Indy YSA Website Documentation

Project Overview

Modern, responsive React website for Indianapolis Young Single Adults (YSA) ward events. Features Firebase real-time data, performant components, and mobile-first design deployed to GitHub Pages. It provides announcements, flyers, events, a contact directory, external YSA links, and an interactive calendar.

Live Site: https://pinkarmy10.github.io/Indy-Stake-YSA/

Quick Start

npm install
npm start # Development: http://localhost:3000
npm run build # Production build
npm run deploy # Deploy to GitHub Pages

Core Features

Feature	Technology	Status
Responsive Navigation	React Router v7 + useLocation	✅ Live
Real-time Calendar	Firebase Firestore + react-big-calendar	✅ Interactive
Live Voting Poll	Firebase real-time updates	✅ Multi-user
Contact Form	Firebase Firestore	✅ No backend
Event Cards	Accordion components	✅ Touch-optimized
Image Carousel	Auto-play slider	✅ Responsive

Project Architecture

The application uses a component-based structure with React Router for navigation and Firebase for real-time data management. The main entry point is `App.js`, which sets up routing through the Layout wrapper component.

Directory Structure

- src/

    - components/ - Reusable UI components (Layout, Header, Footer, Accordion, Carousel)
    - pages/ - Route-specific page components (home, events, calendar, contact)
    - firebase.js - Centralized Firebase configuration
    - App.js - Router and Routes setup
    - App.css - Comprehensive styling (7k+ lines)
    - Layout.js - Responsive navigation wrapper
- public/
    - Images/ - holds all the images of the project
    - Index.html – is the base page
- .gitignore – stops the upload of certain files
- README.md – Holds documentation of site for everyone to see

Key Components

Layout.js - Responsive Navigation

Purpose: Provides responsive mobile-first navigation wrapper for entire application.

Hooks Used:

- `useState` - Manages mobile menu open/close state
- `useEffect` - Auto-closes menu on route change
- `useLocation` - Detects current route from React Router

Props: None (uses context from React Router)

Key Features:

- Mobile hamburger menu (☰) that toggles `isOpen` state
- CSS class conditional rendering: `className={isOpen ? "isOpen" : ""}`
- Automatic menu closure on route navigation via `useEffect([location.pathname])`
- Clean `<Link>` navigation to Home, Events, Calendar, Contact pages

- Outlet component renders nested page content

Performance Note: useEffect dependency on `location.pathname` triggers immediately on route change, preventing mobile menu from staying open across pages.

FirebaseCalendar.js - Interactive Event Calendar

Purpose: Displays YSA events in interactive month/week/agenda calendar views with real-time Firebase sync.

Libraries: react-big-calendar, moment.js

Hooks Used:

- `useState` - Manages local events array and loading state
- `useEffect` - Sets up real-time Firebase listener via `onSnapshot`

Data Flow:

1. Component mounts → `useEffect` subscribes to Firestore "events" collection
2. Firebase `onSnapshot` listener converts Firestore Timestamp to JavaScript Date
3. Events update UI in real-time across all users
4. Click slots to add events → `addDoc` to Firestore
5. Click existing event → `deleteDoc` or update title

Firestore Structure:

Field	Type
id	string (document ID)
title	string
start	Timestamp
end	Timestamp
allDay	boolean

Responsive Design: Calendar height set to 80vh (viewport height) for mobile and desktop.

FirebasePoll.js - Real-time Voting System

Purpose: Displays interactive poll with live vote counts across all users.

Hooks Used:

- `useState` - Manages poll data and loading state
- `useEffect` - Subscribes to real-time poll updates

Firestore Path: `polls/mainPoll`

Data Structure:

Field	Type
question	string
options	array of objects

Performance Optimization:

- Uses `updateDoc` for atomic vote increments
- Single `onSnapshot` listener prevents multiple subscriptions
- Progress bar calculations: `percentage = (votes / totalVotes) * 100`
- Compact design: 300px height with no scrolling

User Experience:

- Click vote button → increment votes → `updateDoc` to Firestore
- All users see votes update instantly via real-time listener
- Progress bars show percentage of total votes
- Vote buttons remain clickable (no disable after vote)

EventsCard.js - Event Display Component

Props Interface:

Prop	Type	Purpose
title	string	Event name
date	string/Date	Event date/time
description	string	Event details
id	string/number	Unique identifier

Features:

- Card-based layout with consistent spacing
- Pairs with Accordion.js for expandable details
- Touch-optimized with 44px+ target areas
- Responsive: stacks on mobile, 2-3 columns on desktop

Accordion.js - Expandable Content Component

Purpose: Reusable expandable/collapsible container for event details.

Hooks Used:

- `useState` - Manages expanded/collapsed state

Props: `children` (ReactNode)

Features:

- Click header to expand/collapse
- Smooth CSS transitions
- Accessible with semantic HTML
- Optimized for mobile touch

Performance Optimizations

The application implements several performance strategies to ensure smooth user experience across devices:

React Hook Optimization

Hook Pattern	Implementation	Benefit
useEffect Dependencies	`[location.pathname]`	Mobile menu closes instantly
Real-time Listeners	Single `onSnapshot` per component	No polling overhead
useState	Local state only	No prop drilling

Firebase Optimization

- Single `onSnapshot` listener per collection prevents duplicate subscriptions
- `updateDoc` with merge strategy prevents data loss
- Firestore timestamp conversion to Date only when needed
- Collection-based queries prevent fetching entire database

CSS and Rendering

- `object-fit: cover` for images prevents layout shifts
- CSS transforms for menu animations (GPU accelerated)
- Conditional rendering based on loading state
- Minimal re-renders via React Router v7

Calendar Virtualization

react-big-calendar includes built-in virtualization that:

- Renders only visible calendar cells
- Handles 1000+ events smoothly
- Adjusts for different month/week/agenda views

Responsive Design Strategy

The application follows mobile-first design principles with breakpoints for tablet and desktop:

Mobile (< 768px)

- Hamburger menu navigation
- Single-column stacked cards
- 16px+ touch targets
- Full-width calendar and forms
- Vertical carousels

Tablet (768px - 1024px)

- Expanded horizontal navigation
- 2-column event card layout
- Medium-width calendar
- Touch-friendly spacing maintained

Desktop (> 1024px)

- Full horizontal navigation bar
- 3-column event layout
- Full-width calendar views
- Optimized for mouse/trackpad
- Hero sections with larger typography

Technology Stack

Frontend Framework

Library	Version	Purpose
React	19.2.1	Component framework
React Router	7.10.1	SPA routing
Firebase	12.6.0	Real-time database
react-big-calendar	1.19.4	Interactive calendar
moment.js	Latest	Date/time parsing

Table 5: Core frontend dependencies

Styling

- App.css - 7000+ lines of custom utilities and component styles
- TailwindCSS - Utility-first CSS framework
- PostCSS - CSS processing with autoprefixer
- CSS Grid/Flexbox - Modern layout patterns

Deployment

- GitHub Pages - Static hosting via gh-pages branch

- npm scripts - Automated build and deploy workflows
- react-scripts - Create React App build configuration

Firebase Setup and Configuration

Project Requirements

1. Create Firebase project at https://console.firebase.google.com/
2. Enable Firestore database
3. Configure security rules (testing: allow read/write)
4. Copy config to `src/firebase.js`

Required Collections

Collection: events

Documents contain event details for calendar display:

- `title` (string) - Event name
- `start` (Timestamp) - Start date/time
- `end` (Timestamp) - End date/time
- `allDay` (boolean) - All-day event flag

Collection: polls

Document: `mainPoll` contains poll data:

- `question` (string) - Poll question
- `options` (array) - Vote options with id, label, votes

Collection: contacts

Messages from contact form:

- `name` (string) - Visitor name
- `email` (string) - Contact email
- `message` (string) - Message content
- `timestamp` (Timestamp) - Submission time

Deployment Workflow

Local Development

npm install # Install dependencies
npm start # Start dev server at http://localhost:3000

Production Build

npm run build # Creates optimized build in /build directory

Deploy to GitHub Pages

npm run deploy # Runs build + pushes to gh-pages branch

After deploy, site is live at: https://pinkarmy10.github.io/Indy-Stake-YSA/

Deployment Scripts (package.json)

Script	Action
npm start	Development server with hot reload
npm run build	Production build optimization
npm run test	Jest test runner
npm run deploy	Build + gh-pages deployment

Development Guidelines

Component Creation Checklist

1. Use functional components with hooks
2. Destructure props at function signature
3. Keep components < 300 lines
4. Extract custom hooks for shared logic
5. Add prop validation if complex
6. Use React.memo for expensive re-renders
7. Handle Firebase errors with try/catch

Firebase Best Practices

1. One `onSnapshot` listener per component
2. Clean up listeners in useEffect cleanup
3. Use `updateDoc` for atomic updates
4. Validate data before Firestore writes
5. Handle loading and error states
6. Use specific document paths (no wildcards)

Performance Checklist

1. Check React DevTools Profiler for re-renders
2. Use Chrome DevTools Performance tab
3. Monitor Firebase read/write operations
4. Test on actual mobile devices
5. Use lighthouse for page speed metrics
6. Optimize images before deployment

Troubleshooting

Common Issues

Mobile menu stays open after navigation: Check `useEffect([location.pathname])` in Layout.js - should trigger setIsOpen(false)

Firebase data not loading: Verify Firestore security rules allow reads, check Firebase config in firebase.js, inspect Network tab for errors

Calendar events showing NaN: Ensure Firestore timestamps convert properly - use `data.start?.toDate()` with optional chaining

Poll votes not updating: Use `updateDoc` not `setDoc` - prevents overwriting entire document

References

- React 19 Hooks: https://react.dev/reference/react/hooks
- React Router v7: https://reactrouter.com/
- Firebase Firestore: https://firebase.google.com/docs/firestore
- react-big-calendar: https://jquense.github.io/react-big-calendar/
- GitHub Pages Deployment: https://create-react-app.dev/deployment/github-pages/
- Help when things break and getting calendar and firebase integration: Perplexity AI Assistant

---