**🗓️ Consultation Scheduling System — Project Plan**

**✅ Overview**

A full-stack application to manage and schedule academic consultations between **students**, **faculty**, and **administrators**.

**👥 User Roles & Permissions**

| **Role** | **Permissions** |
| --- | --- |
| **Student** | - View available consultation slots - Request a slot - View status |
| **Faculty** | - Set availability - Approve/decline consultation requests |
| **Admin** | - Manage users - View all consultations - Assign faculty (optional) |

**🧱 Backend (Laravel)**

**Suggested Models**

* **User**
  + Fields: id, name, email, password, role
* **Consultation**
  + Fields: id, student\_id, faculty\_id, date\_time, status, message, consultation\_area\_id
* **Availability**
  + Fields: id, faculty\_id, start\_time, end\_time, is\_booked
* **ConsultationArea**
  + Fields: id, name, description

**Consultation Area**

* Categorizes consultations by subject or department (e.g., Math, Science, Counseling)
* Helps students select the right faculty and area for their consultation
* Faculty can manage availability per consultation area (optional)
* Admin can create/update/delete consultation areas

**Notification System**

| **Trigger** | **Recipient** | **Notification Type** |
| --- | --- | --- |
| Student requests a consultation | Faculty | New consultation request |
| Faculty approves/declines a request | Student | Request approved/declined |
| Admin creates user or assigns faculty | Faculty/Student | (optional) account alert |

**Implementation Details:**

* Use Laravel's Notification system (php artisan make:notification)
* Supports Email, Database (in-app), and Broadcast (real-time)
* Store unread/read notifications in DB
* API endpoint to fetch notifications (GET /api/notifications)
* API endpoint to mark notifications as read (POST /api/notifications/mark-as-read/:id)

**Recommended Stack**

* Laravel 12
* Sanctum for SPA auth
* spatie/laravel-permission for roles (optional)
* MySQL/PostgreSQL
* RESTful API via routes/api.php

**🎨 Frontend (React + Tailwind + shadcn/ui)**

**Stack**

* React (Vite) + TypeScript
* Tailwind CSS
* shadcn/ui
* Axios for API requests
* React Router
* Zod + react-hook-form for validation

**Folder Structure**

bash

CopyEdit

/frontend

├── src/components/ui/ ← shadcn components

├── src/pages/ ← route pages

├── src/hooks/

├── src/lib/axios.ts ← API config

**🗂 Suggested Pages & Routes**

**👨‍🎓 Student**

* /student/dashboard – My requests
* /student/schedule – Pick consultation area & time slot
* /student/status – Track approval
* **Notifications:** Show consultation updates

**👨‍🏫 Faculty**

* /faculty/dashboard – Incoming requests
* /faculty/availability – Set availability per consultation area
* /faculty/consultations/:id – Approve/decline
* **Notifications:** New requests & updates

**👨‍💼 Admin**

* /admin/users – Manage all users
* /admin/consultations – View all activity
* /admin/consultation-areas – Manage consultation areas
* /admin/settings – System settings
* **Notifications:** Optional alerts for account changes

**🔁 API Communication**

**Axios Base Config**

ts

CopyEdit

import axios from 'axios'

const axiosInstance = axios.create({

baseURL: 'http://localhost:8000/api',

withCredentials: true,

})

export default axiosInstance

**Notification APIs**

* GET /api/notifications – fetch unread notifications
* POST /api/notifications/mark-as-read/:id – mark notification read

**🛠 Key Features**

* ✅ Secure auth with Laravel Sanctum
* ✅ Role-based access (admin, faculty, student)
* ✅ Dynamic consultation request flow with consultation areas
* ✅ Availability management per faculty and consultation area
* ✅ Notifications (in-app and email)
* ✅ Mobile-responsive UI with Tailwind + shadcn
* ✅ Admin dashboard and user management
* ✅ Optional: email confirmations via Laravel Mail