React Roundtrip

Lesson 1: AWS EC2, Cloud 9, node.js and create-react-app setup

EC2 setup, Cloud 9 setup. If you have an existing EC2 instance to connect, do these. (Harder)

1. Cloud 9 must connect to an existing ec2 instance
2. Must upgrade the ec2 instance using https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/setting-up-node-on-ec2-instance.html
3. https://stackoverflow.com/questions/12392598/how-to-add-rsa-key-to-authorized-keys-file set up on the new instance in ec2
4. https://stackoverflow.com/questions/28933310/could-not-execute-node-js-cloud9 to setup node on the new AWS instance
5. Select amazon linux, nano instance
6. Set up an inbound list permitting ssh, http, https, outbound to 0.0.0.0
7. Connect to ec2 using the in-browser connection manager, install authorized keys, append to end of file, then install node legacy, per above
8. You can test the visibility of changes you make as you utilize the cloud 9 terminal to create changes on the server

```
[ec2-user@ip-172-31-41-33 ~]$ ls
[ec2-user@ip-172-31-41-33 ~]$ touch newfile.txt
[ec2-user@ip-172-31-41-33 ~]$ cat newfile.txt
hi hi
[ec2-user@ip-172-31-41-33 ~]$ which node
~/.nvm/versions/node/v13.5.0/bin/node
[ec2-user@ip-172-31-41-33 ~]$ which git
/usr/bin/git
[ec2-user@ip-172-31-41-33 ~]$
```

9. Time to set up create react app, using cloud 9's terminal

Easiest: Choose a **T3 small** instance during the Cloud 9 setup phase. Connect to the instance via terminal in Cloud 9, not via session manager in EC2. Simply change the Security Group setting to include http, https, ssh (already there), but add **TCP ports 0-65536**. Then utilize the public IP address given to access the [public ip address]:8080, should open react after:

1. You've done **`npm init`**
2. **`npx create-react-app`** [name of your app, anything will do, no spaces]
3. Open [public ip]:8080, should show react standard.
4. Navigate to root folder, **`npm start`**
5. Get FQDN from EC2, add 8080 as suffix over http, load in separate browser tab

Lesson 2: negotiating and understanding the create-react-app template system

Layout of the empty application:

1. Index.js, global includes
2. Index.html, root page, invoking the root app
3. App.js, the primary loading page default page

Lesson 3: dependencies and remaking the root App.js

1. Run

ec2-user:~/environment/react-cpp (master) $ npm i -s bootstrap react-icons lodash jquery popper.js moment react-moment

2. Appears inside of **package.json**
3. Create css, components directories, update links to App.js and App.css in their respective locations in App.js
4. Must adjust the default by adding folders (css, components) and update reference links to them
5. New code: index.html loses root, with new code beginning with <header white> all the way to <footer>
6. App.js, add new `render`() creating new divs for add, list, search appts, with className

Lesson 4: adding new components, custom built

1. Add ListAppointments, Search and Add, each as a .js file under components directory. Add an export default [component name] to the end of each, with render statements, each with a closed and complete HTML element within.
2. Import each component into App.js, then use as a JSX element within the render() piece.
3. Save each, should update within the production app

Lesson 5: learning state

1. Must add {`Component`} to top line import from React
2. Add `render`() before the return
3. Change the function App() to `class App extends Component`
4. Study the running app in the react developer tool console, in Inspect, click the Components, then the app, where the variable you placed within the main <Div>

```
import React, { Component} from 'react';
import '../css/App.css';
import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

//function App() {
 class App extends Component {
  constructor ()
  {
    super(); //allows data exchange between self and parent compoentn
    this.state = {
      myName: 'Stefan Bund times'
    };
  }
  render(){
  return (
    <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">
              {this.state.myName}
                < AddAppointments/>
                < SearchAppointments />
                < ListAppointments/>
              </div>
            </div>
          </div>
        </div>
      </main>
  );
}
}
```

```
export default App;
```

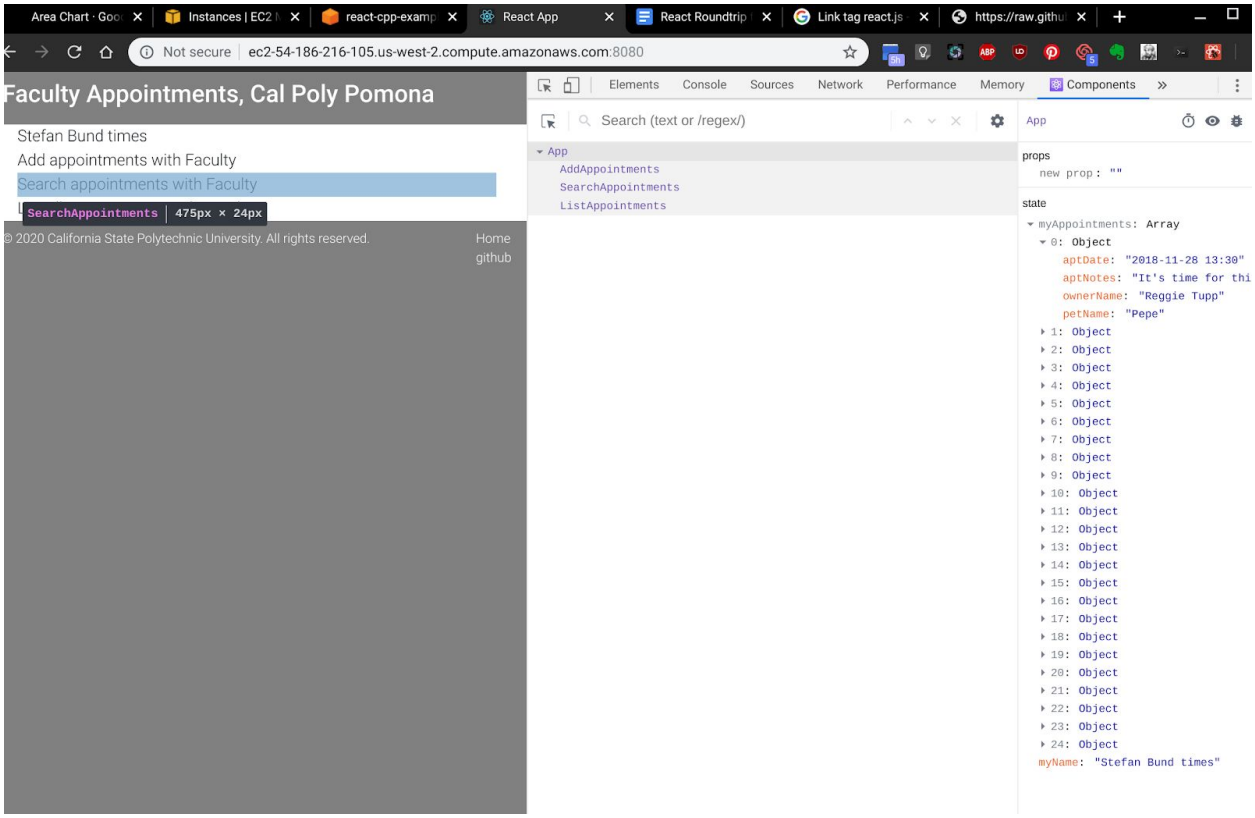
[check component browser for myName appears]

Lesson 6: Lifecycle

Utilize component did mount, get data.json into the public folder. App.js:

```
import React, { Component} from 'react';
import '../css/App.css';
import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

//function App() {
 class App extends Component {
  constructor ()
  {
    super(); //allows data exchange between self and parent compoentn
    this.state = {
      myName: 'Stefan Bund times',
      myAppointments: []
    };
  }

  componentDidMount()
  {
    fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        return item;
      })
      this.setState({
      myAppointments: apts
    });

    });
  }


  render(){
  return (
    <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
```

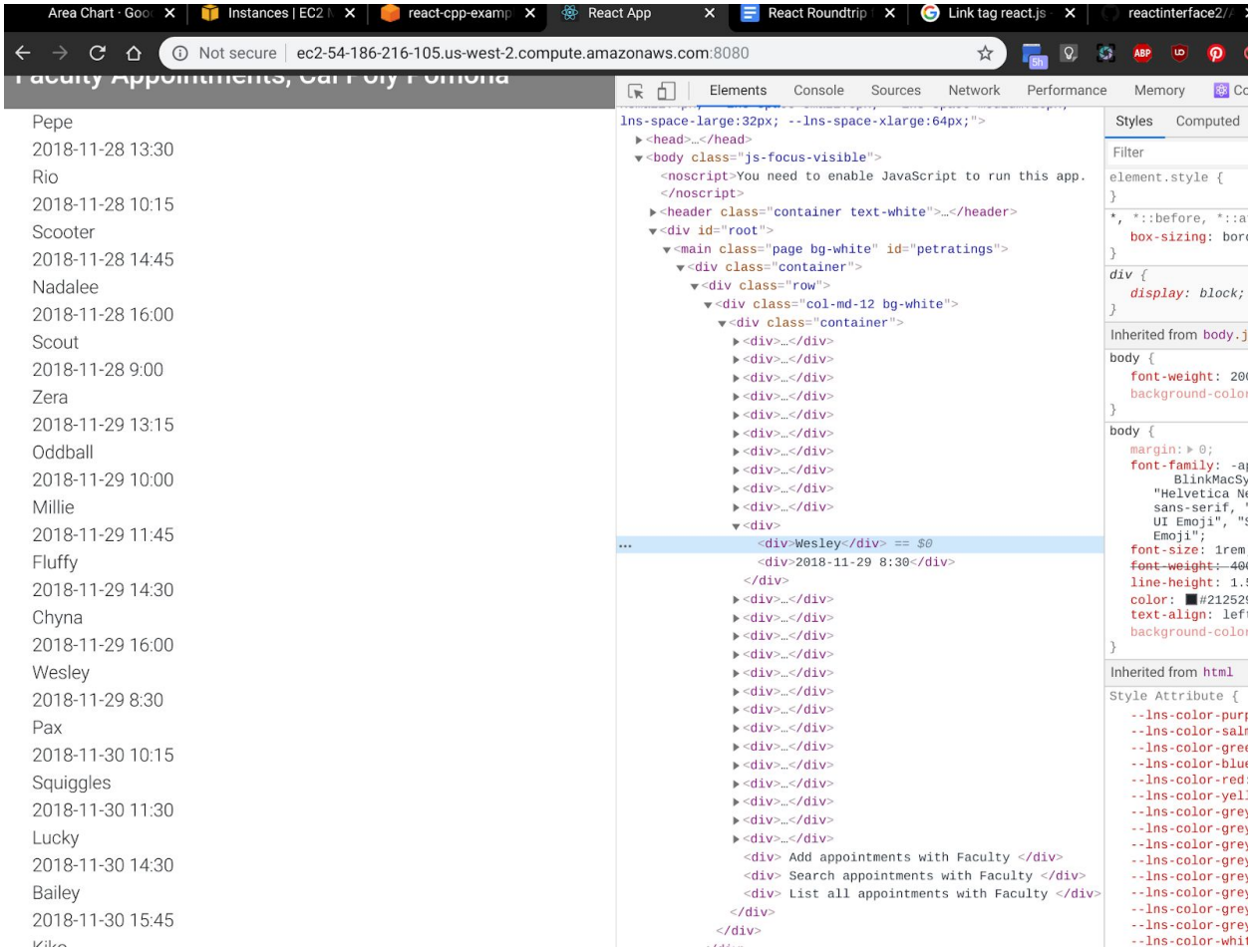
```
            <div className="col-md-12 bg-white">
              <div className="container">
              {this.state.myName}
                < AddAppointments/>
                < SearchAppointments />
                < ListAppointments/>
              </div>
            </div>
          </div>
        </div>
      </main>
  );
}
}

export default App;
```



Should show up in the App Component browser

Lesson 7

Mapping data using inline functions, inside of render(), employing javascript methods inside of render()



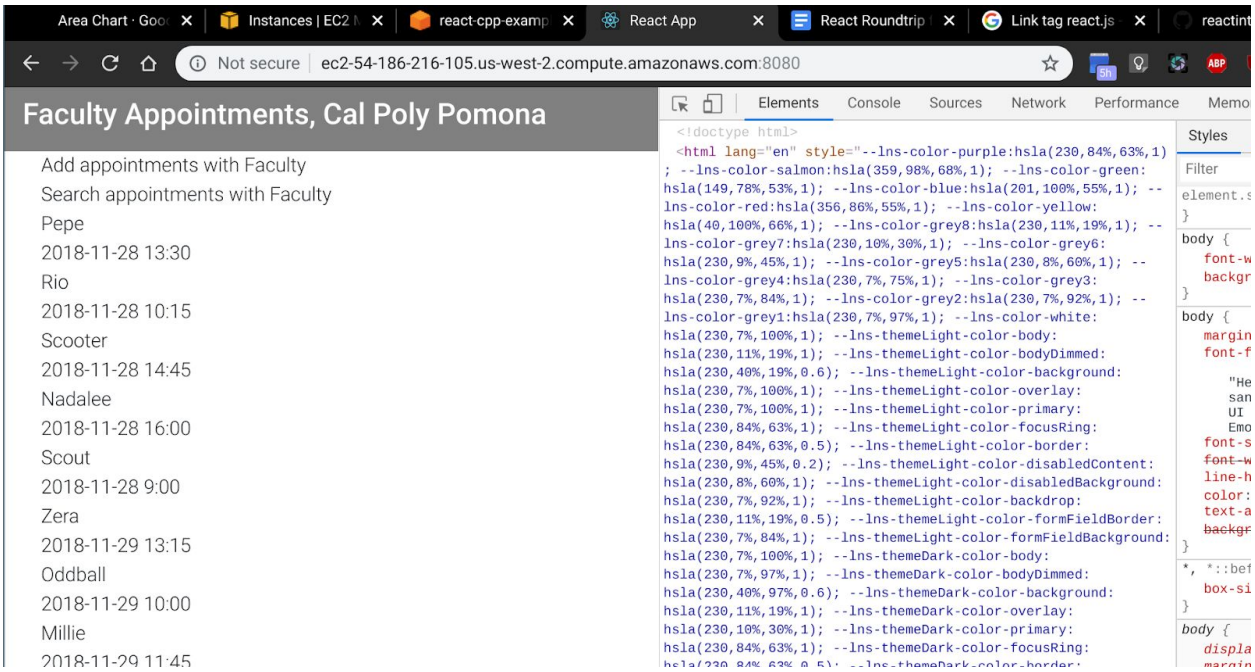

```
import React, { Component} from 'react';
import '../css/App.css';
import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

//function App() {
 class App extends Component {
  constructor ()
  {
    super(); //allows data exchange between self and parent compoentn
    this.state = {
```

```
    myAppointments: []
  };
}

componentDidMount()
{
  fetch('./data.json')
  .then(response => response.json())
  .then(result => {
    const apts = result.map(item => {
      return item;
    })
    this.setState({
    myAppointments: apts
  });

  });
}


render(){

  const listItems = this.state.myAppointments.map(item => (
    <div>
    <div>{item.petName}</div>
    <div>{item.aptDate}</div>

    </div>
  ));

return (
  <main className="page bg-white" id="petratings">
      <div className="container">
        <div className="row">
          <div className="col-md-12 bg-white">
            <div className="container">
            {listItems}
              < AddAppointments/>
              < SearchAppointments />
              < ListAppointments/>
            </div>
          </div>
        </div>
```

```
        </div>
      </main>
  );
}
}

export default App;
```

Lesson 8: props and passing data between components

Updating App.js and ListAppointments.js in order to demonstrate how to fetch data.json in the App.js, pass it down to the subcomponent, then rebuild the UI using the array of data using map(), will do:



App.js:

```
import React, { Component} from 'react';
import '../css/App.css';
import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

//function App() {
 class App extends Component {
  constructor ()
  {
    super(); //allows data exchange between self and parent compoentn
    this.state = {
      myAppointments: []
    };
  }
```

```
  componentDidMount()
  {
    fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        return item;
      })
      this.setState({
      myAppointments: apts
    });

    });
  }

  render(){

  //to demonstrate passing down a value via props to a subcomponent

  return (
    <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">

                < AddAppointments/>
                < SearchAppointments />
                < ListAppointments
appointments={this.state.myAppointments}/>
              </div>
            </div>
          </div>
        </div>
      </main>
  );
}
}

export default App;
```

ListAppointments.js

```
import React, {Component} from 'react';

class ListAppointments extends Component{
    render(){
        //appointments is passed down to ListAppointments via App.js
        const listItems = this.props.appointments.map(item => (
      <div>
      <div>{item.petName}</div>
      <div>{item.aptDate}</div>

      </div>
    ));

    return <div>{listItems}</div>

    }
}
export default ListAppointments;
```

Lesson 9: templating

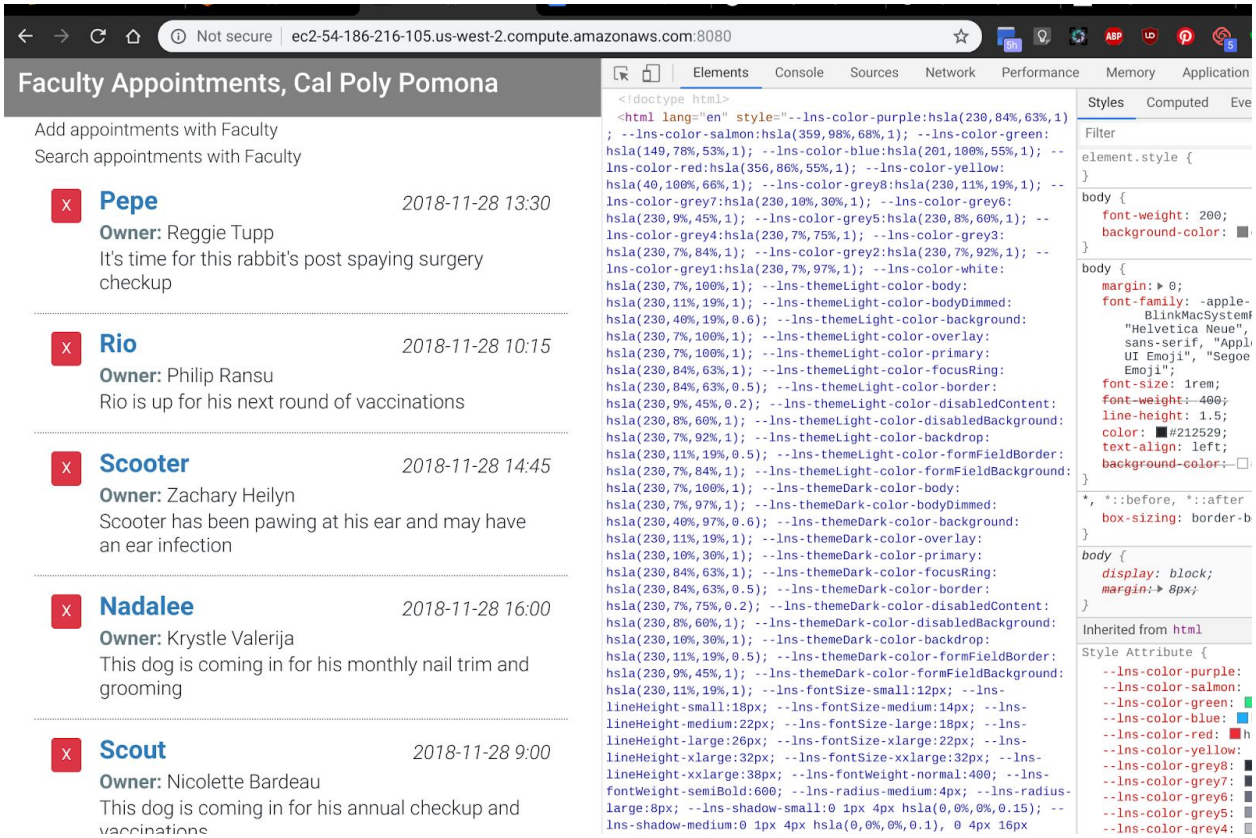
Will comment out listItems in ListAppointments, instead iterating the map of props, and templating div html.

ListAppointments.js, new update

```
import React, {Component} from 'react';

class ListAppointments extends Component{
    render(){
        //appointments is passed down to ListAppointments via App.js
    //     const listItems = this.props.appointments.map(item => (
    //   <div>
    //   <div>{item.petName}</div>
    //   <div>{item.aptDate}</div>

    //   </div>
    // ));

    return (
        <div className="appointment-list item-list mb-3">
        {this.props.appointments.map(item => (
          <div className="pet-item col media py-3">
            <div className="mr-3">
              <button className="pet-delete btn btn-sm
btn-danger">X</button>
            </div>

            <div className="pet-info media-body">
              <div className="pet-head d-flex">
                <span className="pet-name">{item.petName}</span>
                <span className="apt-date
ml-auto">{item.aptDate}</span>
              </div>

              <div className="owner-name">
                <span className="label-item">Owner: </span>
                <span>{item.ownerName}</span>
              </div>
              <div className="apt-notes">{item.aptNotes}</div>
```

```
            </div>
          </div>
        ))}
      </div>
      );

    }
}
export default ListAppointments;
```

Update CSS: take the following new css and replace the contents of **App.css** with it:

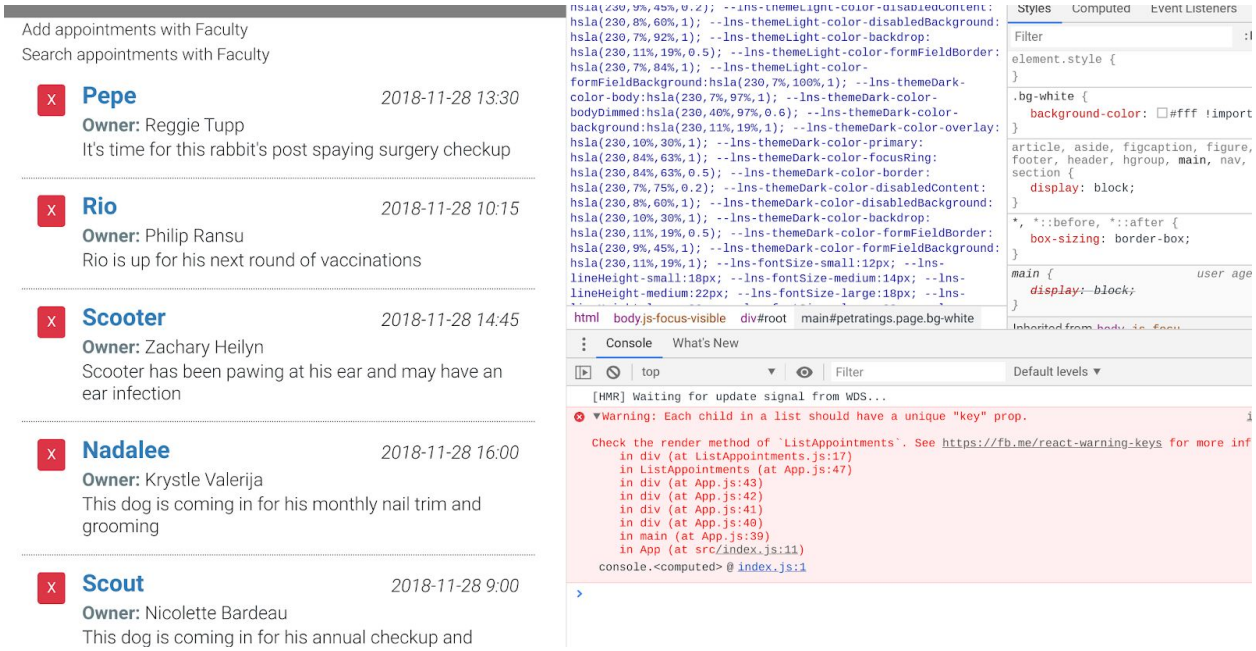



```
.add-appointment .card-body {
  display: none;
```

```
}

.apt-addheading {
  cursor: pointer;
  user-select: none;
}

.appointment-list {
  font-size: 1.1em;
}

.appointment-list .pet-item {
  border-bottom: 1px dotted gray;
}

.appointment-list .pet-item:last-child {
  border-bottom: none;
}

.appointment-list .pet-name {
  font-weight: 600;
  color: #337ab7;
  font-size: 1.3em;
  line-height: 100%;
}

.appointment-list .label-item {
  font-weight: 600;
  color: #667b82;
}

.appointment-list .apt-date {
  font-style: italic;
}

.appointment-list .apt-notes {
  line-height: 120%;
}

.dropdown-item.active,
.dropdown-item:active {
  background-color: #a9aeb2;
}
```

Lesson 10: uniqueness among data

React issues error related to key properties, in array items.



Need to add an index value to the array data, artificially.

Lesson 11: icons, dates via third party libraries

- FaTimes via react-icons
- React-moment
- Already installed at start, look in node_modules and package.json

Changes to ListAppointment.js

```
import React, {Component} from 'react';
import { FaTimes } from 'react-icons/fa';
import Moment from 'react-moment';

class ListAppointments extends Component{
    render(){

    return (
        <div className="appointment-list item-list mb-3">
        {this.props.appointments.map(item => (
          <div className="pet-item col media py-3">
            <div className="mr-3">
             <button className="pet-delete btn btn-sm btn-danger">

              < FaTimes /></button>
            </div>

            <div className="pet-info media-body">
              <div className="pet-head d-flex">
                <span className="pet-name">{item.aptId}:
{item.petName}</span>
                <span className="apt-date
ml-auto">{item.aptDate}</span>
                <Moment
                    date={item.aptDate}
                    parse="YYYY-MM-dd hh:mm"
                    format="MMM-D h:mma"
                  />
              </div>

              <div className="owner-name">
                <span className="label-item">Owner: </span>
```

```
                <span>{item.ownerName}</span>
              </div>
              <div className="apt-notes">{item.aptNotes}</div>
            </div>
          </div>
        ))}
      </div>
      );

    }
}
export default ListAppointments;
```

Lesson 12: clicks, prop handling

App and ListAppointments must add a new method, DeleteAppointments to fire alongside button clicks. Several supporting changes must be made, primarily within App.js, as highlighted below.

App.js

```
import React, { Component} from 'react';
import '../css/App.css';
import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';
import { without } from 'lodash';

//function App() {
 class App extends Component {
  constructor ()
  {
    super(); //allows data exchange between self and parent compoentn
    this.state = {
      myAppointments: [],
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);//must
bind deleteApt to the component
  }

  deleteAppointment(apt) { //must point to this function from within
the component below
    let tempApts = this.state.myAppointments;
    tempApts = without(tempApts, apt);//remove, then

    this.setState({
      myAppointments: tempApts //replace, needs to bind it within the
constructor
    });
  }

  componentDidMount()
  {
    fetch('./data.json')
```

```
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        item.aptId = this.state.lastIndex; //new for 9
        this.setState({ lastIndex: this.state.lastIndex + 1});
        return item;
      })
      this.setState({
      myAppointments: apts
    });

    });
  }



  render(){


  //to demonstrate passing down a value via props to a subcomponent

  return (
    <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">

                < AddAppointments/>
                < SearchAppointments />
                < ListAppointments
appointments={this.state.myAppointments}
                deleteAppointment={this.deleteAppointment}/>
              </div>
            </div>
          </div>
        </div>
      </main>
  );
}
}
```

```
export default App;
```

ListAppointments.js

```
import React, { Component } from 'react';
import { FaTimes } from 'react-icons/fa';
import Moment from 'react-moment';
//needs binding and declaration within the component's use within the
App use
class ListAppointments extends Component {
  render() {
    return (
      <div className="appointment-list item-list mb-3">
        {this.props.appointments.map(item => (
          <div className="pet-item col media py-3" key={item.aptId}>
            <div className="mr-3">
              <button
                className="pet-delete btn btn-sm btn-danger"
                onClick={() => this.props.deleteAppointment(item)}>
                <FaTimes />
              </button>
            </div>

            <div className="pet-info media-body">
              <div className="pet-head d-flex">
                <span className="pet-name">{item.petName}</span>
                <span className="apt-date ml-auto">
                  <Moment
                    date={item.aptDate}
                    parse="YYYY-MM-dd hh:mm"
                    format="MMM-D h:mma"
                  />
                </span>
              </div>

              <div className="owner-name">
                <span className="label-item">Owner: </span>
                <span>{item.ownerName}</span>
              </div>
              <div className="apt-notes">{item.aptNotes}</div>
            </div>
```

```
          </div>
        ))}
      </div>
    );
  }
}

export default ListAppointments;
```

Lesson 13: forDisplay, passing visibility settings via props

Time to enable AddAppointments (AA.js). Load up AA.js with the following, keeping attention on the className, where formDisplay is consumed, via App.js, calling the component.

AddAppointments.js

```
import React, { Component } from 'react';

class AddAppointments extends Component {
  render() {
    return (
      <div
        className={
          'card textcenter mt-3 ' +
          (this.props.formDisplay ? '' : 'add-appointment')
        }
      >
        <div className="apt-addheading card-header bg-primary
text-white">
          Add Appointment
        </div>

        <div className="card-body">
          <form id="aptForm" noValidate>
            <div className="form-group form-row">
              <label
                className="col-md-2 col-form-label text-md-right"
                htmlFor="petName"
                readOnly
              >
                Pet Name
              </label>
              <div className="col-md-10">
                <input
                  type="text"
                  className="form-control"
                  name="petName"
```

```
          placeholder="Pet's Name"
/>
      </div>
    </div>

    <div className="form-group form-row">
      <label
        className="col-md-2 col-form-label text-md-right"
        htmlFor="ownerName"
      >
        Pet Owner
      </label>
      <div className="col-md-10">
        <input
          type="text"
          className="form-control"
          name="ownerName"
          placeholder="Owner's Name"
        />
      </div>
    </div>

    <div className="form-group form-row">
      <label
        className="col-md-2 col-form-label text-md-right"
        htmlFor="aptDate"
      >
        Date
      </label>
      <div className="col-md-4">
        <input
          type="date"
          className="form-control"
          name="aptDate"
          id="aptDate"
        />
      </div>
      <label
        className="col-md-2 col-form-label text-md-right"
        htmlFor="aptTime"
      >
        Time
      </label>
```

```
        <div className="col-md-4">
                <input
                  type="time"
                  className="form-control"
                  name="aptTime"
                  id="aptTime"
                />
              </div>
            </div>

            <div className="form-group form-row">
              <label className="col-md-2 text-md-right"
htmlFor="aptNotes">
                Apt. Notes
              </label>
              <div className="col-md-10">
                <textarea
                  className="form-control"
                  rows="4"
                  cols="50"
                  name="aptNotes"
                  id="aptNotes"
                  placeholder="Appointment Notes"
                />
              </div>
            </div>

            <div className="form-group form-row mb-0">
              <div className="offset-md-2 col-md-10">
                <button
                  type="submit"
                  className="btn btn-primary d-block ml-auto"
                >
                  Add Appointment
                </button>
              </div>
            </div>
          </form>
        </div>
      </div>
    );
  }
}
```

```
export default AddAppointments;
```

App.js, where two small shifts enable AddAppointments to become visible

```
import React, { Component} from 'react';
import '../css/App.css';
import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';
import { without } from 'lodash';

//function App() {
 class App extends Component {
  constructor ()
  {
    super(); //allows data exchange between self and parent compoentn
    this.state = {
      myAppointments: [],
      formDisplay: true,
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);//must
bind deleteApt to the component
  }

  deleteAppointment(apt) { //must point to this function from within
the component below
    let tempApts = this.state.myAppointments;
    tempApts = without(tempApts, apt);//remove, then

    this.setState({
      myAppointments: tempApts //replace, needs to bind it within the
constructor
    });
  }

  componentDidMount()
  {
    fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
```

```
        item.aptId = this.state.lastIndex; //new for 9
        this.setState({ lastIndex: this.state.lastIndex + 1});
        return item;
      })
      this.setState({
      myAppointments: apts
    });

    });
  }




  render(){


  //to demonstrate passing down a value via props to a subcomponent

  return (
    <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">

                < AddAppointments formDisplay={this.state.formDisplay}
/>
                < SearchAppointments />
                < ListAppointments
appointments={this.state.myAppointments}
                deleteAppointment={this.deleteAppointment}/>
              </div>
            </div>
          </div>
        </div>
      </main>
  );
}
}

export default App;
```

Lesson 14: Toggle elements in UI

Update AddAppointments.js with the following bold elements, to enable a (+) sign toggle in the main ui.

AddAppointments.js changes (in bold)

```
import React, { Component } from 'react';
import { FaPlus } from 'react-icons/fa';

class AddAppointments extends Component {
  render() {
    return (
      <div
        className={
          'card textcenter mt-3 ' +
          (this.props.formDisplay ? '' : 'add-appointment')
        }
      >
        <div
          className="apt-addheading card-header bg-primary
text-white"
          onClick={this.props.toggleForm}
        >
          <FaPlus /> Add Appointment
        </div>

        <div className="card-body">
          <form id="aptForm" noValidate>
            <div className="form-group form-row">
              <label
                className="col-md-2 col-form-label text-md-right"
                htmlFor="petName"
                readOnly
              >
                Pet Name
              </label>
              <div className="col-md-10">
                <input
                  type="text"
                  className="form-control"
```

```
      name="petName"
      placeholder="Pet's Name"
    />
  </div>
</div>

<div className="form-group form-row">
  <label
    className="col-md-2 col-form-label text-md-right"
    htmlFor="ownerName"
  >
    Pet Owner
  </label>
  <div className="col-md-10">
    <input
      type="text"
      className="form-control"
      name="ownerName"
      placeholder="Owner's Name"
    />
  </div>
</div>

<div className="form-group form-row">
  <label
    className="col-md-2 col-form-label text-md-right"
    htmlFor="aptDate"
  >
    Date
  </label>
  <div className="col-md-4">
    <input
      type="date"
      className="form-control"
      name="aptDate"
      id="aptDate"
    />
  </div>
  <label
    className="col-md-2 col-form-label text-md-right"
    htmlFor="aptTime"
  >
    Time
```

```
              </label>
              <div className="col-md-4">
                <input
                  type="time"
                  className="form-control"
                  name="aptTime"
                  id="aptTime"
                />
              </div>
            </div>

            <div className="form-group form-row">
              <label className="col-md-2 text-md-right"
htmlFor="aptNotes">
                Apt. Notes
              </label>
              <div className="col-md-10">
                <textarea
                  className="form-control"
                  rows="4"
                  cols="50"
                  name="aptNotes"
                  id="aptNotes"
                  placeholder="Appointment Notes"
                />
              </div>
            </div>

            <div className="form-group form-row mb-0">
              <div className="offset-md-2 col-md-10">
                <button
                  type="submit"
                  className="btn btn-primary d-block ml-auto"
                >
                  Add Appointment
                </button>
              </div>
            </div>
          </form>
        </div>
      </div>
    );
  }
```

```
}

export default AddAppointments;
```

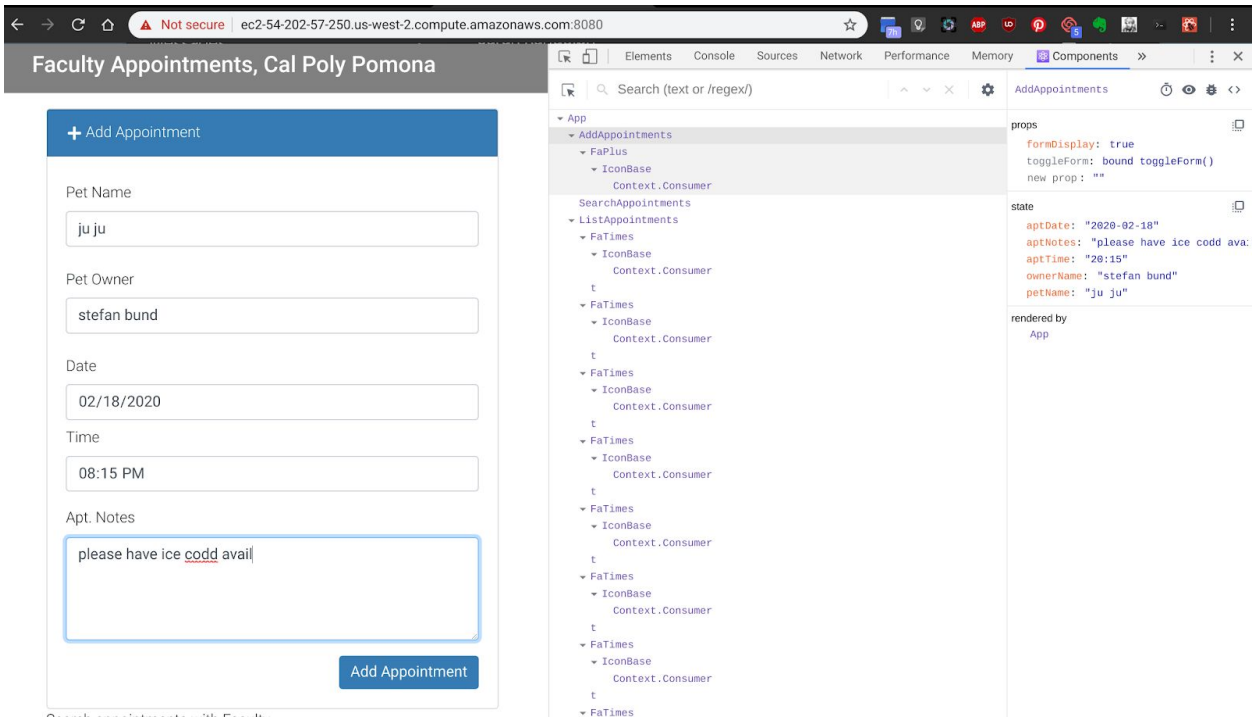
Changes to App.js:

```
import React, { Component } from 'react';
import '../css/App.css';

import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

import { without } from 'lodash';

class App extends Component {
  constructor() {
    super();
    this.state = {
      myAppointments: [],
      formDisplay: false,
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);
    this.toggleForm = this.toggleForm.bind(this);
  }

  toggleForm() {
    this.setState({
      formDisplay: !this.state.formDisplay
    });
  }

  deleteAppointment(apt) {
    let tempApts = this.state.myAppointments;
    tempApts = without(tempApts, apt);

    this.setState({
      myAppointments: tempApts
    });
  }
```

```
 componentDidMount() {
    fetch('./data.json')
      .then(response => response.json())
      .then(result => {
        const apts = result.map(item => {
          item.aptId = this.state.lastIndex;
          this.setState({ lastIndex: this.state.lastIndex + 1 });
          return item;
        });
        this.setState({
          myAppointments: apts
        });
      });
  }

  render() {
    return (
      <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">
                <AddAppointments
                  formDisplay={this.state.formDisplay}
                  toggleForm={this.toggleForm}
                />
                <SearchAppointments />
                <ListAppointments
                  appointments={this.state.myAppointments}
                  deleteAppointment={this.deleteAppointment}
                />
              </div>
            </div>
          </div>
        </div>
      </main>
    );
  }
}

export default App;
```

Lesson 15: capturing user input as state

Let’s shoot for entering in data to each input element, then verifying that state has gotten hold of it, using React Components in the Inspector. Here is where we will be heading today:



State in the component is equipped to store a set of variables, reflecting the data set passed down to it via App.js.
Each input must be able to set its value equal to the state, of one of the variables, then pass that value to handleChange(), where that value can be input, to update the model of the state. Each input area in the UI will correspond to one value inside of the state, and will updated with handle change. The screenshot shows how the states undergoes changes once you begin typing, a fascinating way to study the way the UI element consumes and updates data. However, the master data set won’t be updated in these code changes, yet.

App.js undergoes no changes, but AA.js does:

```
import React, { Component } from 'react';
import { FaPlus } from 'react-icons/fa';

class AddAppointments extends Component {
    constructor() {
    super();
    this.state = {
      petName: '',
      ownerName: '',
      aptDate: '',
      aptTime: '',
```

```
    aptNotes: ''
  };
  this.handleChange = this.handleChange.bind(this);
}

handleChange(e) {
  const target = e.target;
  const value = target.value;
  const name = target.name;

  this.setState({
    [name]: value
  });
}


render() {
  return (
    <div
      className={
        'card textcenter mt-3 ' +
        (this.props.formDisplay ? '' : 'add-appointment')
      }
    >
      <div
        className="apt-addheading card-header bg-primary text-white"
        onClick={this.props.toggleForm}
      >
        <FaPlus /> Add Appointment
      </div>

      <div className="card-body">
        <form id="aptForm" noValidate>
          <div className="form-group form-row">
            <label
              className="col-md-2 col-form-label text-md-right"
              htmlFor="petName"
              readOnly
            >
              Pet Name
            </label>
            <div className="col-md-10">
              <input
                type="text"
                className="form-control"
                name="petName"
                placeholder="Pet's Name"
                value={this.state.petName}
                onChange={this.handleChange}
              />
```

```
  </div>
</div>

<div className="form-group form-row">
  <label
    className="col-md-2 col-form-label text-md-right"
    htmlFor="ownerName"
  >
    Pet Owner
  </label>
  <div className="col-md-10">
    <input
      type="text"
      className="form-control"
      name="ownerName"
      placeholder="Owner's Name"
      value={this.state.ownerName}
      onChange={this.handleChange}
    />
  </div>
</div>

<div className="form-group form-row">
  <label
    className="col-md-2 col-form-label text-md-right"
    htmlFor="aptDate"
  >
    Date
  </label>
  <div className="col-md-4">
    <input
      type="date"
      className="form-control"
      name="aptDate"
      id="aptDate"
      value={this.state.aptDate}
      onChange={this.handleChange}
    />
  </div>
  <label
    className="col-md-2 col-form-label text-md-right"
    htmlFor="aptTime"
  >
    Time
  </label>
  <div className="col-md-4">
    <input
      type="time"
      className="form-control"
      name="aptTime"
```

```
                id="aptTime"
                value={this.state.aptTime}
                onChange={this.handleChange}
              />
            </div>
          </div>

          <div className="form-group form-row">
            <label className="col-md-2 text-md-right" htmlFor="aptNotes">
              Apt. Notes
            </label>
            <div className="col-md-10">
              <textarea
                className="form-control"
                rows="4"
                cols="50"
                name="aptNotes"
                id="aptNotes"
                placeholder="Appointment Notes"
                value={this.state.aptNotes}
                onChange={this.handleChange}
              />
            </div>
          </div>

          <div className="form-group form-row mb-0">
            <div className="offset-md-2 col-md-10">
              <button
                type="submit"
                className="btn btn-primary d-block ml-auto"
              >
                Add Appointment
              </button>
            </div>
          </div>
        </form>
      </div>
    </div>
  );
 }
}

export default AddAppointments;
```

Lesson 16: adding to the master UI's state

Time to add data to our interface, and update the master array of data. We must make multiple additions, below.

AddAppointments.js

```
import React, { Component } from 'react';
import { FaPlus } from 'react-icons/fa';

class AddAppointments extends Component {
  constructor() {
    super();
    this.state = {
      petName: '',
      ownerName: '',
      aptDate: '',
      aptTime: '',
      aptNotes: ''
    };
    this.handleChange = this.handleChange.bind(this);
    this.handleAdd = this.handleAdd.bind(this);
  }

  handleAdd(e) {
    e.preventDefault();
    let tempApt = {
      petName: this.state.petName,
      ownerName: this.state.ownerName,
      aptDate: this.state.aptDate + ' ' + this.state.aptTime,
      aptNotes: this.state.aptNotes
    };

    this.props.addAppointment(tempApt);

    this.setState({
      petName: '',
      ownerName: '',
      aptDate: '',
      aptTime: '',
      aptNotes: ''
    });
    this.props.toggleForm();
  }
```

```
handleChange(e) {
  const target = e.target;
  const value = target.value;
  const name = target.name;

  this.setState({
    [name]: value
  });
}

render() {
  return (
    <div
      className={
        'card textcenter mt-3 ' +
        (this.props.formDisplay ? '' : 'add-appointment')
      }
    >
      <div
        className="apt-addheading card-header bg-primary text-white"
        onClick={this.props.toggleForm}
      >
        <FaPlus /> Add Appointment
      </div>

      <div className="card-body">
        <form id="aptForm" noValidate onSubmit={this.handleAdd}>
          <div className="form-group form-row">
            <label
              className="col-md-2 col-form-label text-md-right"
              htmlFor="petName"
              readOnly
            >
              Pet Name
            </label>
            <div className="col-md-10">
              <input
                type="text"
                className="form-control"
                name="petName"
                placeholder="Pet's Name"
                value={this.state.petName}
                onChange={this.handleChange}
              />
            </div>
          </div>

          <div className="form-group form-row">
            <label
              className="col-md-2 col-form-label text-md-right"
```

```
    htmlFor="ownerName"
  >
    Pet Owner
  </label>
  <div className="col-md-10">
    <input
      type="text"
      className="form-control"
      name="ownerName"
      placeholder="Owner's Name"
      value={this.state.ownerName}
      onChange={this.handleChange}
    />
  </div>
</div>

<div className="form-group form-row">
  <label
    className="col-md-2 col-form-label text-md-right"
    htmlFor="aptDate"
  >
    Date
  </label>
  <div className="col-md-4">
    <input
      type="date"
      className="form-control"
      name="aptDate"
      id="aptDate"
      value={this.state.aptDate}
      onChange={this.handleChange}
    />
  </div>
  <label
    className="col-md-2 col-form-label text-md-right"
    htmlFor="aptTime"
  >
    Time
  </label>
  <div className="col-md-4">
    <input
      type="time"
      className="form-control"
      name="aptTime"
      id="aptTime"
      value={this.state.aptTime}
      onChange={this.handleChange}
    />
  </div>
</div>
```

```
            <div className="form-group form-row">
              <label className="col-md-2 text-md-right" htmlFor="aptNotes">
                Apt. Notes
              </label>
              <div className="col-md-10">
                <textarea
                  className="form-control"
                  rows="4"
                  cols="50"
                  name="aptNotes"
                  id="aptNotes"
                  placeholder="Appointment Notes"
                  value={this.state.aptNotes}
                  onChange={this.handleChange}
                />
              </div>
            </div>

            <div className="form-group form-row mb-0">
              <div className="offset-md-2 col-md-10">
                <button
                  type="submit"
                  className="btn btn-primary d-block ml-auto"
                >
                  Add Appointment
                </button>
              </div>
            </div>
          </form>
        </div>
      </div>
    );
  }
}

export default AddAppointments;
```

App.js

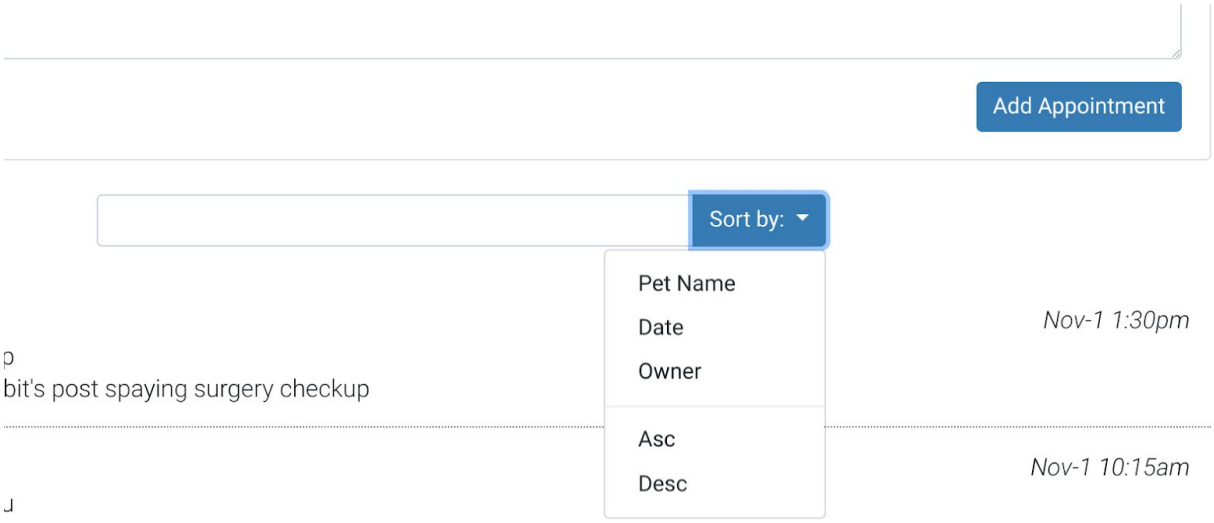
```
import React, { Component} from 'react';
import '../css/App.css';
import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';
import { without } from 'lodash';

//function App() {
```

```
 class App extends Component {
  constructor ()
  {
    super(); //allows data exchange between self and parent compoentn
    this.state = {
      myAppointments: [],
      formDisplay: true,
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);//must bind deleteApt
to the component
    this.toggleForm = this.toggleForm.bind(this);
    this.addAppointment = this.addAppointment.bind(this);

  }

  toggleForm() {
    this.setState({
      formDisplay: !this.state.formDisplay
    });
  }


  deleteAppointment(apt) { //must point to this function from within the component
below
    let tempApts = this.state.myAppointments;
    tempApts = without(tempApts, apt);//remove, then

    this.setState({
      myAppointments: tempApts //replace, needs to bind it within the constructor
    });
  }

  componentDidMount()
  {
    fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        item.aptId = this.state.lastIndex; //new for 9
        this.setState({ lastIndex: this.state.lastIndex + 1});
          return item;
      })
      this.setState({
      myAppointments: apts
    });

    });
  }
```

```
  addAppointment(apt) {
    let tempApts = this.state.myAppointments;
    apt.aptId = this.state.lastIndex;
    tempApts.unshift(apt);//creates a new version of the array
    this.setState({
      myAppointments: tempApts,
      lastIndex: this.state.lastIndex + 1
    });//must rebuild the incrementable index
  }


  render(){
     //to demonstrate passing down a value via props to a subcomponent

  return (
    <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">

                < AddAppointments
                formDisplay={this.state.formDisplay}
                toggleForm={this.toggleForm}
                addAppointment={this.addAppointment}/>
                < SearchAppointments />
                < ListAppointments appointments={this.state.myAppointments}
                deleteAppointment={this.deleteAppointment}/>
              </div>
            </div>
          </div>
        </div>
      </main>
  );
}
}

export default App;
```

Lesson 17: initiating component #3, the search component

We will add new html to the searchAppointments.js component to enable new UI like so:



For this rendition, the dropdown sort terms don't trigger yet, however, you 'll see the listed appointments appear, as ordered by parameters in App.js. You will gain the ability to sort the appointments, on demand in the next lessons.

App.js, helping to sort the existing appointments:

```
import React, { Component } from 'react';
import '../css/App.css';

import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

import { without } from 'lodash';

class App extends Component {
  constructor() {
    super();
    this.state = {
      myAppointments: [],
      formDisplay: false,
      orderBy: 'ownerName',
      orderDir: 'asc',
      lastIndex: 0
    };
```

```
  this.deleteAppointment = this.deleteAppointment.bind(this);
  this.toggleForm = this.toggleForm.bind(this);
  this.addAppointment = this.addAppointment.bind(this);
}

toggleForm() {
  this.setState({
    formDisplay: !this.state.formDisplay
  });
}

addAppointment(apt) {
  let tempApts = this.state.myAppointments;
  apt.aptId = this.state.lastIndex;
  tempApts.unshift(apt);
  this.setState({
    myAppointments: tempApts,
    lastIndex: this.state.lastIndex + 1
  });
}

deleteAppointment(apt) {
  let tempApts = this.state.myAppointments;
  tempApts = without(tempApts, apt);

  this.setState({
    myAppointments: tempApts
  });
}

componentDidMount() {
  fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        item.aptId = this.state.lastIndex;
        this.setState({ lastIndex: this.state.lastIndex + 1 });
        return item;
      });
      this.setState({
        myAppointments: apts
      });
    });
}

render() {
  let order;
  let filteredApts = this.state.myAppointments;
  if (this.state.orderDir === 'asc') {
    order = 1;
```

```
    } else {
      order = -1;
    }

    filteredApts.sort((a, b) => {
      if (
        a[this.state.orderBy].toLowerCase() <
        b[this.state.orderBy].toLowerCase()
      ) {
        return -1 * order;
      } else {
        return 1 * order;
      }
    });

    return (
      <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">
                <AddAppointments
                  formDisplay={this.state.formDisplay}
                  toggleForm={this.toggleForm}
                  addAppointment={this.addAppointment}
                />
                <SearchAppointments />
                <ListAppointments
                  appointments={filteredApts}
                  deleteAppointment={this.deleteAppointment}
                />
              </div>
            </div>
          </div>
        </div>
      </main>
    );
  }
}

export default App;
```

SearchAppointments.js, adding new html for the first time, in bold:

```
import React, { Component } from 'react';

class SearchAppointments extends Component {
  render() {
    return (
      <div className="search-appointments row justify-content-center my-4">
```

```
        <div className="col-md-6">
          <div className="input-group">
            <input
              id="SearchApts"
              type="text"
              className="form-control"
              aria-label="Search Appointments"
            />
            <div className="input-group-append">
              <button
                type="button"
                className="btn btn-primary dropdown-toggle"
                data-toggle="dropdown"
                aria-haspopup="true"
                aria-expanded="false"
              >
                Sort by: <span className="caret" />
              </button>

              <div className="sort-menu dropdown-menu dropdown-menu-right">
                <button className="sort-by dropdown-item" href="#">
                  Pet Name
                </button>
                <button className="sort-by dropdown-item" href="#">
                  Date
                </button>
                <button className="sort-by dropdown-item" href="#">
                  Owner
                </button>
                <div role="separator" className="dropdown-divider" />
                <button className="sort-by dropdown-item" href="#">
                  Asc
                </button>
                <button className="sort-by dropdown-item" href="#">
                  Desc
                </button>
              </div>
            </div>
          </div>
        </div>
      );
    }
}

export default SearchAppointments;
```

Lesson 18: conveying state variables to the component, to facilitate a sorted search

Utilize search parameters in the state of App.js to communicate parameters for search. You will see how SearchAppointments.js captures these stateful preferences, reading the orderBy variable, then activating the search with these parameters. Since we did this in the last lesson, we are now selecting/activating the search terms used in the search. This prepares us to send actual search keywords to the subcomponent, and the sort method (asc/desc).

App.js updates:

```
import React, { Component } from 'react';
import '../css/App.css';

import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

import { without } from 'lodash';

class App extends Component {
  constructor() {
    super();
    this.state = {
      myAppointments: [],
      formDisplay: false,
      orderBy: 'petName',
      orderDir: 'desc',
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);
    this.toggleForm = this.toggleForm.bind(this);
    this.addAppointment = this.addAppointment.bind(this);
  }

  toggleForm() {
    this.setState({
      formDisplay: !this.state.formDisplay
    });
  }
```

```
addAppointment(apt) {
  let tempApts = this.state.myAppointments;
  apt.aptId = this.state.lastIndex;
  tempApts.unshift(apt);
  this.setState({
    myAppointments: tempApts,
    lastIndex: this.state.lastIndex + 1
  });
}

deleteAppointment(apt) {
  let tempApts = this.state.myAppointments;
  tempApts = without(tempApts, apt);

  this.setState({
    myAppointments: tempApts
  });
}

componentDidMount() {
  fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        item.aptId = this.state.lastIndex;
        this.setState({ lastIndex: this.state.lastIndex + 1 });
        return item;
      });
      this.setState({
        myAppointments: apts
      });
    });
}

render() {
  let order;
  let filteredApts = this.state.myAppointments;
  if (this.state.orderDir === 'asc') {
    order = 1;
  } else {
    order = -1;
  }

  filteredApts.sort((a, b) => {
    if (
      a[this.state.orderBy].toLowerCase() <
      b[this.state.orderBy].toLowerCase()
    ) {
      return -1 * order;
    } else {
```

```
        return 1 * order;
      }
    });

    return (
      <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">
                <AddAppointments
                  formDisplay={this.state.formDisplay}
                  toggleForm={this.toggleForm}
                  addAppointment={this.addAppointment}
                />
                <SearchAppointments
                  orderBy={this.state.orderBy}
                  orderDir={this.state.orderDir}
                />
                <ListAppointments
                  appointments={filteredApts}
                  deleteAppointment={this.deleteAppointment}
                />
              </div>
            </div>
          </div>
        </div>
      </main>
    );
  }
}

export default App;
```

SearchAppointments.js updates, extensive:

```
import React, { Component } from 'react';

class SearchAppointments extends Component {
  render() {
    return (
      <div className="search-appointments row justify-content-center my-4">
        <div className="col-md-6">
          <div className="input-group">
            <input
              id="SearchApts"
              type="text"
```

```
  className="form-control"
  aria-label="Search Appointments"
/>
<div className="input-group-append">
  <button
    type="button"
    className="btn btn-primary dropdown-toggle"
    data-toggle="dropdown"
    aria-haspopup="true"
    aria-expanded="false"
  >
    Sort by: <span className="caret" />
  </button>

  <div className="sort-menu dropdown-menu dropdown-menu-right">
    <button
      className={
        'sort-by dropdown-item ' +
        (this.props.orderBy === 'petName' ? 'active' : '')
      }
      href="#"
    >
      Pet Name
    </button>
    <button
      className={
        'sort-by dropdown-item ' +
        (this.props.orderBy === 'aptDate' ? 'active' : '')
      }
      href="#"
    >
      Date
    </button>
    <button
      className={
        'sort-by dropdown-item ' +
        (this.props.orderBy === 'ownerName' ? 'active' : '')
      }
      href="#"
    >
      Owner
    </button>
    <div role="separator" className="dropdown-divider" />
    <button
      className={
        'sort-by dropdown-item ' +
        (this.props.orderDir === 'asc' ? 'active' : '')
      }
      href="#"
    >
```

```
                  Asc
                </button>
                <button
                  className={
                    'sort-by dropdown-item ' +
                    (this.props.orderDir === 'desc' ? 'active' : '')
                  }
                  href="#"
                >
                  Desc
                </button>
              </div>
            </div>
          </div>
        </div>
      </div>
    );
  }
}

export default SearchAppointments;
```

Lesson 18 part 2: fulfilling a complete set of sort operations, in UI

During the last two lessons we established a way to pass sort parameters to the subcomponent, via state, then reaccess the order of the listed items in the appointments array via props. This enabled a two-way communication between the UI in the subcomponent (SearchAppointments) and the master collection of data in App.js.

App.js updates, per normal parent component constructor, body and render() area updates

```
import React, { Component } from 'react';
import '../css/App.css';

import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

import { without } from 'lodash';

class App extends Component {
  constructor() {
    super();
    this.state = {
      myAppointments: [],
      formDisplay: false,
      orderBy: 'petName',
      orderDir: 'asc',
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);
    this.toggleForm = this.toggleForm.bind(this);
    this.addAppointment = this.addAppointment.bind(this);
    this.changeOrder = this.changeOrder.bind(this);
  }

  toggleForm() {
    this.setState({
      formDisplay: !this.state.formDisplay
    });
  }

  changeOrder(order, dir) {
    this.setState({
      orderBy: order,
```

```
    orderDir: dir
  });
}

addAppointment(apt) {
  let tempApts = this.state.myAppointments;
  apt.aptId = this.state.lastIndex;
  tempApts.unshift(apt);
  this.setState({
    myAppointments: tempApts,
    lastIndex: this.state.lastIndex + 1
  });
}

deleteAppointment(apt) {
  let tempApts = this.state.myAppointments;
  tempApts = without(tempApts, apt);

  this.setState({
    myAppointments: tempApts
  });
}

componentDidMount() {
  fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        item.aptId = this.state.lastIndex;
        this.setState({ lastIndex: this.state.lastIndex + 1 });
        return item;
      });
      this.setState({
        myAppointments: apts
      });
    });
}

render() {
  let order;
  let filteredApts = this.state.myAppointments;
  if (this.state.orderDir === 'asc') {
    order = 1;
  } else {
    order = -1;
  }

  filteredApts.sort((a, b) => {
    if (
      a[this.state.orderBy].toLowerCase() <
```

```
        b[this.state.orderBy].toLowerCase()
      ) {
        return -1 * order;
      } else {
        return 1 * order;
      }
    });

    return (
      <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">
                <AddAppointments
                  formDisplay={this.state.formDisplay}
                  toggleForm={this.toggleForm}
                  addAppointment={this.addAppointment}
                />
                <SearchAppointments
                  orderBy={this.state.orderBy}
                  orderDir={this.state.orderDir}
                  changeOrder={this.changeOrder}
                />
                <ListAppointments
                  appointments={filteredApts}
                  deleteAppointment={this.deleteAppointment}
                />
              </div>
            </div>
          </div>
        </div>
      </main>
    );
  }
}

export default App;
```

SearchAppointments.js updates, consuming sort parameters (via state), then adjusting the calling component's data (via props)

```
import React, { Component } from 'react';

class SearchAppointments extends Component {
  render() {
```

```
return (
  <div className="search-appointments row justify-content-center my-4">
    <div className="col-md-6">
      <div className="input-group">
        <input
          id="SearchApts"
          type="text"
          className="form-control"
          aria-label="Search Appointments"
        />
        <div className="input-group-append">
          <button
            type="button"
            className="btn btn-primary dropdown-toggle"
            data-toggle="dropdown"
            aria-haspopup="true"
            aria-expanded="false"
          >
            Sort by: <span className="caret" />
          </button>

          <div className="sort-menu dropdown-menu dropdown-menu-right">
            <button
              className={
                'sort-by dropdown-item ' +
                (this.props.orderBy === 'petName' ? 'active' : '')
              }
              onClick={e =>
                this.props.changeOrder('petName', this.props.orderDir)
              }
              href="#"
            >
              Pet Name
            </button>
            <button
              className={
                'sort-by dropdown-item ' +
                (this.props.orderBy === 'aptDate' ? 'active' : '')
              }
              onClick={e =>
                this.props.changeOrder('aptDate', this.props.orderDir)
              }
              href="#"
            >
              Date
            </button>
            <button
              className={
                'sort-by dropdown-item ' +
                (this.props.orderBy === 'ownerName' ? 'active' : '')
```

```
                  }
                  onClick={e =>
                    this.props.changeOrder('ownerName', this.props.orderDir)
                  }
                  href="#"
                >
                  Owner
                </button>
                <div role="separator" className="dropdown-divider" />
                <button
                  className={
                    'sort-by dropdown-item ' +
                    (this.props.orderDir === 'asc' ? 'active' : '')
                  }
                  onClick={e =>
                    this.props.changeOrder(this.props.orderBy, 'asc')
                  }
                  href="#"
                >
                  Asc
                </button>
                <button
                  className={
                    'sort-by dropdown-item ' +
                    (this.props.orderDir === 'desc' ? 'active' : '')
                  }
                  onClick={e =>
                    this.props.changeOrder(this.props.orderBy, 'desc')
                  }
                  href="#"
                >
                  Desc
                </button>
              </div>
            </div>
          </div>
        </div>
      </div>
    );
  }
}

export default SearchAppointments;
```

Lesson 19: filtering data interactively in parent UI, where state is present

In this segment, we pre-filter the appointments, using a state variable, to prove that we can filter the appointments array based upon one or more variables.

App.js, and new stateful parameters and filtering, bolded

```
import React, { Component } from 'react';
import '../css/App.css';

import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

import { without } from 'lodash';

class App extends Component {
  constructor() {
    super();
    this.state = {
      myAppointments: [],
      formDisplay: false,
      orderBy: 'petName',
      orderDir: 'asc',
      queryText: 'fluffy',
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);
    this.toggleForm = this.toggleForm.bind(this);
    this.addAppointment = this.addAppointment.bind(this);
    this.changeOrder = this.changeOrder.bind(this);
  }

  toggleForm() {
    this.setState({
      formDisplay: !this.state.formDisplay
    });
  }

  changeOrder(order, dir) {
    this.setState({
      orderBy: order,
      orderDir: dir
    });
  }
```

```
addAppointment(apt) {
  let tempApts = this.state.myAppointments;
  apt.aptId = this.state.lastIndex;
  tempApts.unshift(apt);
  this.setState({
    myAppointments: tempApts,
    lastIndex: this.state.lastIndex + 1
  });
}

deleteAppointment(apt) {
  let tempApts = this.state.myAppointments;
  tempApts = without(tempApts, apt);

  this.setState({
    myAppointments: tempApts
  });
}

componentDidMount() {
  fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        item.aptId = this.state.lastIndex;
        this.setState({ lastIndex: this.state.lastIndex + 1 });
        return item;
      });
      this.setState({
        myAppointments: apts
      });
    });
}

render() {
  let order;
  let filteredApts = this.state.myAppointments;
  if (this.state.orderDir === 'asc') {
    order = 1;
  } else {
    order = -1;
  }

  filteredApts = filteredApts
    .sort((a, b) => {
      if (
        a[this.state.orderBy].toLowerCase() <
        b[this.state.orderBy].toLowerCase()
      ) {
```

```
          return -1 * order;
        } else {
          return 1 * order;
        }
      })
      .filter(eachItem => {
        return (
          eachItem['petName']
            .toLowerCase()
            .includes(this.state.queryText.toLowerCase()) ||
          eachItem['ownerName']
            .toLowerCase()
            .includes(this.state.queryText.toLowerCase()) ||
          eachItem['aptNotes']
            .toLowerCase()
            .includes(this.state.queryText.toLowerCase())
        );
      });

    return (
      <main className="page bg-white" id="petratings">
        <div className="container">
          <div className="row">
            <div className="col-md-12 bg-white">
              <div className="container">
                <AddAppointments
                  formDisplay={this.state.formDisplay}
                  toggleForm={this.toggleForm}
                  addAppointment={this.addAppointment}
                />
                <SearchAppointments
                  orderBy={this.state.orderBy}
                  orderDir={this.state.orderDir}
                  changeOrder={this.changeOrder}
                />
                <ListAppointments
                  appointments={filteredApts}
                  deleteAppointment={this.deleteAppointment}
                />
              </div>
            </div>
          </div>
        </div>
      </main>
    );
  }
}

export default App;
```

Lesson 20: capturing UI input to drive search and filtering, on arrays

In the last lesson you successfully drove a filtration event based on hard-coded filter parameters inside of the state constructor of App.js. This enabled a filter to exist when the UI was drawn, independent of UI events.

In this lesson, we take it a bit farther, enabling you to type values into the search bar, and when that value changes (onChange), the text will be used to immediately filter the results. Very little changes inside of SearchAppointments, but several new elements are added to App.js, including a new method to set the queryText inside of the state to update, with the values inside of onChange, in the UI.

SearchAppointments.j updates

```
import React, { Component } from 'react';

class SearchAppointments extends Component {
  render() {
    return (
      <div className="search-appointments row justify-content-center my-4">
        <div className="col-md-6">
          <div className="input-group">
            <input
              id="SearchApts"
              type="text"
              className="form-control"
              aria-label="Search Appointments"
              onChange={e => this.props.searchApts(e.target.value)}
            />
            <div className="input-group-append">
              <button
                type="button"
                className="btn btn-primary dropdown-toggle"
                data-toggle="dropdown"
                aria-haspopup="true"
                aria-expanded="false"
              >
                Sort by: <span className="caret" />
              </button>

              <div className="sort-menu dropdown-menu dropdown-menu-right">
                <button
                  className={
                    'sort-by dropdown-item ' +
                    (this.props.orderBy === 'petName' ? 'active' : '')
                  }
                  onClick={e =>
                    this.props.changeOrder('petName', this.props.orderDir)
                  }
                  href="#"
```

```
              >
                Pet Name
              </button>
              <button
                className={
                  'sort-by dropdown-item ' +
                  (this.props.orderBy === 'aptDate' ? 'active' : '')
                }
                onClick={e =>
                  this.props.changeOrder('aptDate', this.props.orderDir)
                }
                href="#"
              >
                Date
              </button>
              <button
                className={
                  'sort-by dropdown-item ' +
                  (this.props.orderBy === 'ownerName' ? 'active' : '')
                }
                onClick={e =>
                  this.props.changeOrder('ownerName', this.props.orderDir)
                }
                href="#"
              >
                Owner
              </button>
              <div role="separator" className="dropdown-divider" />
              <button
                className={
                  'sort-by dropdown-item ' +
                  (this.props.orderDir === 'asc' ? 'active' : '')
                }
                onClick={e =>
                  this.props.changeOrder(this.props.orderBy, 'asc')
                }
                href="#"
              >
                Asc
              </button>
              <button
                className={
                  'sort-by dropdown-item ' +
                  (this.props.orderDir === 'desc' ? 'active' : '')
                }
                onClick={e =>
                  this.props.changeOrder(this.props.orderBy, 'desc')
                }
                href="#"
              >
                Desc
              </button>
            </div>
          </div>
        </div>
      </div>
    );
  }
}
```

```
export default SearchAppointments;
```

App.js finalization

```
import React, { Component } from 'react';
import '../css/App.css';

import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

import { without } from 'lodash';

class App extends Component {
  constructor() {
    super();
    this.state = {
      myAppointments: [],
      formDisplay: false,
      orderBy: 'petName',
      orderDir: 'asc',
      queryText: '',
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);
    this.toggleForm = this.toggleForm.bind(this);
    this.addAppointment = this.addAppointment.bind(this);
    this.changeOrder = this.changeOrder.bind(this);
    this.searchApts = this.searchApts.bind(this);
  }

  toggleForm() {
    this.setState({
      formDisplay: !this.state.formDisplay
    });
  }

  searchApts(query) {
    this.setState({ queryText: query });
  }

  changeOrder(order, dir) {
```

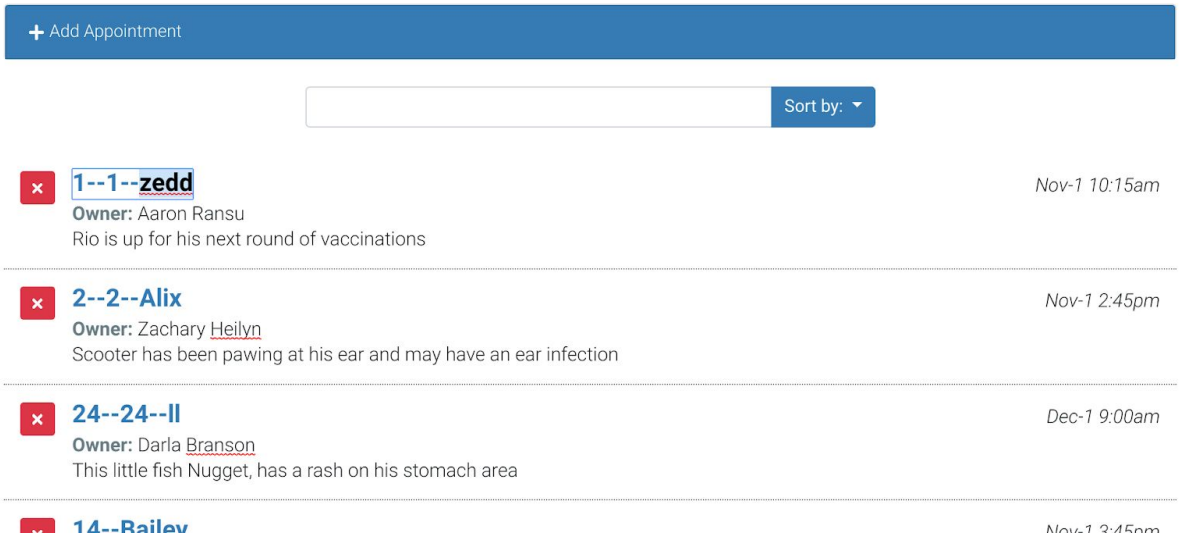
```
  this.setState({
    orderBy: order,
    orderDir: dir
  });
}

addAppointment(apt) {
  let tempApts = this.state.myAppointments;
  apt.aptId = this.state.lastIndex;
  tempApts.unshift(apt);
  this.setState({
    myAppointments: tempApts,
    lastIndex: this.state.lastIndex + 1
  });
}

deleteAppointment(apt) {
  let tempApts = this.state.myAppointments;
  tempApts = without(tempApts, apt);

  this.setState({
    myAppointments: tempApts
  });
}

componentDidMount() {
  fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        item.aptId = this.state.lastIndex;
        this.setState({ lastIndex: this.state.lastIndex + 1 });
        return item;
      });
      this.setState({
        myAppointments: apts
      });
    });
}

render() {
  let order;
  let filteredApts = this.state.myAppointments;
  if (this.state.orderDir === 'asc') {
    order = 1;
  } else {
    order = -1;
  }

  filteredApts = filteredApts
```

```
  .sort((a, b) => {
    if (
      a[this.state.orderBy].toLowerCase() <
      b[this.state.orderBy].toLowerCase()
    ) {
      return -1 * order;
    } else {
      return 1 * order;
    }
  })
  .filter(eachItem => {
    return (
      eachItem['petName']
        .toLowerCase()
        .includes(this.state.queryText.toLowerCase()) ||
      eachItem['ownerName']
        .toLowerCase()
        .includes(this.state.queryText.toLowerCase()) ||
      eachItem['aptNotes']
        .toLowerCase()
        .includes(this.state.queryText.toLowerCase())
    );
  });

return (
  <main className="page bg-white" id="petratings">
    <div className="container">
      <div className="row">
        <div className="col-md-12 bg-white">
          <div className="container">
            <AddAppointments
              formDisplay={this.state.formDisplay}
              toggleForm={this.toggleForm}
              addAppointment={this.addAppointment}
            />
            <SearchAppointments
              orderBy={this.state.orderBy}
              orderDir={this.state.orderDir}
              changeOrder={this.changeOrder}
              searchApts={this.searchApts}
            />
            <ListAppointments
              appointments={filteredApts}
              deleteAppointment={this.deleteAppointment}
            />
          </div>
        </div>
      </div>
    </div>
  </main>
```

```
    );
  }
}

export default App;
```

Lesson 21: modifying array data, using UI inputs in React

The final change will enable the user to select any text element (owner, pet name, description) then change the text on the fly.



After pressing tab, the array is updated, and a sort can be done again, reflecting the new values. This remarkable outcome is a product of several final-stage changes.

ListAppointments.js must be updated, ‘suppress --’

```
import React, { Component } from 'react';
import { FaTimes } from 'react-icons/fa';
import Moment from 'react-moment';

class ListAppointments extends Component {
  render() {
    return (
      <div className="appointment-list item-list mb-3">
        {this.props.appointments.map(item => (
          <div className="pet-item col media py-3" key={item.aptId}>
            <div className="mr-3">
              <button
                className="pet-delete btn btn-sm btn-danger"
                onClick={() => this.props.deleteAppointment(item)}>
                <FaTimes />
              </button>
            </div>
```

```
<div className="pet-info media-body">
  <div className="pet-head d-flex">
    <span
      className="pet-name"
      contentEditable="true"
      suppressContentEditableWarning
      onBlur={e =>
        this.props.updateInfo(
          'petName',
          e.target.innerText,
          item.aptId
        )
      }>
      {item.aptId}--
      {item.petName}
    </span>
    <span className="apt-date ml-auto">
      <Moment
        date={item.aptDate}
        parse="YYYY-MM-dd hh:mm"
        format="MMM-D h:mma"
      />
    </span>
  </div>

  <div className="owner-name">
    <span className="label-item">Owner: </span>
    <span
      contentEditable="true"
      suppressContentEditableWarning
      onBlur={e =>
        this.props.updateInfo(
          'ownerName',
          e.target.innerText,
          item.aptId
        )
      }>
      {item.ownerName}
    </span>
  </div>
  <div
    className="apt-notes"
    contentEditable="true"
    suppressContentEditableWarning
    onBlur={e =>
      this.props.updateInfo(
        'aptNotes',
        e.target.innerText,
        item.aptId
```

```
                  )
                }>
                {item.aptNotes}
              </div>
            </div>
          </div>
        ))}
      </div>
    );
  }
}

export default ListAppointments;
```

App.js, binding new updateInfo feature

```
import React, { Component } from 'react';
import '../css/App.css';

import AddAppointments from './AddAppointments';
import SearchAppointments from './SearchAppointments';
import ListAppointments from './ListAppointments';

import { findIndex, without } from 'lodash'; //critical for finishing

class App extends Component {
  constructor() {
    super();
    this.state = {
      myAppointments: [],
      formDisplay: false,
      orderBy: 'petName',
      orderDir: 'asc',
      queryText: '',
      lastIndex: 0
    };
    this.deleteAppointment = this.deleteAppointment.bind(this);
    this.toggleForm = this.toggleForm.bind(this);
    this.addAppointment = this.addAppointment.bind(this);
    this.changeOrder = this.changeOrder.bind(this);
    this.searchApts = this.searchApts.bind(this);
    this.updateInfo = this.updateInfo.bind(this);
  }

  toggleForm() {
    this.setState({
      formDisplay: !this.state.formDisplay
    });
```

```
}

searchApts(query) {
  this.setState({ queryText: query });
}

changeOrder(order, dir) {
  this.setState({
    orderBy: order,
    orderDir: dir
  });
}

updateInfo(name, value, id) {
  let tempApts = this.state.myAppointments;
  let aptIndex = findIndex(this.state.myAppointments, {
    aptId: id
  });
  tempApts[aptIndex][name] = value;
  this.setState({
    myAppointments: tempApts
  });
}

addAppointment(apt) {
  let tempApts = this.state.myAppointments;
  apt.aptId = this.state.lastIndex;
  tempApts.unshift(apt);
  this.setState({
    myAppointments: tempApts,
    lastIndex: this.state.lastIndex + 1
  });
}

deleteAppointment(apt) {
  let tempApts = this.state.myAppointments;
  tempApts = without(tempApts, apt);

  this.setState({
    myAppointments: tempApts
  });
}

componentDidMount() {
  fetch('./data.json')
    .then(response => response.json())
    .then(result => {
      const apts = result.map(item => {
        item.aptId = this.state.lastIndex;
        this.setState({ lastIndex: this.state.lastIndex + 1 });
```

```
        return item;
      });
      this.setState({
        myAppointments: apts
      });
    });
}

render() {
  let order;
  let filteredApts = this.state.myAppointments;
  if (this.state.orderDir === 'asc') {
    order = 1;
  } else {
    order = -1;
  }

  filteredApts = filteredApts
    .sort((a, b) => {
      if (
        a[this.state.orderBy].toLowerCase() <
        b[this.state.orderBy].toLowerCase()
      ) {
        return -1 * order;
      } else {
        return 1 * order;
      }
    })
    .filter(eachItem => {
      return (
        eachItem['petName']
          .toLowerCase()
          .includes(this.state.queryText.toLowerCase()) ||
        eachItem['ownerName']
          .toLowerCase()
          .includes(this.state.queryText.toLowerCase()) ||
        eachItem['aptNotes']
          .toLowerCase()
          .includes(this.state.queryText.toLowerCase())
      );
    });

  return (
    <main className="page bg-white" id="petratings">
      <div className="container">
        <div className="row">
          <div className="col-md-12 bg-white">
            <div className="container">
              <AddAppointments
                formDisplay={this.state.formDisplay}
```

```
                toggleForm={this.toggleForm}
                addAppointment={this.addAppointment}
              />
              <SearchAppointments
                orderBy={this.state.orderBy}
                orderDir={this.state.orderDir}
                changeOrder={this.changeOrder}
                searchApts={this.searchApts}
              />
              <ListAppointments
                appointments={filteredApts}
                deleteAppointment={this.deleteAppointment}
                updateInfo={this.updateInfo}
              />
            </div>
          </div>
        </div>
      </div>
    </main>
  );
 }
}

export default App;
```