Infusi Take-Home Assignment – TaskFlow (Next.js + RBAC)

Goal: Build a small role-based task management app using Next.js. This assignment evaluates your understanding of authentication, role-based access control (RBAC), and CRUD operations in a real-world context.

Tech Stack: Next.js (App Router), TypeScript (preferred), Tailwind CSS or any modern UI library.

Hosting: Deploy your app on Vercel (preferred) or Netlify.
Submission: Share your hosted link and GitHub repository link.
Include a README file with setup steps, credentials, and a short explanation of your RBAC logic.

Seed Data

You may hardcode this data or store it in a simple JSON file.

users.json:

```
[
  {"email": "admin@taskflow.com", "password": "123456", "role": "admin"},
  {"email": "manager@taskflow.com", "password": "123456", "role": "manager"},
  {"email": "member@taskflow.com", "password": "123456", "role": "member"}
]
```

projects.json:

```
[
  {"id": 1, "name": "Website Revamp", "description": "Marketing site refresh",
"owner": "manager@taskflow.com"},
  {"id": 2, "name": "Mobile App", "description": "v1 onboarding flow",
"owner": "manager@taskflow.com"}
]
```

tasks.json:

```
[
  {"id": 1, "projectId": 1, "title": "Design homepage", "assignedTo":
"member@taskflow.com", "status": "pending"},
  {"id": 2, "projectId": 1, "title": "Build hero section", "assignedTo":
"member@taskflow.com", "status": "pending"},
  {"id": 3, "projectId": 2, "title": "Create login screen", "assignedTo":
"manager@taskflow.com", "status": "done"}
]
```

Question 1 – Authentication & Role Access

Build basic authentication and route protection.
Requirements:
1. Implement login using the seed users.
2. Create /login, /dashboard, and /tasks pages.
3. Protect /dashboard and /tasks so only authenticated users can access them.
4. Show an Admin link in the navbar only if the role is 'admin'.
5. Ensure direct URL access without login is blocked.

Question 2 – Projects & Tasks (Feature Extension)

Extend your app to handle real projects and tasks.
Requirements:
1. Create /projects and /projects/[id] pages.
2. Display all projects and allow CRUD operations on tasks.
3. Enforce these RBAC rules:
   - Admin: Full access.
   - Manager: Create/edit/delete tasks in their projects.
   - Member: Can view and mark their own tasks as done.
4. Apply role restrictions on both UI and API.
5. Add loading, empty, and error states.

Question 3 – Admin Panel & Role Management

Enhance your app with an admin-only control panel.
Requirements:
1. Create /admin page (restricted to 'admin' role).
2. List all users and allow changing roles.
3. Optional: Add analytics (tasks done vs pending).
4. Ensure UI and API enforce role permissions consistently.

Evaluation Criteria

• RBAC correctness (UI + API)
• Code clarity and structure
• User experience (loading/error/empty states)
• Accessibility and responsive layout
• Clean README and working hosted app