Based on the system documentation and the existing test patterns you provided, I have developed additional test cases to cover the remaining critical modules: **Enrollment & Cart**, **Office Hours**, **Communication (Chat)**, and **Transcripts**.

These cases focus on complex business logic like capacity enforcement, conflict prevention, and role-based access.

### 1. Enrollment & Cart Management

*Focuses on the lifecycle of a student's registration process (FR-26 to FR-35).*

| **Test Case ID** | **Module / API Endpoint** | **Scenario** | **Expected Result** |
| --- | --- | --- | --- |
| **CART001** | POST /api/Cart | **Positive:** Add section to cart. | 200 OK; Section added to student's active cart. |
| **CART002** | POST /api/Cart | **Negative:** Add section with time conflict to cart (FR-34). | 400 Bad Request; Error: "Time conflict with existing cart item." |
| **ENROLL001** | POST /api/Enrollment/submit | **Positive:** Submit cart for final enrollment. | 200 OK; Enrollment status set to "Pending" or "Approved." |
| **ENROLL002** | POST /api/Enrollment | **Constraint:** Enroll in a full section (FR-27). | 400 Bad Request; Error: "Section is full." |
| **ENROLL003** | DELETE /api/Enrollment/{id} | **Validation:** Withdraw outside withdrawal period (FR-11). | 403 Forbidden; Error: "Withdrawal period has ended." |

### 2. Office Hours Management

*Focuses on instructor availability and student bookings (FR-40 to FR-44).*

**Test Case ID: OFF001 (Positive)**

* **Endpoint:** POST /api/OfficeHours
* **Role:** Instructor
* **Input:** {"startTime": "2023-10-20T14:00:00", "endTime": "2023-10-20T15:00:00"}
* **Expected Result:** 200 OK; Slot created and visible to students.

**Test Case ID: OFF002 (Conflict Prevention)**

* **Endpoint:** POST /api/OfficeHours/book
* **Scenario:** Two students attempting to book the same slot (FR-43).
* **Expected Result:** First student receives 200 OK; Second student receives 409 Conflict.

### 3. Communication & Notifications

*Focuses on real-time messaging and history (FR-45 to FR-49).*

**Test Case ID: CHAT001 (Real-time Delivery)**

* **Endpoint:** SignalR Hub /chat
* **Steps:** Authenticated Student sends message to Instructor.
* **Expected Result:** Instructor receives message event in real-time; Message stored in database (FR-47).

**Test Case ID: NOTIF001 (Read Status)**

* **Endpoint:** PATCH /api/Notification/{id}/read
* **Input:** id = 5
* **Expected Result:** 200 OK; isRead flag updated to true (FR-49).

### 4. Transcript & Academic Records

*Focuses on grade entry and GPA calculation (FR-36 to FR-39).*

**Test Case ID: TRANS001 (GPA Calculation)**

* **Endpoint:** GET /api/Transcript/me
* **Scenario:** Student with two 3-credit courses (Grades: A and B).
* **Expected Result:** 200 OK; Response body shows calculated GPA of 3.5 (FR-38).

**Test Case ID: TRANS002 (Unauthorized Grade Entry)**

* **Endpoint:** PUT /api/Transcript/{id}/grade
* **Role:** Student
* **Expected Result:** 403 Forbidden; Only Admins or assigned Instructors can modify grades.

### 5. Audit Logging (System Integrity)

*Focuses on accountability and tracking (FR-50 to FR-52).*

**Test Case ID: AUDIT001 (Action Recording)**

* **Scenario:** Admin deletes a user account.
* **Steps:** Execute DELETE /api/User/10. Check GET /api/AuditLogs.
* **Expected Result:** Audit log contains an entry: User 10 deleted by Admin at [Timestamp].

### Visual Workflow: Enrollment Process

To better understand how these test cases fit together, here is the sequence for a student enrolling in a course:

Would you like me to create a **Postman Collection** export (in JSON format) for these test cases so you can run them directly against your API?