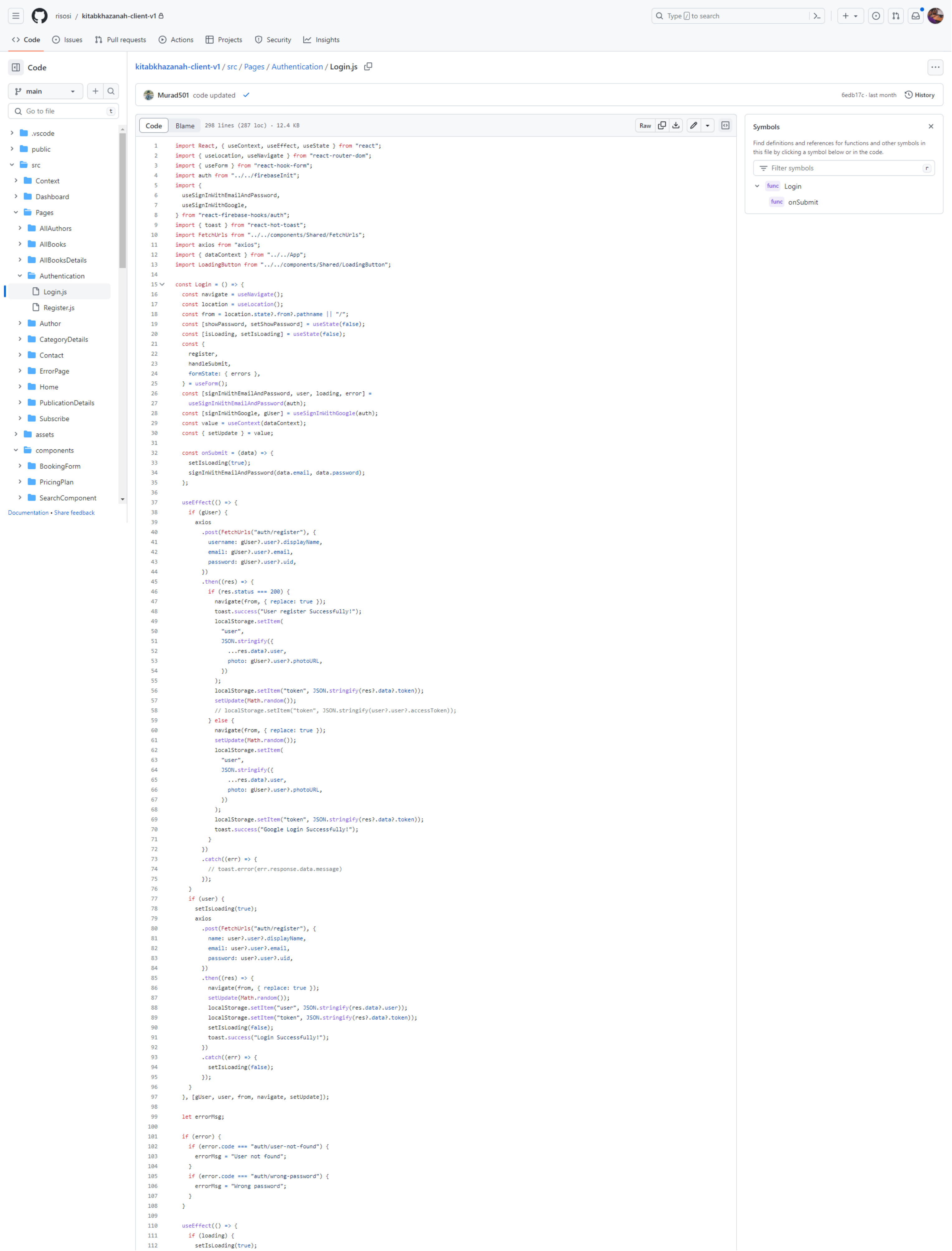
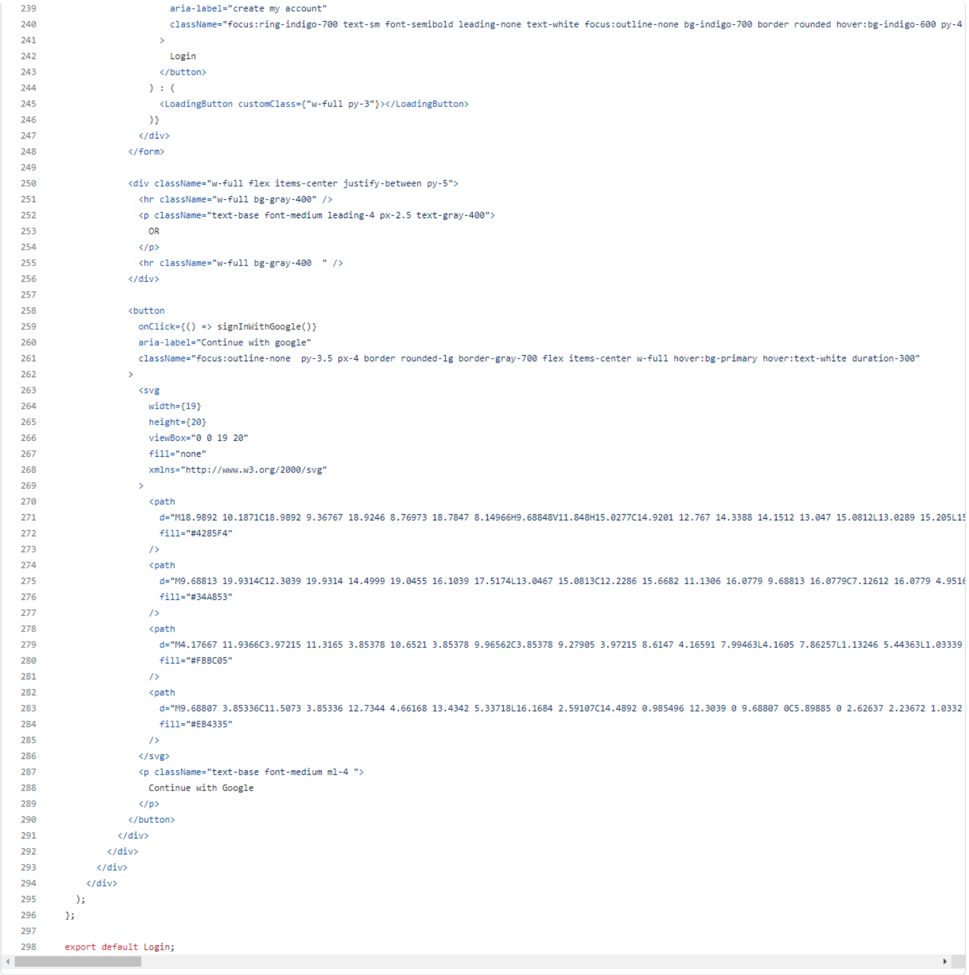


```
import React, { useContext, useEffect, useState } from "react";
import { useLocation, useNavigate } from "react-router-dom";
import { useForm } from "react-hook-form";
import auth from "../../firebaseInit";
import {
  useSignInWithEmailAndPassword,
  useSignInWithGoogle,
} from "react-firebase-hooks/auth";
import { toast } from "react-hot-toast";
import FetchUrls from "../../components/Shared/FetchUrls";
import axios from "axios";
import { dataContext } from "../../App";
import LoadingButton from "../../components/Shared/LoadingButton";

const Login = () => {
  const navigate = useNavigate();
  const location = useLocation();
  const from = location.state?.from?.pathname || "/";
  const [showPassword, setShowPassword] = useState(false);
  const [isLoading, setIsLoading] = useState(false);
  const {
    register,
    handleSubmit,
    formState: { errors },
  } = useForm();
  const [signInWithEmailAndPassword, user, loading, error] =
    useSignInWithEmailAndPassword(auth);
  const [signInWithGoogle, gUser] = useSignInWithGoogle(auth);
  const value = useContext(dataContext);
  const { setUpdate } = value;

  const onSubmit = (data) => {
    setIsLoading(true);
    signInWithEmailAndPassword(data.email, data.password);
  };

  useEffect(() => {
    if (gUser) {
      axios
        .post(FetchUrls("auth/register"), {
          username: gUser?.user?.displayName,
          email: gUser?.user?.email,
          password: gUser?.user?.uid,
        })
        .then((res) => {
          if (res.status === 200) {
            navigate(from, { replace: true });
            toast.success("User register Successfully!");
            localStorage.setItem(
              "user",
              JSON.stringify({
                ...res.data?.user,
                photo: gUser?.user?.photoURL,
              })
            );
            localStorage.setItem("token", JSON.stringify(res?.data?.token));
            setUpdate(Math.random());
            // localStorage.setItem("token", JSON.stringify(user?.user?.accessToken));
          } else {
            navigate(from, { replace: true });
            setUpdate(Math.random());
            localStorage.setItem(
              "user",
              JSON.stringify({
                ...res.data?.user,
                photo: gUser?.user?.photoURL,
              })
            );
            localStorage.setItem("token", JSON.stringify(res?.data?.token));
            toast.success("Google Login Successfully!");
          }
        })
        .catch((err) => {
          // toast.error(err.response.data.message)
        });
    }
    if (user) {
      setIsLoading(true);
      axios
        .post(FetchUrls("auth/register"), {
          name: user?.user?.displayName,
          email: user?.user?.email,
          password: user?.user?.uid,
        })
        .then((res) => {
          navigate(from, { replace: true });
          setUpdate(Math.random());
          localStorage.setItem("user", JSON.stringify(res.data?.user));
          localStorage.setItem("token", JSON.stringify(res?.data?.token));
          setIsLoading(false);
          toast.success("Login Successfully!");
        })
        .catch((err) => {
          setIsLoading(false);
        });
    }
  }, [gUser, user, from, navigate, setUpdate]);

  let errorMsg;

  if (error) {
    if (error.code === "auth/user-not-found") {
      errorMsg = "User not found";
    }
    if (error.code === "auth/wrong-password") {
      errorMsg = "Wrong password";
    }
  }

  useEffect(() => {
    if (loading) {
      setIsLoading(true);
    } else {
      setIsLoading(false);
    }
  }, [loading]);

  return (
    <div>
      <div className=" bg-gradient-to-tl from-green-400 to-indigo-900 w-full pt-10 px-4 h-[93vh]">
        <div className="flex flex-col items-center justify-center">
          <div className="bg-white shadow rounded lg:w-1/3  md:w-1/2 w-full p-10 mt-5 ">
            <p
              tabIndex={0}
              aria-label="Login to your account"
              className="text-2xl font-extrabold leading-6 text-gray-800"
            >
              Login to your account
            </p>
            <p
              onClick={() => navigate("/register")}
              className="text-sm mt-4 font-medium leading-none text-gray-500 "
            >
              Don't have an account?{" "}
              <span
                tabIndex={0}
                role="link"
                aria-label="Sign up here"
                className="text-sm font-medium leading-none underline text-gray-800 cursor-pointer hover:text-primary"
              >
                {" "}
                Register here
              </span>
            </p>

            <form onSubmit={handleSubmit(onSubmit)}>
              <div className="mt-4">
                <label className="text-sm font-medium leading-none text-gray-800">
                  Email
                </label>
                <input
                  aria-label="enter email address"
                  type="email"
                  className="bg-gray-200 border rounded focus:outline-none text-xs font-medium leading-none text-gray-800 py-3 w-full pl-3 mt-2"
                  {...register("email", {
                    required: {
                      value: true,
                      message: "Email is Required",
                    },
                    pattern: {
                      value: /[a-z0-9]+@[a-z]+\.[a-z]{2,3}/,
                      message: "Provide a valid Email",
                    },
                  })}
                />
                <label className="label">
                  {errors.email?.type === "required" && (
                    <span className=" label-text-alt text-red-500">
                      {errors.email.message}
                    </span>
                  )}
                  {errors.email?.type === "pattern" && (
                    <span className=" label-text-alt text-red-500">
                      {errors.email.message}
                    </span>
                  )}
                </label>
              </div>
              <div className="mt-6  w-full">
                <label className="text-sm font-medium leading-none text-gray-800">
                  Password
                </label>
                <div className="relative flex items-center justify-center">
                  <input
                    aria-label="enter Password"
                    type={showPassword ? "text" : "password"}
                    className="bg-gray-200 border rounded focus:outline-none text-xs font-medium leading-none text-gray-800 py-3 w-full pl-3 mt-2"
                    {...register("password", {
                      required: {
                        value: true,
                        message: "Password is Required",
                      },
                      minLength: {
                        value: 6,
                        message: "Must be 6 characters or longer",
                      },
                    })}
                  />
                  <div
                    onClick={() => setShowPassword(!showPassword)}
                    className="absolute right-0 mt-2 mr-3 cursor-pointer"
                  >
                    <svg
                      width={16}
                      height={16}
                      viewBox="0 0 16 16"
                      fill="none"
                      xmlns="http://www.w3.org/2000/svg"
                    >
                      <path
                        d="M7.99978 2C11.5944 2 14.5851 4.58667 15.2124 8C14.5858 11.4133 11.5944 14 7.99978 14C4.40511 14 1.41444 11.4133 0.787109 8C1.41378 4.58667 4.4051
                        fill="#71717A"
                      />
                    </svg>
                  </div>
                </div>
                <label className="label">
                  {errors.password?.type === "required" && (
                    <span className=" label-text-alt text-red-500">
                      {errors.password.message}
                    </span>
                  )}
                  {errors.password?.type === "minLength" && (
                    <span className=" label-text-alt text-red-500">
                      {errors.password.message}
                    </span>
                  )}
                  {error && (
                    <span className=" label-text-alt text-red-500">
                      {errorMsg}
                    </span>
                  )}
                </label>
              </div>
              <div className="mt-8">
                {!isLoading ? (
                  <button
                    type="submit"
```



```
                    aria-label="create my account"
                    className="focus:ring-indigo-700 text-sm font-semibold leading-none text-white focus:outline-none bg-indigo-700 border rounded hover:bg-indigo-600 py-4
                  >
                    Login
                  </button>
                ) : (
                  <LoadingButton customClass={"w-full py-3"}></LoadingButton>
                )}
              </div>
            </form>

            <div className="w-full flex items-center justify-between py-5">
              <hr className="w-full bg-gray-400" />
              <p className="text-base font-medium leading-4 px-2.5 text-gray-400">
                OR
              </p>
              <hr className="w-full bg-gray-400  " />
            </div>

            <button
              onClick={() => signInWithGoogle()}
              aria-label="Continue with google"
              className="focus:outline-none  py-3.5 px-4 border rounded-lg border-gray-700 flex items-center w-full hover:bg-primary hover:text-white duration-300"
            >
              <svg
                width={19}
                height={20}
                viewBox="0 0 19 20"
                fill="none"
                xmlns="http://www.w3.org/2000/svg"
              >
                <path
                  d="M18.9892 10.1871C18.9892 9.36767 18.9246 8.76973 18.7847 8.14966H9.68848V11.848H15.0277C14.9201 12.767 14.3388 14.1512 13.047 15.0812L13.0289 15.205L15
                  fill="#4285F4"
                />
                <path
                  d="M9.68813 19.9314C12.3039 19.9314 14.4999 19.0455 16.1039 17.5174L13.0467 15.0813C12.2286 15.6682 11.1306 16.0779 9.68813 16.0779C7.12612 16.0779 4.9516
                  fill="#34A853"
                />
                <path
                  d="M4.17667 11.9366C3.97215 11.3165 3.85378 10.6521 3.85378 9.96562C3.85378 9.27905 3.97215 8.6147 4.16591 7.99463L4.1605 7.86257L1.13246 5.44363L1.03339
                  fill="#FBBC05"
                />
                <path
                  d="M9.68807 3.85336C11.5073 3.85336 12.7344 4.66168 13.4342 5.33718L16.1684 2.59107C14.4892 0.985496 12.3039 0 9.68807 0C5.89885 0 2.62637 2.23672 1.0332
                  fill="#EB4335"
                />
              </svg>
              <p className="text-base font-medium ml-4 ">
                Continue with Google
              </p>
            </button>
          </div>
        </div>
      </div>
    </div>
  );
};

export default Login;
```