CASE STUDY 3: DISQO COLLECTIVE

EXECUTIVE SUMMARY: Design & implement a Consent-First, Accessible-by-Design Community Platform. DisQo Collective is a full-featured, consent-first digital home for Disabled, Queer, and Neurodivergent people working in movement building and advocacy. Before a single line of the platform has been promoted publicly, 100+ people have already registered on the waitlist, an organic signal of the trust and demand that exists for a platform built by and for this community.

WHAT MAKES THIS CASE STUDY AN EXAMPLE:

Amnesty's roadmap for the Humanity WordPress Theme includes three technical priorities that directly map to what C4AC Labs is building in DisQo: enhanced accessibility compliance at scale, improved CMS publishing and workflow capabilities, and major multilingual improvements. DisQo demonstrates that Kabir and Xai do not just maintain existing accessible, multilingual infrastructure but also design and architect it from scratch, making deliberate decisions at every layer about how accessibility, consent, and safety are built into the system rather than applied to it. We believe that this is the depth of technical judgment Amnesty's platform needs.

Objectives:

- Design and architect a consent-first community platform from first principles: build a full-featured social platform without adapting existing community frameworks, making deliberate architectural decisions at every layer to reject extractive, surveillance-based and algorithmically manipulative design patterns.
- Deliver a full-stack platform across a complex, custom architecture: Build end-to-end across a React frontend, Node.js and Express backend, and PostgreSQL database, with full ownership of every layer from component design to data storage.
- Implement role-based access controls and privacy-first data architecture and design and build a granular permissions system across five user tiers, ensuring member data is protected from surveillance by both other members and administrators.
- Build accessibility into the architecture; develop every component from onboarding flows to direct messaging to the resource library with WCAG 2.1 AA compliance, screen reader compatibility, keyboard navigation and plain language as structural requirements from the start.

Design and implement a proactive moderation infrastructure: Build a moderation system that intercepts and de-escalates harm before it reaches community members, with appeal workflows, transparent suspension notices and audit logs that protect both members and moderators.

- Deliver a performant, accessible landing page and waitlist infrastructure: Build and deploy a server-side rendered Next.js landing page with zero third-party tracking scripts, privacy-compliant waitlist management and notification workflows ahead of the full platform launch.
- Architect for scale and editorial independence: Structure the platform's backend and database to support a growing community, with admin-facing analytics that surface platform health data without exposing individual user behaviour.
- Build for a marginalized community with specific safety requirements: Deliver a platform designed by and for Disabled, Queer and Neurodivergent people, where every technical decision from consent flows to notification systems, reflects the safety and dignity needs of the community it serves.

Approach:

It is being built entirely by C4AC Labs: designed, architected, and developed from first principles as a feminist and disability-affirmative social media platform, with no precedent in existing platforms to borrow from.

Every architectural decision has been made in explicit rejection of the extractive, surveillance-based, algorithmically manipulative design patterns that define mainstream social media.

Technical Architecture:

The **landing page and waitlist infrastructure** is built on **Next.js**, server-side rendered, optimized for performance and accessibility, with a backend handling waitlist signups, notification workflows, and privacy-compliant data storage. The landing page already implements the platform's core design principles: accessible typography, keyboard navigation, WCAG 2.1 AA compliance, and zero third-party tracking scripts.

The **full platform** is being built on **React**, a custom, ground-up architecture rather than an adaptation of an existing community platform framework. This was a deliberate technical decision: no existing platform (Discourse, Circle, Mighty Networks, or otherwise) meets DisQo's requirements for consent-first interaction design, anti-algorithmic content delivery, moderation that protects rather than punishes, and accessibility-by-default at the component level.

Building on React gives the team full architectural control over every layer, from how consent flows are triggered to how notification systems work to how user data is stored and surfaced.

IMPLEMENTATION

The platform architecture includes role-based permissions across five user tiers (Super Admin, Moderators, Verified Members, Pending Members, and Suspended Users), with granular access controls designed to prevent surveillance of members by other members and administrators alike. Core features include solidarity circles, a mutual aid hub, a community resource library, consent-based direct messaging (opt-in, not default-on), community event management, and privacy-first analytics that give administrators platform health data without exposing individual user behavior. The moderation system is architecturally distinct from mainstream platforms: rather than reactive content removal after harm has occurred, DisQo's moderation infrastructure is designed to intercept and de-escalate before harm reaches members, with appeal workflows, transparent suspension notices, and audit logs that protect both community members and moderators from arbitrary decision-making.

Accessibility as architecture, not feature

WCAG 2.1 AA compliance is the floor. Every component being built, from the onboarding flow to the DM interface to the resource library, is developed with screen reader compatibility, keyboard navigation, sensory considerations, and plain language as structural requirements. Accessibility is not being tested at the end of the build but designed into the component library from the start, which is both the harder and the more honest way to build.