house of edtech
WE BUILD BRANDS LIKE NO ONE ELSE COULD DO.

🎓 Fullstack Developer Assignment

(Intern / Junior Level)

Questa Lite – Simple Quiz Platform

🎯 Objective

Build and deploy **Questa**, a simple fullstack application where authenticated users can create quizzes and share them publicly. Other users can access and submit responses via a public link.

This assignment evaluates your foundational skills in fullstack development, backend API integration, authentication, and deployment.

🛠️ Recommended Tech Stack Requirements

- **Framework**: Next.js 15 (App Router) with TypeScript or any other framework like Remix, Tanstack Start/Tanstack Router.
- **Styling**: Tailwind CSS v4 (Use shadcn/ui for better UI.)
- **Database**: Postgres (Supabase, Render & Prisma), MongoDB
- **ORM:** Prisma (Recommended), Drizzle, Mongoose
- **Deployment**: Vercel, Netlify, Cloudflare Workers, or Render

```
# Using CRA or Vanilla React with Vite is prohibited.
# You own your code.
```

🔐 Authentication (Mandatory)

- Implement authentication using:
  - Better Auth (recommended), Supabase Auth , or any secure method like Auth.js, Firebase Auth, etc.

- Only **authenticated users can create, view and edit their quizzes**
- Public users can only view and respond to shared quizzes

Last Updated: May 16, 2025

🔧 Core Functional Requirements

1. User Authentication

- Signup & Login (email-based)
- Auth state persists between refreshes
- Show login/logout options in the UI

2. Quiz Creation (Authenticated Only)

- Authenticated users can:
    - Create a quiz with title, and at least 2 questions
    - Question types:
        - Single-choice (radio)
        - Short text answer
- Store quiz and questions in the database
- Show the public quiz URL after creation

3. Public Quiz Participation

- Public route `/quiz/[id]`
- Anyone (no login) can:
    - View the quiz
    - Submit answers
- Store responses in the database

4. View Submissions (Authenticated Creator Only)

- Logged-in users can see responses to their own quizzes
    - Response list with timestamp and answers

🧱 Backend Requirements

- Real API endpoints/Server Actions (Next.js route handlers)
- Store data securely in a real DB (no mock/local state)
- At minimum, handle:
    - Create quiz
    - Submit response
    - Get quizzes by user
    - Get responses by quiz

Last Updated: May 16, 2025

✅ Must-Haves

- Working **authentication** for quiz creation access
- Fully **live deployment** on Vercel / Netlify / Cloudflare Workers / Render
- GitHub repository with:
  - README.md explaining features, live URL, and setup steps
  - .env.example for env vars
  - Sample login credentials
- App footer must contain:
  - Your name
  - GitHub profile
  - LinkedIn profile

🌟 Bonus (Optional)

- Input validation (e.g. required fields, limits)
- Editable quizzes
- Basic charts for response summary
- CSV export of responses
- Use ShadCN UI for polished design

🧪 Evaluation Criteria

Area	Evaluation Focus
Functionality	Auth, quiz creation, public response, response viewing
UI/UX	Clean layout, intuitive flow, mobile-friendly
Code Quality	Modular file structure, clear logic, proper naming
Backend	Secure API, proper DB design, authentication checks
Deployment	Functional public app, easy to run from README

# Feel free to use any library and AI tools available on the internet.

# Make sure you are aware of the code you have written.

Last Updated: May 16, 2025

This assignment is intended for fullstack intern roles only.

house of edtech

WE BUILD BRANDS LIKE NO ONE ELSE COULD DO.

Last Updated: May 16, 2025