```
import { createServerClient, type CookieOptions } from '@supabase/ssr';
import { cookies } from 'next/headers';
import { Database } from '@/types_db';

// Define a function to create a Supabase client for server-side operations
// The function takes a cookie store created with next/headers cookies as an argument
export const createClient = () => {
  const cookieStore = cookies();

  return createServerClient<Database>(
    // Pass Supabase URL and anonymous key from the environment to the client
    process.env.NEXT_PUBLIC_SUPABASE_URL!,
    process.env.NEXT_PUBLIC_SUPABASE_ANON_KEY!,

     // Define a cookies object with methods for interacting with the cookie store and pass it to the client
    {
      cookies: {
        // The get method is used to retrieve a cookie by its name
        get(name: string) {
          return cookieStore.get(name)?.value;
        },
        // The set method is used to set a cookie with a given name, value, and options
        set(name: string, value: string, options: CookieOptions) {
          try {
            cookieStore.set({ name, value, ...options });
```

```
          } catch (error) {
            // If the set method is called from a Server Component, an error may occur
            // This can be ignored if there is middleware refreshing user sessions
          }
        },
        // The remove method is used to delete a cookie by its name
        remove(name: string, options: CookieOptions) {
          try {
            cookieStore.set({ name, value: '', ...options });
          } catch (error) {
            // If the remove method is called from a Server Component, an error may occur
            // This can be ignored if there is middleware refreshing user sessions
          }
        },
      },
    },
  );
};
```