You are a senior code agent. Your job is to ensure the authentication flow for TitleTesterPro is secure, error-free, and 100% integrated with Supabase as the only Google OAuth provider.  
 You must also ensure there is **no reference to Neon** as a database—only Supabase Postgres is used.

### **Apply the following fixes:**

#### **1. Remove All Legacy/Manual Google OAuth Code**

* Delete or comment out any custom Google OAuth URL construction, code exchange handlers, or old googleAuth.ts files.
* Remove any server or client code that attempts to handle the OAuth code manually or exchanges it for tokens outside Supabase.
* There should be **no manual authUrl construction** or code reading logic in the codebase.

#### **2. Use Supabase Auth Provider Exclusively**

In the frontend, ensure login is only handled with:  
  
 js  
CopyEdit  
supabase.auth.signIn({ provider: 'google' });

or, if using Supabase v2+:  
  
 js  
CopyEdit  
supabase.auth.signInWithOAuth({ provider: 'google' });

* Do **not** allow any manual override or fallback to legacy Google Auth.

#### **3. Confirm Supabase and Google Config Consistency**

* Ensure the following are set and match:  
  + **Supabase Dashboard > Auth > Providers > Google**
    - Client ID and Client Secret exactly match Google Cloud Console.

Callback URL is:  
  
 bash  
CopyEdit  
https://xyehwoacgpsxakhjwglq.supabase.co/auth/v1/callback

* + **Google Cloud Console > Credentials > OAuth 2.0 Client**

Authorized redirect URI(s) include exactly:  
  
 bash  
CopyEdit  
https://xyehwoacgpsxakhjwglq.supabase.co/auth/v1/callback

* + - Add localhost or your production domain if needed for testing.

#### **4. Remove All “no\_code” and Custom OAuth Error Handling**

* Delete or comment out any error handlers that catch a missing code parameter and respond with "no\_code".
* All error handling for OAuth should defer to Supabase’s built-in logic.

#### **5. Session & Token Handling**

* On successful login, the frontend should rely on the Supabase session (using supabase.auth.getSession() or supabase.auth.onAuthStateChange).
* Backend API endpoints should use the sb-access-token cookie/session for user authentication.
* No backend logic should read Google tokens from the URL, request body, or custom headers.

#### **6. Database Cleanup**

* Remove all references, configs, imports, or environment variables relating to **Neon database**.
* Ensure the only database used is the integrated Supabase Postgres instance.
* Remove any old DATABASE\_URL or related env variables for Neon from code, config, and deployment scripts.

#### **7. Test and Document**

* Test the full login and dashboard load with a real Google account.
* Ensure no “no\_code” or OAuth redirect errors are possible.

Update README.md and/or replit.md to clarify that all Google login is through Supabase Auth only, with callback URL:  
  
 bash  
CopyEdit  
https://xyehwoacgpsxakhjwglq.supabase.co/auth/v1/callback

* and that only Supabase Postgres is used for data.