Backend - Full API & Models Documentation

Authentication Endpoints

POST /accounts

- Create a new user account with email, username, password, and role.

- Uses bcrypt for password hashing.

POST /login

- Authenticates user with email and password.

- Returns user details (email, username, role) on success.

Class Management Endpoints

POST /classes

- Create a new class with course ID, name, professor email, and description.

GET /classes

- Retrieve all available classes.

GET /classes/{course_id}

- Get details of a specific class by course ID.

DELETE /classes/{course_id}

- Delete a class by ID.

Enrollment & Student Management

POST /enrollments

- Enroll a student in a class using email, username, and course ID.

GET /classes/{course_id}/students

- List all students enrolled in a given class.

GET /enrollments/student/{student_email}

- Fetch all classes a student is enrolled in.

Appointments API

POST /appointments

- Book a 30-minute appointment. Removes the matching slot if available.

GET /appointments/student/{email}

- Get all appointments for a student.

GET /appointments/professor/{course_id}

- Retrieve all appointments for a professor's class.

GET /appointments

- Get appointments by student email.

DELETE /appointments/{appointment_id}

- Cancel an appointment. Restores the slot if it exists.

POST /appointments/{appointment_id}/reschedule

- Reschedule an appointment with new datetime, course ID, and student email.

Slot Management Endpoints

POST /slots

- Add a new available slot for a professor and course.

GET /slots/{course_id}

- List all available slots for a class.

POST /available-slots

- Save a slot using 24-hour format. Prevents overlapping slots across classes.

GET /available-slots

- Get available slots for a specific professor, course, and date.

DELETE /available-slots/delete

- Remove an existing slot.

DELETE /available-slots

- Alternative delete with identical logic.

User Info Utilities

GET /users/{email}

- Fetch a user by email.

GET /users/username/{username}

- Fetch a user by username.

GET /professor-email/from-course/{course_id}

- Get the professors email based on the course ID.

Chat and WebSocket Endpoints

GET /chat/history?user1=A&user2=B

- Retrieve chat messages exchanged between two users.

WebSocket: /ws/chat/{user_id}

- Real-time message handler.

- Stores messages in MongoDB and forwards if recipient is online.

Notification Endpoints

GET /notifications/{email}

- Fetch all notifications for a user.

POST /notifications/{notification_id}/mark-read

- Mark a notification as read.

Data Models (Pydantic)

User:

- email: EmailStr

- username: min 5 chars

- password: min 4 chars

- role: student or professor

Class:

- course_id, course_name, professor_name, description

Appointment:

- student_name, student_email, course_id, professor_name, appointment_date (datetime)

Enrollment:

- student_email, student_username, course_id

AvailableSlot:

- professor_email, course_id, date (YYYY-MM-DD), time (HH:MM 24hr)

Notification:

- recipient_email, title, message, type, timestamp, read