ALX-Polly: Next.js Polling Application

Remaining Features & Enhancements

- Poll Analytics & Result Charts
- User Roles (Admin/Regular)
- Comments/Threads on Polls
- Email Notifications (poll closing alerts, password reset)
- QR Code Generation for Polls
- Unit & Integration Tests (Jest, React Testing Library)
- AI-powered Reviews & Automated Release Notes (CodeRabbit)
- Mobile-Responsiveness & Accessibility Improvements

Technology Stack

- **Frontend:** Next.js 14, React 18, TypeScript
- **Styling:** Tailwind CSS, Shadcn/ui
- **Backend/DB:** Supabase (Postgres, Auth, Realtime)
- **Testing:** Jest, React Testing Library
- **DevOps:** GitHub, CodeRabbit (AI PR reviews)
- **AI Integration:** GitHub Copilot (note: while AI Native IDEs like Cursor or Trae are popular, Copilot will be used for this project)
- **Schema Awareness:** MCP servers for context-driven code generation

4-Day Feasible Timeline (AI Assisted)

Day	Tasks
1	Poll analytics, charts, user roles, comments
2	Email notifications, QR code generation, mobile/accessibility
3	Write and run tests, optimize code, integrate AI review (CodeRabbit)
4	Finalize docs, polish UI/UX, deploy

AI Integration Plan

- **Prompting Strategy:** Use GitHub Copilot for code generation, refactoring, and documentation. Integrate MCP servers for schema awareness and context-driven prompts, ensuring generated code aligns with database and business logic.
- **Database Migrations:** Use Supabase migrations for schema changes; update TypeScript types accordingly.
- **Business Logic:** Implement in Next.js API routes and React components; use Supabase for CRUD and real-time features.
- **Testing:** Generate and run unit/integration tests with Copilot and Jest.

- **Code Review:** Use CodeRabbit for automated PR reviews, release notes, and optimization suggestions.
- **Continuous Optimization:** Prompt Copilot (with MCP context) for performance, accessibility, and security improvements after each feature.
- **Deployment:** Use Vercel or similar for fast CI/CD; automate deployment steps.

Core Flow

- User Authentication
  - Register, login, email verification, password reset (Supabase Auth)
  - Auth context manages session and access
- Poll Lifecycle
  - Create poll: Authenticated users submit questions, options, end date
  - Browse polls: Filter by active/ended, public/private, user's own polls
  - Vote: Authenticated users select options, one vote per poll enforced
- Role-Based Access
  - Admins: Manage all polls, moderate comments, view analytics
  - Regular users: Create, vote, comment on polls
- Comments & Threads
  - Users add comments/discussion to polls
  - Moderation by admins
- Analytics & Charts
  - Real-time vote counts, poll result charts (charting library)
  - Insights for poll creators and admins
- Notifications
  - Email alerts for poll closing, password reset, registration
- QR Code Generation
  - Each poll has a QR code for easy sharing
- Protected Routes
  - Only authenticated users can access poll creation, voting, commenting
- Mobile & Accessibility
  - Responsive UI, accessible components, optimized for all devices

---

Data & API Integration

- Supabase Database
  - Tables: users, polls, poll_options, votes, comments
  - Row Level Security for data protection
- Next.js API Routes
  - CRUD operations for polls, votes, comments
  - Business rules enforced server-side
- Real-Time Updates
  - Supabase subscriptions for live vote and comment updates

Orchestration

- Context Providers
  - Auth, poll, and comment contexts for state management
- Business Logic Layer
  - Centralized validation, permission checks, and workflow orchestration

- **Testing & Review**
    - **Jest/React Testing Library for logic validation**
    - **CodeRabbit for AI-powered PR review and release notes**