# Requirements Gathering — Student Course Registration System (Java)

This document contains the requirements gathering for the Student Course Registration System implemented in Java. It covers scope, functional and non-functional requirements, data model, user stories, API surface, constraints, risks, test strategies, and an implementation roadmap.

## 1. Project summary

Build a Student Course Registration microservice in Java that lets students sign up, log in, view profiles, browse courses, enroll, drop, join/opt-out waitlists, and automatically promote waitlisted students when seats open. Data will be persisted in AWS DynamoDB. The system must enforce enrollment and waitlist limits, respect enrollment/drop cutoffs, and log all key actions for audit.

## 2. Goals and success criteria

- Students can reliably enroll in courses, respecting seat limits and cutoffs.

- Waitlist behaves FIFO, supports opt-out, and auto-enroll promotes eligible students.

- No overbooking — atomic seat updates under concurrent load.

- System records all key actions in StudentLogs for audit.

- Service is testable locally (DynamoDB Local) and deployable to AWS.

## 3. Scope (in-scope / out-of-scope)

In-scope:

- Student sign-up, login, profile view.

- Course listing and details with seat availability.

- Enroll, Drop flows with cutoffs and limits.

- Waitlist join/opt-out, waitlist positions.

- Auto-enroll logic from waitlist when seats open.

- Audit logs for actions.

- Java Spring Boot service using AWS SDK (DynamoDB).

- Unit and integration tests (DynamoDB Local).

Out-of-scope (for MVP):

- Admin UI to create/update courses (courses are preloaded).

- Payment, grading, or course content delivery.

- Complex authentication (we’ll include simple token/JWT option as add-on).

- Multi-region DB redundancy or cross-region failover.

## 4. Functional requirements (summary)

Student Management:

- FR1.1: Student sign-up with studentId, name, email, password.

- FR1.2: Student login by studentId + password → returns session token.

- FR1.3: Student profile includes basic details and list of course applications with statuses.

Course Management:

- FR2.1: Courses preloaded in DynamoDB.

- FR2.2: Course attributes: courseId, courseName, maxSeats, currentEnrolledCount, startDate, endDate, duration (computed), latestEnrollmentBy.

- FR2.3: View courses with currentEnrolledCount and remainingSeats.

- FR2.4: Enroll allowed only when now <= latestEnrollmentBy.

- FR2.5: Drop allowed only when now <= endDate.

Enrollment:

- FR3.1: Enroll if now <= latestEnrollmentBy, seats available, student active enrollments < 5.

- FR3.2: Drop if now <= endDate. On drop decrement currentEnrolledCount.

- FR3.3: Maintain accurate counters.

- FR3.4: Prevent duplicate enroll/waitlist.

Waitlist:

- FR4.1: FIFO waitlist. Student can be on max 3 waitlists.

- FR4.2: Auto-enroll from waitlist when seat opens.

- FR4.3: Simultaneous openings processed sequentially.

- FR4.4: View waitlist position, opt-out feature.

Audit Logging:

- FR6.1/6.2: Log every key action (signup, login, enroll, drop, waitlist join, auto-enroll).

## 5. Non-functional requirements (NFRs)

- Consistency & correctness: atomic updates, no overbooking.

- Scalability: thousands of students/courses.

- Security: hashed passwords, secure tokens.

- Auditability: StudentLogs for compliance.

- Testability: JUnit + DynamoDB Local.

## 6. Data model (DynamoDB)

Students(studentId, name, email, passwordHash, activeEnrollments, waitlistCount, createdAt...)

Courses(courseId, courseName, maxSeats, currentEnrolledCount, startDate, endDate, latestEnrollmentBy, waitlistSeq)

Enrollments(studentId, courseId, status, enrolledAt, droppedAt, waitlistSeq)

Waitlists(courseId, sequence, studentId, joinedAt, optedOut)

StudentLogs(logId, studentId, action, courseId, timestamp)

## 7. User stories & acceptance criteria

US-1: Sign Up

US-2: Login

US-3: View Profile

US-4: List Courses

US-5: Enroll in Course

US-6: Join Waitlist

US-7: Drop Course

US-8: Auto-enroll from waitlist

US-9: Opt-out Waitlist

## 8. Audit, monitoring & observability

- StudentLogs table records all actions.

- Metrics: enroll attempts, waitlist joins, auto-enroll events.

- Monitor DynamoDB throttling, transaction failures.

## 9. Security & privacy

- Hash passwords with BCrypt.