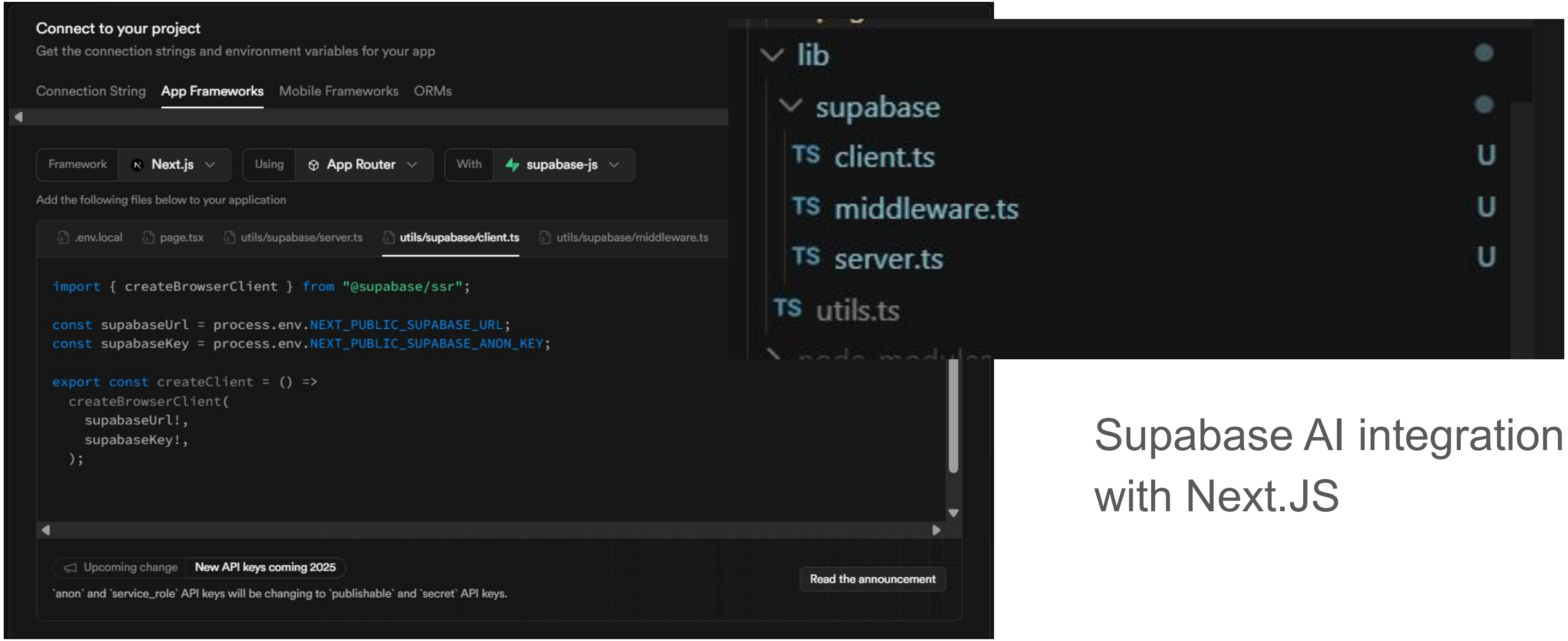
ALX_AIDEV

#doinghardthings

Building Features with AI - Authentication Feature



Connect to your project

Get the connection strings and environment variables for your app

Framework: Next.js · Using: App Router · With: supabase-js

Add the following files below to your application

```
import { createBrowserClient } from "@supabase/ssr";

const supabaseUrl = process.env.NEXT_PUBLIC_SUPABASE_URL;
const supabaseKey = process.env.NEXT_PUBLIC_SUPABASE_ANON_KEY;

export const createClient = () =>
  createBrowserClient(
    supabaseUrl!,
    supabaseKey!,
  );
```

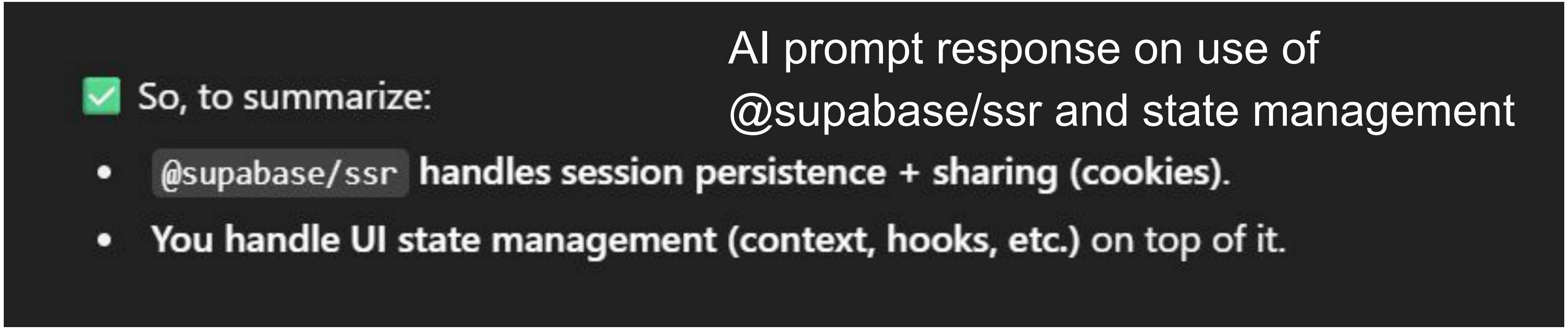
Upcoming change · New API keys coming 2025

`anon` and `service_role` API keys will be changing to `publishable` and `secret` API keys.

```
lib
  supabase
    client.ts       U
    middleware.ts   U
    server.ts       U
  utils.ts
```

Supabase AI integration with Next.JS

Building Features with AI - Authentication Feature

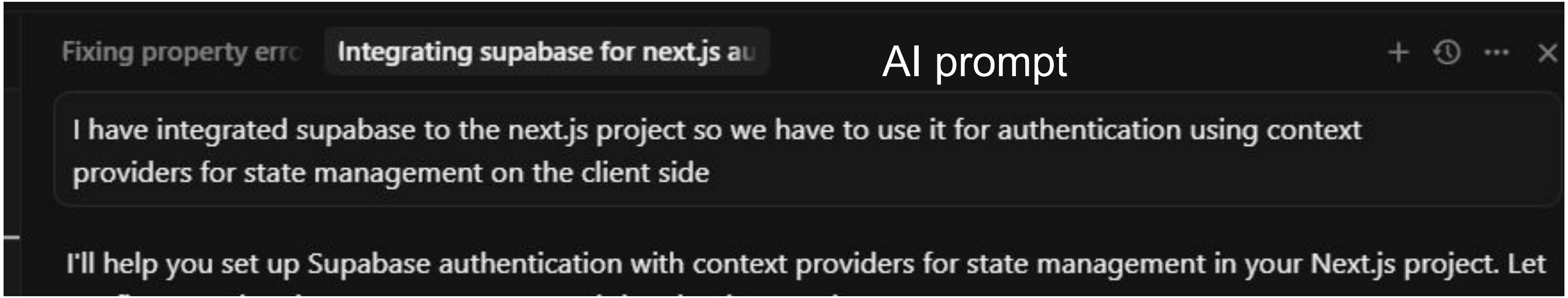


AI prompt response on use of @supabase/ssr and state management

✅ So, to summarize:

- `@supabase/ssr` **handles session persistence + sharing (cookies)**.
- **You handle UI state management (context, hooks, etc.)** on top of it.

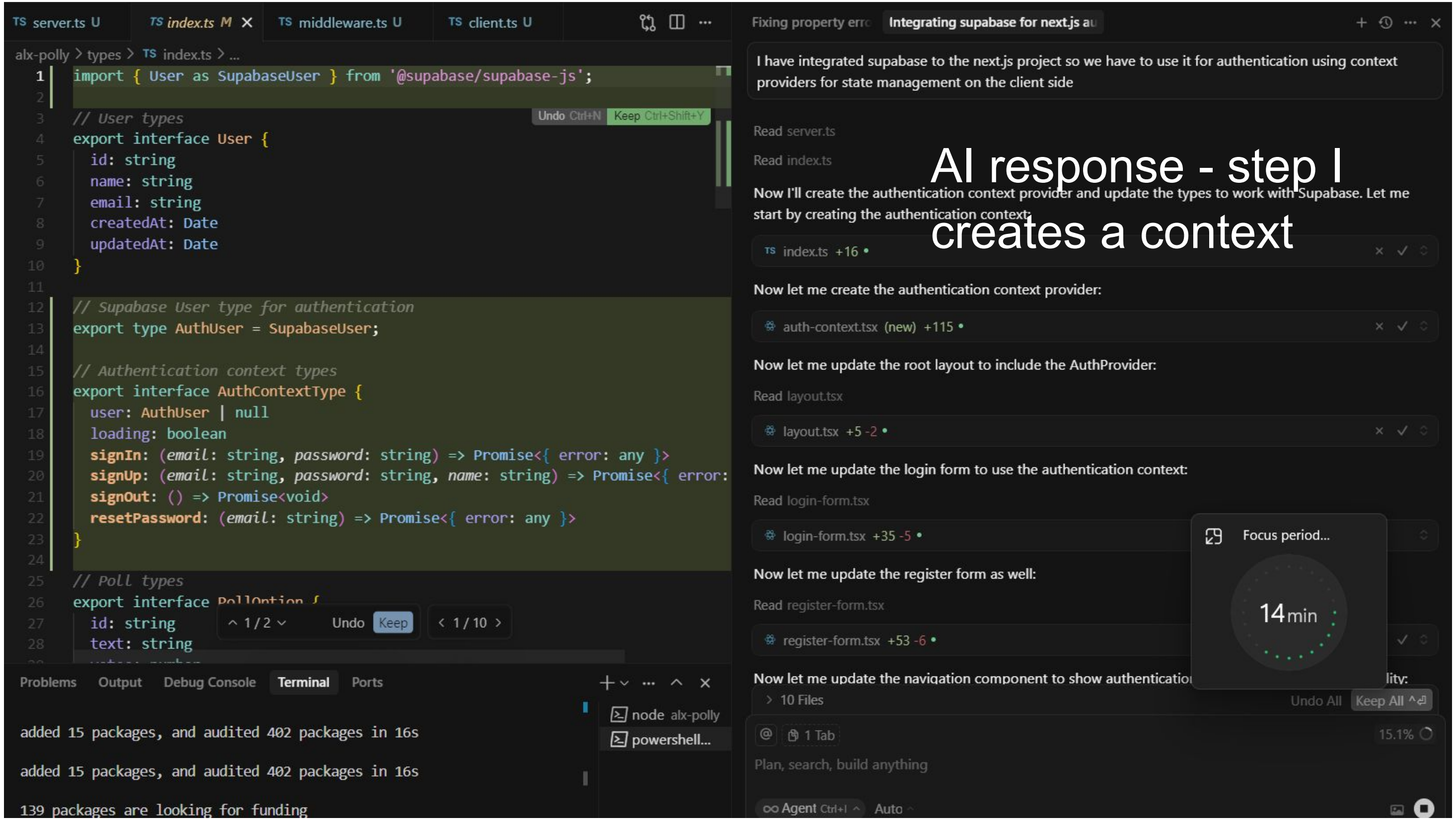
Building Features with AI - Authentication Feature



I have integrated supabase to the next.js project so we have to use it for authentication using context providers for state management on the client side

I'll help you set up Supabase authentication with context providers for state management in your Next.js project. Let

Building Features with AI - Authentication Feature



```typescript
import { User as SupabaseUser } from '@supabase/supabase-js';

// User types
export interface User {
  id: string
  name: string
  email: string
  createdAt: Date
  updatedAt: Date
}

// Supabase User type for authentication
export type AuthUser = SupabaseUser;

// Authentication context types
export interface AuthContextType {
  user: AuthUser | null
  loading: boolean
  signIn: (email: string, password: string) => Promise<{ error: any }>
  signUp: (email: string, password: string, name: string) => Promise<{ error:
  signOut: () => Promise<void>
  resetPassword: (email: string) => Promise<{ error: any }>
}

// Poll types
export interface PollOption {
  id: string
  text: string
```

```
added 15 packages, and audited 402 packages in 16s

added 15 packages, and audited 402 packages in 16s

139 packages are looking for funding
```

I have integrated supabase to the next.js project so we have to use it for authentication using context providers for state management on the client side

Read server.ts

Read index.ts

Now I'll create the authentication context provider and update the types to work with Supabase. Let me start by creating the authentication context:

TS index.ts +16 •

Now let me create the authentication context provider:

auth-context.tsx (new) +115 •

Now let me update the root layout to include the AuthProvider:

Read layout.tsx

layout.tsx +5 -2 •

Now let me update the login form to use the authentication context:

Read login-form.tsx

login-form.tsx +35 -5 •

Now let me update the register form as well:

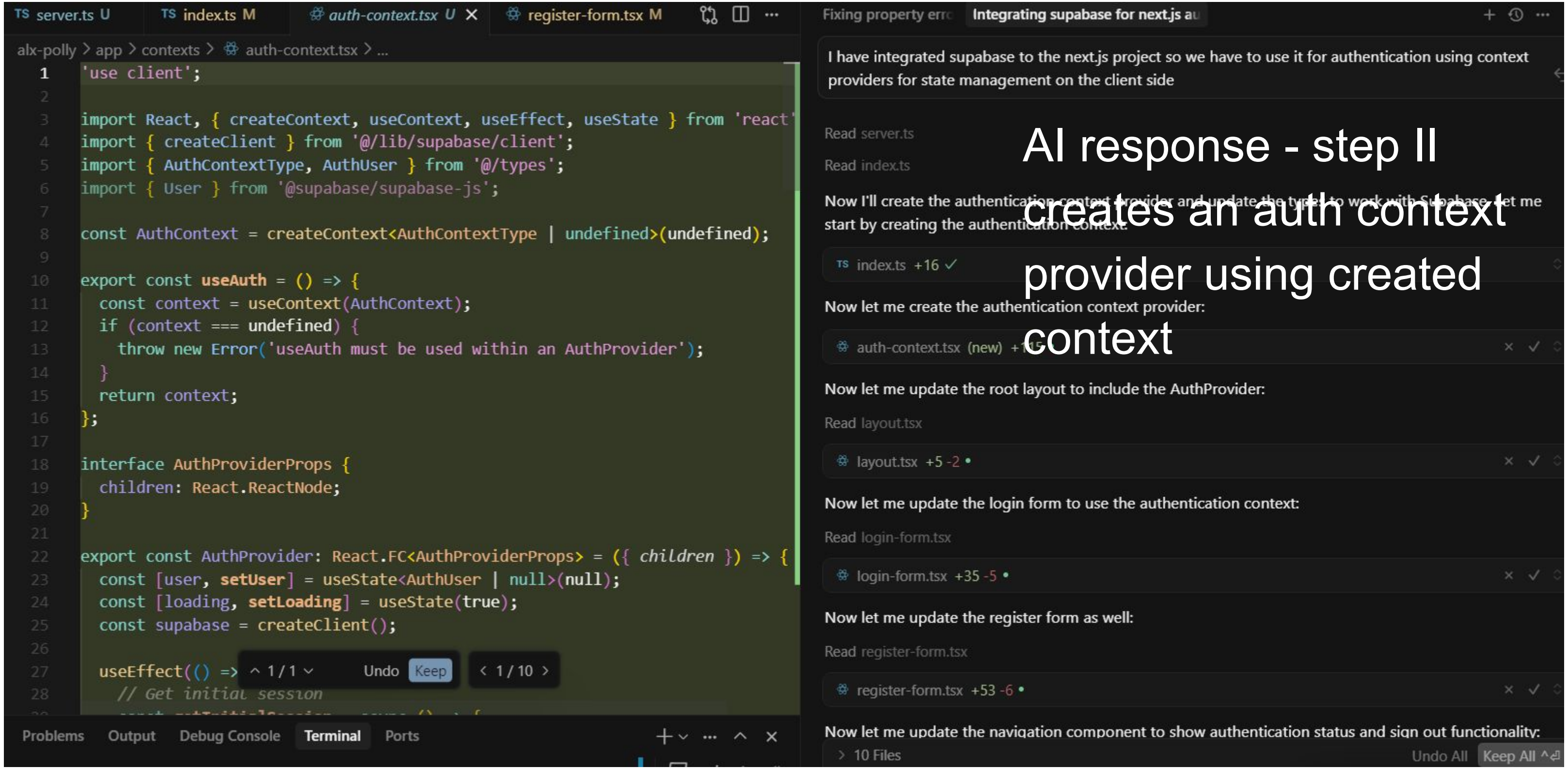
Read register-form.tsx

register-form.tsx +53 -6 •

Now let me update the navigation component to show authenticatio lity:

AI response - step I creates a context

Building Features with AI - Authentication Feature



```tsx
'use client';

import React, { createContext, useContext, useEffect, useState } from 'react'
import { createClient } from '@/lib/supabase/client';
import { AuthContextType, AuthUser } from '@/types';
import { User } from '@supabase/supabase-js';

const AuthContext = createContext<AuthContextType | undefined>(undefined);

export const useAuth = () => {
  const context = useContext(AuthContext);
  if (context === undefined) {
    throw new Error('useAuth must be used within an AuthProvider');
  }
  return context;
};

interface AuthProviderProps {
  children: React.ReactNode;
}

export const AuthProvider: React.FC<AuthProviderProps> = ({ children }) => {
  const [user, setUser] = useState<AuthUser | null>(null);
  const [loading, setLoading] = useState(true);
  const supabase = createClient();

  useEffect(() =>
    // Get initial session
```

I have integrated supabase to the next.js project so we have to use it for authentication using context providers for state management on the client side

Read server.ts

Read index.ts

Now I'll create the authentication context provider and update the types to work with Supabase. Let me start by creating the authentication context.

index.ts +16 ✓

Now let me create the authentication context provider:

auth-context.tsx (new) +115

Now let me update the root layout to include the AuthProvider:

Read layout.tsx

layout.tsx +5 -2 •

Now let me update the login form to use the authentication context:

Read login-form.tsx

login-form.tsx +35 -5 •

Now let me update the register form as well:

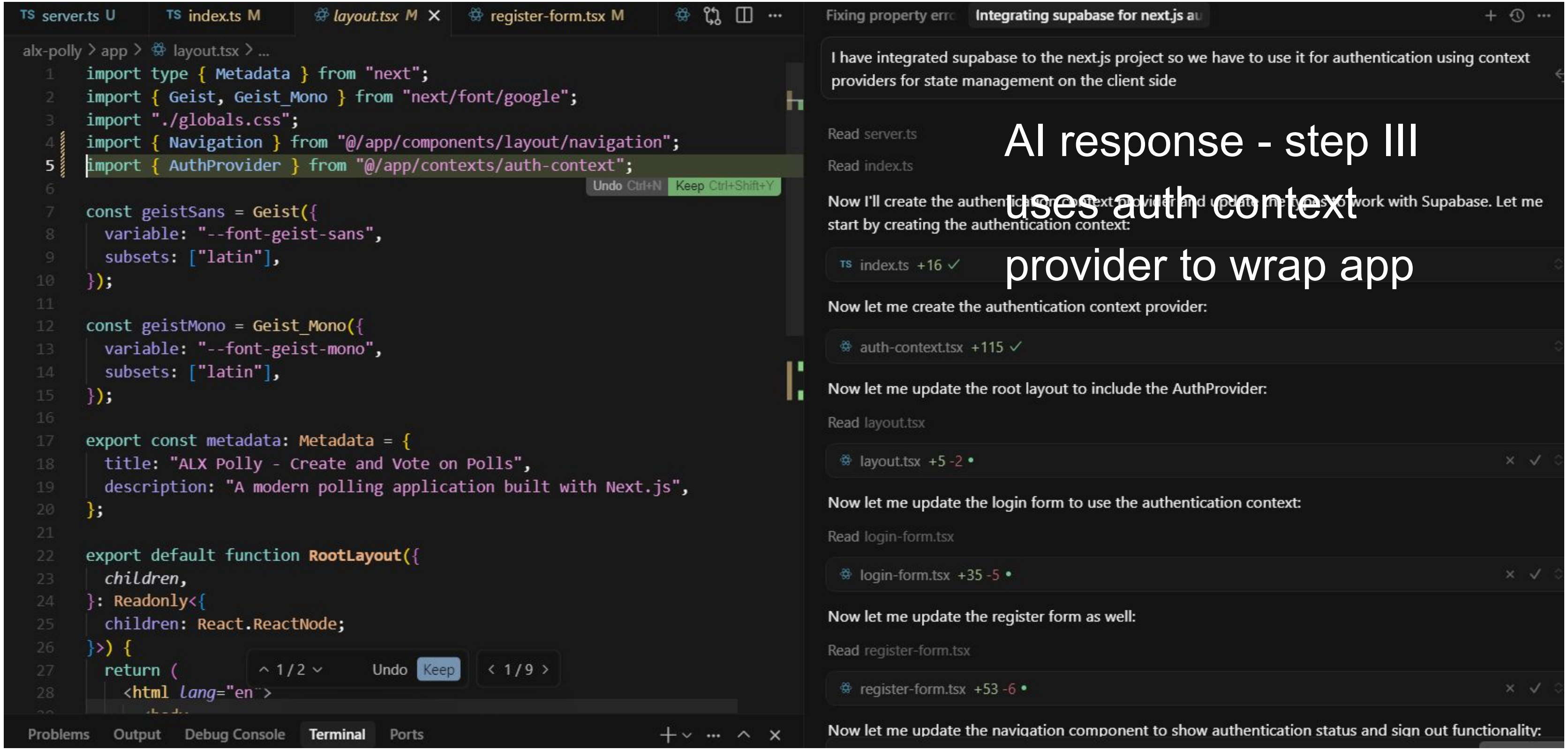
Read register-form.tsx

register-form.tsx +53 -6 •

Now let me update the navigation component to show authentication status and sign out functionality:

AI response - step II creates an auth context provider using created context

Building Features with AI - Authentication Feature



```tsx
import type { Metadata } from "next";
import { Geist, Geist_Mono } from "next/font/google";
import "./globals.css";
import { Navigation } from "@/app/components/layout/navigation";
import { AuthProvider } from "@/app/contexts/auth-context";

const geistSans = Geist({
  variable: "--font-geist-sans",
  subsets: ["latin"],
});

const geistMono = Geist_Mono({
  variable: "--font-geist-mono",
  subsets: ["latin"],
});

export const metadata: Metadata = {
  title: "ALX Polly - Create and Vote on Polls",
  description: "A modern polling application built with Next.js",
};

export default function RootLayout({
  children,
}: Readonly<{
  children: React.ReactNode;
}>) {
  return (
    <html lang="en">
```

I have integrated supabase to the next.js project so we have to use it for authentication using context providers for state management on the client side

Read server.ts

Read index.ts

Now I'll create the authentication context provider and update the types to work with Supabase. Let me start by creating the authentication context:

index.ts +16 ✓

Now let me create the authentication context provider:

auth-context.tsx +115 ✓

Now let me update the root layout to include the AuthProvider:

Read layout.tsx

layout.tsx +5 -2 •

Now let me update the login form to use the authentication context:

Read login-form.tsx

login-form.tsx +35 -5 •

Now let me update the register form as well:

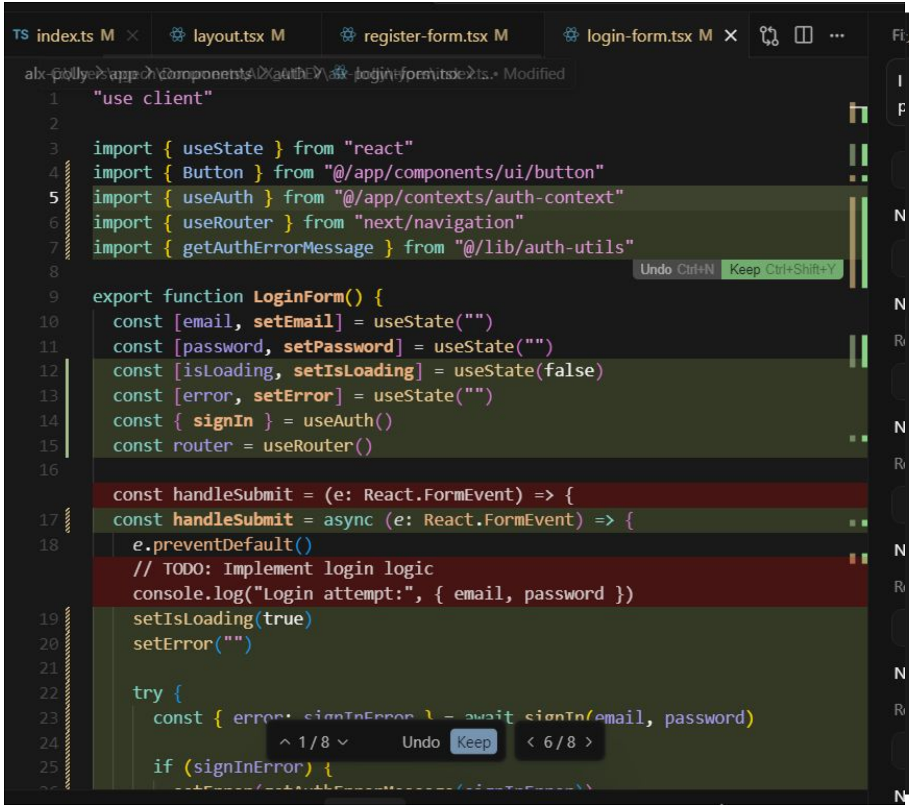
Read register-form.tsx

register-form.tsx +53 -6 •

Now let me update the navigation component to show authentication status and sign out functionality:

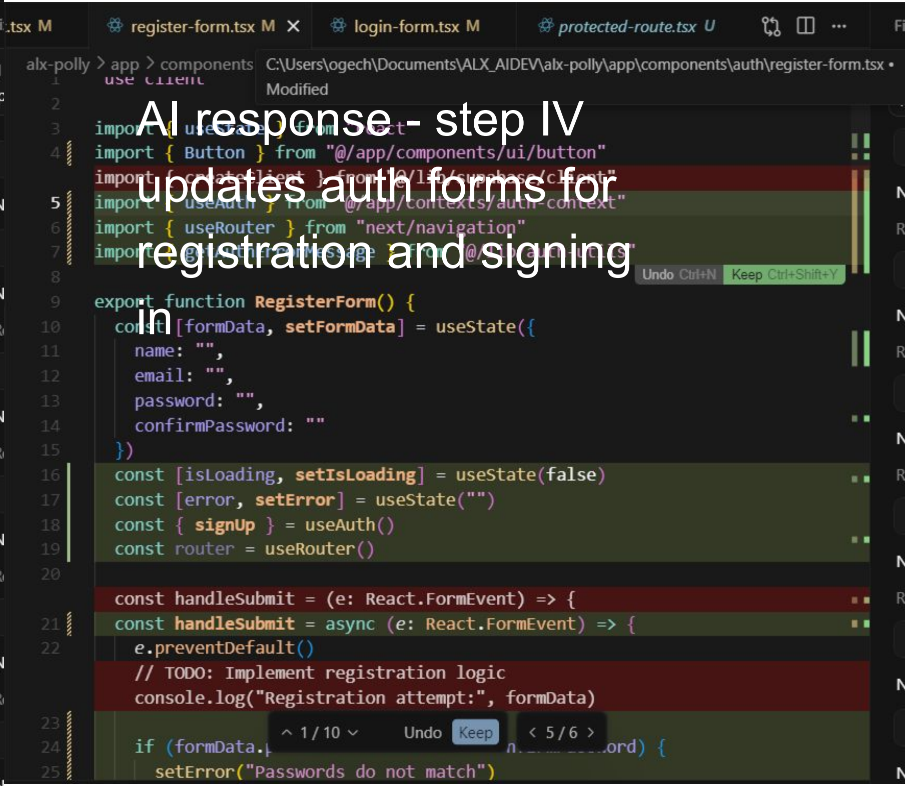
AI response - step III uses auth context provider to wrap app

Building Features with AI - Authentication Feature



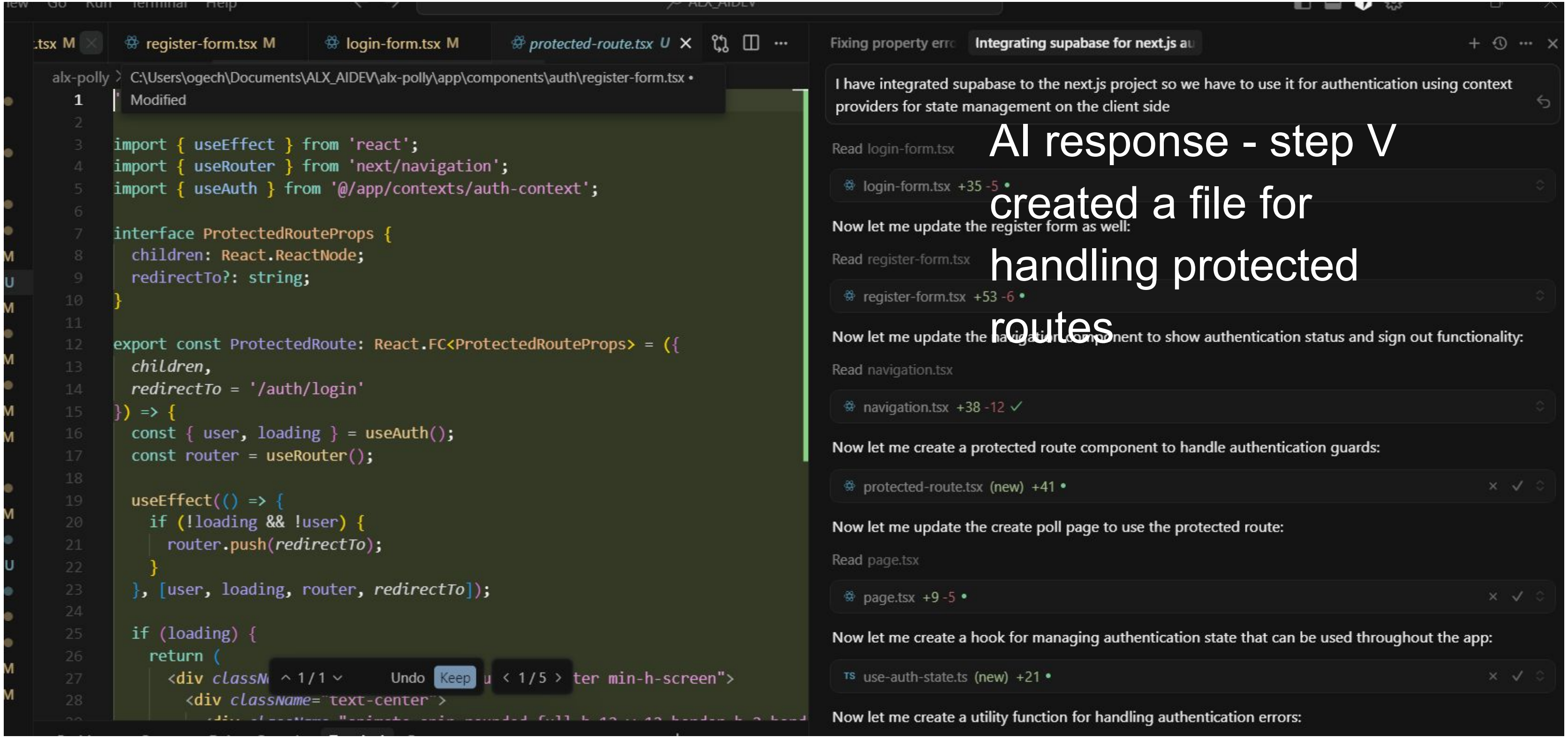
```
"use client"

import { useState } from "react"
import { Button } from "@/app/components/ui/button"
import { useAuth } from "@/app/contexts/auth-context"
import { useRouter } from "next/navigation"
import { getAuthErrorMessage } from "@/lib/auth-utils"

export function LoginForm() {
  const [email, setEmail] = useState("")
  const [password, setPassword] = useState("")
  const [isLoading, setIsLoading] = useState(false)
  const [error, setError] = useState("")
  const { signIn } = useAuth()
  const router = useRouter()

  const handleSubmit = (e: React.FormEvent) => {
  const handleSubmit = async (e: React.FormEvent) => {
    e.preventDefault()
    // TODO: Implement login logic
    console.log("Login attempt:", { email, password })
    setIsLoading(true)
    setError("")

    try {
      const { error: signInError } = await signIn(email, password)

      if (signInError) {
```

AI response - step IV updates auth forms for registration and signing in


Fi .tsx M
register-form.tsx M
login-form.tsx M
protected-route.tsx U
alx-polly > app > components
C:\Users\ogech\Documents\ALX_AIDEV\alx-polly\app\components\auth\register-form.tsx • Modified
Undo Ctrl+N
Keep Ctrl+Shift+Y
1 / 10
Undo
Keep
5 / 6

```
"use client"

import { useState } from "react"
import { Button } from "@/app/components/ui/button"
import { createClient } from "@/lib/supabase/client"
import { useAuth } from "@/app/contexts/auth-context"
import { useRouter } from "next/navigation"
import { getAuthErrorMessage } from "@/lib/auth-utils"

export function RegisterForm() {
  const [formData, setFormData] = useState({
    name: "",
    email: "",
    password: "",
    confirmPassword: ""
  })
  const [isLoading, setIsLoading] = useState(false)
  const [error, setError] = useState("")
  const { signUp } = useAuth()
  const router = useRouter()

  const handleSubmit = (e: React.FormEvent) => {
  const handleSubmit = async (e: React.FormEvent) => {
    e.preventDefault()
    // TODO: Implement registration logic
    console.log("Registration attempt:", formData)

    if (formData.p                          ord) {
      setError("Passwords do not match")
```

Building Features with AI - Authentication Feature


.tsx M
register-form.tsx M
login-form.tsx M
protected-route.tsx U
alx-polly
C:\Users\ogech\Documents\ALX_AIDEV\alx-polly\app\components\auth\register-form.tsx • Modified
^ 1 / 1 ⌄
Undo
Keep
< 1 / 5 >
Fixing property erro
Integrating supabase for next.js au

```tsx
'

import { useEffect } from 'react';
import { useRouter } from 'next/navigation';
import { useAuth } from '@/app/contexts/auth-context';

interface ProtectedRouteProps {
  children: React.ReactNode;
  redirectTo?: string;
}

export const ProtectedRoute: React.FC<ProtectedRouteProps> = ({
  children,
  redirectTo = '/auth/login'
}) => {
  const { user, loading } = useAuth();
  const router = useRouter();

  useEffect(() => {
    if (!loading && !user) {
      router.push(redirectTo);
    }
  }, [user, loading, router, redirectTo]);

  if (loading) {
    return (
      <div classN                          ter min-h-screen">
        <div className="text-center">
```

I have integrated supabase to the next.js project so we have to use it for authentication using context providers for state management on the client side

Read login-form.tsx

login-form.tsx +35 -5 •

Now let me update the register form as well:

Read register-form.tsx

register-form.tsx +53 -6 •

Now let me update the navigation component to show authentication status and sign out functionality:

Read navigation.tsx

navigation.tsx +38 -12 ✓

Now let me create a protected route component to handle authentication guards:

protected-route.tsx (new) +41 •

Now let me update the create poll page to use the protected route:

Read page.tsx

page.tsx +9 -5 •

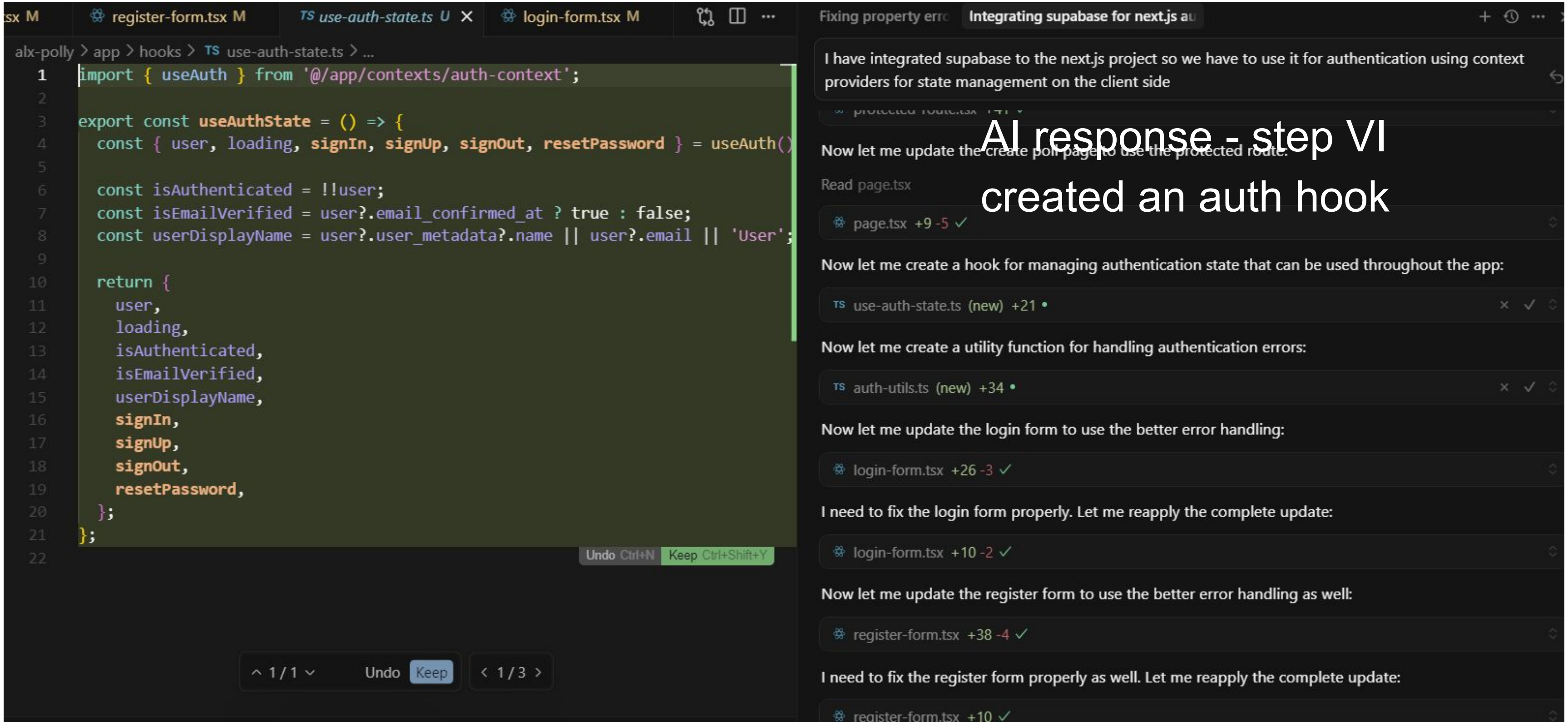
Now let me create a hook for managing authentication state that can be used throughout the app:

use-auth-state.ts (new) +21 •

Now let me create a utility function for handling authentication errors:

AI response - step V created a file for handling protected routes

Building Features with AI - Authentication Feature



```
import { useAuth } from '@/app/contexts/auth-context';

export const useAuthState = () => {
  const { user, loading, signIn, signUp, signOut, resetPassword } = useAuth()

  const isAuthenticated = !!user;
  const isEmailVerified = user?.email_confirmed_at ? true : false;
  const userDisplayName = user?.user_metadata?.name || user?.email || 'User';

  return {
    user,
    loading,
    isAuthenticated,
    isEmailVerified,
    userDisplayName,
    signIn,
    signUp,
    signOut,
    resetPassword,
  };
};
```

I have integrated supabase to the next.js project so we have to use it for authentication using context providers for state management on the client side

Now let me update the create poll page to use the protected route:

Read page.tsx

page.tsx +9 -5 ✓

Now let me create a hook for managing authentication state that can be used throughout the app:

use-auth-state.ts (new) +21 •

Now let me create a utility function for handling authentication errors:

auth-utils.ts (new) +34 •

Now let me update the login form to use the better error handling:

login-form.tsx +26 -3 ✓

I need to fix the login form properly. Let me reapply the complete update:

login-form.tsx +10 -2 ✓

Now let me update the register form to use the better error handling as well:

register-form.tsx +38 -4 ✓

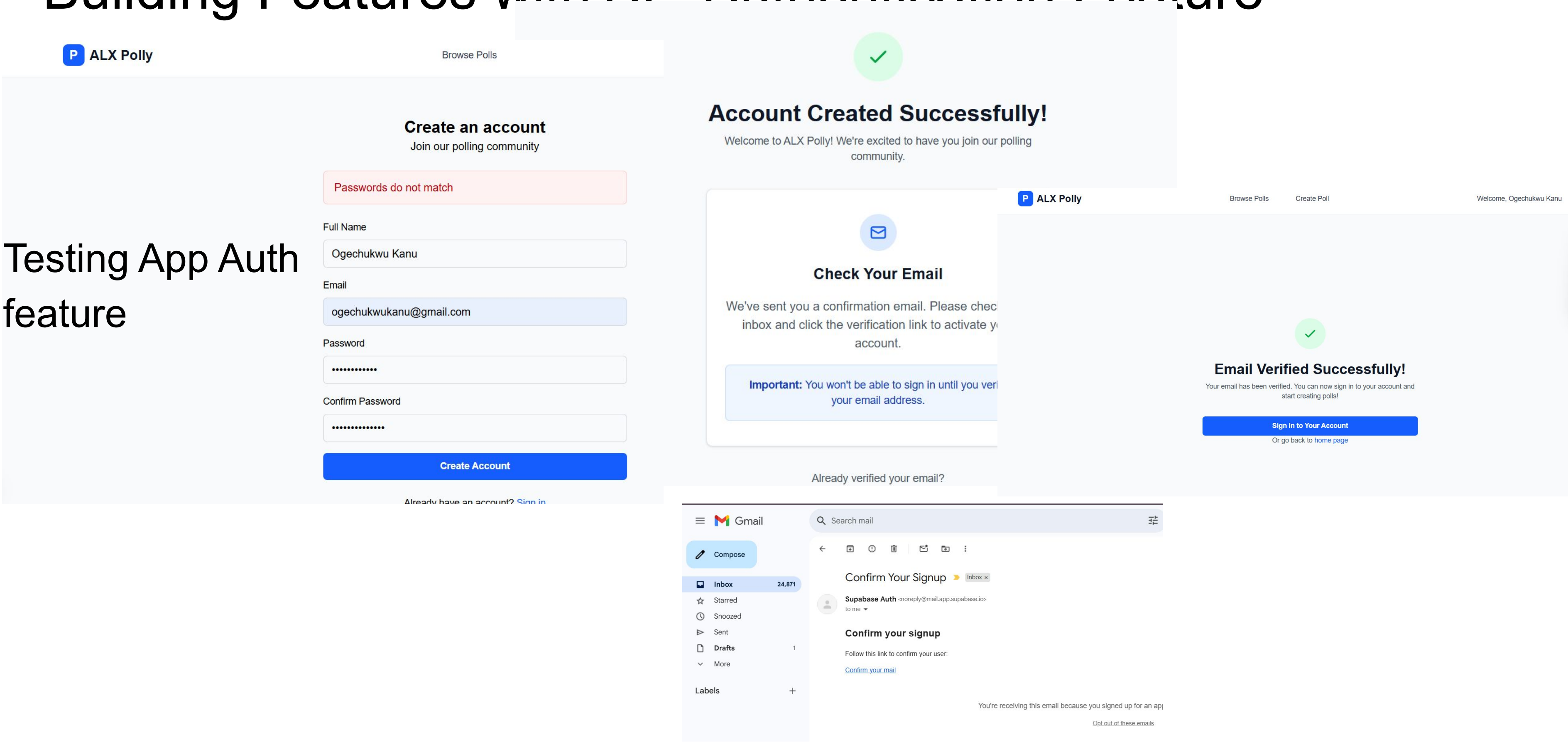
I need to fix the register form properly as well. Let me reapply the complete update:

register-form.tsx +10 ✓

AI response - step VI
created an auth hook

Building Features with AI - Authentication Feature

Testing App Auth feature



Create an account

Join our polling community

Passwords do not match

Full Name: Ogechukwu Kanu

Email: ogechukwukanu@gmail.com

Password: ••••••••••••

Confirm Password: ••••••••••••••

Already have an account? Sign in

Account Created Successfully!

Welcome to ALX Polly! We're excited to have you join our polling community.

Check Your Email

We've sent you a confirmation email. Please chec
inbox and click the verification link to activate y
account.

Important: You won't be able to sign in until you veri
your email address.

Already verified your email?

Email Verified Successfully!

Your email has been verified. You can now sign in to your account and start creating polls!

Or go back to home page

Confirm Your Signup

Supabase Auth <noreply@mail.app.supabase.io>
to me

Confirm your signup

Follow this link to confirm your user:

Confirm your mail

You're receiving this email because you signed up for an ap

Opt out of these emails