Vanesha Richardson
Preparing For Your Capstone: Step 2-Project Proposal

## Project Proposal 1: Classroom Job Manager App

The goal of this project is to create an intuitive and efficient digital platform that helps teachers assign, manage, and provide feedback on classroom jobs. The application will ensure fairness by randomly assigning jobs to students while preventing repeats until all students have experienced each role. Each job assignment will last for two weeks, with two substitute students available in case of absences. At the beginning of each cycle, teachers will be able to display the assigned jobs on a device for students to see. The system will also allow teachers to provide feedback during and after job completion, fostering accountability and skill development.

## Goal & Description

**Type:** A website that can be accessible from any mobile device, not a mobile app

**Goal:** helps teachers assign, manage, and provide feedback on classroom jobs. The system will ensure fairness by randomly assigning jobs to students while preventing repeats until all students have experienced each role at least once within a school year.

**Users:** Elementary School teachers (Middle school and high schools have the option to use the app.)

## Data

**Teacher & Classroom Data**

-**Teacher Account Information** (name, email, password for authentication)
-**Classroom Name** (to manage multiple classes if needed)
-**Number of Students in the Class**

**Student Data**

-**Student Names or class roster** (or unique identifiers)
-**Student Participation History** (to track who has completed which jobs)
-**Substitute Status** (identifies students designated as substitutes in case of absences)

**Job Assignment Data**

-**List of Jobs** (job titles, descriptions, and responsibilities)
-**Assignment Records** (which student was assigned to which job and for what dates)
-**Rotation Tracking** (to ensure jobs are fairly distributed and prevent repeats before all students have experienced each job)
-**Job Start & End Dates** (for automation of new assignment 2 week cycles)

**Absence & Substitute Data**

-**Absence Marking System** (so teachers can mark absent students)
-**Substitute Activation** (to ensure substitute students take over when needed)

**Feedback & Performance Tracking**

**-Teacher Feedback on Student Performance** (comments, ratings, or checkboxes for completion)

**-Student Reflections** (optional—students can self-reflect on what they learned from the job)

**Display & Interaction Data**

**-Job Display Settings** (whether assignments are visible on student devices or only on the teacher's device)

-**Randomization Preferences** (allow teachers to customize the random selection process)

**Potential Existing APIs to Use**

**Backend & Data Management**
*Instead of building custom APIs, I will integrate third-party services and focus on:

**Database & Storage**
-Firebase Firestore (for real-time data storage and automatic scaling)
-Supabase (open-source alternative to Firebase with PostgreSQL support)

**Authentication & User Management**
-Firebase Authentication or Auth0 for user authentication (login/signup for teachers)
-Supabase Auth for simplified user management

**Data Access**
-Use RESTful APIs or GraphQL provided by Firebase or Supabase to access classroom, student, and job data.

**Frontend Stack**
*Since I'm working on a web-based application, I will focus on the following:

**Frontend Framework**
-React.js (for dynamic components and state management)
-Tailwind CSS or ShadCN/UI (for flexible and responsive UI)

**State Management & API Integration**
-React Query (for handling API calls and caching)
-Axios or Fetch API (for interacting with third-party APIs)

**API Integrations**
*These APIs support the application's functionality.

**Authentication API**
-Firebase Authentication or Auth0 (authentication, registration, password resets)

**Database API**
-Firebase Firestore API (real-time data syncing and CRUD operations)
-Supabase API (SQL-based database access for data management)

**Job Assignment & Randomization**

-For job randomization, the logic can be handled directly in your frontend (React) using JavaScript algorithms, like Fisher-Yates, or Firebase Functions for backend processing if needed.

**Notifications & Reminders**

-Firebase Cloud Messaging or Twilio API (to send push notifications or SMS to teachers and students about job updates or feedback)

## Potential Issues

1. **Job Assignment Randomization**

**Problem:**

**-**Students getting the same job repeatedly
-Bias in job assignments

**Solution:**

-Use **Fisher-Yates Shuffle Algorithm** to ensure fairness.
-Implement **state tracking** so students do not repeat jobs before all have had a turn.
-Provide an **override feature** so teachers can manually assign jobs if needed.

2. **Handling Student Absences & Substitute Activation**

**Problem:**

-If an assigned student is absent, their job may remain unfilled.
-If substitutes are pre-assigned, they may also be absent.

**Solution:**

-Allow teachers to mark absences so substitutes are assigned automatically.
-Have a fallback mechanism where a new random substitute can be selected if needed.

3. **Browser & Device Compatibility Issues**

**Problem:**

-The site may not display correctly on different browsers (Chrome, Safari, Edge).
-If displayed on a smartboard or tablet, the UI might not be fully responsive.

**Solution:**

-Use responsive design (CSS media queries, Tailwind CSS).
-Test the site in multiple browsers and devices (use BrowserStack or Chrome DevTools).

4. **Teachers Forgetting to Start New Job Cycles**

**Problem:**

-Teachers might forget to rotate jobs every 2 weeks, leading to job assignments not updating.

**Solution:**

-Implement automated reminders (Firebase Cloud Messaging, email alerts).
-Provide a one-click "Start New Cycle" button for teachers.

Vanesha Richardson
Preparing For Your Capstone: Step 2-Project Proposal