**Инициализация проекта**

**1. Создать новый Next.js проект**

**В терминале выполни:**

**npx create-next-app@latest blog-app**

**При установке ответь:**

* **✅ TypeScript → Yes**
* **✅ ESLint → Yes**
* **✅ Tailwind → Yes**
* **✅ App Router → Yes (важно!)**
* **✅ Import alias → Yes (@/\*)**

**После этого у тебя будет папка blog-app с готовым проектом.**

**2. Запуск проекта**

**Перейди в папку и запусти:**

**cd blog-app**

**npm run dev**

**Теперь проект откроется на** [**http://localhost:3000**](http://localhost:3000)**.**

**3. Проверка Tailwind**

**Открой app/page.tsx и замени содержимое на тестовый код:**

**export default function Home() {**

**return (**

**<main className="flex min-h-screen items-center justify-center bg-gray-100">**

**<h1 className="text-3xl font-bold text-blue-600">Блог работает 🚀</h1>**

**</main>**

**)**

**}**

**Если на экране надпись синяя и по центру → Tailwind подключен.**

**Зависимости для Supabase**

**1. Основная библиотека**

npm install @supabase/supabase-js

👉 Это **клиент для работы с Supabase**.  
Через него мы будем подключаться к базе данных (posts), авторизации и storage.

**2. Хелперы для Next.js**

npm install @supabase/auth-helpers-nextjs

👉 Это **хелперы для интеграции Supabase и Next.js (App Router)**.

* Позволяют легко получать текущую сессию (getSession()).
* Держат пользователя в контексте.
* Упрощают защиту маршрутов.

**3. UI-компоненты для авторизации**

npm install @supabase/auth-ui-react @supabase/auth-ui-shared

👉 Это готовый **UI для формы входа/регистрации**.

* Уже есть стили и кнопки (Войти с Google, email + password).
* Можно кастомизировать под Tailwind.
* Очень удобно для /auth страницы.

**4. (опционально) dev tools**

npm install -D @types/node @types/react

👉 Если используем TypeScript, то это для подсказок типов (чаще всего уже есть).

**🔹 Установка всех зависимостей сразу**

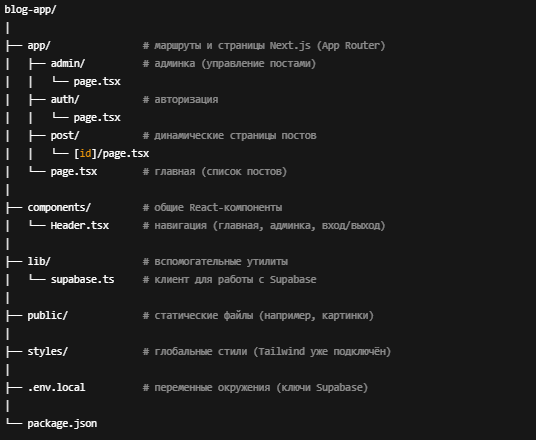
Чтобы не запускать по одному, можно так:

npm install @supabase/supabase-js @supabase/auth-helpers-nextjs @supabase/auth-ui-react @supabase/auth-ui-shared

📌 После установки у нас есть всё необходимое, чтобы:

* подключить клиент Supabase (lib/supabase.ts),
* сделать авторизацию через Google,
* получать данные из таблицы posts.

Структура папок:



**Создать проект в Supabase**

1. **Зайди в Supabase**  
   👉 https://supabase.com/  
   Авторизуйся через GitHub или Google.
2. **Создай новый проект**
   * Нажми **New Project**.
   * Выбери организацию (или создай новую).
   * Укажи название проекта (например, blog-app).
   * Выбери регион поближе (например, West Europe).
   * Установи пароль для базы данных (сохрани его!).
3. **Ждём развёртывания (~1 минута).**
4. **Найди ключи и URL проекта**
   * Перейди в меню **Project Settings → API**.
   * Скопируй:
     + Project URL → это **SUPABASE\_URL**
     + anon public key → это **SUPABASE\_ANON\_KEY**
5. **Добавь ключи в проект Next.js**  
   В корне проекта создай файл **.env.local** и вставь туда:
6. NEXT\_PUBLIC\_SUPABASE\_URL=https://your-project-url.supabase.co
7. NEXT\_PUBLIC\_SUPABASE\_ANON\_KEY=your-anon-key

⚠️ Используем префикс NEXT\_PUBLIC\_, чтобы ключ был доступен на клиенте.

**1. Переходим в Supabase**

1. Заходим в https://supabase.com/.
2. В проекте → вкладка **Database** → **SQL Editor**.

**2. Создаем таблицу posts**

Вставляем SQL-запрос:

create table if not exists posts (

id uuid primary key default gen\_random\_uuid(),

title text not null,

content text,

author text,

created\_at timestamp with time zone default now()

);

**3. Комментарии по полям**

* id — уникальный идентификатор записи (UUID, генерируется автоматически).
* title — заголовок поста (обязательное поле).
* content — текст статьи.
* author — автор поста (пока как text, позже можно связать с пользователем).
* created\_at — дата и время создания.

**4. Дополнительно (по желанию, на будущее)**

* Добавить **ограничения** (например, title уникальный).
* Добавить **связь** с пользователем через auth.users. Тогда author будет uuid references auth.users (id).

**1. Где взять ключи**

1. Заходим в Supabase → **Project Settings** → **API**.
2. Там будут:
   * **Project URL** (например, https://xyzcompany.supabase.co)
   * **anon public key**

**2. Создаём файл окружения**

В корне проекта Next.js создаём файл **.env.local** (он не попадёт в git).

Прописываем туда:

NEXT\_PUBLIC\_SUPABASE\_URL=https://xyzcompany.supabase.co

NEXT\_PUBLIC\_SUPABASE\_ANON\_KEY=ваш\_anon\_key

**3. Важные моменты**

* Переменные начинающиеся с NEXT\_PUBLIC\_ доступны и на клиенте, и на сервере.
* В supabase-js достаточно **anon key**, токен не нужен для базовых операций (чтение/запись в публичные таблицы, если настроены политики).
* **Service key** (секретный ключ) **никогда не используем на клиенте**, только на сервере.

Создай файл **/lib/supabase.ts** с таким кодом:

// /lib/supabase.ts

import { createClient } from '@supabase/supabase-js'

const supabaseUrl = process.env.NEXT\_PUBLIC\_SUPABASE\_URL as string

const supabaseAnonKey = process.env.NEXT\_PUBLIC\_SUPABASE\_ANON\_KEY as string

export const supabase = createClient(supabaseUrl, supabaseAnonKey)

**Объяснение:**

* createClient — подключает наш проект к Supabase.
* Берём URL и ключ из .env.local.
* Экспортируем supabase, чтобы везде можно было использовать:

import { supabase } from '@/lib/supabase'