

Les 10 Huiswerk

Supabase Auth Uitbreiden & Eindopdracht Brainstorm

Vak: AI-Assisted Development

Opleiding: NOVI Hogeschool Utrecht

Deadline: Voor de volgende les

Onderwerp: Supabase Auth, Google OAuth, RLS, Eindopdracht

Overzicht

In les 10 hebben we Supabase Auth toegevoegd aan onze Poll App: email/password login, magic links en basis Row Level Security (RLS). In dit huiswerk ga je de authenticatie uitbreiden met Google OAuth, polls koppelen aan gebruikers, en nadenken over je eindexamenopdracht.

Wat je al hebt na de les:

- Supabase project met Auth ingeschakeld
- Email/password registratie en login
- Magic link login
- Basis RLS policies op de polls tabel
- Login- en registratiepagina in je app

Opdracht 1: Google OAuth toevoegen

Voeg Google OAuth toe als derde inlogmethode naast email/password en magic link.

Stappen:

1. Maak een Google Cloud project aan (zie Appendix A)
2. Configureer OAuth credentials in Google Cloud Console
3. Voeg de credentials toe in je Supabase dashboard
4. Bouw een "Inloggen met Google" knop in je app

Checklist Opdracht 1:

- 1 Google Cloud project aangemaakt
- 2 OAuth 2.0 Client ID en Client Secret gegenereerd
- 3 Redirect URL correct ingesteld in Google Cloud Console
- 4 Google provider ingeschakeld in Supabase Auth settings
- 5 Google login knop toegevoegd aan je login pagina
- 6 Google OAuth werkend in je app (je kunt inloggen met je Google account)

Code voorbeeld — Google login knop:

```
// JavaScript
async function signInWithGoogle() {
  const { data, error } = await supabase.auth.signInWithOAuth({
    provider: 'google',
    options: {
      redirectTo: window.location.origin
    }
  })

  if (error) {
    console.error('Google login error:', error.message)
  }
}
```

Styling tip: Gebruik de officiële Google kleuren en logo voor je knop. Google heeft richtlijnen voor hoe hun login knop eruit moet zien: wit of blauw met het Google "G" logo.

Opdracht 2: User-gekoppelde polls

Koppel elke poll aan de gebruiker die hem heeft aangemaakt. Zo kun je laten zien wie een poll heeft gemaakt en ervoor zorgen dat alleen de eigenaar zijn eigen polls kan bewerken of verwijderen.

Stappen:

1. Voeg een `user_id` kolom toe aan de polls tabel
2. Werk de RLS policies bij zodat gebruikers alleen hun eigen polls kunnen bewerken/verwijderen
3. Update je app zodat bij het aanmaken van een poll automatisch de `user_id` wordt meegegeven

4. Toon in de app wie elke poll heeft aangemaakt

Checklist Opdracht 2:

- 1 `user_id` kolom toegevoegd aan `polls` tabel (uuid, references auth.users)
- 2 Bestaande polls een `user_id` gegeven (mag je eigen user id zijn)
- 3 RLS policy: iedereen kan polls lezen (SELECT)
- 4 RLS policy: ingelogde gebruikers kunnen polls aanmaken (INSERT)
- 5 RLS policy: alleen de eigenaar kan zijn poll bewerken (UPDATE)
- 6 RLS policy: alleen de eigenaar kan zijn poll verwijderen (DELETE)
- 7 App stuurt `user_id` mee bij het aanmaken van een poll
- 8 App toont wie elke poll heeft aangemaakt

Code voorbeeld — Poll aanmaken met `user_id`:

```
// JavaScript
async function createPoll(question, options) {
  const { data: { user } } = await supabase.auth.getUser()

  const { data, error } = await supabase
    .from('polls')
    .insert({
      question: question,
      options: options,
      user_id: user.id
    })
    .select()

  if (error) {
    console.error('Error creating poll:', error.message)
    return null
  }
  return data
}
```

UI tip: Toon de email van de maker onder elke poll. Je kunt `user_id` gebruiken om de gebruikersinfo op te halen, of je slaat de email direct op bij de poll.

Opdracht 3: Eindexamenopdracht brainstorm

De eindexamenopdracht is een zelfgekozen app die je bouwt met Supabase en AI-assisted development. Begin nu met nadenken over wat je wilt bouwen.

Schrijf het volgende op (geen code nodig):

1. **App naam** — Hoe heet je app?
2. **Beschrijving** — Wat doet je app? (2-3 zinnen)
3. **3 hoofdfuncties** — Wat zijn de drie belangrijkste features?
4. **Supabase features** — Welke Supabase onderdelen ga je gebruiken?

Supabase onderdelen om over na te denken:

- Database (welke tabellen?)
- Auth (welke methoden?)
- Storage (voor wat?)
- RLS (welke regels?)
- Realtime (waarvoor?)

Checklist Opdracht 3:

- 1 App naam gekozen
- 2 Beschrijving geschreven (2-3 zinnen)
- 3 3 hoofdfuncties beschreven
- 4 Supabase features opgelijst
- 5 Eindexamenopdracht idee uitgeschreven

Voorbeeld format:

```
# Voorbeeld
App naam: StudyBuddy
Beschrijving: Een app waar studenten samen kunnen leren
door flashcards te maken en te delen. Gebruikers kunnen
elkaars kaarten beoordelen en favorieten opslaan.

Hoofdfuncties:
1. Flashcards aanmaken met vraag/antwoord
2. Kaarten delen met andere gebruikers
3. Quiz-modus om jezelf te testen

Supabase features:
- Database: tabellen voor users, flashcards, favorites, scores
- Auth: email/password + Google OAuth
```

- Storage: afbeeldingen bij flashcards
- RLS: eigen kaarten bewerken, gedeelde kaarten alleen lezen
- Realtime: live quiz-sessies met meerdere spelers

Wat lever je in?

#	Onderdeel	Details
1	GitHub repository link	Public, of nodig Tim uit als collaborator
2	Korte reflectie (150 woorden)	Wat ging goed, wat was lastig, wat wil je bouwen voor de eindopdracht?
3	Screenshots	Google login werkend, polls met user info, RLS test
4	Eindexamenopdracht brainstorm	Mag in een apart bestand of in je README

Tips

- **Google OAuth testen:** Je kunt je eigen Gmail account gebruiken. Je hoeft geen "productie" goedkeuring van Google te hebben voor development.
- **RLS testen:** Open twee browsers (bijv. Chrome en Firefox) met verschillende accounts om te testen of je echt alleen je eigen polls kunt bewerken.
- **Supabase Dashboard:** Gebruik de SQL Editor om je policies te testen voordat je ze in je app gebruikt.
- **AI gebruiken:** Gebruik Claude of ChatGPT om je te helpen met SQL queries en RLS policies. Geef de AI je huidige tabelstructuur en vraag om hulp.
- **Eindopdracht:** Kies iets dat je zelf leuk vindt om te bouwen. Motivatie is de belangrijkste factor voor een goed eindproject.

Appendix A: Google OAuth Setup stap-voor-stap

Stap	Actie
1. Google Cloud Project	Ga naar console.cloud.google.com , maak een nieuw project aan (bijv. "Poll App NOVI")
2. OAuth Consent Screen	APIs & Services > OAuth consent screen. Kies "External", vul app name en email in, voeg je Gmail toe als test user
3. OAuth Credentials	APIs & Services > Credentials > Create > OAuth client ID. Type: Web application. Redirect URI: <code>https://<project-id>.supabase.co/auth/v1/callback</code>

Stap	Actie
4. Supabase configureren	Authentication > Providers > Google > Enabled. Plak Client ID en Client Secret
5. Code toevoegen	Voeg de signInWithGoogle() functie en een login knop toe aan je app (zie code hieronder)

```
// JavaScript
// Google login functie
async function signInWithGoogle() {
  const { data, error } = await supabase.auth.signInWithOAuth({
    provider: 'google',
    options: { redirectTo: window.location.origin }
  })
  if (error) console.error('Google login error:', error.message)
}
```

Appendix B: SQL voor user_id kolom en policies

Stap 1: user_id kolom toevoegen + Stap 2: Oude policies verwijderen

```
-- SQL
-- Voeg user_id kolom toe aan polls tabel
ALTER TABLE polls ADD COLUMN user_id UUID REFERENCES auth.users(id);
-- Geef bestaande polls jouw user_id (vind je ID in Supabase > Auth > Users)
UPDATE polls SET user_id = 'jouw-user-id-hier' WHERE user_id IS NULL;
ALTER TABLE polls ALTER COLUMN user_id SET NOT NULL;

-- Verwijder oude policies
DROP POLICY IF EXISTS "Allow all select" ON polls;
DROP POLICY IF EXISTS "Allow all insert" ON polls;
DROP POLICY IF EXISTS "Allow all update" ON polls;
DROP POLICY IF EXISTS "Allow all delete" ON polls;
```

Stap 3: Nieuwe RLS policies + Stap 4: Verifieer

```
-- SQL
CREATE POLICY "Polls zijn zichtbaar voor iedereen"
ON polls FOR SELECT USING (true);

CREATE POLICY "Ingelogde gebruikers kunnen polls aanmaken"
ON polls FOR INSERT WITH CHECK (auth.uid() = user_id);

CREATE POLICY "Eigenaar kan eigen poll updaten"
```

```

ON polls FOR UPDATE USING (auth.uid() = user_id)
WITH CHECK (auth.uid() = user_id);

CREATE POLICY "Eigenaar kan eigen poll verwijderen"
ON polls FOR DELETE USING (auth.uid() = user_id);

-- Verifieer: bekijk alle policies
SELECT policyname, cmd, qual, with_check
FROM pg_policies WHERE tablename = 'polls';

```

Appendix C: Troubleshooting

Fout	Oorzaak	Oplossing
RLS policy violation	user_id komt niet overeen met ingelogde user	Check supabase.auth.getUser() en zorg dat user_id = user.id
Invalid redirect URL	Redirect URL in Google Console matcht niet	Moet exact zijn: https://<id>.supabase.co/auth/v1/callback
Access blocked	OAuth consent screen niet goed ingesteld	Voeg je email toe als test user, status moet "Testing" zijn
Polls onzichtbaar na RLS	Geen SELECT policy aanwezig	Maak een SELECT policy: USING (true)
Andermans poll bewerken	UPDATE/DELETE policy mist of RLS uit	ALTER TABLE polls ENABLE ROW LEVEL SECURITY
Google knop doet niets	JS error of Supabase client fout	Open console (F12), check of supabase is geïnitieerd

Veel succes! Denk goed na over je eindexamenopdracht — het wordt het leukste deel van de cursus.