

INTERVUE.IO - SDE INTERN ROLE ASSIGNMENT - ROUND 1

Dear Candidate,

Thank you for your interest in the **SDE Intern** position at Interview.io.

As part of the evaluation process, we are sharing the details for **Assignment Round 1**. This task will help us assess your **problem-solving ability, technical skills, and alignment with the role expectations**.

Please read the instructions carefully and follow the **provided design layout and guidelines**. This will ensure your submission is evaluated fairly and efficiently.

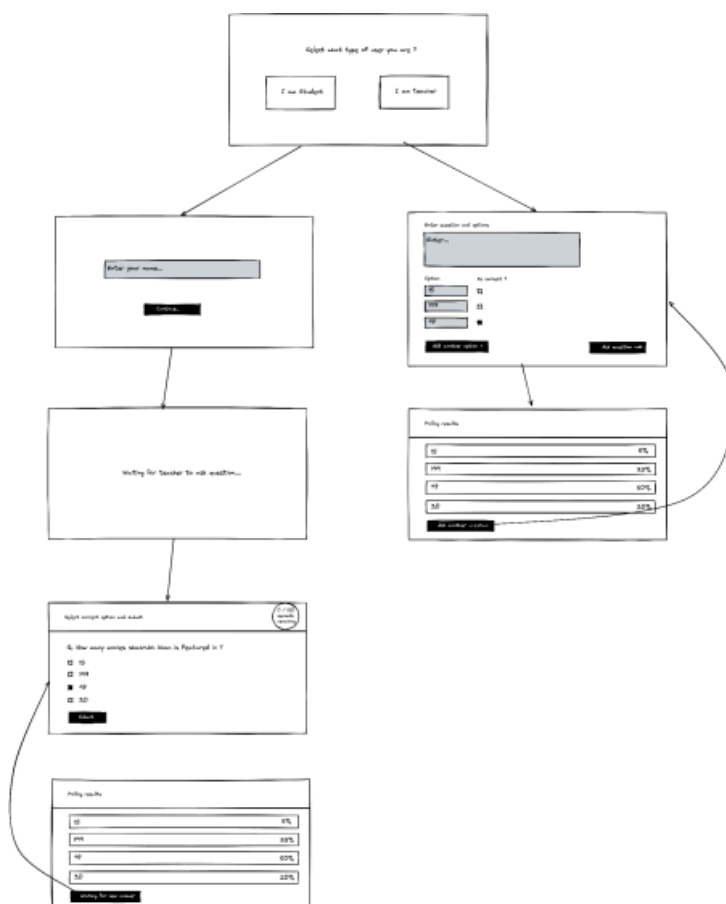
We recommend reviewing this document thoroughly before you begin. If you have any questions, feel free to reach out to us via email.

We kindly request that every candidate read through this [DOCUMENT](#) first. It covers most of the important details and will help clear up any doubts. If you still have questions afterwards, don't hesitate to drop us an email. We'll be more than happy to help!

We kindly request you not to attempt the assignment if you have participated in our hiring process within the last 6 months and were not shortlisted, as per our policy candidates are eligible to reapply only after 6 months from their previous application date.

Assignment Details

Title: Live Polling System



Design Reference

- **Design Link:**
<https://www.figma.com/design/uhinheFgWssbxvII7wtf59/Intervue-Assignment--Poll-system?node-id=0-1&t=Y5>
- **Design Preview:** *(Insert design image screenshot here — ideally from the Figma file for quick reference)*

You are required to create a **"Resilient Live Polling System"** with two personas: **Teacher** and **Student**.

Unlike a basic todo-list app, this system must handle state recovery. If a teacher refreshes their browser mid-poll, the poll should not disappear. If a student joins 30 seconds late to a 60-second question, their timer must start at 30 seconds, not 60.

Technology Stack

- **Frontend:** React.js (Hooks required; Redux/Context API optional but preferred).
 - **Backend:** Node.js with Express.
 - **Real-time Communication:** Socket.io.
 - **Database:** MongoDB or PostgreSQL (Required for persistence).
 - **Languages:** TypeScript
-

Functional Requirements

1. Teacher Persona (Admin)

- **Poll Creation:** Ability to create a question with options and a timer duration (e.g., 60 seconds).
- **Live Dashboard:** View real-time updates as students submit votes (e.g., "Option A: 40%, Option B: 60%").
- **Poll History (DB Integration):** View a list of previously conducted polls and their final aggregate results, fetched from the database.
- Create a new poll
- View live polling results
- Ask a new question only if:
 - No question has been asked yet, **or**
 - All students have answered the previous question

2. Student Persona (User)

- **Onboarding:** Enter a name on the first visit (unique per session/tab).
- **Real-time Interaction:** Receive the question instantly when the teacher asks it.
- **Timer Synchronization:** The timer must remain in sync with the server.
 - *Scenario:* If the time limit is 60s and a student joins 10 seconds late, their timer must show **50s**, not 60s.
- **Voting:** Submit an answer within the time limit.
- Enter name on first visit (unique to each tab)

- Submit answers once a question is asked
- View live polling results after submission
- Maximum of 60 seconds to answer a question, after which results are shown

3. System Behavior (The "Resilience" Factor)

- **State Recovery:** If the Teacher or Student refreshes the page during an active poll, the application must fetch the current state from the backend and resume the UI exactly where it left off.
 - **Race Conditions:** Ensure that a student cannot vote more than once per question, even if they spam the API or manipulate the client-side code.
-

Code Quality & Architecture Standards

We place a high value on clean, maintainable code. Your submission will be evaluated on the following strict criteria:

1. Backend Architecture (Separation of Concerns)

- Do not write business logic directly inside Socket listeners or Express routes.
- Use a **Controller-Service** pattern.
 - *Example:* `PollSocketHandler.js` handles the connection, while `PollService.js` handles the logic and DB interaction.

2. Frontend Architecture

- Use **Custom Hooks** (e.g., `useSocket`, `usePollTimer`) to separate logic from UI components.
- Implement **Optimistic UI** updates where appropriate (UI updates immediately, reverts on error).

3. Error Handling

- The app should not crash if the database is temporarily unreachable.
- Provide user feedback (toasts/alerts) for connection errors or submission failures.

4. Data Integrity

- Use the Database to persist Polls, Options, and Votes.
- Ensure the server is the "Source of Truth" for the timer and vote counts.

Must-Have Requirements

- Functional system with all core features working
- Hosting for both frontend and backend
- Teacher can create polls and students can answer them
- Both teacher and student can view poll results
- **Please ensure the UI in your assignment submission follows the shared Figma design without any deviations.**

Good to Have

- Configurable poll time limit by teacher

- Option for teacher to remove a student
- Well-designed user interface
- System behavior (**The "Resilience" Factor**)

Bonus Features (Brownie Points)

- Chat popup for interaction between students and teachers
 - Teacher can view past poll results (not stored locally)
-

Submission Guidelines

- **Deadline:** As per email instruction
- **Submission Method:** Email only (*Submissions in any other format will not be considered*)
- **Email To:** pallavi@interview.io
- **Email Subject:** SDE INTERN ASSIGNMENT SUBMISSION

Name: [Your Full Name]

Phone Number: [Your Contact Number]

Email ID: [Your Email Address]

LinkedIn URL: [Your LinkedIn Profile Link]

Codebase Link: [Direct codebase Link of the assignment]

Assignment Link: [Hosted/Deployed Link]

ATTACH CV

Please review our company socials attached below for a better understanding:

- **Website-** <https://interview.io>
- **LinkedIn-** <https://www.linkedin.com/company/interview-io/>

Good luck with the assignment.