Learning Management System - Module Documentation

Table of Contents

Introduction

This Learning Management System (LMS) provides a comprehensive platform for educational institutions to manage classes, subjects, lectures, assessments, and track student progress. The system is designed to facilitate both teaching and learning experiences with features like study plans, detailed progress tracking, and guardian oversight.

Each module in this document explains what functionality it provides, what data it stores, and how it connects with other parts of the system. This overview will help you understand the complete functionality of the LMS platform.

User Management

Overview

The User Management module handles all user-related operations, including student, teacher, admin, and guardian accounts.

Key Features

- User registration and account verification via OTP
- Password reset functionality
- User profile management
- Role-based access control

Main Endpoints

- `POST /users/register`: Register a new user
- `POST /users/verify-otp/:userId`: Verify email with OTP
- `POST /users/forgot-password`: Request password reset
- `POST /users/reset-password`: Reset password with token
- `GET /users/profile`: Get current user profile with extended information
- `GET /users`: Get all users
- `GET /users/type/:type`: Get users by type (student, teacher, admin, guardian)

User Schema

- **name**: User's full name
- **email**: User's email address (unique)
- **password**: Hashed password
- **type**: User type (student, teacher, admin, guardian)
- **roleId**: Reference to user's role
- **isVerified**: Whether email is verified
- **otp**: One-time password for verification (temporary)

Integration Points

- Connected to Roles module for permissions
- Used by Auth module for authentication
- Connected to Profile module for additional user data
- Guardian-Student module links guardian users to student users

Authentication

Overview

The Authentication module handles user login, token generation, and authorization across the platform.

Key Features

- Secure JWT token-based authentication
- Role-based authorization
- Public/private route protection

Main Endpoints

- `POST /auth/login`: Authenticate user and generate token

Integration Points

- Validates credentials against User module
- Provides JwtAuthGuard used across the application
- Integrated with Profiles for complete user information during login

Class and Subject Management

Overview

This module handles the organization of educational content into classes and subjects, forming the foundational structure of the educational content.

Key Features

- Create and manage classes (like grade levels or cohorts)
- Create and manage subjects within classes
- Configure assessment criteria for classes and subjects

Main Endpoints

Classes

- `POST /classes`: Create a new class
- `GET /classes`: Get all classes
- `GET /classes/:id`: Get a class by ID
- `PUT /classes/:id`: Update a class
- `DELETE /classes/:id`: Delete a class
- `POST /classes/:classId/subjects/:subjectId`: Add a subject to a class
- `DELETE /classes/:classId/subjects/:subjectId`: Remove a subject from a class

Subjects

- `POST /subjects`: Create a new subject
- `GET /subjects`: Get all subjects
- `GET /subjects/class/:classId`: Get subjects by class
- `GET /subjects/:id`: Get a subject by ID
- `PUT /subjects/:id`: Update a subject
- `POST /subjects/:subjectId/chapters/:chapterId`: Add a chapter to a subject
- `DELETE /subjects/:subjectId/chapters/:chapterId`: Remove a chapter from a subject
- `DELETE /subjects/:id`: Delete a subject

Schemas

Class Schema

- **name**: Unique identifier for the class (e.g., "class-2024-batch-a")
- **displayName**: Human-readable name (e.g., "Class 2024 Batch A")
- **subjects**: List of subject IDs associated with this class
- **assessmentCriteria**: Configuration for assessments, including:
  - **aptitudeTest**: Settings for aptitude tests
  - **chapterTests**: Settings for chapter tests
  - **finalExam**: Settings for final exams

- **isActive**: Whether the class is currently active
- **metadata**: Additional information about the class

Subject Schema

- **name**: Unique identifier for the subject (e.g., “mathematics-2024”)
- **displayName**: Human-readable name (e.g., “Mathematics”)
- **classId**: Reference to the class this subject belongs to
- **chapters**: List of chapter IDs associated with this subject
- **assessmentTypes**: Configuration for different assessment types within the subject
- **currentVersion**: Version of the subject
- **isActive**: Whether the subject is currently active
- **metadata**: Additional information about the subject
- **imageUrl**: Optional URL for subject image

Integration Points

- Chapters and Lectures are linked to Subjects
- Classes and Subjects are linked to Enrollments
- Assessments are created for Subjects within Classes

Chapter and Lecture Management

Overview

This module manages educational content, organizing it into chapters and lectures for structured learning.

Key Features

- Create and organize chapters within subjects
- Create and organize lectures within chapters
- Manage chapter prerequisites
- Manage lecture content and resources
- Support for video lectures with transcripts and timestamps

Main Endpoints

Chapters

- `POST /chapters`: Create a new chapter
- `GET /chapters`: Get all chapters
- `GET /chapters/subject/:subjectId`: Get chapters by subject
- `GET /chapters/:id`: Get a chapter by ID
- `PUT /chapters/:id`: Update a chapter
- `PUT /chapters/:id/order`: Update chapter order
- `POST /chapters/reorder`: Reorder multiple chapters within a subject
- `POST /chapters/:chapterId/lectures/:lectureId`: Add a lecture to a chapter
- `DELETE /chapters/:chapterId/lectures/:lectureId`: Remove a lecture from a chapter
- `DELETE /chapters/:id`: Delete a chapter

Lectures

- `POST /lectures`: Create a new lecture
- `GET /lectures`: Get all lectures
- `GET /lectures/byChapter/:chapterId`: Get lectures by chapter
- `GET /lectures/available`: Get lectures not assigned to any chapter
- `GET /lectures/:id`: Get a lecture by ID
- `GET /lectures/:id/details`: Get detailed lecture information
- `GET /lectures/:id/resources`: Get resources for a lecture
- `GET /lectures/:id/transcript`: Get transcript for a lecture
- `POST /lectures/:id/complete`: Mark a lecture as completed
- `POST /lectures/:id/progress`: Update student progress for a lecture
- `PUT /lectures/:id`: Update a lecture
- `DELETE /lectures/:id`: Delete a lecture
- `POST /lectures/:id/resources`: Add a resource to a lecture

- `DELETE /lectures/:id/resources/:index`: Remove a resource from a lecture

Schemas

Chapter Schema

- **name**: Unique identifier for the chapter
- **displayName**: Human-readable name
- **subjectId**: Reference to the subject this chapter belongs to
- **order**: Position of the chapter within the subject
- **lectures**: List of lecture IDs associated with this chapter
- **chapterTest**: Configuration for chapter tests
- **isLocked**: Whether the chapter is locked (requires prerequisites)
- **prerequisites**: List of chapter IDs that must be completed before this one
- **description**: Detailed description of the chapter
- **duration**: Estimated time to complete (in minutes)
- **isActive**: Whether the chapter is currently active
- **metadata**: Additional information about the chapter

Lecture Schema

- **title**: Title of the lecture
- **description**: Detailed description of the lecture
- **classId**: Reference to the class this lecture belongs to
- **subjectId**: Reference to the subject this lecture belongs to
- **chapterId**: Reference to the chapter this lecture belongs to
- **order**: Position of the lecture within the chapter
- **estimatedDuration**: Estimated duration in minutes
- **prerequisites**: List of lecture IDs that should be viewed before this one
- **content**: Configuration for lecture content (type, data)
- **isPublished**: Whether the lecture is published and available
- **tags**: Array of tags for categorizing the lecture
- **metadata**: Additional information about the lecture
- **resources**: Array of resources associated with the lecture
- **hasTranscript**: Whether the lecture has a transcript
- **transcript**: Array of transcript segments with timestamps

Integration Points

- Chapters belong to Subjects
- Lectures belong to Chapters
- Student Progress tracks completion of Lectures and Chapters
- Notes can be created for Lectures

Student Progress Tracking

Overview

This module tracks all aspects of student learning progress, including chapter completion, lecture viewing, assessment results, and study time.

Key Features

- Track student progress through chapters and lectures
- Record time spent on lectures
- Track assessment scores
- Generate detailed progress statistics and analytics
- Create and manage study plans

Main Endpoints

- `PUT /student-progress/:studentId/resource/:resourceId`: Update progress for a specific resource
- `GET /student-progress/:studentId/subject/:subjectId`: Get detailed progress for a specific subject
- `GET /student-progress/:studentId/class/:classId`: Get student progress for a class
- `POST /student-progress/:studentId/study-plan`: Create a new study plan
- `PUT /student-progress/:studentId/study-plan/:planId/item/:itemId`: Update study plan item status
- `GET /student-progress/:studentId/study-plan/:planId/analytics`: Get study plan analytics
- `GET /student-progress/:studentId/overview`: Get overall student progress overview
- `GET /student-progress/:studentId/statistics`: Get detailed progress statistics

Progress Schema

- **studentId**: Reference to the student
- **classId**: Reference to the class
- **subjectId**: Reference to the subject
- **chapterId**: Reference to the chapter
- **lectureId**: Reference to the lecture (optional)
- **status**: Progress status (not_started, in_progress, completed)
- **timeSpentMinutes**: Time spent on the resource
- **progressPercentage**: Percentage of completion
- **lastAccessedAt**: When the resource was last accessed
- **completedAt**: When the resource was completed
- **assessmentScores**: Map of assessment scores

Study Plan Schema

- **studentId**: Reference to the student
- **classId**: Reference to the class
- **planItems**: Array of study plan items, each containing:
  - **itemId**: Reference to the item (chapter, lecture, assessment)
  - **itemType**: Type of item
  - **plannedStartDate**: When study should start
  - **plannedEndDate**: When study should end
  - **actualStartDate**: When study actually started
  - **actualEndDate**: When study actually ended
  - **status**: Status of the item (pending, in_progress, completed, overdue)
  - **reminderEnabled**: Whether reminders are enabled

Integration Points

- Connected to Lectures and Chapters for tracking content progress
- Connected to Assessments for tracking test results
- Used by Guardian features to monitor student progress
- Connected to Study Plans for progress against planned study

Study Plans and Sessions

Overview

This module helps students organize their learning with structured study plans and track actual study sessions.

Key Features

- Create weekly study schedules with time slots
- Set study benchmarks and goals
- Record and track study sessions
- Analyze study time and efficiency
- Compare actual study time against planned schedule

Main Endpoints

Study Plans

- `POST /study-plans/schedules/:studentId`: Create a new study schedule
- `PUT /study-plans/schedules/:studentId/:scheduleId`: Update a study schedule
- `GET /study-plans/schedules/:studentId/:scheduleId`: Get a specific schedule
- `GET /study-plans/schedules/:studentId/active`: Get active schedule
- `GET /study-plans/schedules/:studentId`: Get all schedules

Study Sessions

- `POST /study-sessions/:studentId/start`: Start a new study session
- `PUT /study-sessions/:studentId/end/:sessionId`: End an active study session
- `GET /study-sessions/:studentId/active`: Get active study session
- `GET /study-sessions/:studentId/session/:sessionId`: Get specific study session details
- `GET /study-sessions/:studentId/list`: Get list of study sessions
- `GET /study-sessions/:studentId/analytics`: Get study session analytics

Study Progress

- `GET /study-progress/:studentId/weekly/:year/:week`: Get weekly progress details
- `GET /study-progress/:studentId/summary`: Get progress summary

Schemas

Study Plan Schedule Schema

- **studentId**: Reference to the student
- **weeklySchedule**: Array of time slots for study, each containing:
  - **dayOfWeek**: Day of the week (0-6)
  - **startTime**: Start time (e.g., "18:00")
  - **endTime**: End time (e.g., "19:30")

    - **subjectId**: Reference to the subject
    - **isActive**: Whether this slot is active
- **benchmarks**: Array of study benchmarks/goals, each containing:
    - **type**: Type of benchmark (daily, weekly, monthly)
    - **target**: Target value
    - **metric**: Metric type (hours, topics, assessments)
    - **isActive**: Whether this benchmark is active
    - **guardianId**: Reference to guardian monitoring this benchmark
    - **note**: Optional note about this benchmark
- **isActive**: Whether this schedule is currently active
- **effectiveFrom**: When this schedule becomes effective
- **effectiveUntil**: When this schedule expires (optional)
- **preferences**: Map of user preferences for study

Study Session Schema

- **studentId**: Reference to the student
- **subjectId**: Reference to the subject being studied
- **startTime**: When the session started
- **endTime**: When the session ended (if completed)
- **durationMinutes**: Duration of the session in minutes
- **isCompleted**: Whether the session is completed
- **chaptersStudied**: Array of chapter IDs studied in this session
- **assessmentsTaken**: Array of assessment IDs taken in this session
- **progress**: Object tracking detailed progress:
    - **topicsCompleted**: Number of topics completed
    - **exercisesSolved**: Number of exercises solved
    - **assessmentScore**: Assessment score (if applicable)
- **notes**: Session notes
- **scheduleId**: Reference to the study schedule

Study Progress Schema

- **studentId**: Reference to the student
- **weekNumber**: Week number
- **year**: Year
- **totalStudyMinutes**: Total minutes studied
- **sessionsCompleted**: Number of sessions completed
- **topicsCompleted**: Number of topics completed
- **assessmentsTaken**: Number of assessments taken
- **subjectProgress**: Map of subject IDs to minutes spent
- **benchmarkAchievement**: Map of benchmark IDs to achievement percentages
- **adherenceMetrics**: Metrics comparing planned vs. actual study:

    - **scheduledMinutes**: Minutes scheduled for study
    - **actualMinutes**: Actual minutes studied
    - **adherenceRate**: Percentage of scheduled time actually studied
- **sessions**: Array of session IDs

Integration Points

- Connected to Subjects for organizing study by subject
- Relies on Student Progress for tracking completion
- Guardian features can set and monitor benchmarks
- Notification system can send study reminders

Assessment System

Overview

This module manages all types of assessments, including aptitude tests, chapter tests, and final exams.

Key Features

- Create and manage different types of assessments
- Create and manage assessment questions
- Dynamic assessment generation from templates
- Track assessment results and analytics
- Configure assessment criteria (time limits, attempts allowed, etc.)

Main Endpoints

Assessments

- `POST /assessments`: Create a new assessment
- `POST /assessments/questions`: Create a new question
- `GET /assessments`: Get all assessments with filters
- `GET /assessments/questions`: Get all questions
- `GET /assessments/pending/:studentId`: Get pending assessments for a student
- `GET /assessments/questions/:id`: Get a question by ID
- `GET /assessments/:id`: Get an assessment by ID
- `PUT /assessments/questions/:id`: Update a question
- `PUT /assessments/:id`: Update an assessment
- `DELETE /assessments/questions/:id`: Delete a question
- `DELETE /assessments/:id`: Delete an assessment

Assessment Templates

- `POST /assessment-templates`: Create a new assessment template
- `GET /assessment-templates`: Get all assessment templates
- `GET /assessment-templates/:id`: Get an assessment template by ID
- `PUT /assessment-templates/:id`: Update an assessment template
- `DELETE /assessment-templates/:id`: Delete an assessment template
- `POST /assessment-templates/generate/:templateId/:studentId`: Generate an assessment from a template
- `GET /assessment-templates/for-subject/:classId/:subjectId`: Find a suitable template for a class/subject

Assessment Results

- `POST /assessment-results/:studentId`: Create a new assessment result
- `GET /assessment-results/student/:studentId`: Get assessment results for a student

- `GET /assessment-results/analytics/:assessmentId`: Get analytics for an assessment

Schemas

Assessment Schema

- **title**: Title of the assessment
- **description**: Description of the assessment
- **type**: Type of assessment (aptitude, lecture-activity, chapter-test, final-exam)
- **classId**: Reference to the class
- **subjectId**: Reference to the subject (required for aptitude tests)
- **questions**: Array of question IDs
- **totalPoints**: Total points possible
- **passingScore**: Minimum passing score
- **settings**: Configuration settings:
  - **timeLimit**: Time limit in minutes
  - **shuffleQuestions**: Whether to shuffle questions
  - **showResults**: Whether to show results immediately
  - **attemptsAllowed**: Number of attempts allowed
  - **isPublished**: Whether the assessment is published
  - **isRequired**: Whether the assessment is required
- **startDate**: When the assessment starts (optional)
- **endDate**: When the assessment ends (optional)
- **metadata**: Additional information about the assessment

Question Schema

- **text**: Question text
- **options**: Array of answer options, each containing:
  - **text**: Option text
  - **isCorrect**: Whether this option is correct
  - **explanation**: Explanation for this option (optional)
- **type**: Question type (mcq, true-false, short-answer, essay)
- **explanation**: Explanation for the correct answer (optional)
- **difficultyLevel**: Difficulty level (beginner, intermediate, advanced)
- **tags**: Array of tags for categorizing
- **points**: Points awarded for correct answer
- **subjectId**: Reference to the subject (optional)
- **topics**: Array of topics covered by this question
- **cognitiveLevel**: Cognitive skill being tested (optional)
- **estimatedTimeSeconds**: Estimated time to answer
- **metadata**: Additional information about the question

Assessment Template Schema

- **title**: Title of the template
- **description**: Description of the template
- **type**: Type of assessment this template is for
- **classId**: Reference to the class
- **subjectId**: Reference to the subject (required for aptitude tests)
- **questionCriteria**: Criteria for selecting questions:
  - **totalQuestions**: Total number of questions to include
  - **difficultyDistribution**: Distribution by difficulty level
  - **topicDistribution**: Distribution by topic
  - **skillsToAssess**: Skills to assess with this test (optional)
- **totalPoints**: Total points possible
- **passingScore**: Minimum passing score
- **settings**: Configuration settings (same as Assessment)
- **isActive**: Whether the template is active

Assessment Result Schema

- **studentId**: Reference to the student
- **assessmentId**: Reference to the assessment
- **classId**: Reference to the class
- **subjectId**: Reference to the subject (optional)
- **chapterId**: Reference to the chapter (optional)
- **totalScore**: Total score achieved
- **maxPossibleScore**: Maximum possible score
- **percentageScore**: Percentage score
- **timeSpentMinutes**: Time spent in minutes
- **attemptNumber**: Attempt number
- **questionResponses**: Array of responses to questions, each containing:
  - **questionId**: Reference to the question
  - **selectedAnswer**: Selected answer
  - **isCorrect**: Whether the answer was correct
  - **score**: Score for this question
  - **timeSpentSeconds**: Time spent on this question (optional)
- **status**: Result status (completed, partial, timeout, submitted)
- **skillScores**: Map of skills to scores
- **metadata**: Additional information about the result

Integration Points

- Assessments are linked to Classes and Subjects
- Assessment results feed into Student Progress
- Enrollment module uses aptitude test results

- Guardian module can receive notifications about assessment results

Enrollment Management

Overview

This module handles student enrollment in classes and subjects, including aptitude test requirements.

Key Features

- Enroll students in classes and subjects
- Manage aptitude test requirements for subjects
- Track enrollment status and history
- Check subject access based on test status

Main Endpoints

- `POST /enrollment`: Enroll a student in a specific subject
- `GET /enrollment`: Get all enrollments with optional filtering
- `POST /enrollment/bulk`: Enroll a student in multiple subjects
- `POST /enrollment/assign-tests/:studentId`: Assign aptitude tests for enrolled subjects
- `GET /enrollment/student/:studentId`: Get student enrollments
- `GET /enrollment/student/:studentId/class/:classId/subject/:subjectId`: Get a specific enrollment
- `PUT /enrollment/student/:studentId/class/:classId/subject/:subjectId`: Update an enrollment
- `PUT /enrollment/update-test-status/:studentId/:assessmentResultId`: Update aptitude test status
- `GET /enrollment/access/:studentId/:subjectId`: Check if student has access to a subject
- `GET /enrollment/pending-tests/:studentId`: Get pending aptitude tests for a student
- `DELETE /enrollment/student/:studentId/class/:classId/subject/:subjectId`: Delete an enrollment
- `GET /enrollment/status/all-required-tests-passed/:studentId`: Check if all required tests are passed

Student Subject Enrollment Schema

- **studentId**: Reference to the student
- **classId**: Reference to the class
- **subjectId**: Reference to the subject
- **aptitudeTestCompleted**: Whether the aptitude test has been completed
- **aptitudeTestPassed**: Whether the aptitude test has been passed
- **aptitudeTestId**: Reference to the aptitude test assessment (optional)
- **aptitudeTestResultId**: Reference to the aptitude test result (optional)

- **testCompletedDate**: Date when the test was completed (optional)
- **isEnrolled**: Whether the student is currently enrolled
- **enrollmentDate**: Date when the student enrolled
- **metadata**: Additional information about the enrollment

Integration Points

- Connected to Classes and Subjects for enrollment
- Connected to Assessments for aptitude tests
- Student Progress tracks progress within enrolled subjects
- Guardian features can monitor student enrollments

Notes System

Overview

This module allows students to take and manage notes during lectures, with features like timestamps for video content.

Key Features

- Create and manage notes for lectures
- Add timestamps to link notes to specific points in videos
- Tag and categorize notes
- Export notes in different formats
- Search and filter notes

Main Endpoints

- `POST /notes/lectures/:lectureId`: Create a new note for a lecture
- `GET /notes/lectures/:lectureId`: Get all notes for a lecture
- `GET /notes/:id`: Get a note by ID
- `PUT /notes/:id`: Update a note
- `DELETE /notes/:id`: Delete a note
- `GET /notes/lectures/:lectureId/export`: Export notes for a lecture

Note Schema

- **userId**: Reference to the user who created the note
- **lectureId**: Reference to the lecture this note is for
- **content**: Content of the note
- **timestamp**: Timestamp in the video where the note was taken (in seconds, optional)
- **tags**: Array of tags for organizing notes
- **isPinned**: Whether the note is pinned to the top
- **formatting**: Custom formatting and style for the note (optional)
- **createdAt**: Creation timestamp
- **updatedAt**: Last update timestamp

Integration Points

- Notes are connected to Lectures
- Only the note creator can view their notes

Guardian-Student Relationship

Overview

This module manages relationships between guardians and students, allowing guardians to monitor student progress.

Key Features

- Create and manage guardian-student relationships
- Set permission levels for what guardians can access
- Track student progress from guardian perspective
- Set study benchmarks for students
- Receive notifications about student performance

Main Endpoints

- `POST /guardian-student`: Create a new guardian-student relationship
- `GET /guardian-student/guardian/:guardianId`: Get all students for a guardian
- `GET /guardian-student/student/:studentId`: Get all guardians for a student
- `GET /guardian-student/:guardianId/:studentId`: Get a specific relationship
- `PUT /guardian-student/:guardianId/:studentId`: Update a relationship
- `DELETE /guardian-student/:guardianId/:studentId`: Delete a relationship
- `PATCH /guardian-student/:guardianId/:studentId/deactivate`: Deactivate a relationship
- `GET /guardian-student/:guardianId/:studentId/permissions`: Get guardian permissions for a student
- `GET /guardian-student/guardian/:guardianId/students-progress`: Get all students with progress for a guardian
- `POST /guardian-student/guardian/:guardianId/student/:studentId/benchmark`: Set a study benchmark for a student

Guardian-Student Relationship Schema

- **guardianId**: Reference to the guardian
- **studentId**: Reference to the student
- **relationship**: Relationship type (parent, legal_guardian, relative, other)
- **isPrimary**: Whether this is the primary guardian
- **permissionLevel**: Permission level (full, readonly, progress_only, custom)
- **isActive**: Whether the relationship is active
- **metadata**: Additional information about the relationship

Integration Points

- Connects Guardian users to Student users
- Accesses Student Progress for monitoring
- Can set benchmarks in Study Plans

- Works with Notification system for alerts

Notification System

Overview

This module handles all notifications across the platform, including system notifications, assessment reminders, and guardian alerts.

Key Features

- Create and manage notifications for users
- Support multiple notification channels (in-app, email, etc.)
- Template-based notification generation
- Scheduled notifications
- Read/unread status tracking

Main Endpoints

- `POST /notifications`: Create a new notification
- `GET /notifications`: Get all notifications for the current user
- `GET /notifications/unread/count`: Get count of unread notifications
- `GET /notifications/summary`: Get notification summary
- `GET /notifications/:id`: Get a notification by ID
- `PATCH /notifications/:id/read`: Mark a notification as read
- `PATCH /notifications/mark-all-read`: Mark all notifications as read
- `DELETE /notifications/:id`: Delete a notification

Notification Schema

- **userId**: Reference to the user this notification is for
- **title**: Title of the notification
- **content**: Content of the notification
- **type**: Type of notification (system, payment, course, assessment, etc.)
- **isRead**: Whether the notification has been read
- **priority**: Priority level (high, medium, low)
- **channels**: Channels the notification was sent through
- **deliveryStatus**: Map of channel to delivery status
- **metadata**: Additional information related to the notification
- **scheduledFor**: Scheduled time for the notification (optional)
- **expiresAt**: Expiration time for the notification (optional)

Integration Points

- Used by Assessment module for test notifications
- Used by Guardian features for student performance alerts
- Used by Study Plans for reminders
- Connected to Email module for email notifications

Content Versioning

Overview

This module manages different versions of educational content, allowing for curriculum updates while preserving previous versions.

Key Features

- Create and manage content versions
- Track version history and changes
- Assign specific versions to students
- Support for multiple active versions

Main Endpoints

- `POST /content-versions`: Create a new content version
- `GET /content-versions`: Get all content versions
- `GET /content-versions/entity/:entityType/:entityId`: Get content versions for a specific entity
- `GET /content-versions/:id`: Get a content version by ID
- `PUT /content-versions/:id`: Update a content version
- `PATCH /content-versions/:id/deactivate`: Deactivate a content version
- `GET /content-versions/:id/history`: Get version history
- `POST /content-versions/assign`: Assign a version to a student
- `POST /content-versions/assign/bulk`: Bulk assign a version to multiple students
- `GET /content-versions/student/:studentId`: Get all active version assignments for a student
- `DELETE /content-versions/student/:studentId/version/:versionId`: Remove a version assignment

Schemas

Content Version Schema

- **version**: Version name or number
- **entityType**: Entity type this version applies to (subject, chapter, lecture, assessment)
- **entityId**: Reference to the entity
- **startDate**: Start date for this version
- **endDate**: End date for this version (optional)
- **isActive**: Whether this version is currently active
- **changes**: Content changes in this version
- **metadata**: Additional information about the version

Version History Schema

- **contentVersionId**: Reference to the content version

- **previousVersionId**: Reference to the previous version
- **userId**: Reference to the user who made the change
- **action**: Action type (create, update, delete, publish, retire)
- **changeDetails**: Detailed changes made
- **comment**: User comment about the changes

Student Version Schema

- **studentId**: Reference to the student
- **contentVersionId**: Reference to the content version
- **entityType**: Entity type this version applies to
- **entityId**: Reference to the entity
- **assignedAt**: Assignment date
- **assignedBy**: Reference to the user who assigned this version
- **isActive**: Whether this assignment is active

Integration Points

- Can version any content type (Subject, Chapter, Lecture, Assessment)
- Students can be assigned specific content versions
- Tracks history of content changes

File Upload System

Overview

This module handles file uploads for the platform, including lecture resources, profile images, and assignments.

Key Features

- Secure file upload and storage
- File type validation
- File metadata tracking
- File access control

Main Endpoints

- `POST /upload`: Upload a file
- `GET /upload`: Get all files for the current user
- `GET /upload/:id`: Get a file by ID
- `DELETE /upload/:id`: Delete a file

Upload Schema

- **filename**: Name of the file on the server
- **originalname**: Original name of the file
- **mimetype**: MIME type of the file
- **size**: Size of the file in bytes
- **path**: Path to the file on the server
- **uploadedBy**: Reference to the user who uploaded the file
- **type**: Type of file (image, video, document, other)
- **metadata**: Additional information about the file

Integration Points

- Used by Lectures for resource files
- Can be used for profile images
- Can be used for assignment submissions