Shane Wilkerson

Cruz Arreola

Proposal -

Hi,

Imagine you're at a party or club, and instead of shouting song requests at the DJ, you just scan a QR code and instantly join a live music session with everyone else in the crowd. Our app lets guests queue songs in real time, kind of like a shared playlist for the event. Other guests can upvote songs they love, pushing them higher in the queue, while the DJ still keeps full control to skip or remove tracks if needed. It turns every event into an interactive experience, giving the crowd a voice while helping DJs read the room effortlessly. Whether it's a house party, wedding, or nightclub, we make music social, democratic, and fun.

Framework: swiftUI (we both used it for project 1 and we want to learn more about it)
Database: Firebase Firestore (seems to be good for real time data)
Version Control: Github

Features that we want:
- Create event
- Join event
- Submit request
- Upvote
- DJ moderating controls
- Real-time updates

Entities: admin (dj), users, events

Models: events, users, requests(?)

PREPARATION PHASE:

1. **User Profiles/Stories**

   Users: A user is anyone who attends an event, scans the QR code (or enters in code), and joins the session. They can add recommendations to the queue.

Goals:

- Request songs they want to hear.
- Upvote songs to influence the queue.
- See the live music queue update in real time.

DJ's: The DJ creates an event session and controls the music queue. They can approve, reorder, or remove songs to maintain flow and energy.

Goals:

- Create a new event session quickly.
- Display a QR code for guests to join instantly.
- View incoming requests in real time.

2. Features/Requirements

Users:

1. Join Session with QR Code
   - Guest scans a QR code generated by the DJ
   - Automatically joins the event
2. Submit Song Requests
   - Guests can search for a song
   - Request is added to the queue
3. Upvote Songs
   - Guests can upvote any song in the queue
   - Upvotes make requests go to the top of queue

DJ:

1. Create a New Event
   - DJ creates session
   - New event is made with unique ID
2. Display QR Code
   - App generates a QR code for that specific event
3. Manage Queue
   - DJ sees all incoming requests in real time
   - Can skip, remove, or reorder songs

3. Relational DB Schema/Diagram

How to run -

npm run dev

Folders where we edit -

App (layout.tsx, page.tsx, etc.)

Folders ive edited -

- Globals.css (this is where we will reuse all of our cards and text and backgrounds)
- models.ts (inside the lib folder inside app), this is where we make all of our models (

He might be looking for models and also controllers during presentation. We used a lot of these during labs so he will look for them

Pronounce it as jsx

Created models, home page.tsx, also made the global.css

- And then we made sub files for create and join, learned that it has to be named page.tsx in those subfiles.

Then we installed supabase, and added the supabaseClient.ts to the lib folder.

Then we created the .env.local. And updated it with our supabase URL and API key.

Okay, made the create-event API route in app/api/create-event/route.ts. Basically, this

- Validates input
- Inserts event into database
- Returns event ID so we can redirect the DJ

Then, we added those functions to the top of app/create/page.tsx.

Made adminQueue/[eventId]/page.tsx

- This file lives inside a folder named [eventId], which makes the URL dynamic
- any URL like /adminQueue/12345 will load this page
- 12345 becomes params.eventId
- The page reads params.eventId to know which event to load
- Receives the event ID from the URL automatically

- Fetches event data from Supabase

Made the joinerPage.tsx in joinerPage/[eventId]/page.tsx

- Loads event info by eventId
- Displays event name
- Shows requested songs list placeholder
- Button for request a song
- Guest-only version (no admin controls)

Okay, made the join-event API route in app/api/join-event/route.ts. Basically, this

- Accepts name and event code
- Looks up event by its code
- Creates a Guest row in Supabase
- Returns the event ID and guest ID
- Used on the Join Page

12-8

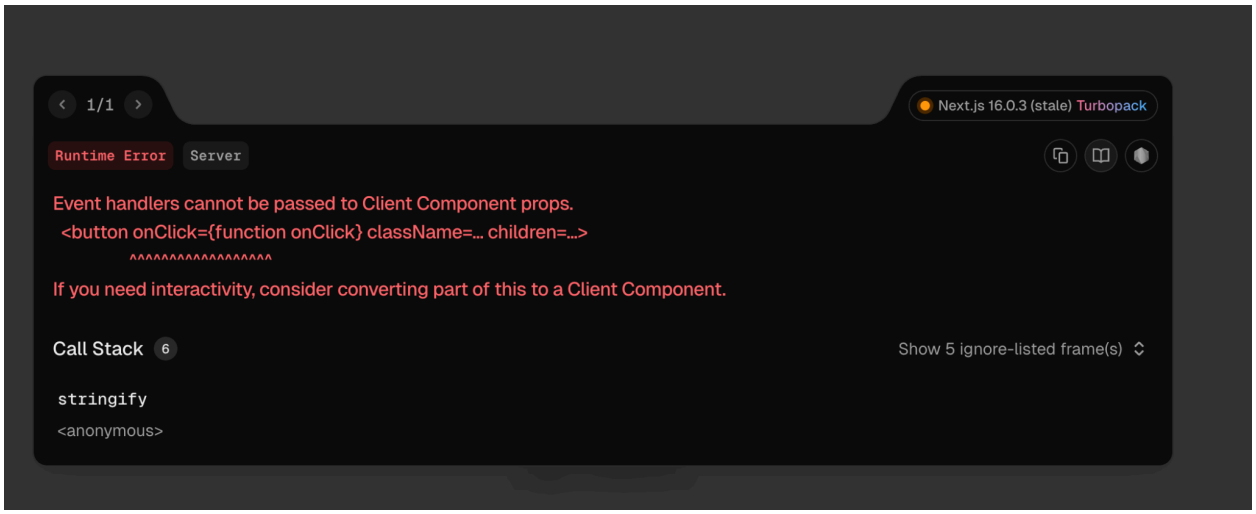
Ahhh yep — that error is 100% expected because:

Your adminQueue/[eventId]/page.tsx is a SERVER COMPONENT

…but we just added:

<button onClick={...}>

Server components cant use onClick
Any interactivity must be inside a client component



CRUZ

Created “create/join” initial pages

Added admin queue page to pop up for an admin after they create an event, still need to add actual queue display

Added a joiner home page with a request button to bring up a different page to request a song, also need to add other joiner’s requests (*stretch goal*: with like and dislike features)