CAD_Phase4

SIGN-IN FORM: (For signing in the app for the first time)

```
import "./sign-in-form.styles.scss";
import Button from "../button/button.component";
import FormInput from "../form-input/form-input.component";
import {
  signInWithGooglePopup,
  signInAuthUserFromEmailAndPassword,
  passwordReset,
} from "../../utils/firebase/firebase.utils";
import { useState } from "react";
import Alert from "../alert-menu/alert.component";

const defaultFormFields = {
  email: "",
  password: "",
};
const statusMessages = {
  1: "Please enter an email address for password reset.",
  2: "Invalid email address.",
  3: "User not found.",
  4: "Error resetting password.",
  5: "Wrong Password",
  6: "Password reset email sent. Please check your inbox.",
};

const SignInForm = () => {
  const [formFields, setFormFields] = useState(defaultFormFields);
  const { email, password } = formFields;
  const [error, setError] = useState(null);
  const [showAlert, setShowAlert] = useState(false);

  const handleHideAlert = () => {
    setShowAlert(false);
  };

  const handleForgotPassword = async () => {
    try {
      await passwordReset(email);
```

```
    setError(statusMessages[6]);
    setShowAlert(true);
  } catch (error) {
    switch (error.code) {
      case "auth/missing-email":
        setError(statusMessages[1]);
        setShowAlert(true);
        break;
      case "auth/invalid-email":
        setError(statusMessages[2]);
        setShowAlert(true);
        break;
      case "auth/user-not-found":
        setError(statusMessages[3]);
        setShowAlert(true);
        break;
      default:
        setError(statusMessages[4]);
        setShowAlert(true);
    }
    console.log(error.message);
  }
};

const onChangeHandler = (event) => {
  const { name, value } = event.target;
  setFormFields({ ...formFields, [name]: value });
};

const signInWithGoogle = async (event) => {
  try {
    await signInWithGooglePopup();
  } catch (error) {
    console.log("Error signing in with google", error.message);
  }
};

const resetFormFields = () => {
  setFormFields(defaultFormFields);
};
```

```
const handleSubmit = async (event) => {
  event.preventDefault();
  try {
    await signInAuthUserFromEmailAndPassword(email, password);
    resetFormFields();
  } catch (error) {
    if (error.code === "auth/wrong-password") {
      setError(statusMessages[5]);
      setShowAlert(true);
    }
    if (error.code === "auth/user-not-found") {
      setError(statusMessages[3]);
      setShowAlert(true);
    }
    console.log("Error signing in", error.message);
  }
};
return (
  <div className="sign-up-container">
    <h2>Already have an account?</h2>
    <span>Sign in with your email and password</span>
    <form onSubmit={handleSubmit}>
      <FormInput
        label="Email"
        type="email"
        name="email"
        required
        onChange={onChangeHandler}
        value={email}
      />
      <FormInput
        label="Password"
        type="password"
        name="password"
        required
        onChange={onChangeHandler}
        value={password}
      />
```

```
        <div className="buttons-container">
          <Button type="submit">SIGN IN</Button>
          <Button type="button" onClick={signInWithGoogle} buttonType="google">
            SIGN IN WITH GOOGLE
          </Button>
        </div>
      </form>
      <a onClick={handleForgotPassword} className="forgot-pass" href="#">
        Forgot Password?
      </a>
      {showAlert &&
        (error === statusMessages[6] ? (
          <Alert
            message={error}
            onClose={handleHideAlert}
            alertType="message"
          />
        ) : (
          <Alert message={error} onClose={handleHideAlert} alertType="error" />
        ))}
    </div>
  );
};export default SignInForm;
```

SIGN-UP-FORM: **(For user , already with an account)**

```
import { useState } from "react";
import {
  createUserDocumentFromAuth,
  createAuthUserFromEmailAndPassword,
} from "../../utils/firebase/firebase.utils";
import FormInput from "../form-input/form-input.component";
import "./sign-up-form.styles.scss";
import Button from "../button/button.component";
import Alert from "../alert-menu/alert.component";

const statusMessages = {
  1: "Email already in use",
  2: "Passwords don't match",
```

```
  3: "Password should be at least 6 characters",
};
const defaultFormFields = {
  displayName: "",
  email: "",
  password: "",
  confirmPassword: "",
};

const SignUpForm = () => {
  const [formFields, setFormFields] = useState(defaultFormFields);
  const { displayName, email, password, confirmPassword } = formFields;
  const [error, setError] = useState(null);
  const [showAlert, setShowAlert] = useState(false);

  const handleHideAlert = () => {
    setShowAlert(false);
  };

  const resetFormFields = () => {
    setFormFields(defaultFormFields);
  };

  const handleSubmit = async (event) => {
    event.preventDefault();
    if (password !== confirmPassword) {
      setError(statusMessages[2]);
      setShowAlert(true);
      return;
    }
    try {
      const { user } = await createAuthUserFromEmailAndPassword(
        email,
        password
      );
      await createUserDocumentFromAuth(user, { displayName });
      resetFormFields();
    } catch (error) {
      if (error.code === "auth/email-already-in-use") {
        setError(statusMessages[1]);
```

```
      setShowAlert(true);
    } else if (error.code === "auth/weak-password") {
      setError(statusMessages[3]);
      setShowAlert(true);
    }
    console.log("User creation encountered an error", error);
  }
};

const handleChange = (event) => {
  const { name, value } = event.target;
  setFormFields({ ...formFields, [name]: value });
};
return (
  <div className="sign-up-container">
    <h2>Don't have an account?</h2>
    <span>Sign up with your email and password</span>
    <form onSubmit={handleSubmit}>
      <FormInput
        label="Display Name"
        type="text"
        name="displayName"
        required
        onChange={handleChange}
        value={displayName}
      />

      <FormInput
        label="Email"
        type="email"
        name="email"
        required
        onChange={handleChange}
        value={email}
      />

      <FormInput
        label="Password"
        type="password"
        name="password"
```

```
          required
          onChange={handleChange}
          value={password}
        />

        <FormInput
          label="Confirm Password"
          type="password"
          name="confirmPassword"
          required
          onChange={handleChange}
          value={confirmPassword}
        />
        <Button type="submit">Sign Up</Button>
        {showAlert && (
          <Alert message={error} onClose={handleHideAlert} alertType="error" />
        )}
      </form>
    </div>
  );
};

export default SignUpForm;
```

CHECKOUT ITEM: **(For checking out the final items in the cart before payment)**

```
import {
  CheckoutItemContainer,
  CheckoutItemImage,
  CheckoutItemName,
  CheckoutItemPrice,
  CheckoutItemQuantity,
  ItemRemoveButton,
} from "./checkout-item.styles";
import { useContext } from "react";
import { CartContext } from "../../contexts/cart.context";
```

```
const CheckOutItem = ({ cartItem }) => {
  const { addItemToCart, deleteItemFromCart, removeItemFromCart } =
    useContext(CartContext);

  const { name, imageUrl, price, quantity } = cartItem;

  const addItemHandler = () => {
    addItemToCart(cartItem);
  };
  const removeItemHandler = () => {
    removeItemFromCart(cartItem);
  };
  const deleteItemHandler = () => {
    deleteItemFromCart(cartItem);
  };

  return (
    <CheckoutItemContainer>
      <CheckoutItemImage>
        <img src={imageUrl} alt={`${name}`} />
      </CheckoutItemImage>
      <CheckoutItemName className="name">{name}</CheckoutItemName>
      <CheckoutItemQuantity className="quantity">
        <div className="arrow" onClick={removeItemHandler}>
          &lt;
        </div>
        <span className="value">{quantity}</span>
        <div className="arrow" onClick={addItemHandler}>
          &gt;
        </div>
      </CheckoutItemQuantity>
      <CheckoutItemPrice className="price">₹{price}</CheckoutItemPrice>
      <ItemRemoveButton className="remove-button" onClick={deleteItemHandler}>
        &#10005;
      </ItemRemoveButton>
    </CheckoutItemContainer>
  );
};

export default CheckOutItem;
```

ALERT COMPONENT:(Showing the updated notifications)

```
import "./alert.styles.scss";
import Button from "../button/button.component";
import { Fragment } from "react";
const Alert = ({ message, onClose, alertType }) => {
  const handleOk = () => {
    onClose();
  };


  return (
    <div className="alert-overlay">
      <div className="alert-container">
        {alertType === "error" ? (
          <Fragment>
            <img
              src="https://i.ibb.co/HgzrzKf/3588294.png"
              alt="alert"
              className="alert-icon"
            />
            <div className="alert-title">Oops....!</div>
          </Fragment>
        ) : (
          <Fragment>
            <img

              alt="alert"
              className="alert-icon"
            />
          </Fragment>
        )}
        <div className="alert-message">{message}</div>
        <Button buttonType="alertBox" onClick={handleOk}>
          OK
```

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


export default Alert;
```

AUTHENTICATION:**(Get the user's login credentials for verification)**

```
import SignUpForm from "../../components/sign-up-form/sign-up-form.component";
import SignInForm from "../../components/sign-in-form/sign-in-form.component";
import { AuthenticationContainer } from "./authentication.styles";

const Authentication = () => {
 return (
  <AuthenticationContainer>
   <SignInForm />
   <SignUpForm />
  </AuthenticationContainer>
 );
};

export default Authentication;
```

CART DROPDOWN:**(Checking out for payment)**

```
import "./cart-dropdown.styles.scss";
import { useNavigate } from "react-router-dom";
import { useContext } from "react";
import { CartContext } from "../../contexts/cart.context";
import Button from "../button/button.component";
import CartItem from "../cart-item/cart-item.component";

const CartDropdown = () => {
 const { isCartOpen, setIsCartOpen, cartItems } = useContext(CartContext);
 const navigate = useNavigate();
```

```
  const handleCheckout = () => {
    setIsCartOpen(!isCartOpen);
    navigate("/checkout");
  };

  return (
    <div className="cart-dropdown-container">
      {cartItems.length === 0 ? (
        <div className="empty-message">Nothing in the cart</div>
      ) : (
        <div className="cart-items">
          {cartItems.map((item) => (
            <CartItem key={item.id} cartItem={item} />
          ))}
        </div>
      )}
      <Button onClick={handleCheckout}>GO TO CHECKOUT</Button>
    </div>
  );
};

export default CartDropdown;
```