Password Reset

Let's learn how to reset user passwords using Firebase!

WE'LL COVER THE FOLLOWING

Let’s take a step back from the **higher-order components**, **React Context API**, and **session handling**. In this section, we will implement two additional features available in the Firebase authentication API:

- Retrieve password (password forget)
- Change password.

We’ll begin with the first feature, password retrieval.

Forgot Password

In this lesson, we’ll implement the **Forgot Password** feature. Since we’ve already implemented the interface in our Firebase class, we can use it in our application’s components.

The following file adds most of the password reset logic in a form again. We’ve already used a couple of those forms before, so it shouldn’t be a problem.

Implementation

Add this in the `src/components/PasswordForget/index.js` file:

```
import React, { Component } from 'react';
import { Link } from 'react-router-dom';

import { withFirebase } from '../Firebase';
```

```
import * as ROUTES from '../../constants/routes';

const PasswordForgetPage = () => (

  <div>
    <h1>PasswordForget</h1>
    <PasswordForgetForm />
  </div>
);

const INITIAL_STATE = {
  email: '',
  error: null,
};

class PasswordForgetFormBase extends Component {
  constructor(props) {
    super(props);

    this.state = { ...INITIAL_STATE };
  }

  onSubmit = event => {
    const { email } = this.state;

    this.props.firebase
      .doPasswordReset(email)
      .then(() => {
        this.setState({ ...INITIAL_STATE });
      })
      .catch(error => {
        this.setState({ error });
      });

    event.preventDefault();
  };

  onChange = event => {
    this.setState({ [event.target.name]: event.target.value });
  };

  render() {
    const { email, error } = this.state;

    const isInvalid = email === '';

    return (
      <form onSubmit={this.onSubmit}>
        <input
          name="email"
          value={this.state.email}
          onChange={this.onChange}
          type="text"
          placeholder="Email Address"
        />
        <button disabled={isInvalid} type="submit">
          Reset My Password
        </button>

        {error && <p>{error.message}</p>}
      </form>
    );
  }
```

```
}

const PasswordForgetLink = () => (
  <p>
    <Link to={ROUTES.PASSWORD_FORGET}>Forgot Password?</Link>
  </p>
);

export default PasswordForgetPage;

const PasswordForgetForm = withFirebase(PasswordForgetFormBase);

export { PasswordForgetForm, PasswordForgetLink };
```

PasswordForget/index.js

Explanation

The code is verbose, but it's no different from the sign-up and sign-in forms we implemented earlier. The `PasswordForgetPage` uses a form to submit the information (email address) needed by the Firebase authentication API to reset the password.

The class method, `onSubmit`, ensures that the information is sent to the API. It also resets the form's input field on a successful request and shows an error in case of an erroneous request.

The form is also validated before submission. The file implements a `PasswordForgetLink` as a component which isn't used directly in the form component. It is similar to the `SignUpLink` component that we used in the `SignInPage` component.

This link is the same and still usable. If a user forgets the password after sign up, the `PasswordForgetPage` uses the link in the `src/components/SignIn/index.js` file:

```
import React, { Component } from 'react';
import { withRouter } from 'react-router-dom';
import { compose } from 'recompose';

import { SignUpLink } from '../SignUp';
import { PasswordForgetLink } from '../PasswordForget';
import { withFirebase } from '../Firebase';
import * as ROUTES from '../../constants/routes';

const SignInPage = () => (
  <div>
    <h1>SignIn</h1>
    <SignInForm />
```

```
    <PasswordForgetLink />
    <SignUpLink />
  </div>
);


...
```

SignIn/index.js

The `PasswordForgetPassword` page is already matched in the `App` component. Hence, we can drop the `PasswordForgetLink` component in the sign-in page, ensuring that the mapping between the route and the component is complete.

Let's start the application and reset our password. It doesn't matter if we are authenticated or not. Once we send the request, we should get an email from Firebase requesting us to update our password.

In the next lesson, we will implement the **password change** functionality.