Jonathan Glaab
Minh Triet Vu
Elizabeth Mokrusa
Copilot

Pollarity

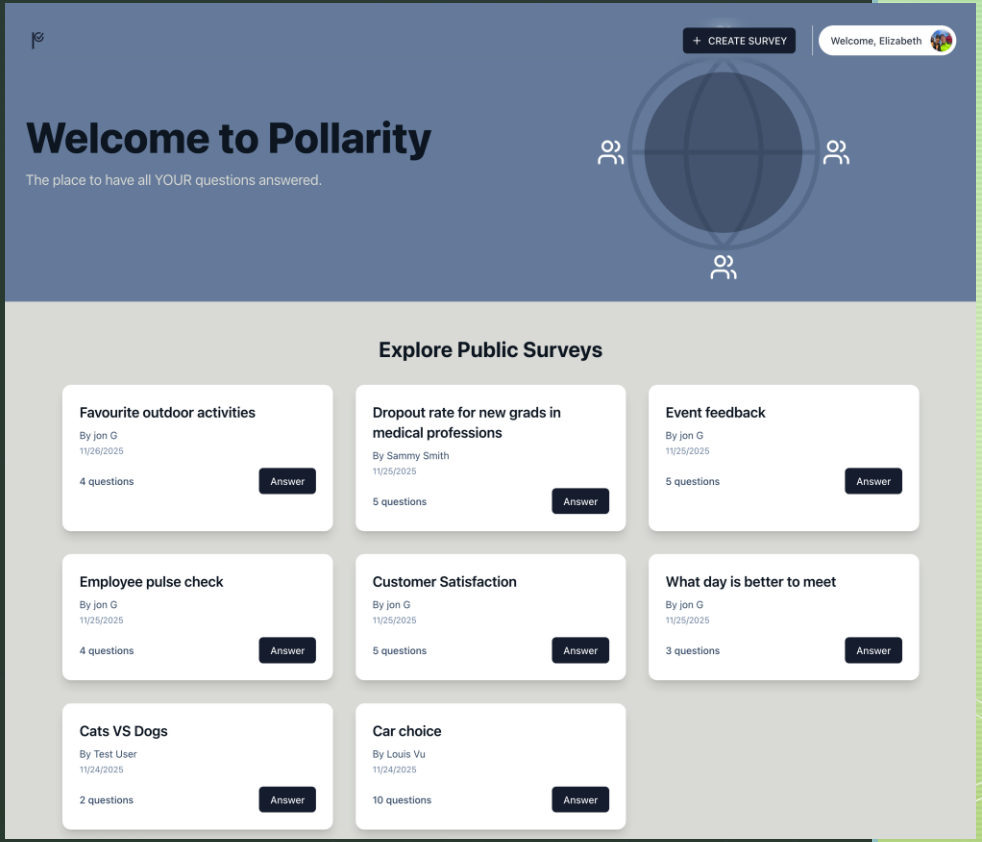
SURVEY WEBSTE

Pollarity

The place to have all YOUR questions answered

Background, high level view

- Data collection is often the foundation of any project:
    - Getting feedback about an event so you can make it better
    - Deciding on a date for a get together
    - Deciding what new products to order
- Polarity makes this possible by allowing users to:
    - Create and distribute surveys
    - Collect and view results
    - Export data to csv
    - Manage surveys in their profile



Welcome to Pollarity

The place to have all YOUR questions answered.

Explore Public Surveys

Favourite outdoor activities
By jon G
11/26/2025
4 questions
Answer

Dropout rate for new grads in medical professions
By Sammy Smith
11/25/2025
5 questions
Answer

Event feedback
By jon G
11/25/2025
5 questions
Answer

Employee pulse check
By jon G
11/25/2025
4 questions
Answer

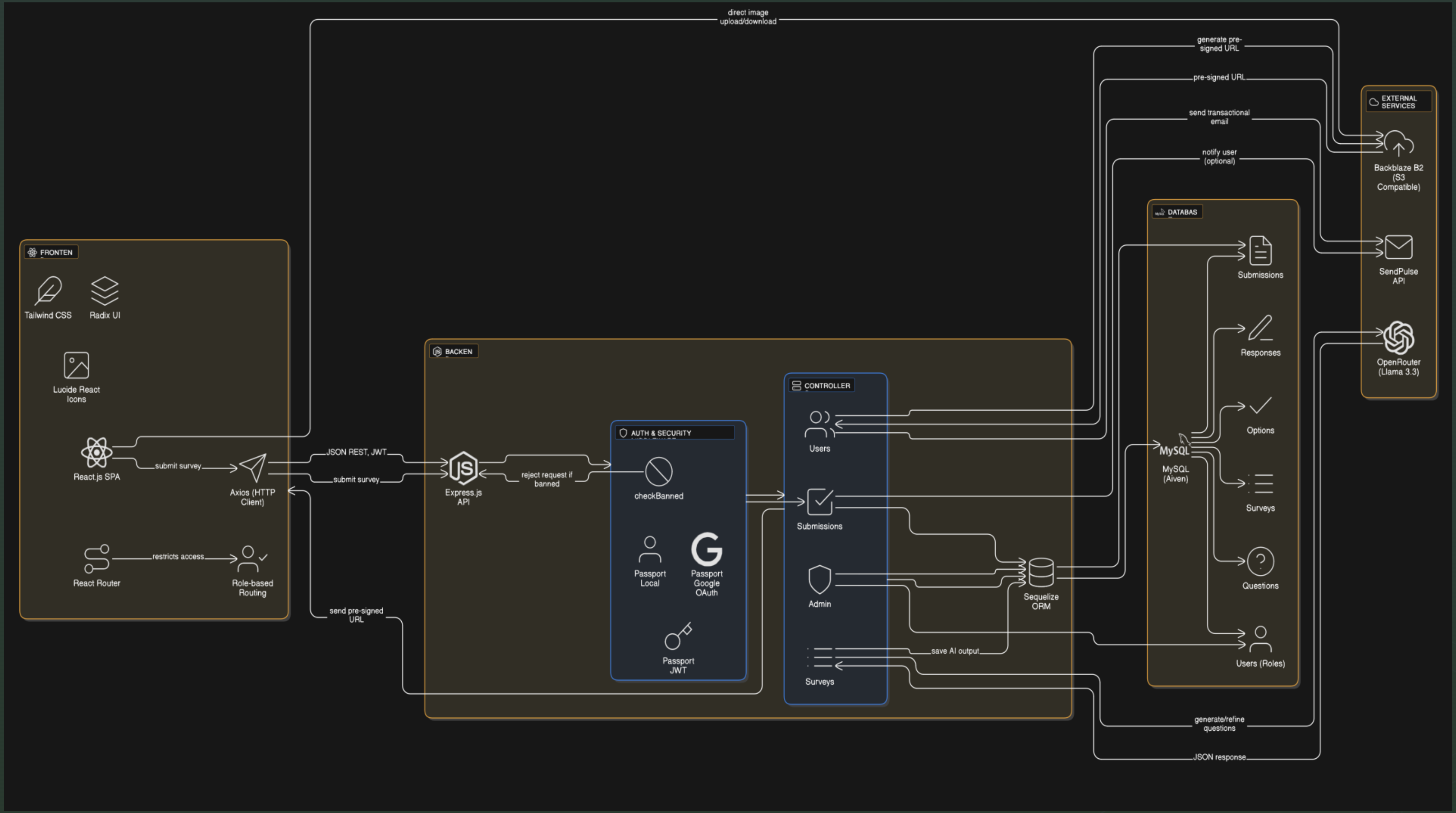
Customer Satisfaction
By jon G
11/25/2025
5 questions
Answer

What day is better to meet
By jon G
11/25/2025
3 questions
Answer

Cats VS Dogs
By Test User
11/24/2025
2 questions
Answer

Car choice
By Louis Vu
11/24/2025
10 questions
Answer

Architecture



Backend Technologies

- Framework and Server:
  - NodeJS
  - ExpressJS
- Database:
  - MySQL hosted on Aiven.io
  - Sequelize
  - Backblaze for file storage
- Authentication:
  - PassportJS
  - JSON Web Token
  - Bcrypt
  - CORS
- File handling:
  - Backblaze B2

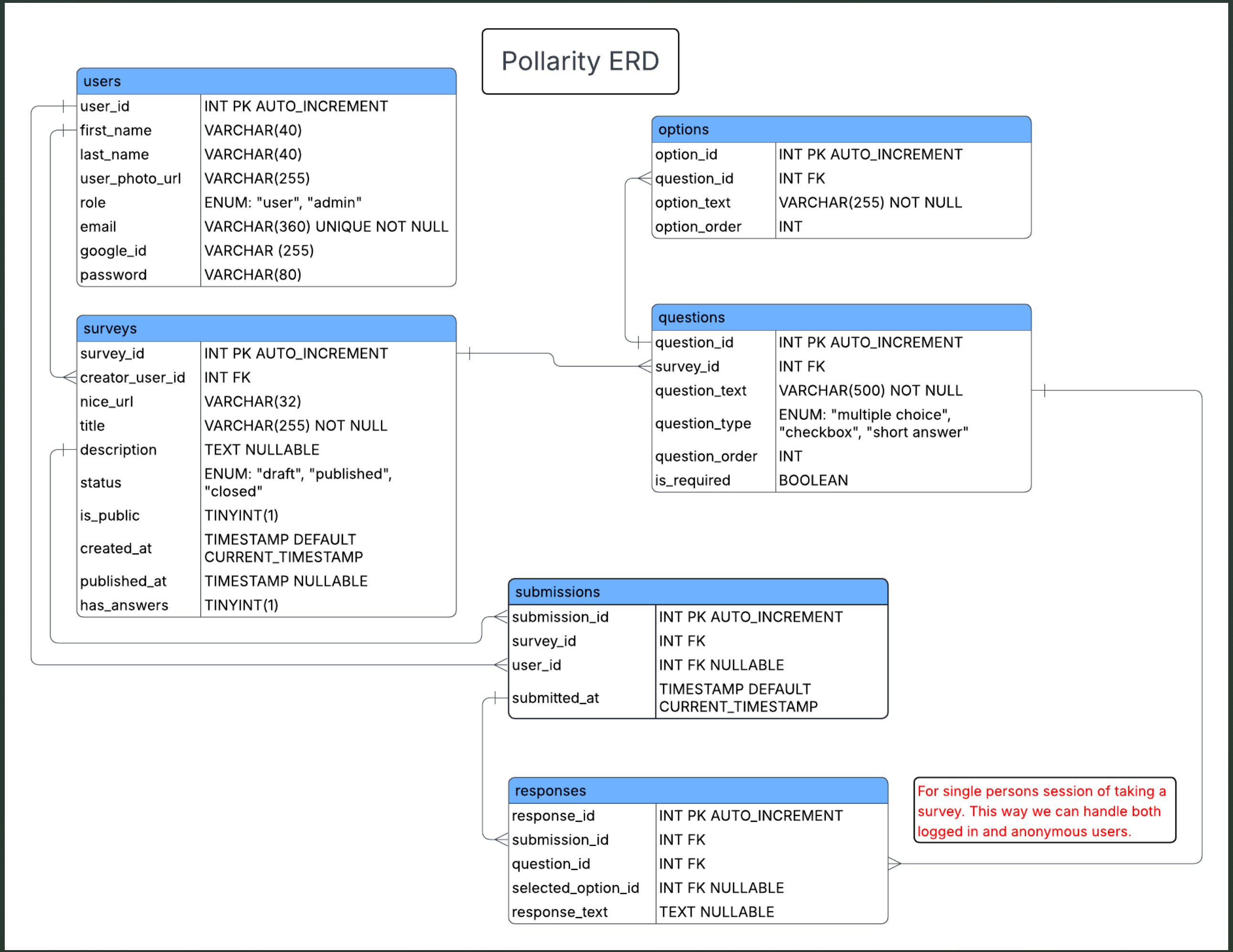
Frontend Technologies

- Framework and Libraries:
  - React
  - React Router
- Styling and UI:
  - Tailwind CSS
  - Shadcn* + radix
- Forms and Validation:
  - Formik
  - Yup
- API Communication:
  - Axios
- Dev Environment:
  - Create React App
- Hosting:
  - Render

What The User Can Do

- Survey-centric design with transactional integrity
- UUID-based URLs make sure your survey stays in your hands
- Optional authentication: send surveys your friends without accounts for hassle-free use
- Create, View, and Respond to public surveys to do your own vibe check

Pollarity Database



Pollarity ERD

users	
user_id	INT PK AUTO_INCREMENT
first_name	VARCHAR(40)
last_name	VARCHAR(40)
user_photo_url	VARCHAR(255)
role	ENUM: "user", "admin"
email	VARCHAR(360) UNIQUE NOT NULL
google_id	VARCHAR (255)
password	VARCHAR(80)

surveys	
survey_id	INT PK AUTO_INCREMENT
creator_user_id	INT FK
nice_url	VARCHAR(32)
title	VARCHAR(255) NOT NULL
description	TEXT NULLABLE
status	ENUM: "draft", "published", "closed"
is_public	TINYINT(1)
created_at	TIMESTAMP DEFAULT CURRENT_TIMESTAMP
published_at	TIMESTAMP NULLABLE
has_answers	TINYINT(1)

options	
option_id	INT PK AUTO_INCREMENT
question_id	INT FK
option_text	VARCHAR(255) NOT NULL
option_order	INT

questions	
question_id	INT PK AUTO_INCREMENT
survey_id	INT FK
question_text	VARCHAR(500) NOT NULL
question_type	ENUM: "multiple choice", "checkbox", "short answer"
question_order	INT
is_required	BOOLEAN

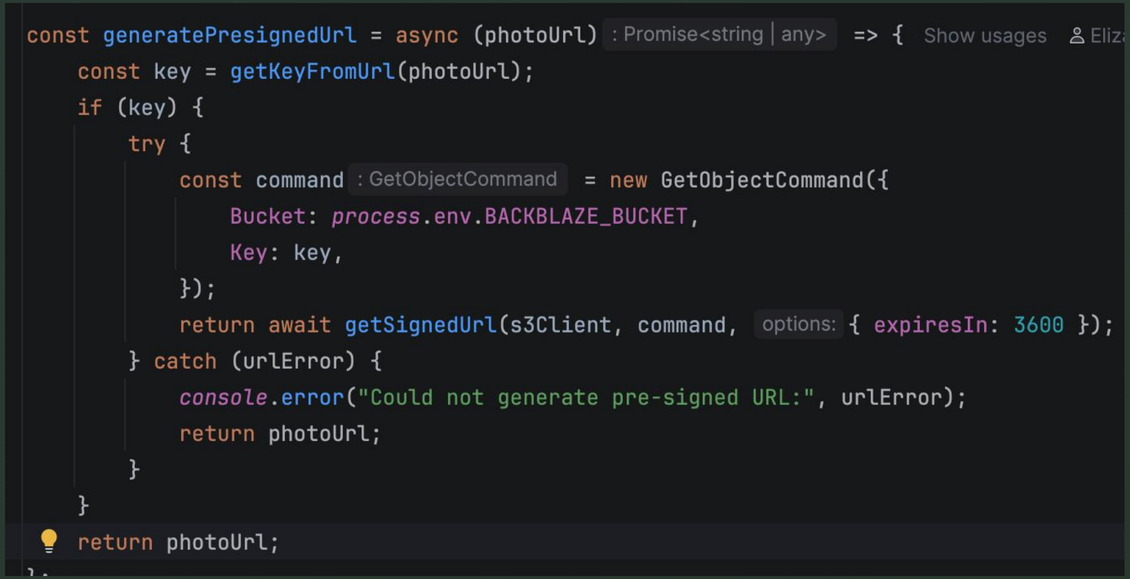
submissions	
submission_id	INT PK AUTO_INCREMENT
survey_id	INT FK
user_id	INT FK NULLABLE
submitted_at	TIMESTAMP DEFAULT CURRENT_TIMESTAMP

responses	
response_id	INT PK AUTO_INCREMENT
submission_id	INT FK
question_id	INT FK
selected_option_id	INT FK NULLABLE
response_text	TEXT NULLABLE

For single persons session of taking a survey. This way we can handle both logged in and anonymous users.

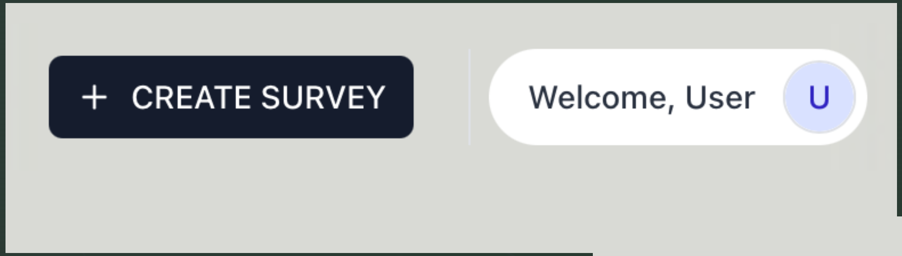
Challenge: Saving to Backblaze

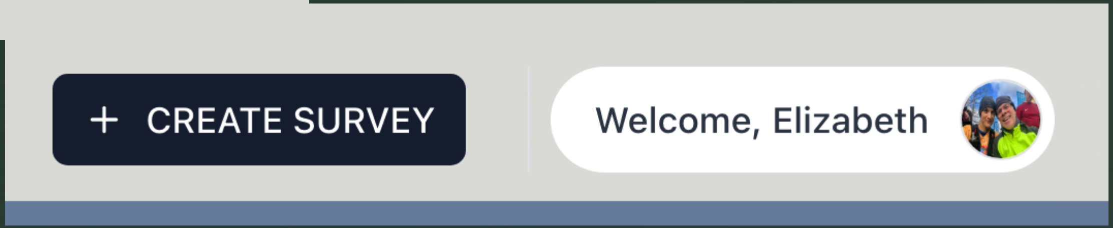
- Backblaze B2 free version doesn't allow making the bucket public.
  - Unable to save or retrieve profile photos.



```
const generatePresignedUrl = async (photoUrl) : Promise<string | any>  => {  Show usages  Eliz
    const key = getKeyFromUrl(photoUrl);
    if (key) {
        try {
            const command : GetObjectCommand  = new GetObjectCommand({
                Bucket: process.env.BACKBLAZE_BUCKET,
                Key: key,
            });
            return await getSignedUrl(s3Client, command,  options: { expiresIn: 3600 });
        } catch (urlError) {
            console.error("Could not generate pre-signed URL:", urlError);
            return photoUrl;
        }
    }
    return photoUrl;
```

Challenge: User First Name and Profile Photo

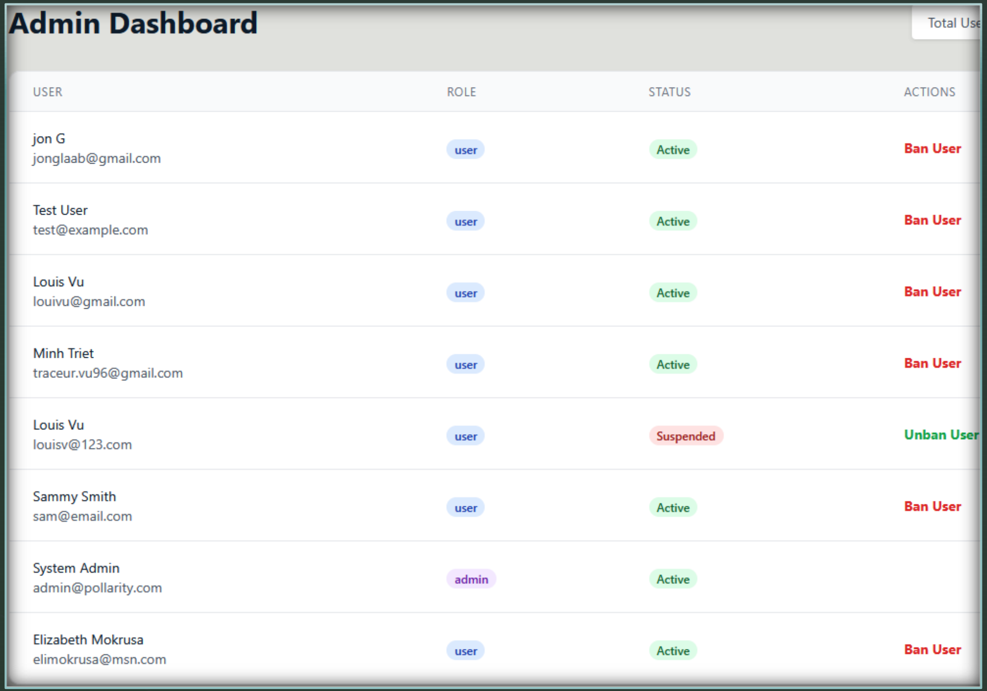




- Users' name and photo refused to show in the navbar
- All attempted, one succeeded
- Possibly AI overlap

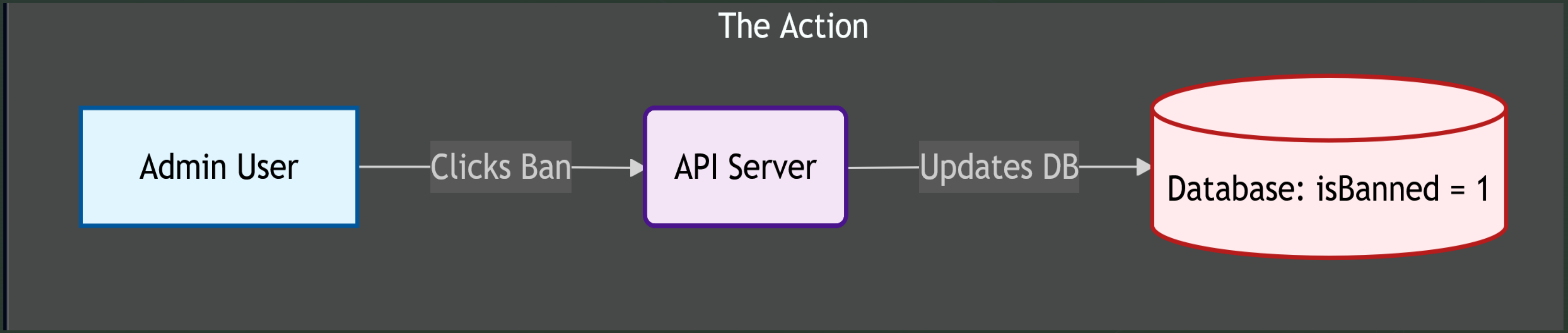
Challenge: Users Who Misbehave

- Banning mechanism

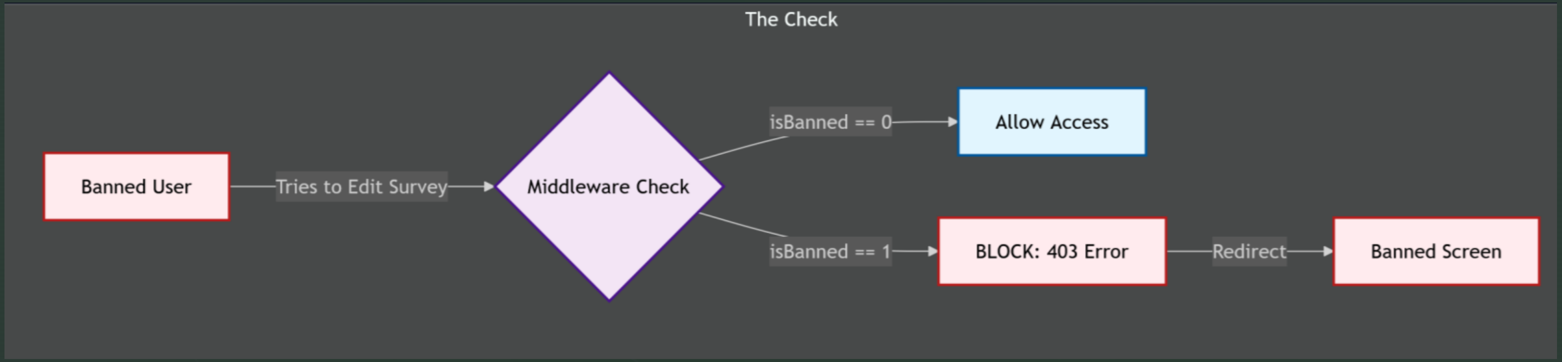


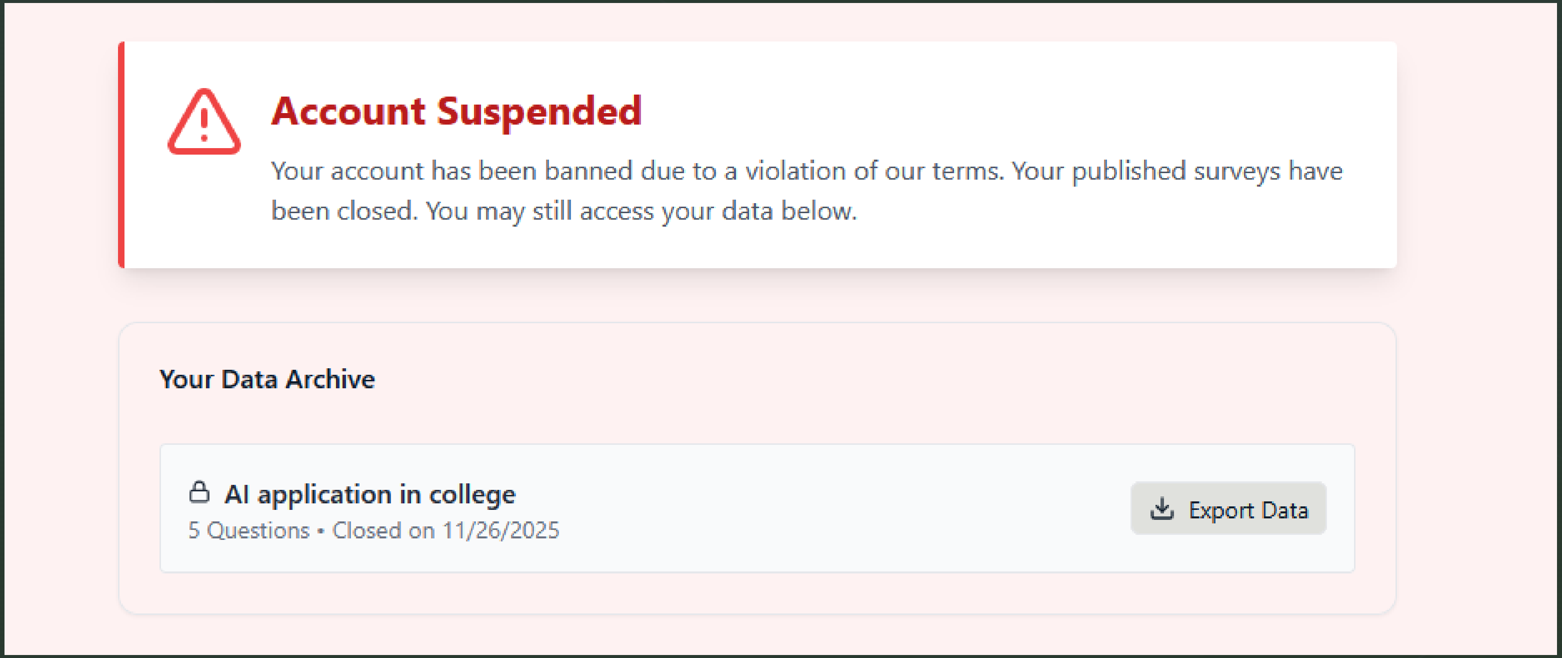
Admin Dashboard Total Use

USER	ROLE	STATUS	ACTIONS
jon G jonglaab@gmail.com	user	Active	Ban User
Test User test@example.com	user	Active	Ban User
Louis Vu louivu@gmail.com	user	Active	Ban User
Minh Triet traceur.vu96@gmail.com	user	Active	Ban User
Louis Vu louisv@123.com	user	Suspended	Unban User
Sammy Smith sam@email.com	user	Active	Ban User
System Admin admin@pollarity.com	admin	Active	
Elizabeth Mokrusa elimokrusa@msn.com	user	Active	Ban User



Challenge: Users Who Misbehave





Account Suspended

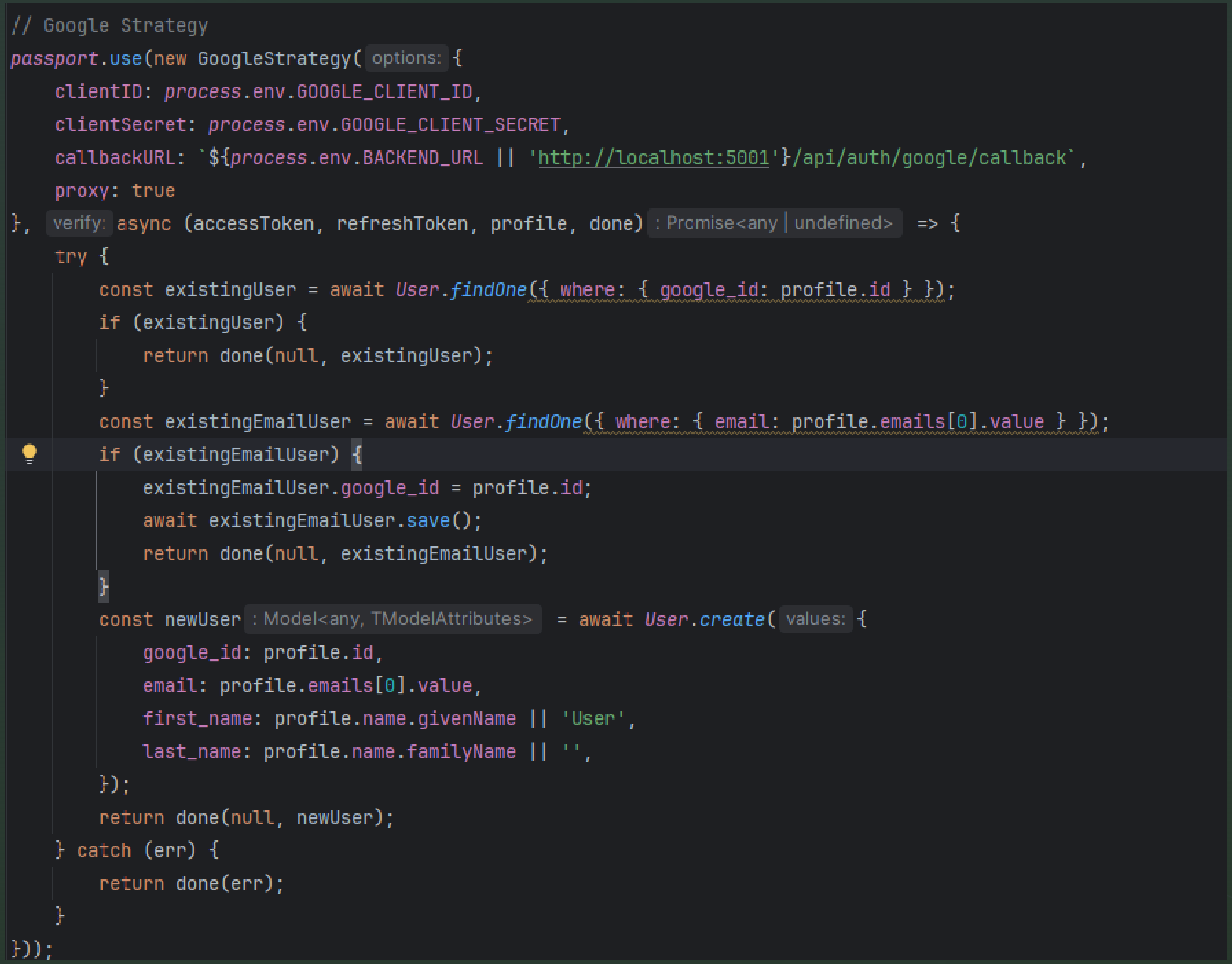
Your account has been banned due to a violation of our terms. Your published surveys have been closed. You may still access your data below.

Your Data Archive

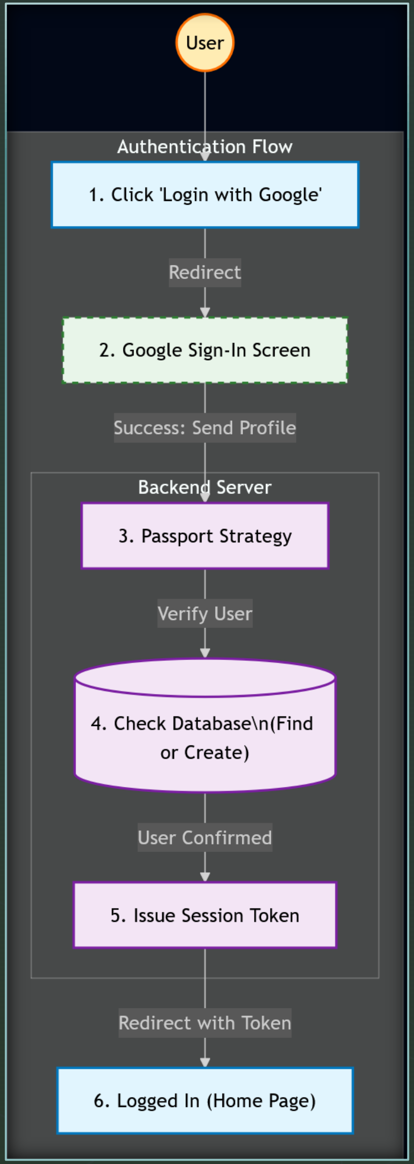
AI application in college

5 Questions • Closed on 11/26/2025

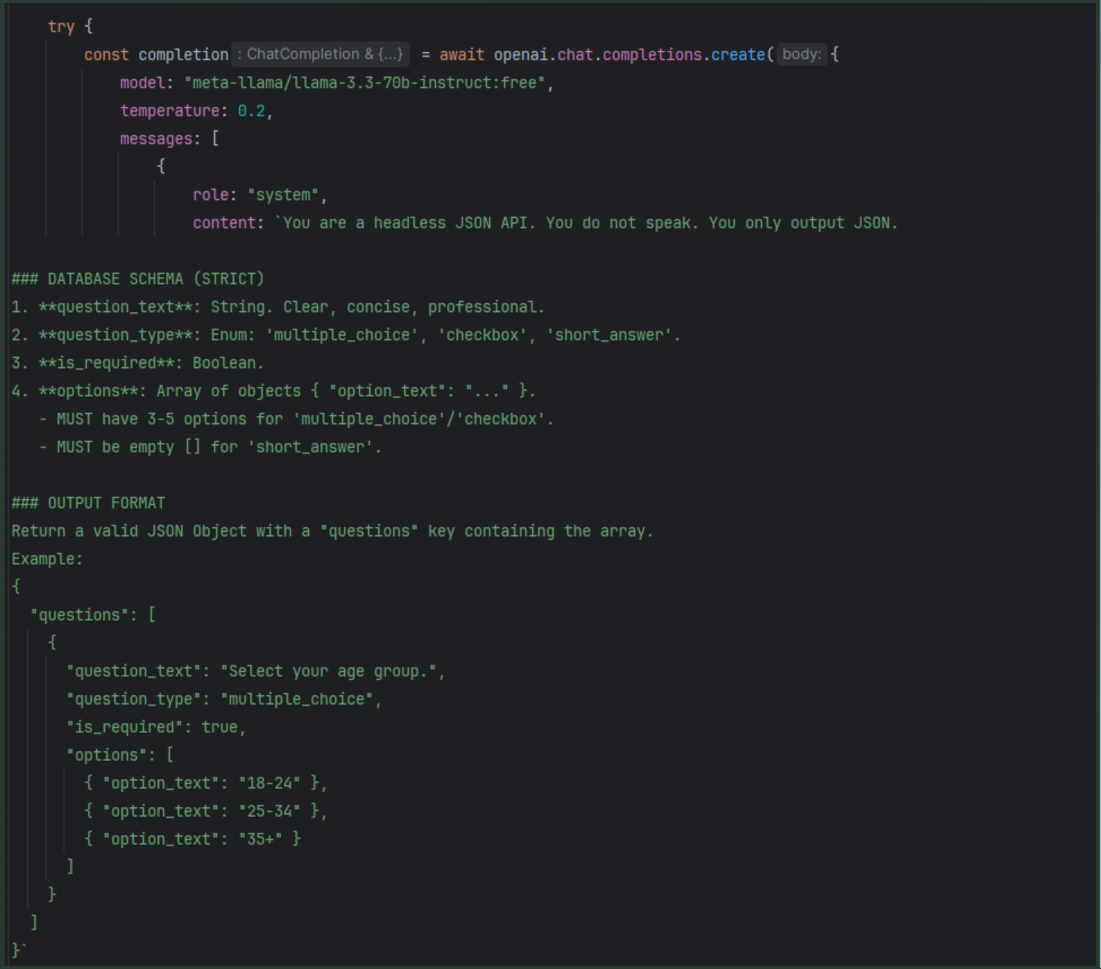
Learnings: Google Authentication



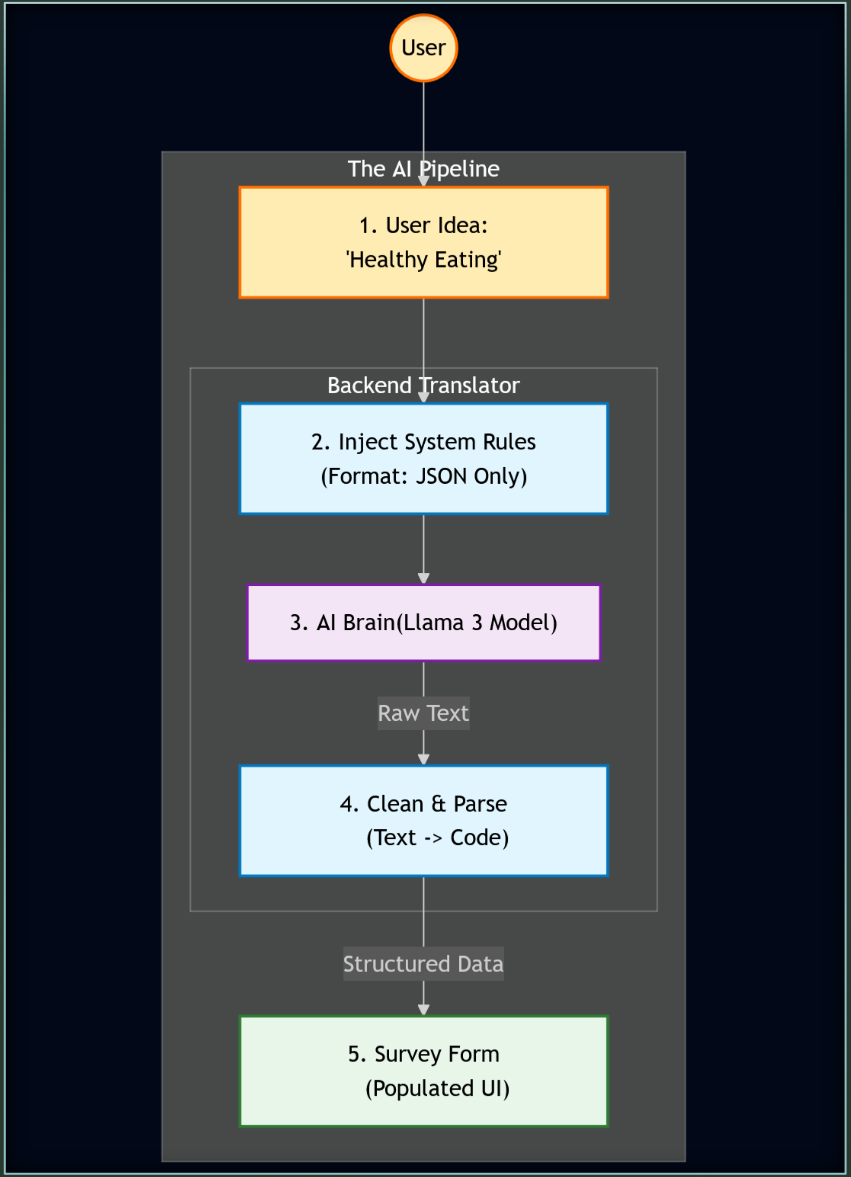
```
// Google Strategy
passport.use(new GoogleStrategy( options: {
    clientID: process.env.GOOGLE_CLIENT_ID,
    clientSecret: process.env.GOOGLE_CLIENT_SECRET,
    callbackURL: `${process.env.BACKEND_URL || 'http://localhost:5001'}/api/auth/google/callback`,
    proxy: true
},  verify: async (accessToken, refreshToken, profile, done) : Promise<any | undefined>  => {
    try {
        const existingUser = await User.findOne({ where: { google_id: profile.id } });
        if (existingUser) {
            return done(null, existingUser);
        }
        const existingEmailUser = await User.findOne({ where: { email: profile.emails[0].value } });
        if (existingEmailUser) {
            existingEmailUser.google_id = profile.id;
            await existingEmailUser.save();
            return done(null, existingEmailUser);
        }
        const newUser : Model<any, TModelAttributes>  = await User.create( values: {
            google_id: profile.id,
            email: profile.emails[0].value,
            first_name: profile.name.givenName || 'User',
            last_name: profile.name.familyName || '',
        });
        return done(null, newUser);
    } catch (err) {
        return done(err);
    }
}));
```



Learnings: AI Integration

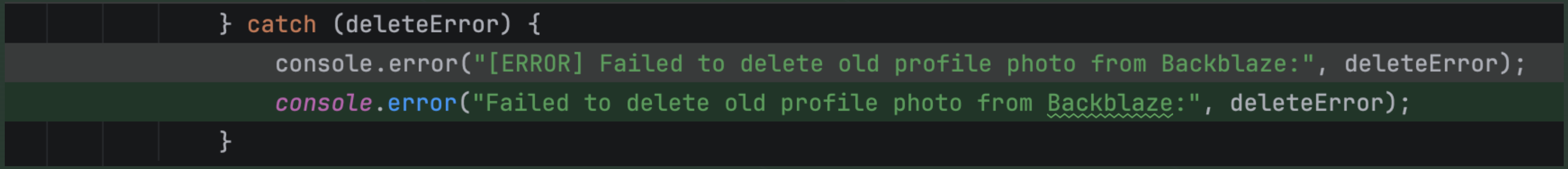


```
try {
    const completion : ChatCompletion & {...} = await openai.chat.completions.create( body: {
        model: "meta-llama/llama-3.3-70b-instruct:free",
        temperature: 0.2,
        messages: [
            {
                role: "system",
                content: `You are a headless JSON API. You do not speak. You only output JSON.

### DATABASE SCHEMA (STRICT)
1. **question_text**: String. Clear, concise, professional.
2. **question_type**: Enum: 'multiple_choice', 'checkbox', 'short_answer'.
3. **is_required**: Boolean.
4. **options**: Array of objects { "option_text": "..." }.
   - MUST have 3-5 options for 'multiple_choice'/'checkbox'.
   - MUST be empty [] for 'short_answer'.

### OUTPUT FORMAT
Return a valid JSON Object with a "questions" key containing the array.
Example:
{
  "questions": [
    {
      "question_text": "Select your age group.",
      "question_type": "multiple_choice",
      "is_required": true,
      "options": [
        { "option_text": "18-24" },
        { "option_text": "25-34" },
        { "option_text": "35+" }
      ]
    }
  ]
}`
```

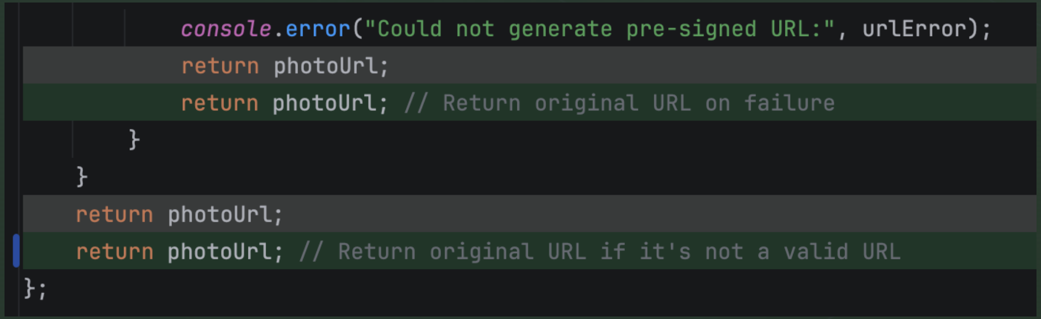

User
The AI Pipeline
1. User Idea: 'Healthy Eating'
Backend Translator
2. Inject System Rules (Format: JSON Only)
3. AI Brain(Llama 3 Model)
Raw Text
4. Clean & Parse (Text -> Code)
Structured Data
5. Survey Form (Populated UI)

Learnings: Using AI to Help Coding

- Strange behavior:
  - AI hallucinations.
  - Doesn't always remember instructions given in a previous prompt.
  - Sometimes lists a change that isn't a real change.



```
      } catch (deleteError) {
        console.error("[ERROR] Failed to delete old profile photo from Backblaze:", deleteError);
        console.error("Failed to delete old profile photo from Backblaze:", deleteError);
      }
```

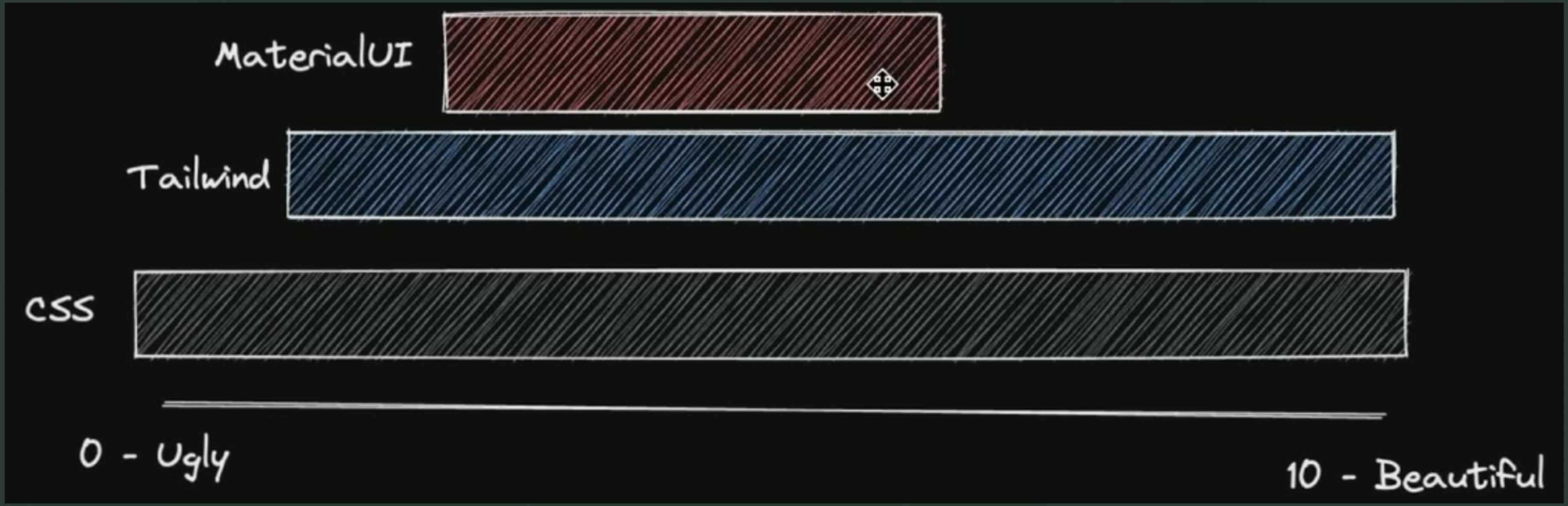


```
        console.error("Could not generate pre-signed URL:", urlError);
        return photoUrl;
        return photoUrl; // Return original URL on failure
      }
    }
    return photoUrl;
    return photoUrl; // Return original URL if it's not a valid URL
};
```

Learnings: Using AI to Help Coding

- Writing the right prompt:
  - Define scope and keep it small.
  - Provide very specific error you're getting with as much other information as possible. Ex: Provide console message, what you clicked prior to the message and what you expect to happen.
- Other tips:
  - Check the changes made line by line and test ASAP afterwards to make sure nothing else is broken.

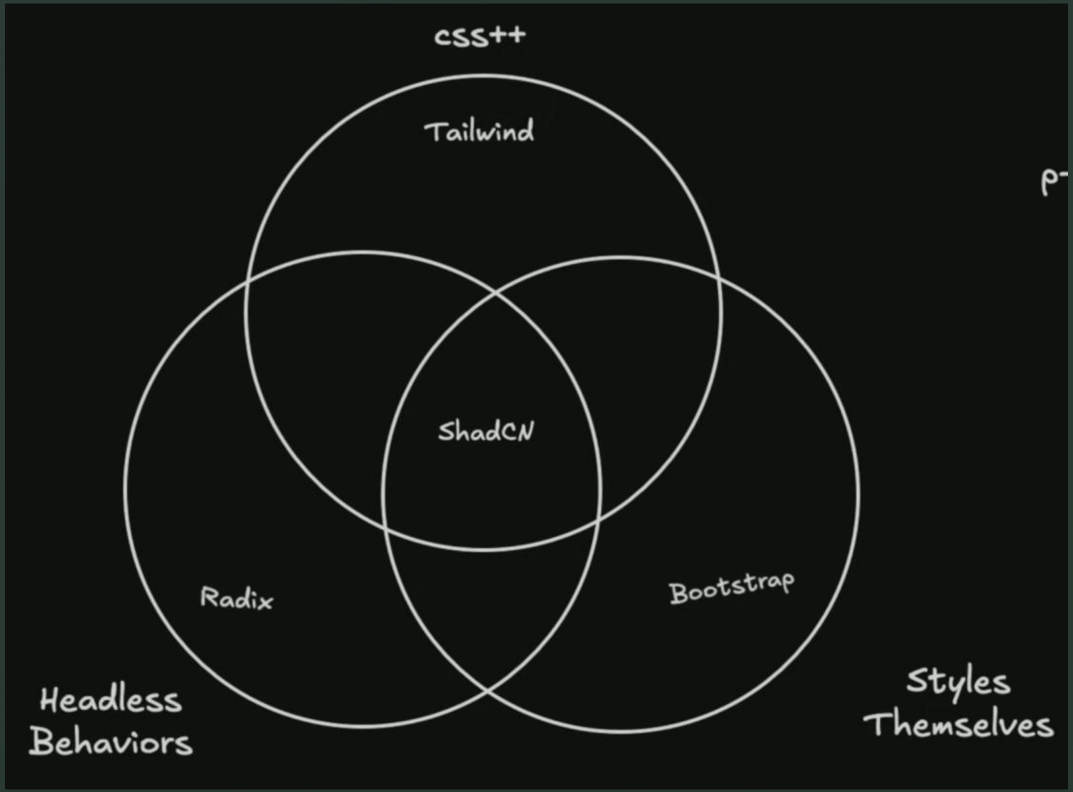
Learnings: Component Based Front End



- The skill floor and the skill ceiling of different styling options
- Higher floor options may also be highly opinionated and complex to modify
- If you hate front end, you can have packages do the heavy lifting
- If you have a design in mind, consider something else...

- https://www.youtube.com/watch?v=CQuTF-bkOgc - source Theo t3.gg

Learnings: What is Shadcn?



- https://ui.shadcn.com/
- Mildly opinionated component structure for maximum adaptability
- https://www.youtube.com/watch?v=lIUp8bdKiq4 -source Theo t3.gg

Future work

- Ability to have themes and change colors of the site
- Facebook login
- Page break for longer surveys
- Leaderboard:
  - Who has the most surveys
  - Who took the most surveys
- Coauthor on surveys

Summary

- Achievements:
  - Google sign in and authentication.
  - AI implementation for question generation.
  - Data visualization and export to csv.
- Result:
  - Functional survey creation and data collection website.

Thank You!

Distribution of Work

- Jonathan
  - UI
  - Survey replies
  - User dashboard
  - Bug fixes
- Minh
  - Google Auth
  - Setup of backend models
  - Survey results display
  - UI Components
- Elizabeth
  - Initial setup with Render, Backblaze, and Aiven.io (MySQL)
  - Profile
  - Create survey
  - Bug fixes