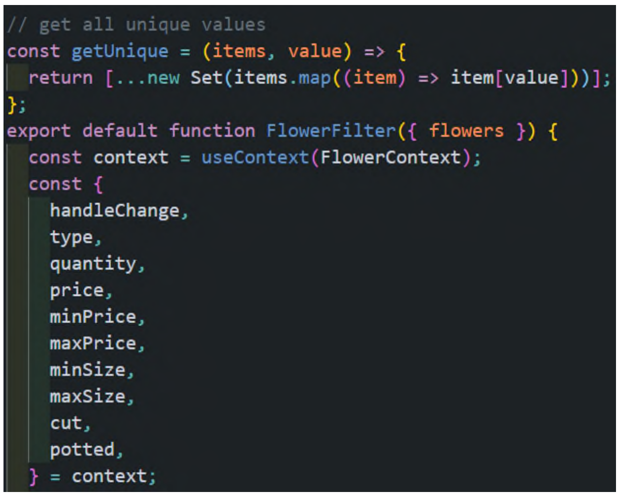
Quantity Per Crate: 400

Flower Price £252

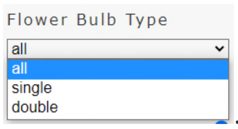
Flower Size: 14 | 19

☐ Cut Flower Bulbs

☐ Potted Flower Bulbs



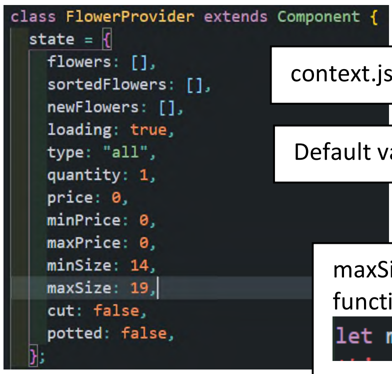
```
// get all unique values
const getUnique = (items, value) => {
  return [...new Set(items.map((item) => item[value]))];
};
export default function FlowerFilter({ flowers }) {
  const context = useContext(FlowerContext);
  const {
    handleChange,
    type,
    quantity,
    price,
    minPrice,
    maxPrice,
    minSize,
    maxSize,
    cut,
    potted,
  } = context;
```

handleChange in the State reflects whatever there is in the fields. .

Flower Bulb Type: all

all
single
double

The values are set in the context state and passed down in order to be accessed by each filter component. Only the values that relate to every filter component, in order to control the input.

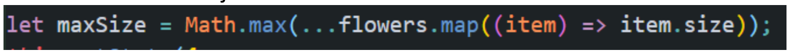


```
class FlowerProvider extends Component {
  state = {
    flowers: [],
    sortedFlowers: [],
    newFlowers: [],
    loading: true,
    type: "all",
    quantity: 1,
    price: 0,
    minPrice: 0,
    maxPrice: 0,
    minSize: 14,
    maxSize: 19,
    cut: false,
    potted: false,
  };
```

context.js

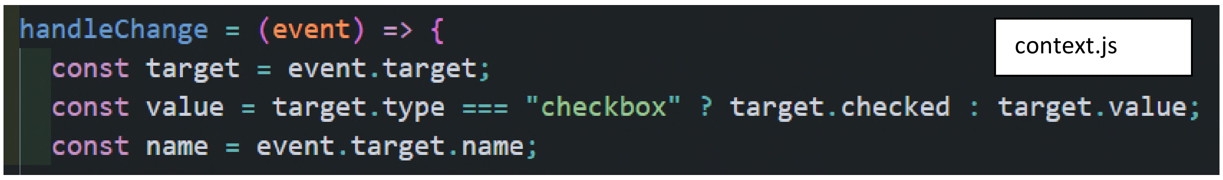
Default values

maxSize can be given by default: 19, but can be also calculated using a Math function in context.js



```
let maxSize = Math.max(...flowers.map((item) => item.size));
```

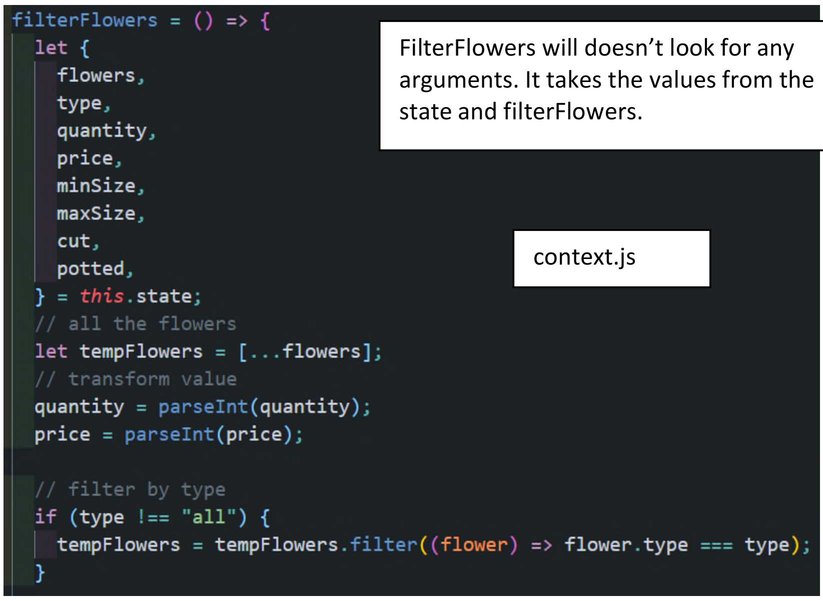
It is passed as a value, not as an array, that's why is used the Spread operator **…flowers,** then mapped over and the size for each and every item is returned.



```
handleChange = (event) => {
  const target = event.target;
  const value = target.type === "checkbox" ? target.checked : target.value;
  const name = event.target.name;
```

context.js

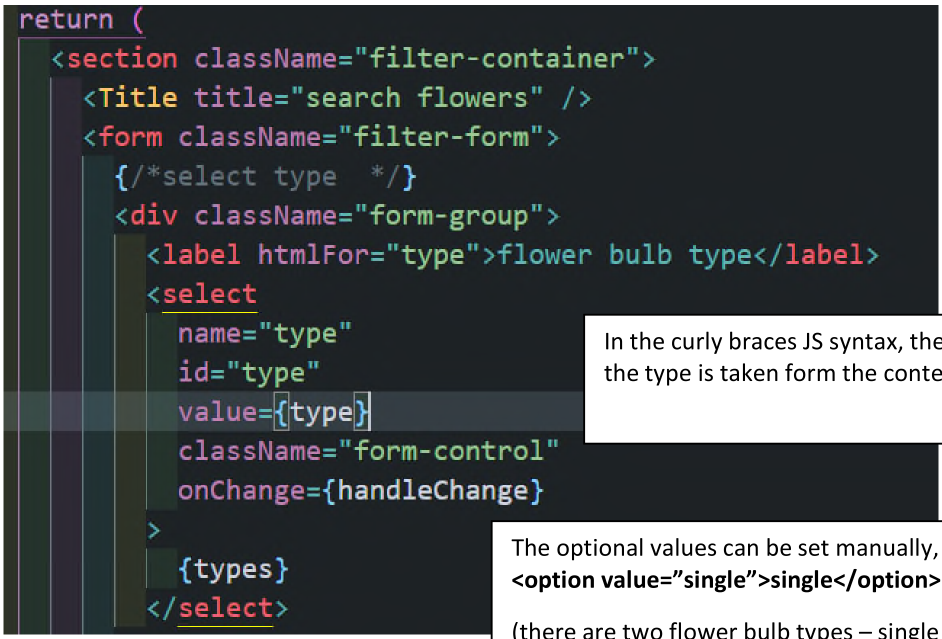
The function handleChange is set. This function grab everything that control the inputs. In this way the values in the state are fixed. The values in the state will be accessed and changed within the handleChange function.



```
filterFlowers = () => {
  let {
    flowers,
    type,
    quantity,
    price,
    minSize,
    maxSize,
    cut,
    potted,
  } = this.state;
  // all the flowers
  let tempFlowers = [...flowers];
  // transform value
  quantity = parseInt(quantity);
  price = parseInt(price);

  // filter by type
  if (type !== "all") {
    tempFlowers = tempFlowers.filter((flower) => flower.type === type);
  }
```

FilterFlowers will doesn't look for any arguments. It takes the values from the state and filterFlowers.

context.js



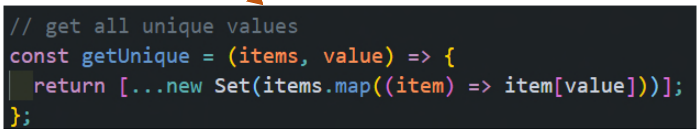
```
return (
  <section className="filter-container">
    <Title title="search flowers" />
    <form className="filter-form">
      {/*select type  */}
      <div className="form-group">
        <label htmlFor="type">flower bulb type</label>
        <select
          name="type"
          id="type"
          value={type}
          className="form-control"
          onChange={handleChange}
        >
          {types}
        </select>
```

In the curly braces JS syntax, the value of the type is taken form the context.

The optional values can be set manually, by writing
<option value=”single”>single</option>

(there are two flower bulb types – single and double).

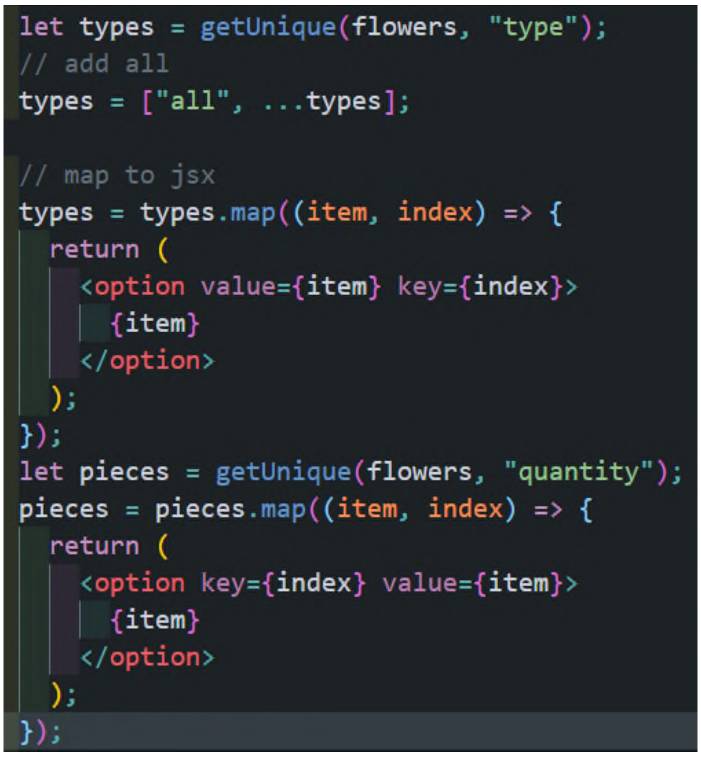
Instead of this manual approach, a function is set that returns only the **unique values**.



```
// get all unique values
const getUnique = (items, value) => {
  return [...new Set(items.map((item) => item[value]))];
};
```

The getUnique function returns an array spread it out new Set. Within the Set there is mapping over and return of item[value].

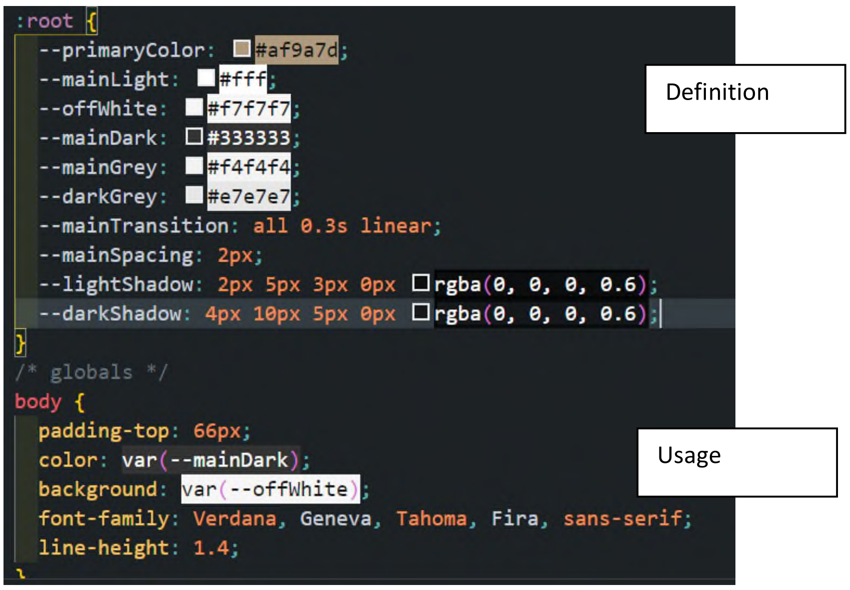
item[value] is a dynamic property. Value on the row above is a string. When passing on the string there will be a check, what kind of value there is for the type and if that value is not in the state it will be included. If its in the Set by default, set is not going to included.



```
let types = getUnique(flowers, "type");
// add all
types = ["all", ...types];

// map to jsx
types = types.map((item, index) => {
  return (
    <option value={item} key={index}>
      {item}
    </option>
  );
});
let pieces = getUnique(flowers, "quantity");
pieces = pieces.map((item, index) => {
  return (
    <option key={index} value={item}>
      {item}
    </option>
  );
});
```

Some Notes on CSS and Audit

1. Mobile first approach is applied for responsibility reason.
2. Mainly flex box method is used.
3. 100 SEO SEO Audit checks achieved score of 100%.
4. Images’ size may affect the Performance rate, since all the images were taken from The Flower Council of Holland - Image Bank and there are some restriction on changing them.
5. During audit checks Accessibility – 100%, SEO – 100% score were achieved.
6. **aria-label=””** aria-label was included after requirement from **Audits** for **Accessibility purposes.**
7. There was not statistical research done for defining the media query break points.
8. A common best practice is to define custom properties on the :root pseudo-class, so that it can be applied globally across the HTML document: (Mozilla and individual contributors)
   In the following example colors can be changed easily only on the :root.



```css
:root {
  --primaryColor: #af9a7d;
  --mainLight: #fff;
  --offWhite: #f7f7f7;
  --mainDark: #333333;
  --mainGrey: #f4f4f4;
  --darkGrey: #e7e7e7;
  --mainTransition: all 0.3s linear;
  --mainSpacing: 2px;
  --lightShadow: 2px 5px 3px 0px rgba(0, 0, 0, 0.6);
  --darkShadow: 4px 10px 5px 0px rgba(0, 0, 0, 0.6);
}
/* globals */
body {
  padding-top: 66px;
  color: var(--mainDark);
  background: var(--offWhite);
  font-family: Verdana, Geneva, Tahoma, Fira, sans-serif;
  line-height: 1.4;
```

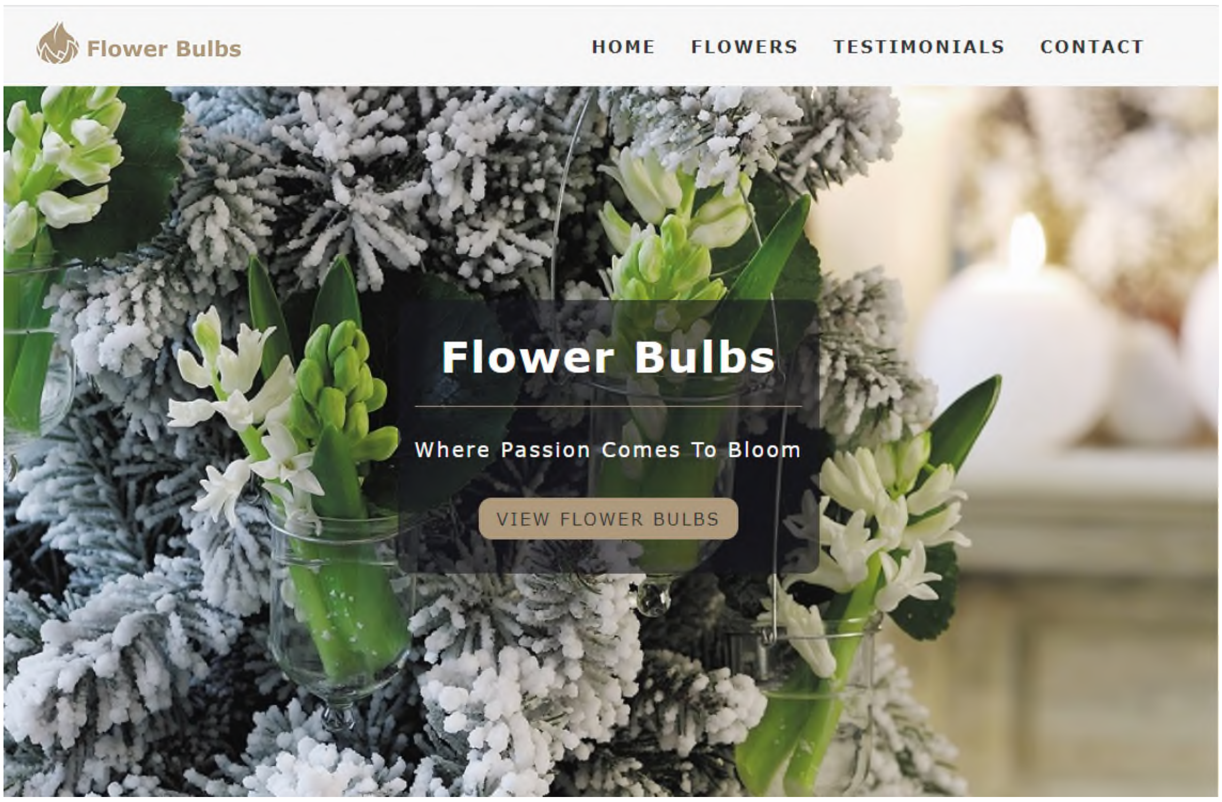
9. The usage of z-index place the container in front or behind other visual objects. The maximum value is 9999 and can be used to ensure placing the contaner in front of all other containers.

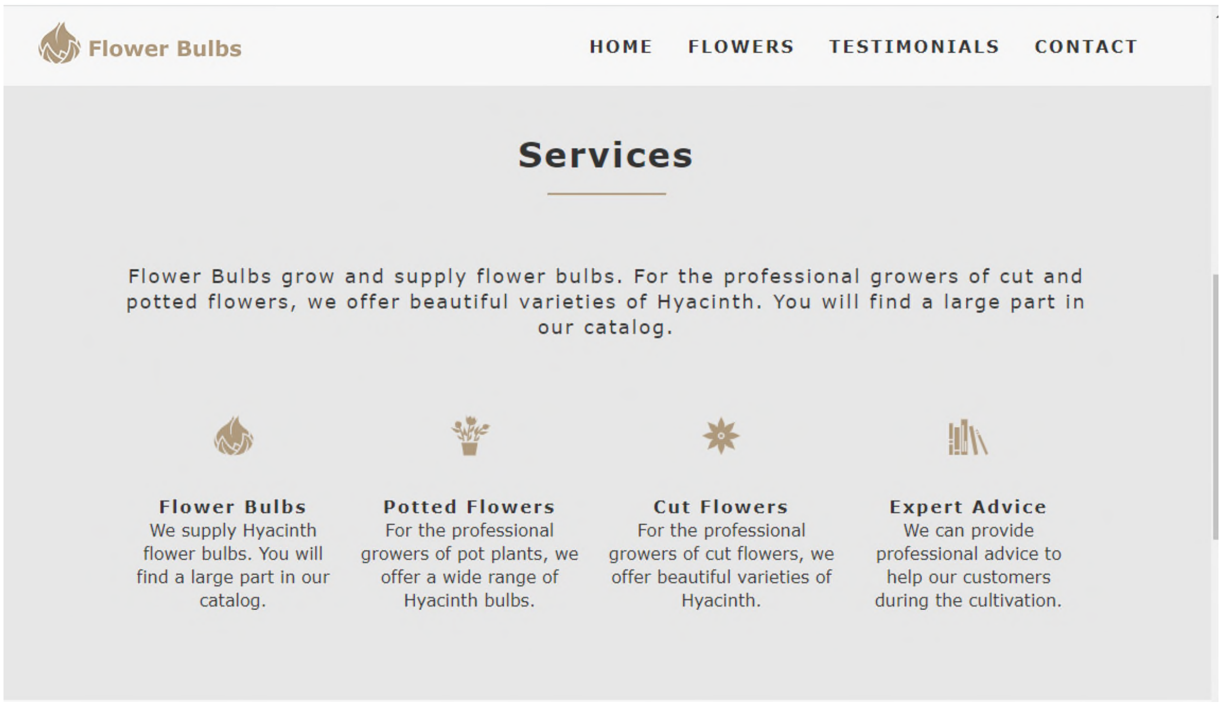
Appendixes:

1. Project website’s screenshots in Mobile and Desktop views.

References:

1. The Flower Council of Holland - Image Bank. [Viewed 12 July 2020] (https://www.flowercouncil.co.uk/image-bank).
2. React icons [Viewed 12 July 2020]. https://react-icons.github.io/react-icons/
3. React training/React Router [Viewed 12 July 2020] https://reacttraining.com/react-router/web/guides/quick-start
4. (Facebook open source, 2020) [Viewed 12 July 2020] https://reactjs.org/docs/context.html
5. Mozilla and Contributors [Viewed 12 July 2020] Available form: https://developer.mozilla.org/en-US/docs/Web/CSS/Using_CSS_custom_properties
6. 0to255 Studio [Viewed 12 July 2020] https://www.0to255.com/
7. Real Favicon Generator [Viewed 12 July 2020] Available from: (https://realfavicongenerator.net/).
8. W3Scools [Viewed 12 July 2020] Available from: https://www.w3schools.com/html/





Services

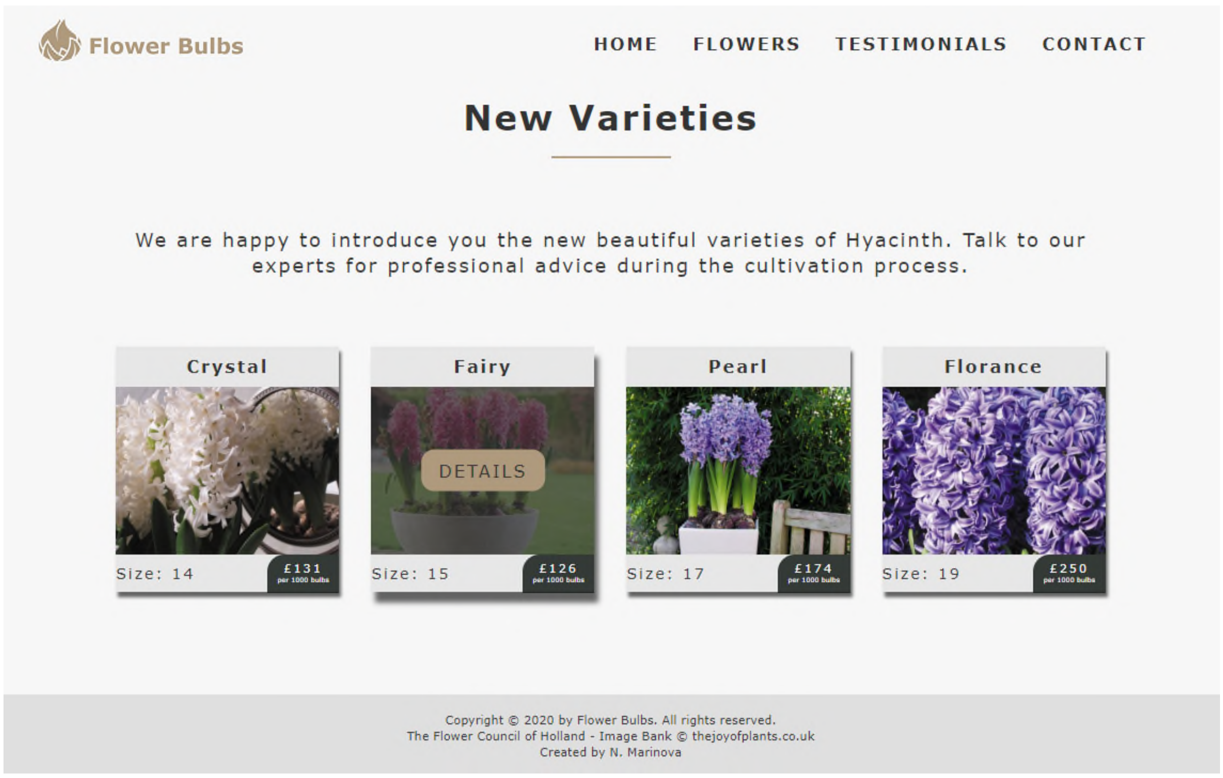
Flower Bulbs grow and supply flower bulbs. For the professional growers of cut and potted flowers, we offer beautiful varieties of Hyacinth. You will find a large part in our catalog.

Flower Bulbs
We supply Hyacinth flower bulbs. You will find a large part in our catalog.

Potted Flowers
For the professional growers of pot plants, we offer a wide range of Hyacinth bulbs.

Cut Flowers
For the professional growers of cut flowers, we offer beautiful varieties of Hyacinth.

Expert Advice
We can provide professional advice to help our customers during the cultivation.

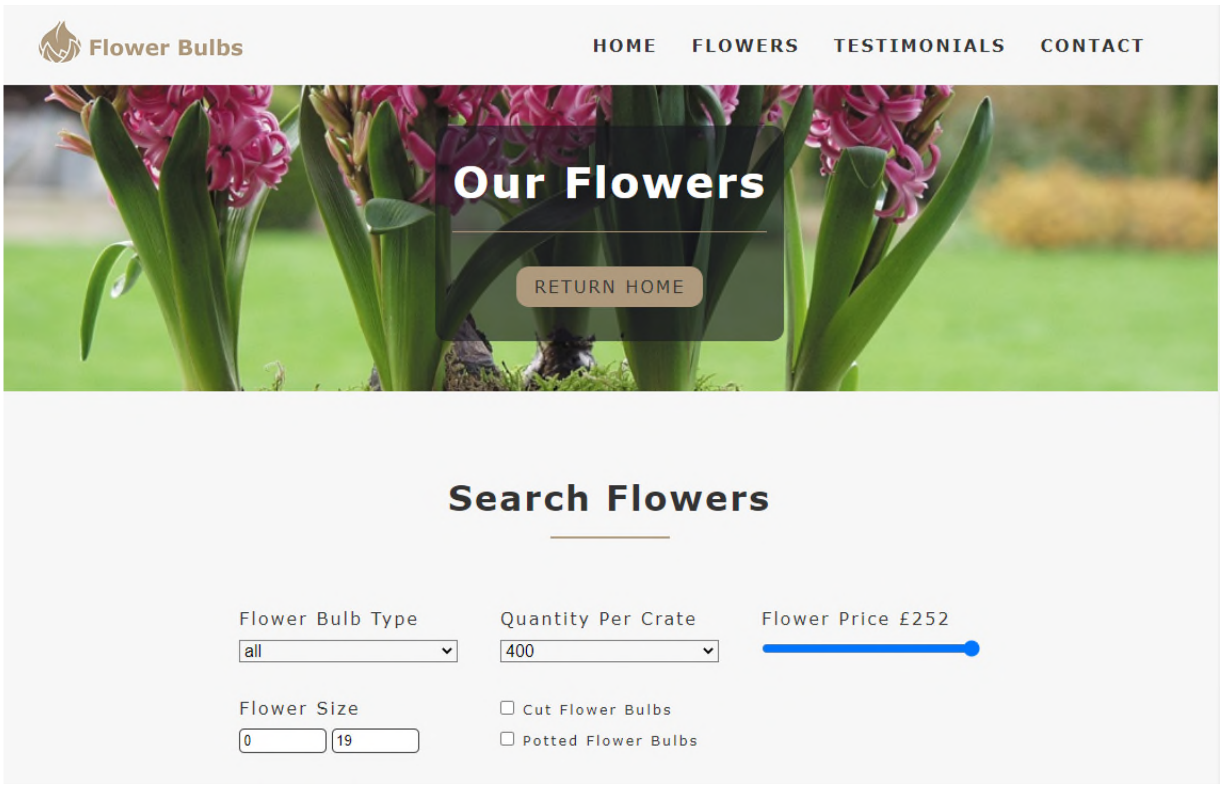


New Varieties

We are happy to introduce you the new beautiful varieties of Hyacinth. Talk to our experts for professional advice during the cultivation process.

Name	Size	Price
Crystal	Size: 14	£131 per 1000 bulbs
Fairy	Size: 15	£126 per 1000 bulbs
Pearl	Size: 17	£174 per 1000 bulbs
Florance	Size: 19	£250 per 1000 bulbs

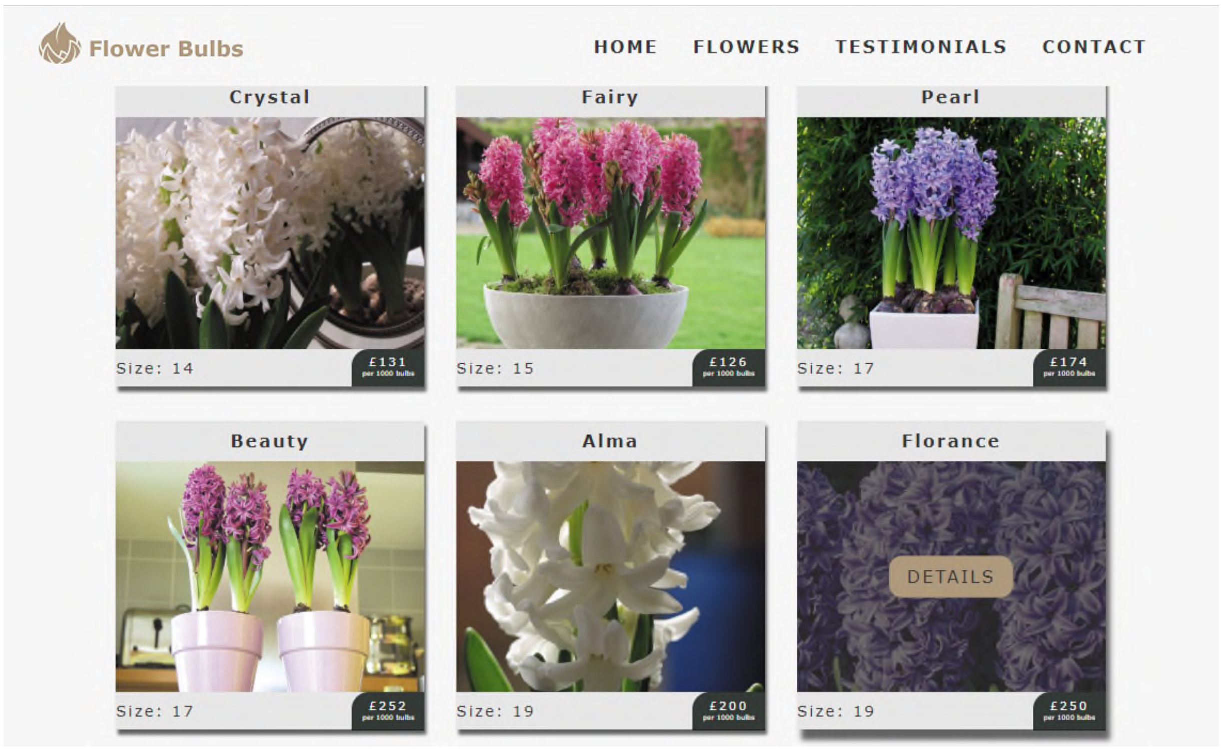
Copyright © 2020 by Flower Bulbs. All rights reserved.
The Flower Council of Holland - Image Bank © thejoyofplants.co.uk
Created by N. Marinova

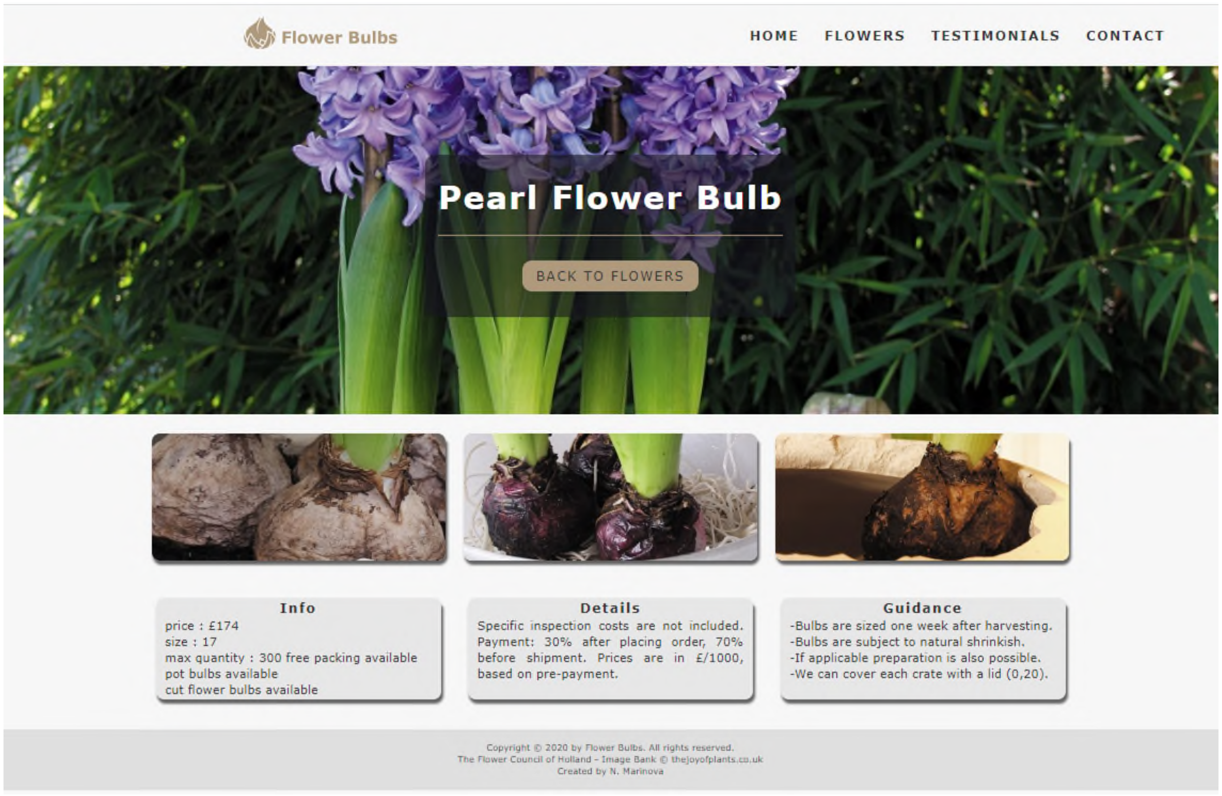


Our Flowers

Search Flowers

Flower Bulb Type: all
Quantity Per Crate: 400
Flower Price £252
Flower Size: 0 – 19
☐ Cut Flower Bulbs
☐ Potted Flower Bulbs





Info

price : £174
size : 17
max quantity : 300 free packing available
pot bulbs available
cut flower bulbs available

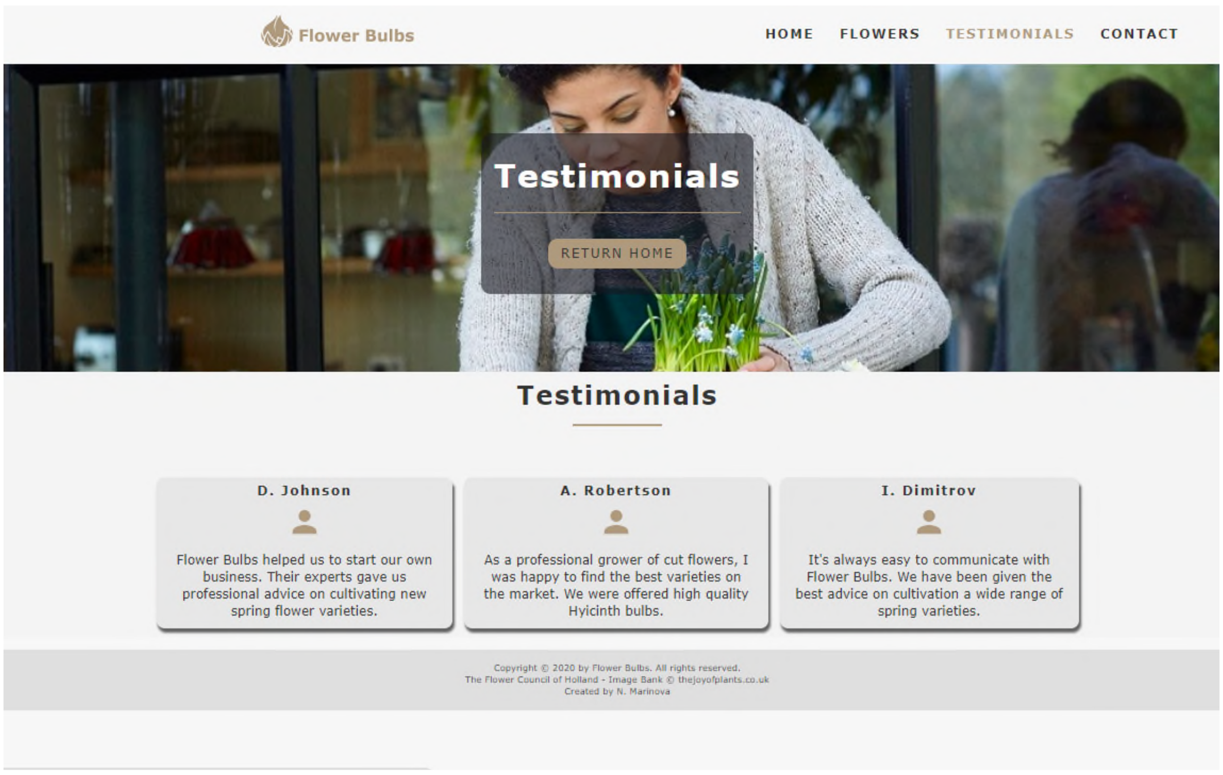
Details

Specific inspection costs are not included. Payment: 30% after placing order, 70% before shipment. Prices are in £/1000, based on pre-payment.

Guidance

-Bulbs are sized one week after harvesting.
-Bulbs are subject to natural shrinkish.
-If applicable preparation is also possible.
-We can cover each crate with a lid (0,20).

Copyright © 2020 by Flower Bulbs. All rights reserved.
The Flower Council of Holland – Image Bank © thejoyofplants.co.uk
Created by N. Marinova



Flower Bulbs

HOME FLOWERS TESTIMONIALS CONTACT

Testimonials

RETURN HOME

Testimonials

D. Johnson

Flower Bulbs helped us to start our own business. Their experts gave us professional advice on cultivating new spring flower varieties.

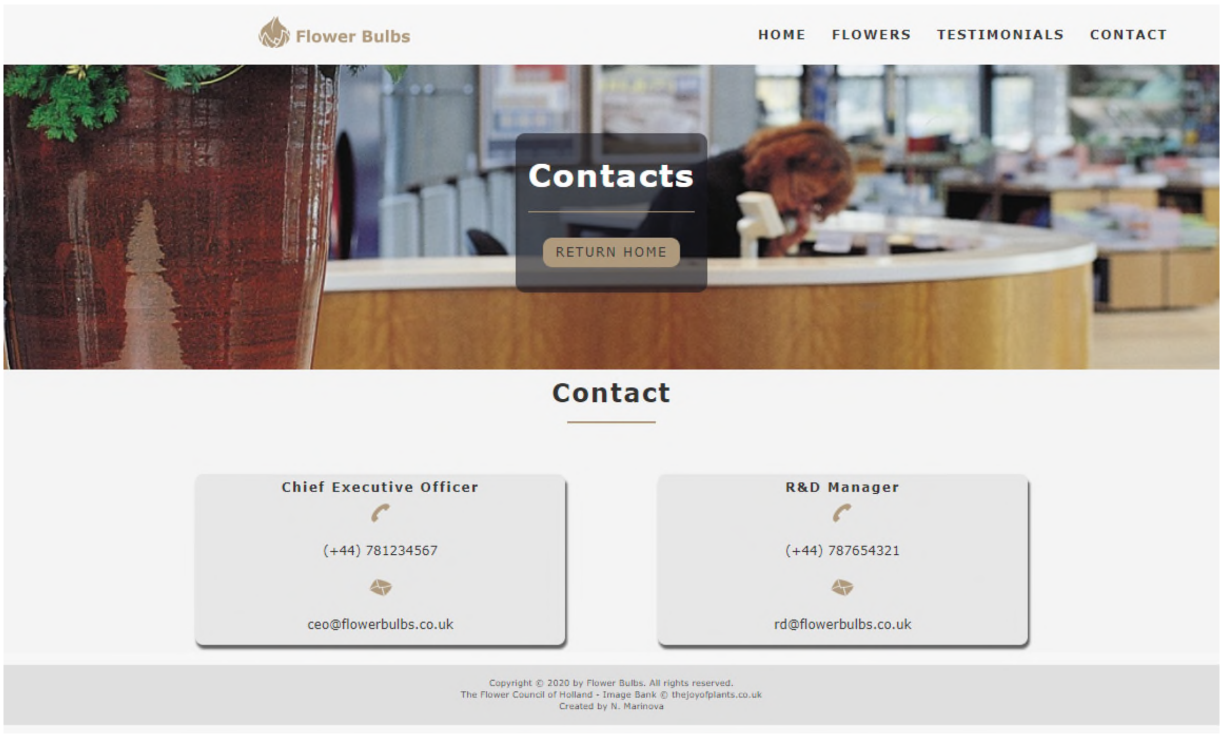
A. Robertson

As a professional grower of cut flowers, I was happy to find the best varieties on the market. We were offered high quality Hyicinth bulbs.

I. Dimitrov

It's always easy to communicate with Flower Bulbs. We have been given the best advice on cultivation a wide range of spring varieties.

Copyright © 2020 by Flower Bulbs. All rights reserved.
The Flower Council of Holland - Image Bank © thejoyofplants.co.uk
Created by N. Marinova



Flower Bulbs

HOME FLOWERS TESTIMONIALS CONTACT

Contacts

RETURN HOME

Contact

Chief Executive Officer

(+44) 781234567

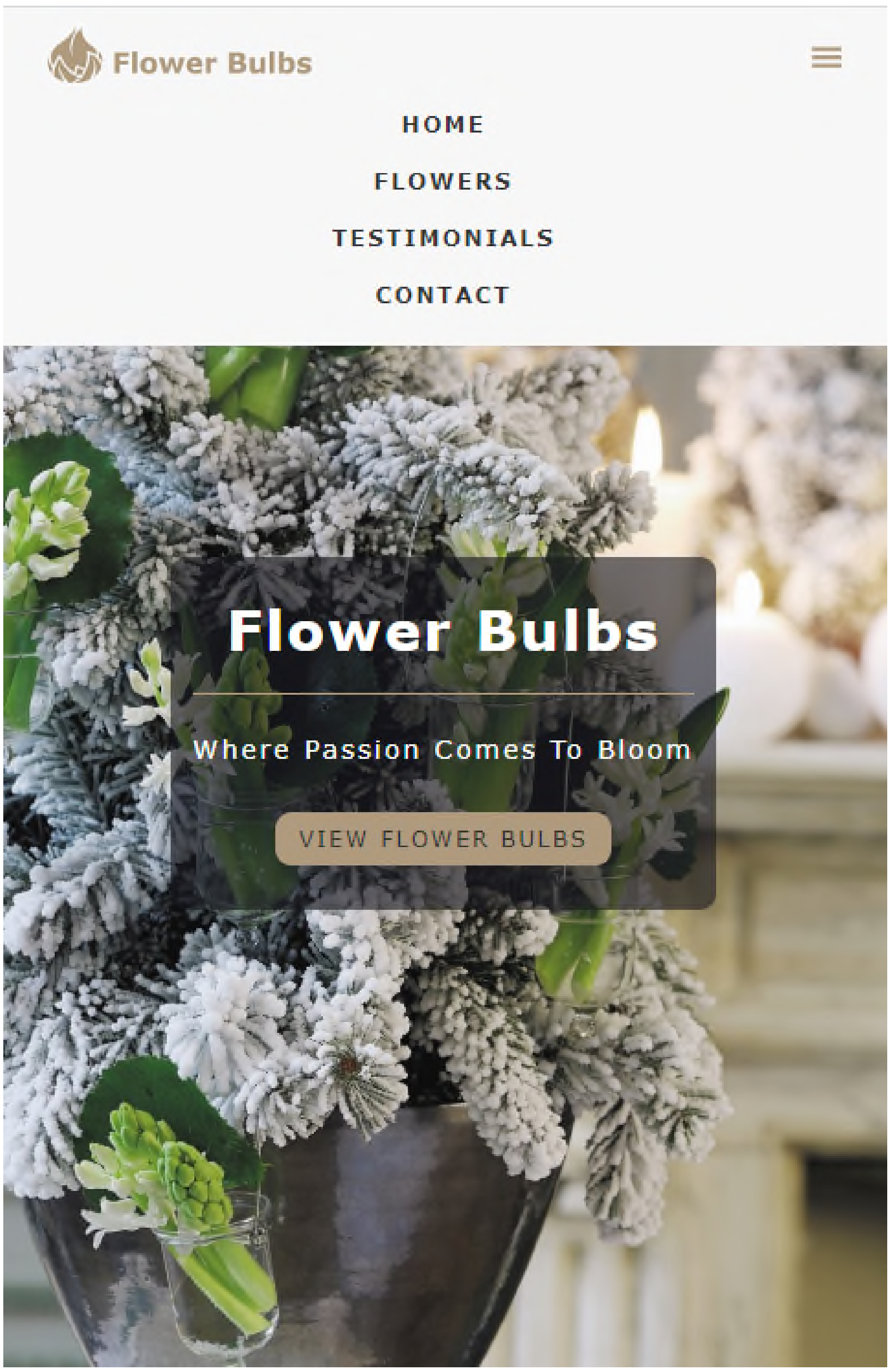
ceo@flowerbulbs.co.uk

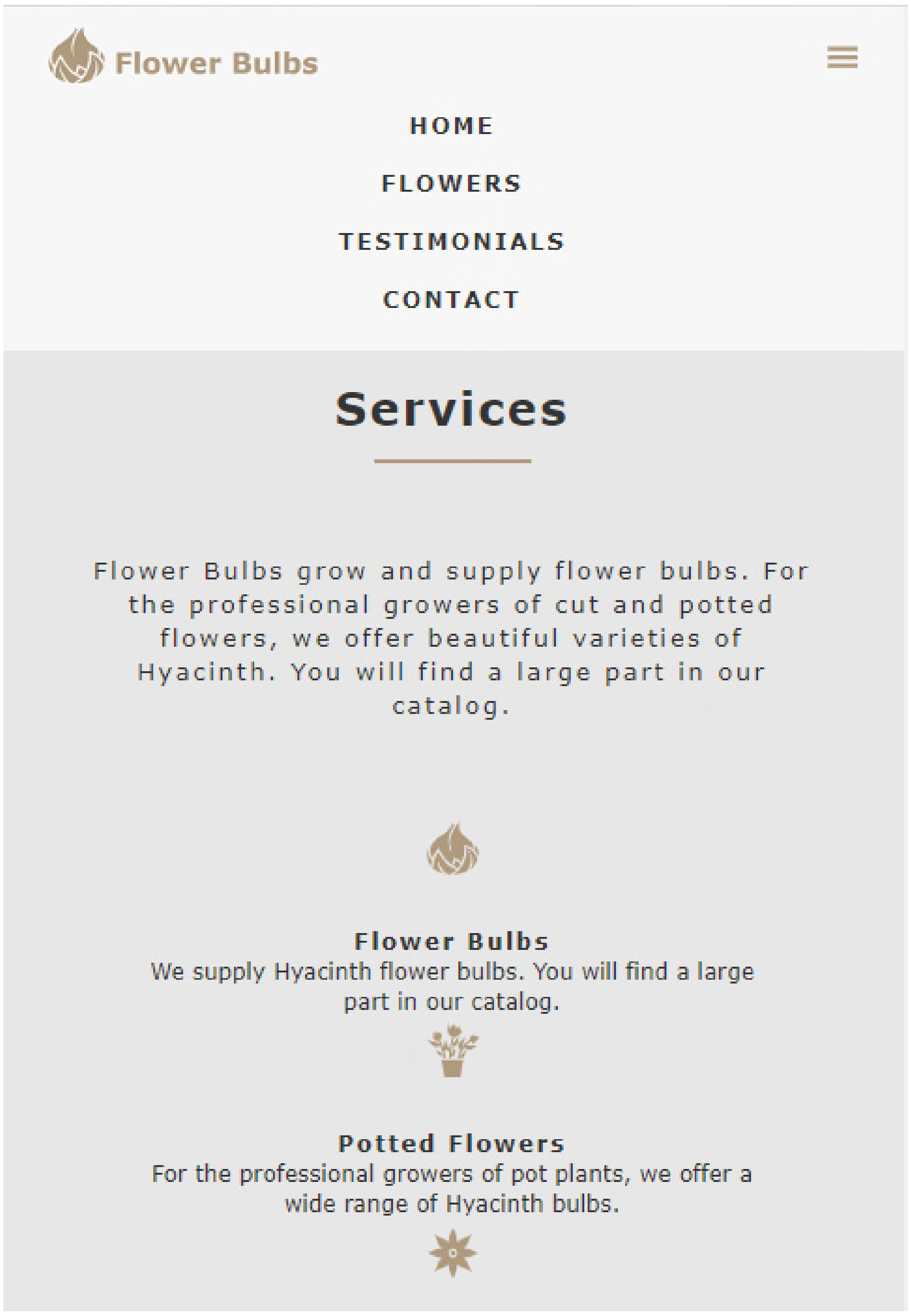
R&D Manager

(+44) 787654321

rd@flowerbulbs.co.uk

Copyright © 2020 by Flower Bulbs. All rights reserved.
The Flower Council of Holland - Image Bank © thejoyofplants.co.uk
Created by N. Marinova


Flower Bulbs
HOME
FLOWERS
TESTIMONIALS
CONTACT
Flower Bulbs
Where Passion Comes To Bloom
VIEW FLOWER BULBS



Flower Bulbs

HOME

FLOWERS

TESTIMONIALS

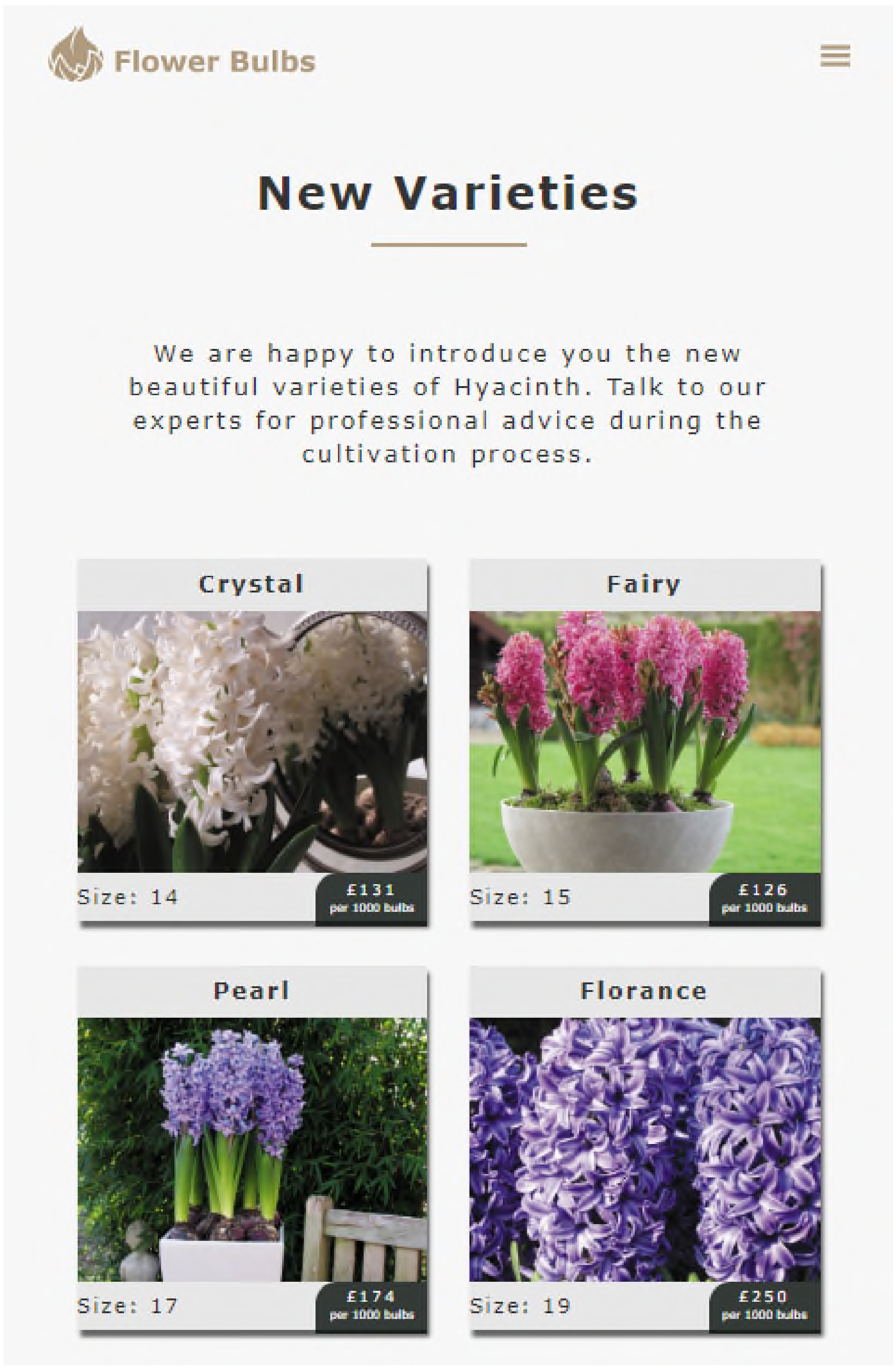
CONTACT

Services

Flower Bulbs grow and supply flower bulbs. For the professional growers of cut and potted flowers, we offer beautiful varieties of Hyacinth. You will find a large part in our catalog.

Flower Bulbs
We supply Hyacinth flower bulbs. You will find a large part in our catalog.

Potted Flowers
For the professional growers of pot plants, we offer a wide range of Hyacinth bulbs.



Flower Bulbs

New Varieties

We are happy to introduce you the new beautiful varieties of Hyacinth. Talk to our experts for professional advice during the cultivation process.

Crystal

Size: 14

£131
per 1000 bulbs

Fairy

Size: 15

£126
per 1000 bulbs

Pearl

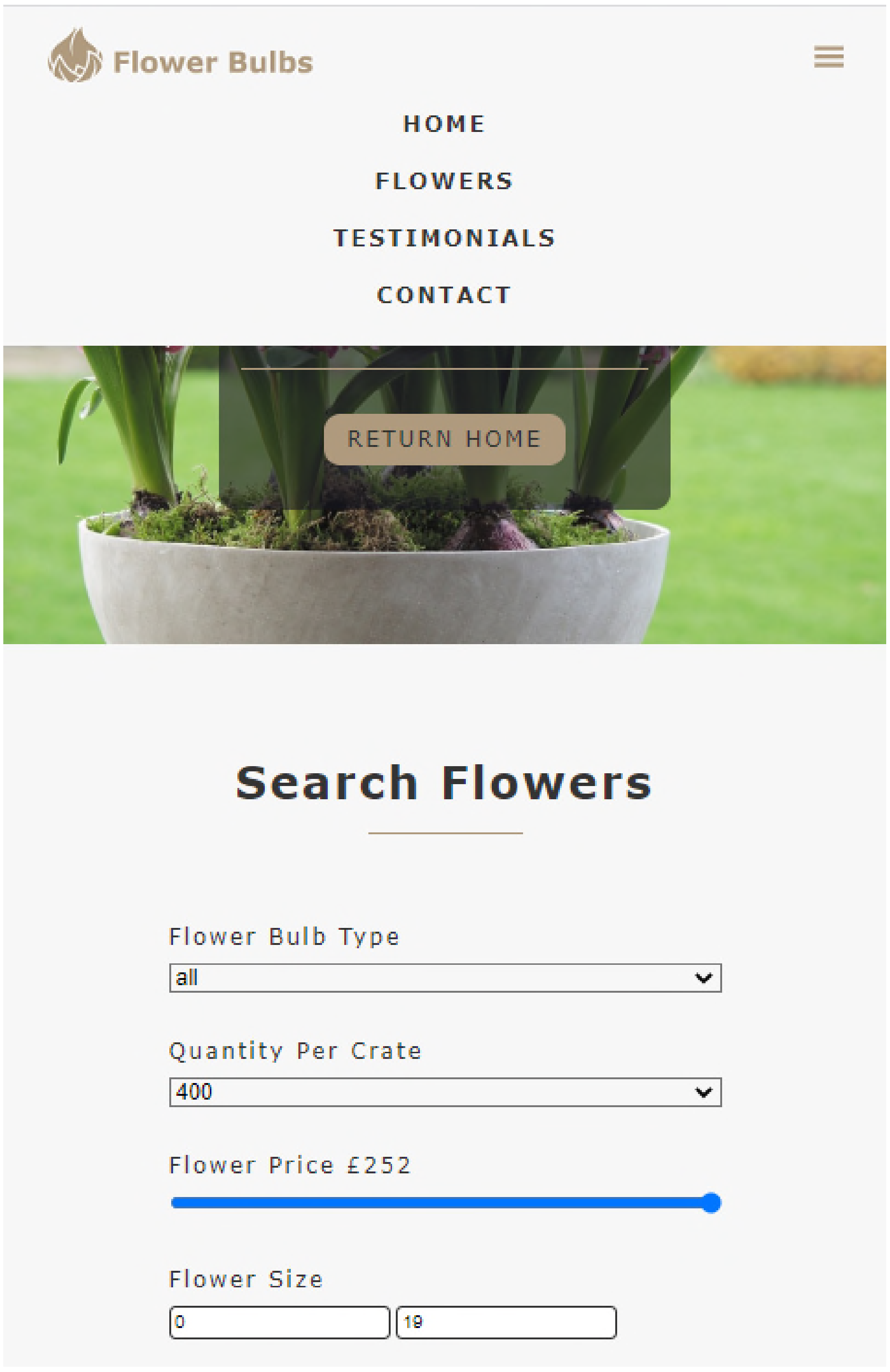
Size: 17

£174
per 1000 bulbs

Florance

Size: 19

£250
per 1000 bulbs


Flower Bulbs

HOME

FLOWERS

TESTIMONIALS

CONTACT

RETURN HOME

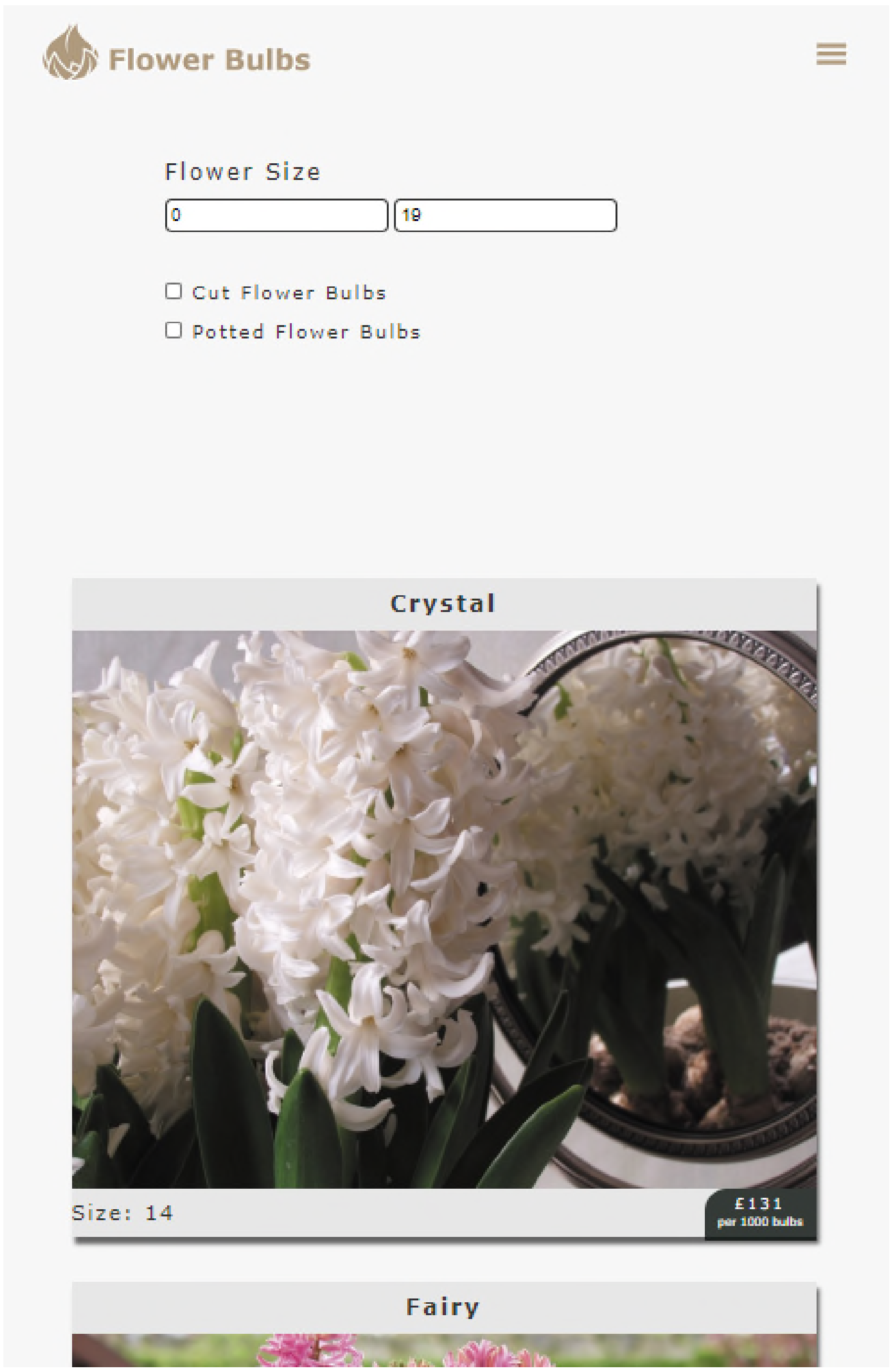
Search Flowers

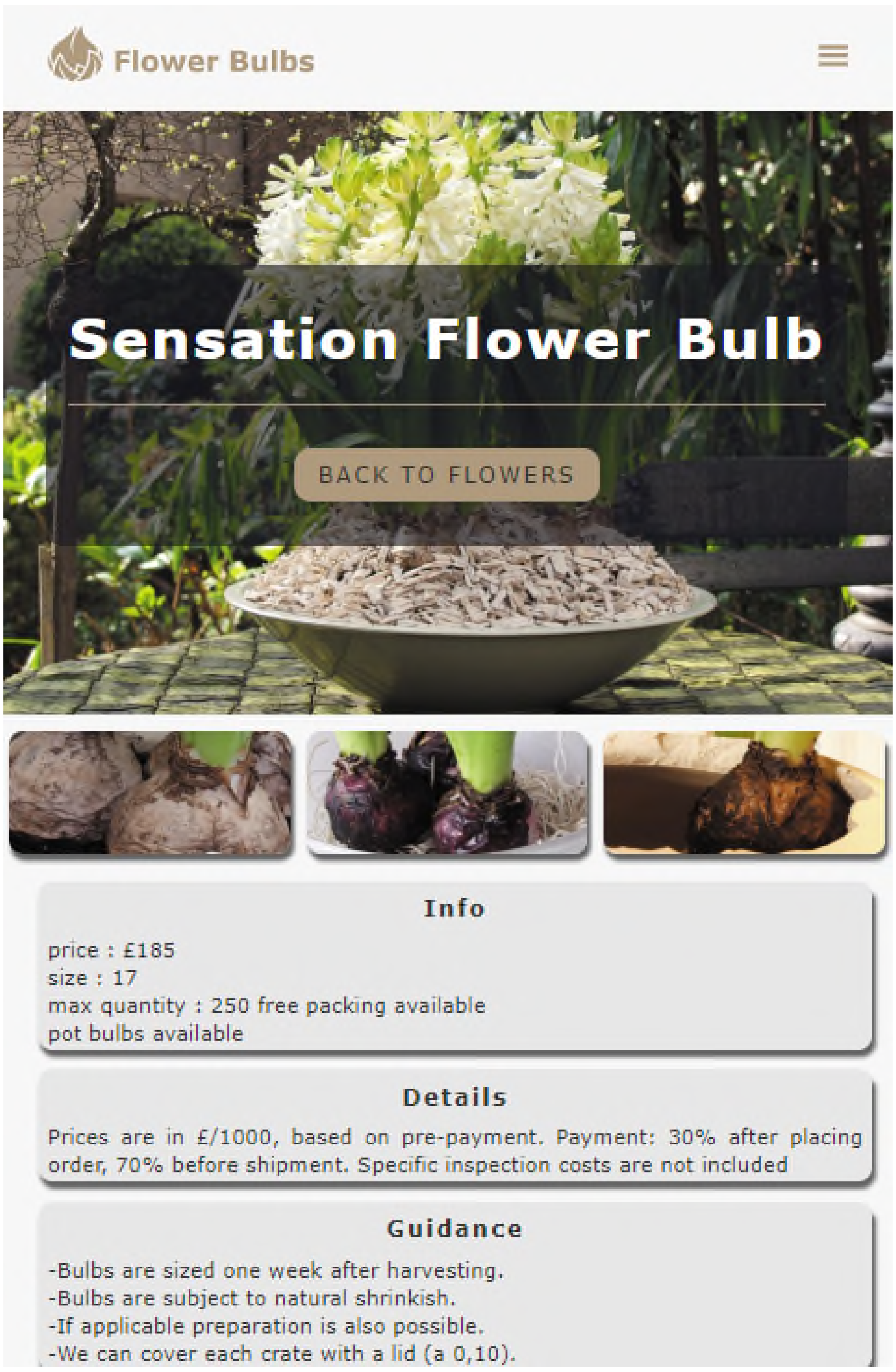
Flower Bulb Type
all

Quantity Per Crate
400

Flower Price £252

Flower Size
0
19


Flower Bulbs
Flower Size
0
19
Cut Flower Bulbs
Potted Flower Bulbs
Crystal
Size: 14
£131
per 1000 bulbs
Fairy



Info

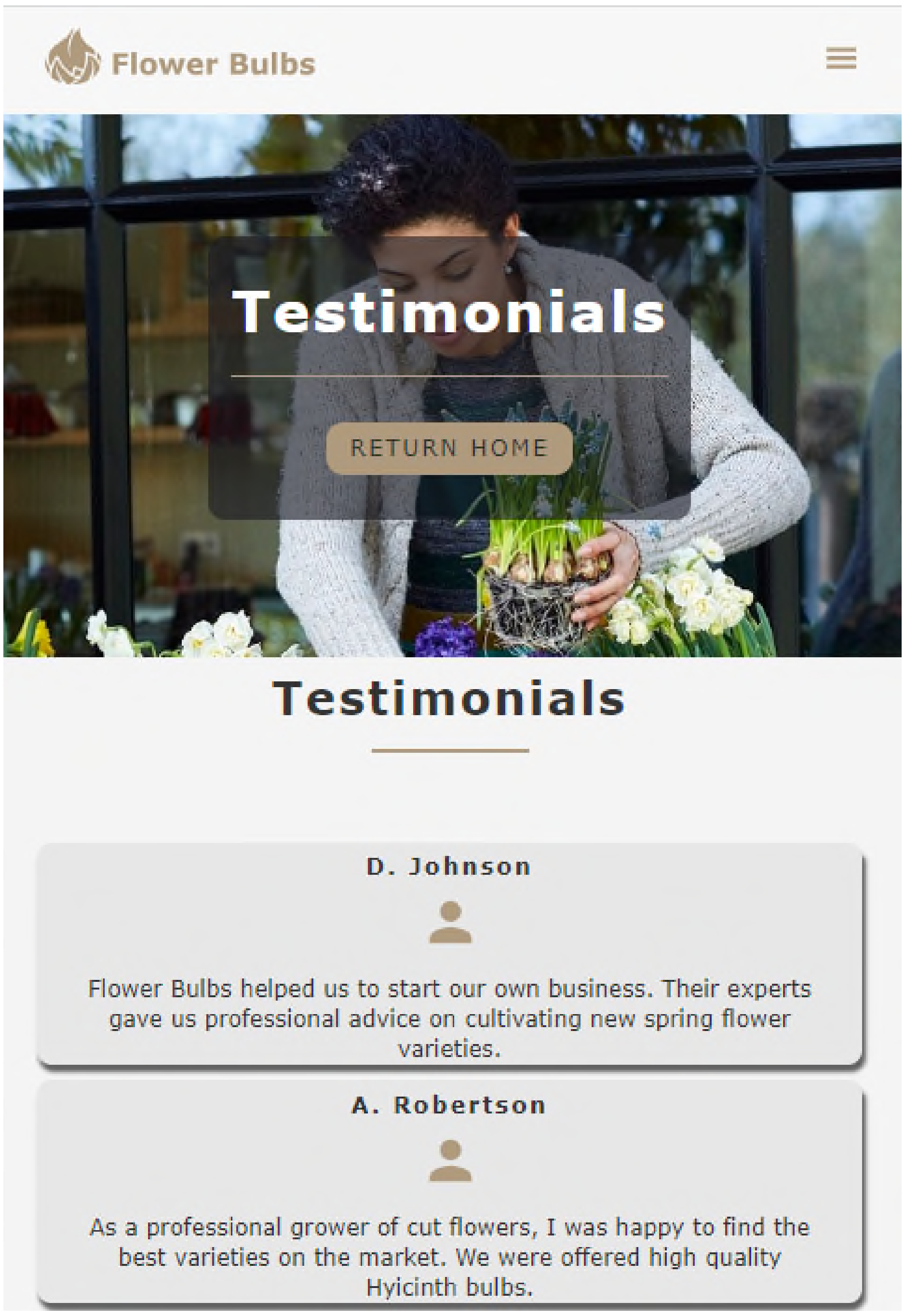
price : £185
size : 17
max quantity : 250 free packing available
pot bulbs available

Details

Prices are in £/1000, based on pre-payment. Payment: 30% after placing order, 70% before shipment. Specific inspection costs are not included

Guidance

-Bulbs are sized one week after harvesting.
-Bulbs are subject to natural shrinkish.
-If applicable preparation is also possible.
-We can cover each crate with a lid (a 0,10).



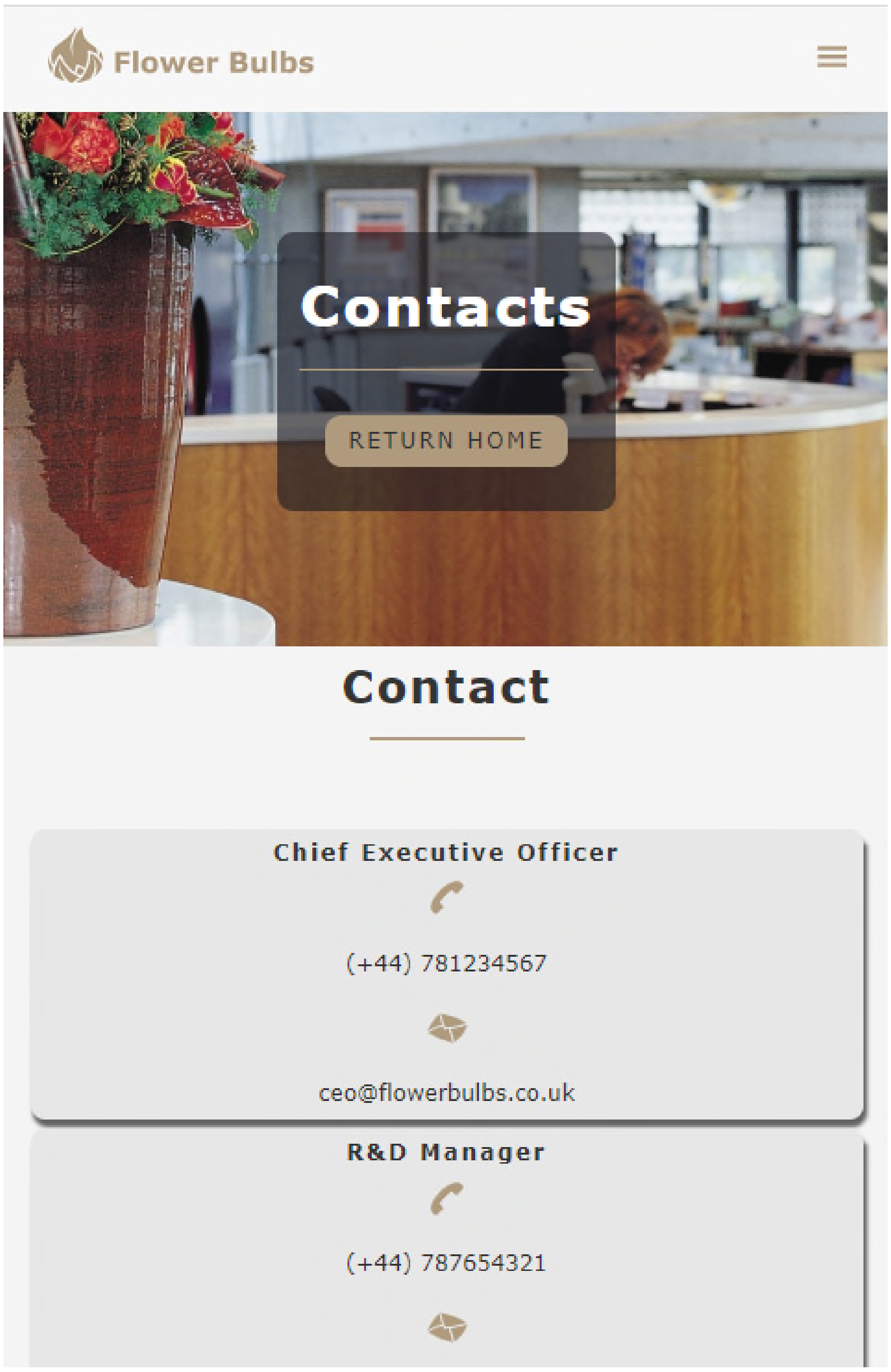
Testimonials

D. Johnson

Flower Bulbs helped us to start our own business. Their experts gave us professional advice on cultivating new spring flower varieties.

A. Robertson

As a professional grower of cut flowers, I was happy to find the best varieties on the market. We were offered high quality Hyicinth bulbs.


Flower Bulbs

Contacts

RETURN HOME

Contact

Chief Executive Officer

(+44) 781234567

ceo@flowerbulbs.co.uk

R&D Manager

(+44) 787654321