**Frontend Development Task**

**Objective:** Build a task management application using **React**/**Next.js (Next.js is preferred because our all tasks will be done by Next.js )** that supports user account creation, organization management, and individual/group task handling.

**Requirements:**

* **User and Organization Management:**
  + Users can create an account and log in.
  + Users can create organizations.
  + An organization admin/owner can send an invitation to users to join the organization.
  + Users can accept or decline the invitation.
* **Task Management:**
  + Tasks can be created, updated, deleted, and viewed.
  + Tasks can be assigned to individuals or groups within the organization.
  + Include filtering functionality for tasks by **priority** (High, Medium, Low).
* **Deadline and Notification(If possible):**
  + When creating a task, set a **deadline**.
  + Provide notifications (Dashboard alerts or UI indications) before the deadline approaches.
* **UI/UX Guidelines:**
  + Keep the design simple and intuitive.
  + Provide clear navigation between user account, organization, and task management pages.

**Technical Constraints:**

* Use **React** and **Next.js** as the core technologies.
* State management can be implemented using local state, Context API, or Redux (optional).
* Use basic CSS or a library like Tailwind CSS for styling (optional).

**Submission Details:**

* Host the application on a free platform (e.g., Vercel).
* Share the GitHub repository link with clear instructions to run the project locally in the README file.

**Timeline:**You have **48 hours** to complete this task.

**Evaluation Criteria:**

1. Functionality as per the requirements.
2. Code quality, structure, and comments.
3. Simplicity and intuitiveness of the design.
4. Effective use of React and Next.js features.

**Scenario for Task Explanation**

Imagine a company named **"TaskFlow Solutions"**. It is a growing organization with multiple departments, and each department needs a system to manage tasks effectively. As a developer, you are tasked with building a **Task Management Application** for this company. Here’s how the application should work:

### **Story**

1. **Setting Up the System**
   * The admin of **TaskFlow Solutions**, John, logs into the system and creates an account.
   * After logging in, John creates an organization named **"TaskFlow Solutions"** within the app.
2. **Inviting Team Members**
   * John, as the organization admin, invites team members (e.g., Alice and Bob) to join the organization by sending them an invitation through the app.
   * Alice and Bob receive the invitation, accept it, and join the organization.
3. **Creating Tasks**
   * John creates a new task: **"Prepare Monthly Sales Report"**, assigns it to Alice, and sets the priority as **High**.
   * John also creates a group task: **"Organize Annual Meeting"**, assigning it to both Alice and Bob, and sets the priority as **Medium**.
4. **Filtering Tasks**
   * Alice logs in to her account, views her assigned tasks, and filters them by priority to focus on high-priority tasks first.
   * Bob does the same for his tasks.
5. **Managing Deadlines**
   * While creating tasks, John sets a deadline for each task. For example, **"Prepare Monthly Sales Report"** must be completed by **January 20, 2025**.
   * The system sends a notification to Alice (a dashboard alert) reminding her that the task deadline is approaching.
6. **Task Completion and Plagiarism Check**
   * Alice completes her task and submits the details through the system.
   * The task is marked as **Complete**.

### **What the Frontend Developer Needs to Do**

1. Create an intuitive and user-friendly interface for the story above using **React** or **Next.js**.
2. Build pages for:
   * User login and registration.
   * Organization creation and invitation management.
   * Task creation, filtering, and notifications.

### **Final Workflow**

1. John logs in → Creates an organization → Invites Alice and Bob.
2. Alice and Bob accept the invitations → View their tasks.
3. Tasks are created, filtered, updated, and managed based on the requirements.
4. Notifications ensure deadlines are met, and tasks are checked for originality.