Online Course Registration System

SW301 - Course Project – Phase 1

Ahmed Wael 202201415
Ahmed Sameh 202202151
Seif Eldin 202201510

System overview

The Online Course Registration System is a course registration management system at the university; its goal is to make academic life for students, teachers, and administrators easy. It enables students to browse courses, view schedules and degree progress, register and drop courses within registration rules, and submit special requests. Instructors can manage course content and view/approve requests from students. Administrators control course, academic terms, user accounts, and registration business rules.

Objectives

- Provide a single, consistent interface for course discovery and registration.
- Apply academic rules (prerequisites, capacity, schedule conflict, credit limits).
- Reduce administrative overhead through self-service and automated checks/notifications.
- Maintain audit logs of registration actions.

Actors and their roles

Primary actors

1. Student
    - browses courses – registers and drops courses – views schedule and transcript – requests overrides.

2. Instructor
    - Creates and edits course details – posts schedules – views enrolled students – approves or denies requests.
3. Administrator
    - manages course – academic terms/registration windows –manage user accounts – registration rules and reports.

Secondary/system actors

1. Authentication System
    - validates user identity and roles.
2. Two-factor authentication (2FA)
    - validates user identity for more and better security
3. Notification Service
    - sends email/SMS/app notifications for confirmations, announcements, approvals.
4. Payment Gateway
    - Enabling students to pay tuition fees in multiple ways
5. Registrar Office
    - handles special cases or manual approval tasks.

Functional requirements

1. **Browse Courses:** Students and instructors can search and filter the course list.
2. **View Course Details:** Any user can view course syllabus, prerequisites, capacity, scheduled times, and instructor information.
3. **Register for Course:** Students can add a course to their schedule; the system performs checks for prerequisites, time conflicts, capacity, and credit limits.
4. **Drop Course:** Students can drop courses within allowed drop windows; waitlist rules apply if applicable.
5. **Special Requests:** Students can submit requests (prerequisite override, instructor permission) and instructors or administrator can approve or deny.
6. **Manage Courses (Instructor):** Instructors can create or update course descriptions, capacities, and upload syllabus materials (administrator approval if needed).

7. **Manage Courses & Terms (Admin):** Administrators create academic terms, set registration time, enable or disable courses, and set global rules (max credits, max seats).
8. **Manage Users & Roles (Administrator):** Admins onboard or offboard users and assign roles (student, instructor, administrator).
9. **Notifications:** System send confirmation or failure messages for registration events and send notifications for approvals changes and send announcements notifications.
10. **Reporting:** System logs registration events and provides reports for administrators.
11. **Authentication & Authorization:** All access requires authenticated session and role-based authorization checks.
12. **Conflict Resolution:** When the system detects a conflict (time or capacity), provide clear feedback and suggested alternatives.

Non-functional requirements

1. **Usability:** The UI must be intuitive; new students should complete basic registration tasks with minimal instructions.
2. **Performance:** Typical registration operations (search, add/drop) should be completed within a reasonable user-perceivable time (under 3 seconds under normal load).
3. **Availability:** The system should be available during registration time with scheduled maintenance outside busy times and aims for high availability during registration times.
4. **Scalability:** The system must be scaled to support the student population of the university and big load at the start of the semester.
5. **Security & Privacy:** All personal and academic data must be stored and transmitted securely and role-based access controls enforced.
6. **Reliability & Integrity:** Transactions (registrations/drops) must be atomic to prevent inconsistent enrollments.
7. **Maintainability:** The system must be modular with well-documented interfaces to ease future development.
8. **Auditability:** All actions must be logged with timestamps and actor identity.

System constraints

- Regulatory or policy constraints from the registrar (e.g., maximum credits per term) are enforced by system rules.

System assumptions

- Student transcript (completed courses, grades) is available through the system.
- Time data for schedules is provided in a consistent time zone and format.
- Notifications (email/SMS) use an external service; delivery is not guaranteed.
- Instructors and admins have desktop/laptop access during normal working hours for management tasks; students may use mobile devices.

Use case list and descriptions

1. Browse Courses (Student / Instructor)
   - Find courses by search and preview course details.
2. View Course Details (All)
   - Display full course metadata and requirements.
3. Register for Course (Student)
   - Enroll a student into a course after validation.
4. Drop Course (Student)
   - Remove student enrollment within allowed drop time.
5. Request Override (Student)
   - Request permission for exceptions (prerequisite or overload – instructor approval).
6. Approve/Deny Request (Instructor / Admin)
   - Review and approve or deny student requests.
7. Manage Course (Instructor)
   - Create and update course info, syllabuses, capacity.
8. Manage Courses & Terms (Administrator)
   - Control term setup, course availability, global rules.
9. Manage Users (Administrator)
   - Create user accounts and assign roles.

10. Send Notifications (external service)
    - Send confirmation and status messages to users.
11. Generate Reports (Administrator)
    - Generate system and logs reports.

UML Use Case Diagram