

Resilient Live Polling System

Live Link: <https://livepollingsystem-kappa.vercel.app>

Project Overview

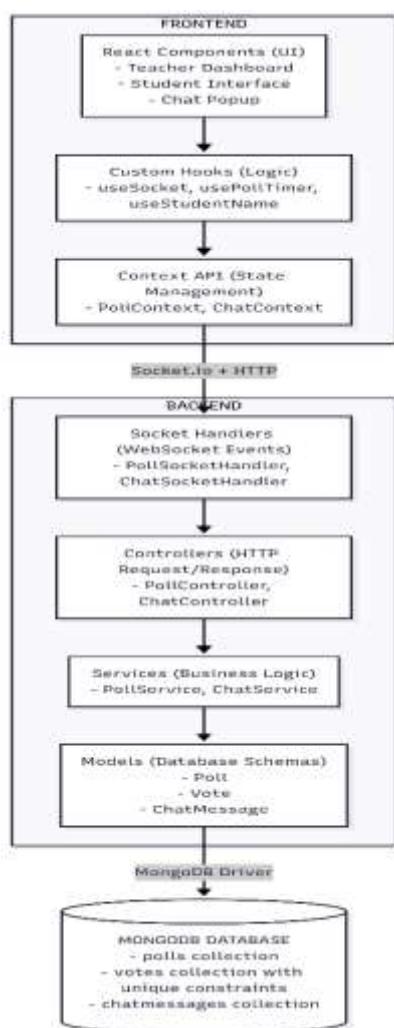
The **Resilient Live Polling System** is a real-time web application that enables teachers to create and manage live polls while students participate in real-time. The system is designed with resilience as a core principle, ensuring that users can recover from page refreshes, network interruptions, and late joins without losing state or data.

Key Differentiators

- **State Recovery:** Survive page refreshes during active polls
- **Timer Synchronization:** Server-authoritative time with client sync
- **Race Condition Prevention:** Database-level duplicate vote prevention
- **Real-time Communication:** Socket.io for instant updates
- **Persistent Storage:** MongoDB for poll history and chat

System Architecture

Architecture Pattern: Clean Architecture + MVC



Backend Architecture Principles

1. Separation of Concerns

- **Models:** Only define data schemas, no logic
- **Services:** All business logic and database operations
- **Controllers:** Only handle HTTP request/response
- **Socket Handlers:** Only handle WebSocket events, delegate to services

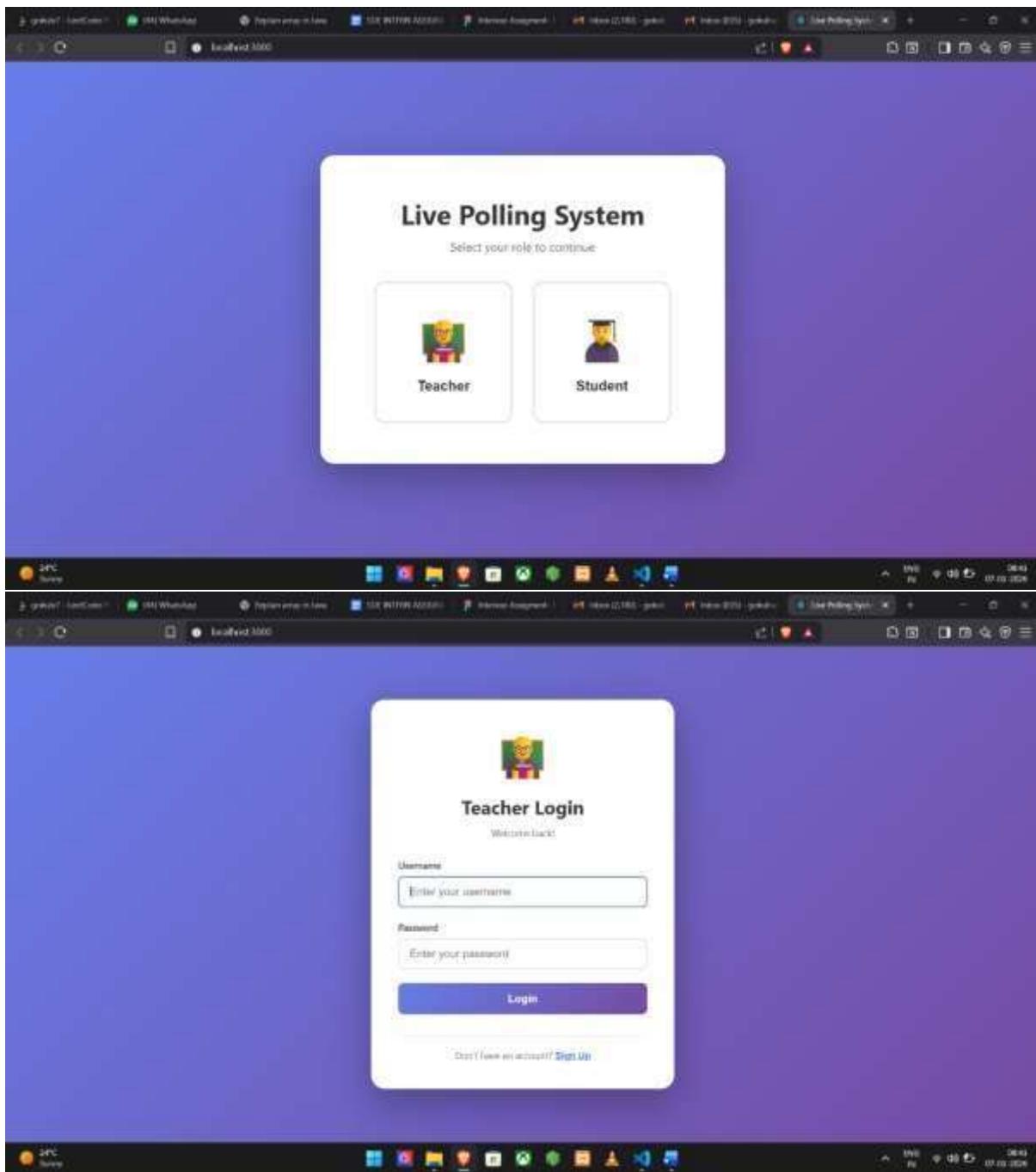
2. Single Responsibility

- PollService.js: Create polls, record votes, fetch results
- PollController.js: Parse HTTP requests, validate, respond
- PollSocketHandler.js: Manage WebSocket connections, emit events

3. DRY (Don't Repeat Yourself)

- Services are shared between HTTP controllers and Socket handlers
- No duplicate business logic

Teacher Persona



1. Poll Creation

- Create question with text input
- Add 2-6 options dynamically
- Set timer duration (10-300 seconds)
- Validation: Question and at least 2 options required

The screenshot shows the 'Teacher Dashboard' interface. At the top, there are three tabs: 'Create Poll' (highlighted in green), 'Live Results', and 'Poll History'. Below the tabs, the main area is titled 'Create New Poll'. It contains fields for 'Question' (with placeholder 'Enter your question...'), 'Options' (with two input fields labeled 'Option 1' and 'Option 2'), a button '+ Add Option', and a 'Duration (seconds)' field set to '60'. The background of the dashboard is purple.

2. Live Dashboard

- Real-time vote counts per option
- Percentage calculation
- Visual progress bars
- Auto-updates via Socket.io

The screenshot shows the 'Teacher Dashboard' with the 'Live Results' tab selected. The main content area displays a poll question: 'A person lent a personal loan at 8.5% per annum at simple interest. In 20 years, the interest amounted to ₹6,300 more than the loan lent. What was the sum lent?'. Below the question, there is a timer showing '0:52'. The results table shows three options: '₹ 9000' (0 votes, 0%), '₹ 8500' (0 votes, 0%), and '₹ 10000' (0 votes, 0%). The background of the dashboard is purple.

3. Poll History

- Fetch past polls from MongoDB

- Display question, options, final results
- Timestamps (created, ended)
- No local storage - all from database

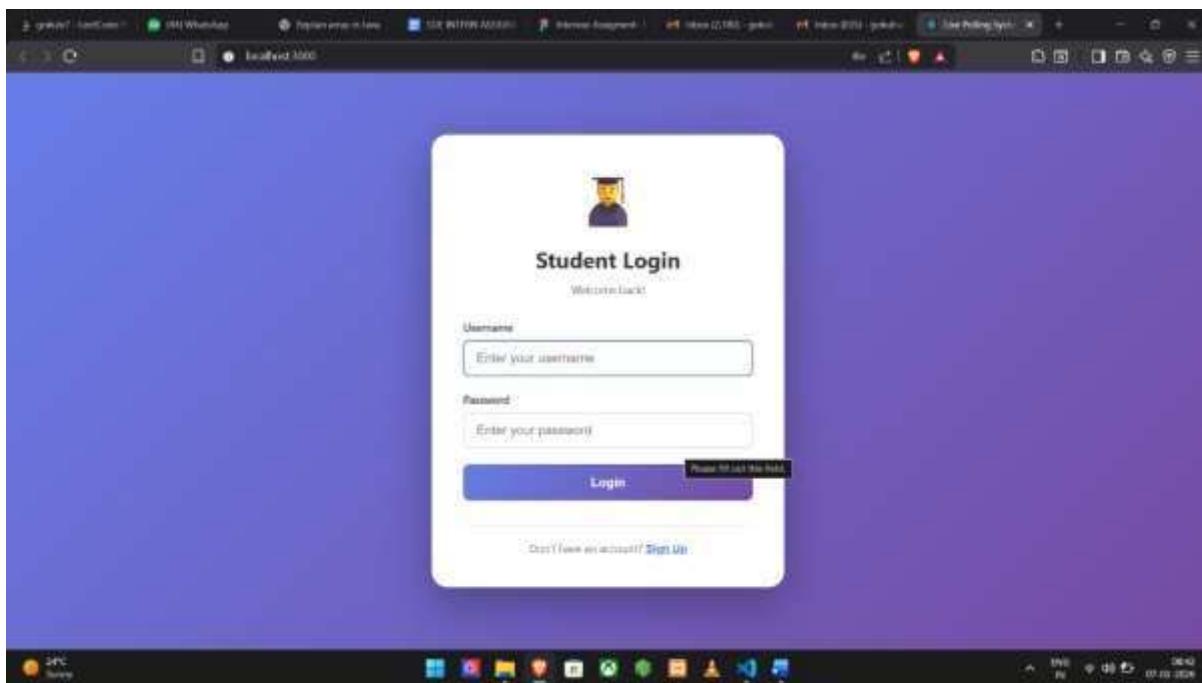
The screenshot shows a web browser window titled "Teacher Dashboard" with a purple header bar. The header includes "Welcome, author", "Logout", and three buttons: "Create Poll", "Live Results", and "Poll History". The "Poll History" button is highlighted. Below the header, the page title is "Poll History". A math question is displayed: "A person lent a personal loan at 8.5% per annum at simple interest. In 20 years, the interest amounted to ₹6,300 more than the loan lent. What was the sum lent?". Below the question, it says "1/1/2016, 8:09:04 AM Total Votes: 1". Three options are listed with their percentages:

- ₹ 9000: 0.0%
- ₹ 8500: 100.0%
- ₹ 10000: 0.0%

The screenshot shows a second instance of the "Teacher Dashboard" with a purple header bar. The header includes "Welcome, author", "Logout", and three buttons: "Create Poll", "Live Results", and "Poll History". The "Poll History" button is highlighted. Below the header, the page title is "Poll History". A geography question is displayed: "which is the longest Country?". Below the question, it says "1/1/2016, 8:09:04 AM Total Votes: 1". Three options are listed with their percentages:

- China: 0.0%
- India: 0.0%
- Russia: 100.0%

Student Persona

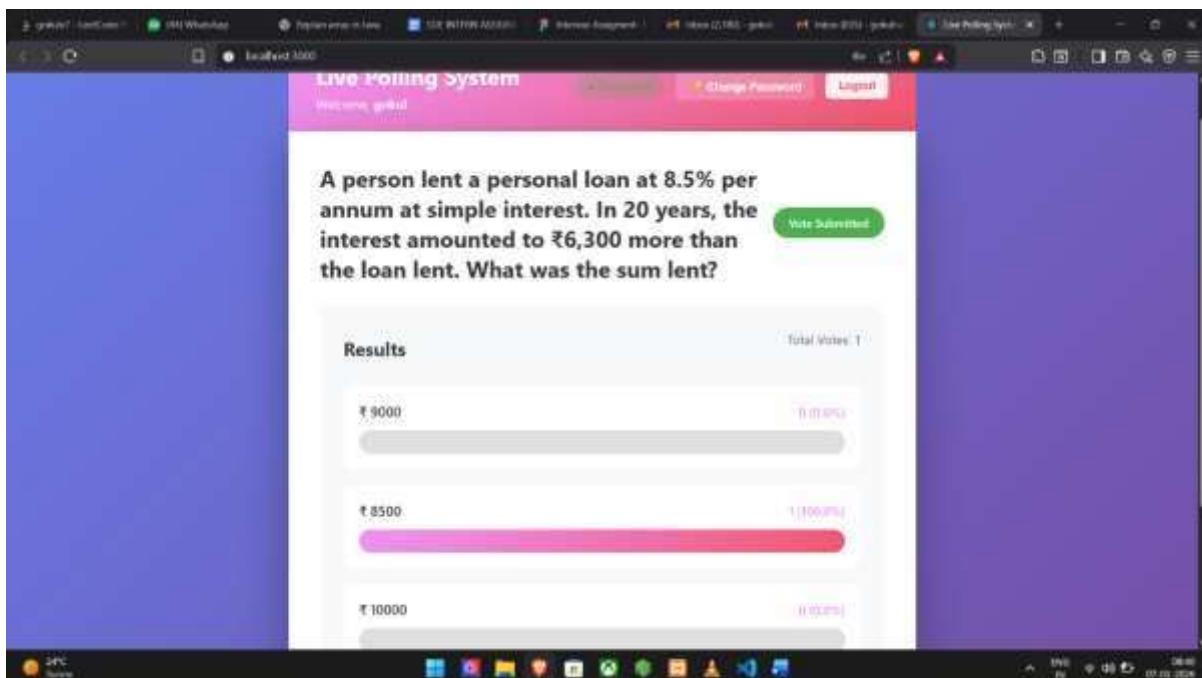


1. Onboarding

- Enter name on first visit
- Unique per browser tab (sessionStorage)
- Name persists until tab closes

2. Real-time Interaction

- Receive poll instantly when teacher creates it
- See all options
- Submit vote within time limit



- Late joiners see accurate countdown
- Example: 60s poll, join at 10s → shows 50s

The screenshot shows a web-based application titled "Live Polling System". At the top, there are buttons for "Logout", "Change Password", and "Logout". Below the title, it says "Welcome, gopal". A math problem is displayed: "A person lent a personal loan at 8.5% per annum at simple interest. In 20 years, the interest amounted to ₹6,300 more than the loan lent. What was the sum lent?". To the right of the question is a timer showing "0:19". Below the question are four answer options, each with a radio button:

- ₹ 9000
- ₹ 8500
- ₹ 10000
- ₹ 12000

4. Voting

- One vote per poll per student
- Immediate feedback on submission
- Cannot vote after time expires

The screenshot shows the same "Live Polling System" interface. The math question and options from the previous screenshot are no longer visible. Instead, there is a large white area with a small hourglass icon in the center. Below the icon, the text "Waiting for teacher to start a poll..." is displayed.

Resilience Features

1. State Recovery

Scenario: Teacher refreshes browser during active poll

Expected Behavior:

- Poll remains active on server
- On refresh, frontend fetches current poll state via `/api/polls/current`
- UI resumes showing live results
- Timer continues from actual remaining time

2. Timer Synchronization

Scenario: Student joins 30s into a 60s poll

Expected Behavior:

- Server stores poll creation time in MongoDB
- Server calculates: $\text{remainingTime} = \text{duration} - (\text{now} - \text{createdAt})$
- Student receives remainingTime : 30 seconds
- Client countdown starts from 30s, not 60s

3. Race Condition Prevention

Scenario: Student spams vote button or opens dev tools to call API multiple times

Expected Behavior:

- First vote succeeds and is recorded
- Subsequent votes fail with error
- Final results show only 1 vote

Last Updated: January 11, 2026

Version: 1.0.0

