LMS API Specification

> Learning Management System Backend API 문서
>
> **작성자** : 송명주 (최종수정 2025-12-22)

목차

도메인	설명	문서
Auth	인증/인가 (회원가입, 로그인, 토큰)	7개 API
Profile	프로필 관리 (조회, 수정, 이미지)	4개 API
User Search	유저 검색 (학생/교수 탐색)	4개 API
Course	강의 관리 (조회, 개설, 주차/콘텐츠)	16개 API
Subject	과목 관리 (목록, 상세, 검색)	3개 API
Enrollment	수강신청 (장바구니, 신청, 취소)	9개 API
Attendance	출석 관리 (학생/교수)	6개 API
Board	게시판 (게시글, 댓글, 첨부파일)	15개 API
Assignment	과제 (등록, 제출, 채점)	10개 API
Message	메시지 (대화방, 메시지, SSE)	12개 API
Notification	알림 (조회, 읽음처리, 삭제)	8개 API
Dashboard	대시보드 (학생용)	4개 API
Course Notice	강의 공지사항 (공지, 댓글)	9개 API

공통 사항

Base URL

/api/v1/...
/api/auth/... (인증 관련)
/api/notifications/... (알림)

인증 방식

- JWT Bearer Token

- Header: Authorization: Bearer {accessToken}
- 토큰 갱신: Refresh-Token: {refreshToken}

공통 응답 형식

성공 응답

```
{
  "success": true,
  "data": { ... },
  "message": "처리 완료"
}
```

에러 응답

```
{
  "success": false,
  "message": "에러 메시지"
}
```

HTTP 상태 코드

코드	설명
200	성공
201	생성 성공
204	삭제 성공 (No Content)
400	잘못된 요청
401	인증 필요
403	권한 없음
404	리소스 없음
500	서버 에러

사용자 타입

타입	설명
STUDENT	학생
PROFESSOR	교수

Quick Reference

API 엔드포인트

1 인증 API (/api/auth)

Method	Endpoint	설명
POST	/signup/email-verification	이메일 인증 코드 발송
POST	/signup/verify-code	인증 코드 확인
POST	/signup	회원가입
POST	/login	로그인
POST	/refresh	토큰 갱신
POST	/logout	로그아웃
GET	/check-email	이메일 중복 확인

2 프로필 API (/api/v1/profile)

Method	Endpoint	설명
GET	/me	내 프로필 조회
PATCH	/me	프로필 수정
POST	/me/image	프로필 이미지 업로드
DELETE	/me/image	프로필 이미지 삭제

3 사용자 검색 API (/api/v1/users)

Method	Endpoint	설명
GET	/search	사용자 검색
GET	/colleges	단과대학 목록
GET	/colleges/{id}/departments	단과대학별 학과 목록
GET	/departments	전체 학과 목록

4 강의 API (/api/v1/courses)

Method	Endpoint	설명
GET	/	강의 목록 조회
GET	/{courseId}	강의 상세 조회

5 교수 강의 관리 API (/api/v1/professor/courses)

Method	Endpoint	설명
POST	/	강의 개설
PUT	/{courseId}	강의 수정
DELETE	/{courseId}	강의 삭제
GET	/	내 강의 목록
GET	/{courseId}	내 강의 상세

6 주차별 콘텐츠 API (/api/v1/professor/courses/{courseId}/weeks)

Method	Endpoint	설명
GET	/	주차 목록 조회
POST	/	주차 등록
PUT	/{weekId}	주차 수정
DELETE	/{weekId}	주차 삭제
GET	/{weekId}/contents	콘텐츠 목록
POST	/{weekId}/contents	콘텐츠 등록
PUT	/{weekId}/contents/{contentId}	콘텐츠 수정
DELETE	/{weekId}/contents/{contentId}	콘텐츠 삭제
PUT	/{weekId}/contents/reorder	콘텐츠 순서 변경

7 과목 API (/api/v1/subjects)

Method	Endpoint	설명
GET	/	과목 목록 조회
GET	/{subjectId}	과목 상세 조회
GET	/search	과목 검색

8 수강신청 API (/api/v1/enrollments)

Method	Endpoint	설명
GET	/periods/current	현재 수강신청 기간 조회
GET	/courses	수강신청 가능 강의 조회

Method	Endpoint	설명
POST	/bulk	수강신청 (일괄)
DELETE	/bulk	수강 취소 (일괄)
GET	/my	내 수강 목록

9 장바구니 API (/api/v1/carts)

Method	Endpoint	설명
GET	/	장바구니 조회
POST	/bulk	장바구니 추가 (일괄)
DELETE	/bulk	장바구니 삭제 (일괄)
DELETE	/	장바구니 전체 삭제

10 게시판 API (/api/v1/board)

Method	Endpoint	설명
POST	/{boardType}/posts	게시글 작성
GET	/{boardType}/posts	게시글 목록
GET	/{boardType}/posts/{id}	게시글 상세
PUT	/posts/{id}	게시글 수정
DELETE	/posts/{id}	게시글 삭제
POST	/posts/{id}/like	좋아요 토글
GET	/posts/{id}/liked	좋아요 여부 조회

11 댓글 API (/api/v1/board)

Method	Endpoint	설명
POST	/comments	댓글 작성
GET	/comments	댓글 목록 (postId 필터)
PUT	/comments/{id}	댓글 수정
DELETE	/comments/{id}	댓글 삭제

12 첨부파일 API (/api/v1/attachments)

Method	Endpoint	설명
POST	/upload	단일 파일 업로드
POST	/upload/multiple	다중 파일 업로드

Method	Endpoint	설명
GET	/{attachmentId}/download	파일 다운로드
GET	/{attachmentId}	첨부파일 정보 조회
DELETE	/{attachmentId}	첨부파일 삭제

13 과제 API (/api/v1/assignments)

Method	Endpoint	설명
POST	/	과제 등록 (교수)
GET	/	강의별 과제 목록
GET	/{id}	과제 상세
PUT	/{id}	과제 수정 (교수)
DELETE	/{id}	과제 삭제 (교수)
POST	/{id}/submit	과제 제출 (학생)
GET	/{id}/submissions	제출 목록 (교수)
PUT	/submissions/{submissionId}/grade	채점 (교수)
GET	/{id}/my-submission	내 제출 조회 (학생)

14 알림 API (/api/notifications)

Method	Endpoint	설명
GET	/	알림 목록 (커서 기반)
GET	/{notificationId}	알림 상세
GET	/unread-count	안읽은 알림 개수
PATCH	/{notificationId}/read	읽음 처리
PATCH	/read-all	모든 알림 읽음 처리
DELETE	/{notificationId}	알림 삭제
DELETE	/read	읽은 알림 삭제
DELETE	/all	모든 알림 삭제

15 대화방 API (/api/v1/conversations)

Method	Endpoint	설명
GET	/	대화방 목록
POST	/with/{otherUserId}	대화방 생성/조회
GET	/{conversationId}	대화방 상세
DELETE	/{conversationId}	대화방 삭제
POST	/{conversationId}/read	읽음 처리
GET	/unread-count	전체 안읽음 수

16 메시지 API (/api/v1/messages)

Method	Endpoint	설명
POST	/	메시지 전송
POST	/bulk	메시지 일괄 전송
GET	/conversations/{conversationId}	대화 내역 조회
DELETE	/{messageId}	메시지 삭제
POST	/conversations/{conversationId}/read	읽음 처리

17 SSE API (/api/v1/sse)

Method	Endpoint	설명
GET	/subscribe	SSE 연결 (실시간 알림/메시지)

18 강의 공지사항 API (/api/v1/courses/{courseId}/notices)

Method	Endpoint	설명
POST	/	공지사항 생성 (교수)
GET	/	공지사항 목록 조회
GET	/{noticeId}	공지사항 상세 조회
PUT	/{noticeId}	공지사항 수정 (교수)

Method	Endpoint	설명
DELETE	/{noticeId}	공지사항 삭제 (교수)
POST	/{noticeId}/comments	댓글 작성
POST	/{noticeId}/comments/{parentId}/replies	대댓글 작성
PUT	/{noticeId}/comments/{commentId}	댓글 수정
DELETE	/{noticeId}/comments/{commentId}	댓글 삭제

---

19 Video Streaming API (Port: 8090)

> **Video Streaming Server** 전용 API

영상 업로드 (TUS Protocol)

Method	Endpoint	설명
OPTIONS	/api/v1/videos/upload	TUS 프로토콜 지원 확인
POST	/api/v1/videos/upload	업로드 생성 (Location 헤더 반환)
HEAD	/api/v1/videos/upload/**	업로드 상태 확인
PATCH	/api/v1/videos/upload/**	청크 업로드
DELETE	/api/v1/videos/upload/**	업로드 취소

영상 스트리밍

Method	Endpoint	설명
GET	/api/v1/videos/stream/{videoId}	영상 스트리밍 (Range 헤더 지원)

시청 세션

Method	Endpoint	설명
POST	/api/v1/sessions	시청 세션 시작
DELETE	/api/v1/sessions/{sessionId}	시청 세션 종료

Method	Endpoint	설명
GET	/api/v1/sessions/{sessionId}	세션 정보 조회
GET	/api/v1/sessions/active	활성 세션 조회 (userId)

진도 보고

Method	Endpoint	설명
POST	/api/v1/progress	진도 보고 (5초 간격)
GET	/api/v1/progress/{contentId}	진도 조회 (studentId)

시청 이벤트

Method	Endpoint	설명
POST	/api/v1/watch-events	이벤트 기록
GET	/api/v1/watch-events/session/{sessionId}	세션별 이벤트 조회
GET	/api/v1/watch-events/session/{sessionId}/type	이벤트 유형별 조회

Auth API

> 인증/인가 관련 API

목차

- 1. 이메일 인증 코드 발송
- 2. 이메일 인증 코드 확인
- 3. 회원가입
- 4. 로그인
- 5. 토큰 갱신
- 6. 로그아웃
- 7. 이메일 중복 확인

---

1. 이메일 인증 코드 발송

회원가입을 위한 이메일 인증 코드를 발송합니다.

Request

POST /api/auth/signup/email-verification

Request Body

```
{
  "email": "user@example.com"
}
```

Response

```
{
  "success": true,
  "message": "인증 코드가 발송되었습니다."
}
```

Error Response

상태 코드	메시지
400	이미 가입된 이메일입니다.
500	인증 코드 발송에 실패했습니다.

---

2. 이메일 인증 코드 확인

발송된 인증 코드를 확인합니다.

Request

POST /api/auth/signup/verify-code

Request Body

```
{
  "email": "user@example.com",
  "code": "123456"
}
```

Response

```
{
  "success": true,
  "message": "이메일 인증이 완료되었습니다."
}
```

Error Response

상태 코드	메시지
400	인증 코드가 올바르지 않거나 만료되었습니다.

---

3. 회원가입

새로운 사용자를 등록합니다.

Request

POST /api/auth/signup

Request Body

```
{
  "email": "user@example.com",
  "password": "password123",
  "name": "홍길동",
  "userType": "STUDENT",
  "departmentId": 1,
  "phoneNumber": "010-1234-5678"
}
```

Response

```
{
  "success": true,
  "message": "회원가입이 완료되었습니다.",
  "userId": "2025010001"
}
```

Error Response

상태 코드	메시지
400	이메일 인증이 필요합니다.
400	이미 가입된 이메일입니다.

---

4. 로그인

사용자 인증 후 JWT 토큰을 발급합니다.

Request

POST /api/auth/login

Request Body

```
{
  "email": "user@example.com",
  "password": "password123"
}
```

Response

```
{
  "accessToken": "eyJhbGciOiJIUzI1NiJ9...",
  "refreshToken": "eyJhbGciOiJIUzI1NiJ9...",
  "tokenType": "Bearer",
  "expiresIn": 3600,
  "userType": "STUDENT",
  "userId": 2025010001,
  "userName": "홍길동"
}
```

Error Response

상태 코드	메시지
400	이메일 또는 비밀번호가 올바르지 않습니다.

---

5. 토큰 갱신

Refresh Token을 사용하여 새로운 Access Token을 발급합니다.

Request

POST /api/auth/refresh

Request Body

```
{
  "refreshToken": "eyJhbGciOiJIUzI1NiJ9..."
}
```

Response


```
{
  "accessToken": "eyJhbGciOiJIUzI1NiJ9...",
  "refreshToken": "eyJhbGciOiJIUzI1NiJ9...",
  "tokenType": "Bearer",
  "expiresIn": 3600
}
```


Error Response

상태 코드	메시지
401	유효하지 않은 Refresh Token입니다.
401	만료된 Refresh Token입니다.

---

6. 로그아웃

사용자 로그아웃 처리 (Refresh Token 무효화)

Request

POST /api/auth/logout

Headers

Refresh-Token: {refreshToken}

Response


```
{
  "success": true,
  "message": "로그아웃되었습니다."
}
```


---

7. 이메일 중복 확인

이메일 사용 가능 여부를 확인합니다.

Request

GET /api/auth/check-email?email={email}

Query Parameters

파라미터	타입	필수	설명
email	string	O	확인할 이메일

Response

```
{
  "available": true,
  "message": "사용 가능한 이메일입니다."
}

{
  "available": false,
  "message": "이미 사용 중인 이메일입니다."
}
```

Profile API

> 프로필 관리 API

목차

- 1. 내 프로필 조회
- 2. 프로필 수정
- 3. 프로필 이미지 업로드
- 4. 프로필 이미지 삭제

---

1. 내 프로필 조회

로그인한 사용자의 프로필 정보를 조회합니다.

Request

GET /api/v1/profile/me

Headers

Authorization: Bearer {accessToken}

Response


```
{
  "userId": 2025010001,
  "userNumber": "2025010001",
  "name": "홍길동",
  "email": "hong@example.com",
  "userType": "STUDENT",
  "phoneNumber": "010-1234-5678",
  "departmentName": "컴퓨터공학과",
  "collegeName": "공과대학",
  "profileImageUrl": "https://storage.example.com/profiles/123.jpg",
  "createdAt": "2025-01-01T00:00:00"
}
```


---

2. 프로필 수정

프로필 정보를 수정합니다.

Request

PATCH /api/v1/profile/me

Headers

Authorization: Bearer {accessToken}
Content-Type: application/json

Request Body


```
{
  "phoneNumber": "010-9876-5432",
  "introduce": "안녕하세요. 컴퓨터공학과 학생입니다."
}
```


Response


```
{
  "userId": 2025010001,
  "userNumber": "2025010001",
  "name": "홍길동",
  "email": "hong@example.com",
  "userType": "STUDENT",
  "phoneNumber": "010-9876-5432",
  "introduce": "안녕하세요. 컴퓨터공학과 학생입니다.",
  "departmentName": "컴퓨터공학과",
  "collegeName": "공과대학",
```

```
  "profileImageUrl": "https://storage.example.com/profiles/123.jpg"
}
```


---

3. 프로필 이미지 업로드

프로필 이미지를 업로드합니다.

Request

POST /api/v1/profile/me/image

Headers

Authorization: Bearer {accessToken}
Content-Type: multipart/form-data

Request Body (Form Data)

필드	타입	필수	설명
image	file	O	이미지 파일 (jpg, png, gif)

Response

HTTP/1.1 202 Accepted

> 이미지는 비동기로 처리되며, 처리 완료 후 프로필 조회 시 반영됩니다.

---

4. 프로필 이미지 삭제

프로필 이미지를 삭제합니다.

Request

DELETE /api/v1/profile/me/image

Headers

Authorization: Bearer {accessToken}

Response

HTTP/1.1 200 OK

User Search API

> 유저 검색 API (커서 기반 무한스크롤)

목차

- 1. 유저 검색
- 2. 단과대 목록 조회
- 3. 학과 목록 조회 (단과대별)
- 4. 전체 학과 목록 조회

---

1. 유저 검색

단과대, 학과, 이름, 사용자 타입으로 유저를 검색합니다.

Request

GET /api/v1/users/search

Headers

Authorization: Bearer {accessToken}

Query Parameters

파라미터	타입	필수	설명
keyword	string	X	검색어 (이름)
collegeId	long	X	단과대 ID
departmentId	long	X	학과 ID
userType	string	X	사용자 타입 (STUDENT, PROFESSOR)
cursor	long	X	커서 (이전 응답의 nextCursor)
size	int	X	페이지 크기 (기본: 20)

Response


```
{
  "users": [
    {
      "userId": 2025010001,
      "userNumber": "2025010001",
      "name": "홍길동",
      "userType": "STUDENT",
      "departmentName": "컴퓨터공학과",
      "collegeName": "공과대학",
      "profileImageUrl": "https://storage.example.com/profiles/123.jpg"
    }
  ],
  "nextCursor": 2025010002,
  "hasNext": true,
  "totalCount": 150
}
```


---

2. 단과대 목록 조회

유저 검색 필터용 단과대 목록을 조회합니다.

Request

GET /api/v1/users/colleges

Headers

Authorization: Bearer {accessToken}

Response


```
[
  {
    "collegeId": 1,
    "collegeName": "공과대학"
  },
  {
    "collegeId": 2,
    "collegeName": "경영대학"
  }
]
```


---

3. 학과 목록 조회 (단과대별)

특정 단과대의 학과 목록을 조회합니다.

Request

GET /api/v1/users/colleges/{collegeId}/departments

Path Parameters

파라미터	타입	설명
collegeId	long	단과대 ID

Headers

Authorization: Bearer {accessToken}

Response

```
[
  {
    "departmentId": 1,
    "departmentName": "컴퓨터공학과"
  },
  {
    "departmentId": 2,
    "departmentName": "전자공학과"
  }
]
```

---

4. 전체 학과 목록 조회

모든 학과 목록을 조회합니다.

Request

GET /api/v1/users/departments

Headers

Authorization: Bearer {accessToken}

Response


```
[
  {
    "departmentId": 1,
    "departmentName": "컴퓨터공학과",
    "collegeName": "공과대학"
  },
  {
    "departmentId": 2,
    "departmentName": "전자공학과",
    "collegeName": "공과대학"
  }
]
```


Course API

> 강의 관리 API

목차

강의 조회 (공통)

- 1. 강의 목록 검색
- 2. 강의 상세 조회

교수 강의 관리

- 3. 강의 개설
- 4. 강의 수정
- 5. 강의 취소
- 6. 내 강의 목록 조회
- 7. 교수 강의 상세 조회

주차/콘텐츠 관리

- 8. 주차 목록 조회
- 9. 주차 생성
- 10. 주차 수정
- 11. 주차 삭제
- 12. 주차별 콘텐츠 목록 조회
- 13. 콘텐츠 등록
- 14. 콘텐츠 수정
- 15. 콘텐츠 삭제
- 16. 콘텐츠 단건 수정/삭제

1. 강의 목록 검색

개설된 강의를 검색합니다.

Request

GET /api/v1/courses

Query Parameters

파라미터	타입	필수	설명
enrollmentPeriodId	long	O	수강신청 기간 ID
keyword	string	X	검색어 (강의명, 교수명)
departmentId	long	X	학과 ID
courseType	int	X	강의 유형 (1: 전공, 2: 교양)
credits	int	X	학점
page	int	X	페이지 번호 (기본: 0)
size	int	X	페이지 크기 (기본: 20)
sort	string	X	정렬 기준

Response


```
{
  "success": true,
  "data": {
    "courses": [
      {
        "courseId": 1,
        "courseName": "자료구조",
        "professorName": "김교수",
        "departmentName": "컴퓨터공학과",
        "credits": 3,
        "courseType": "전공필수",
        "schedule": "월 09:00-10:30, 수 09:00-10:30",
        "classroom": "공학관 301호",
        "currentEnrollment": 25,
        "maxEnrollment": 40
      }
    ],
    "totalPages": 5,
    "totalElements": 100,
    "currentPage": 0
```

```
  }
}
```

---

2. 강의 상세 조회

강의 상세 정보를 조회합니다.

Request

GET /api/v1/courses/{courseId}

Response

```
{
  "success": true,
  "data": {
    "courseId": 1,
    "courseName": "자료구조",
    "professorName": "김교수",
    "professorId": 2024010001,
    "departmentName": "컴퓨터공학과",
    "credits": 3,
    "courseType": "전공필수",
    "description": "자료구조의 기본 개념과 알고리즘을 학습합니다.",
    "schedule": "월 09:00-10:30, 수 09:00-10:30",
    "classroom": "공학관 301호",
    "currentEnrollment": 25,
    "maxEnrollment": 40,
    "syllabus": "..."
  }
}
```

---

3. 강의 개설

새로운 강의를 개설합니다. (교수 전용)

Request

POST /api/v1/professor/courses

Request Body

```
{
  "subjectId": 1,
  "academicTermId": 1,
```

```
  "maxEnrollment": 40,
  "schedule": [
    {
      "dayOfWeek": "MONDAY",
      "startTime": "09:00",
      "endTime": "10:30"
    }
  ],
  "classroom": "공학관 301호",
  "description": "강의 설명"
}
```

Response

```
{
  "success": true,
  "data": {
    "courseId": 1,
    "courseName": "자료구조",
    "message": "강의가 개설되었습니다."
  }
}
```

---

4. 강의 수정

강의 정보를 수정합니다. (교수 전용)

Request

PUT /api/v1/professor/courses/{courseId}

Request Body

```
{
  "maxEnrollment": 50,
  "classroom": "공학관 401호",
  "description": "수정된 강의 설명"
}
```

Response

```
{
  "success": true,
  "data": {
    "courseId": 1,
    "courseName": "자료구조",
    "maxEnrollment": 50
```

```
  }
}
```

---

5. 강의 취소

개설한 강의를 취소합니다. (교수 전용)

Request

DELETE /api/v1/professor/courses/{courseId}

Response

```
{
  "success": true,
  "data": null,
  "message": "강의가 취소되었습니다."
}
```

---

6. 내 강의 목록 조회 (교수)

교수가 개설한 강의 목록을 조회합니다.

Request

GET /api/v1/professor/courses

Query Parameters

파라미터	타입	필수	설명
academicTermId	long	X	학기 ID (미지정 시 전체)

Response

```
{
  "success": true,
  "data": {
    "courses": [
      {
        "courseId": 1,
        "courseName": "자료구조",
        "credits": 3,
        "currentEnrollment": 25,
```

```
      "maxEnrollment": 40,
      "academicTermName": "2025학년도 1학기"
    }
   ]
  }
}
```


---

7. 교수 강의 상세 조회

교수용 강의 상세 정보를 조회합니다.

Request

GET /api/v1/professor/courses/{courseId}

Response


```
{
  "success": true,
  "data": {
    "courseId": 1,
    "courseName": "자료구조",
    "credits": 3,
    "currentEnrollment": 25,
    "maxEnrollment": 40,
    "enrolledStudents": [...],
    "weeks": [...]
  }
}
```


---

8. 주차 목록 조회

강의의 주차 목록을 조회합니다.

Request

GET /api/v1/professor/courses/{courseId}/weeks

Response


```
{
  "success": true,
  "data": [
    {
      "weekId": 1,
```

```
    "weekNumber": 1,
    "title": "오리엔테이션",
    "startDate": "2025-03-02",
    "endDate": "2025-03-08",
    "contentCount": 3
   }
  ]
}
```


---

9. 주차 생성

새로운 주차를 생성합니다. (교수 전용)

Request

POST /api/v1/professor/courses/{courseId}/weeks

Request Body


```
{
  "weekNumber": 1,
  "title": "오리엔테이션",
  "startDate": "2025-03-02",
  "endDate": "2025-03-08"
}
```


Response


```
{
  "success": true,
  "data": {
    "weekId": 1,
    "weekNumber": 1,
    "title": "오리엔테이션"
  },
  "message": "주차가 생성되었습니다"
}
```


---

10. 주차 수정

주차 정보를 수정합니다. (교수 전용)

Request

PUT /api/v1/professor/courses/{courseId}/weeks/{weekId}

Request Body


```
{
  "title": "수정된 제목",
  "startDate": "2025-03-02",
  "endDate": "2025-03-08"
}
```


---

11. 주차 삭제

주차를 삭제합니다. (교수 전용)

Request

DELETE /api/v1/professor/courses/{courseId}/weeks/{weekId}

---

12. 주차별 콘텐츠 목록 조회

특정 주차의 콘텐츠 목록을 조회합니다.

Request

GET /api/v1/professor/courses/{courseId}/weeks/{weekId}/contents

Response


```
{
  "success": true,
  "data": {
    "weekId": 1,
    "weekNumber": 1,
    "title": "오리엔테이션",
    "contents": [
      {
        "contentId": 1,
        "contentType": "VIDEO",
        "title": "강의 소개 영상",
        "duration": 600,
        "order": 1
      },
      {
        "contentId": 2,
        "contentType": "DOCUMENT",
        "title": "강의계획서",
        "order": 2
      }
```

```
    ]
  }
}
```

---

13. 콘텐츠 등록

주차에 콘텐츠를 등록합니다. (교수 전용)

Request

POST /api/v1/professor/courses/{courseId}/weeks/{weekId}/contents

Request Body

```
{
  "contentType": "VIDEO",
  "title": "1주차 강의",
  "description": "강의 설명",
  "videoId": "abc123",
  "duration": 3600,
  "order": 1
}
```

Content Types

타입	설명
VIDEO	영상 콘텐츠
DOCUMENT	문서/자료
QUIZ	퀴즈
ASSIGNMENT	과제

---

14. 콘텐츠 수정

콘텐츠를 수정합니다. (교수 전용)

Request

PUT /api/v1/professor/courses/{courseId}/weeks/{weekId}/contents/{contentId}

---

15. 콘텐츠 삭제

콘텐츠를 삭제합니다. (교수 전용)

Request

DELETE /api/v1/professor/courses/{courseId}/weeks/{weekId}/contents/{contentId}

---

16. 콘텐츠 단건 수정/삭제

콘텐츠 ID만으로 수정/삭제합니다. (교수 전용)

수정

PUT /api/v1/professor/contents/{contentId}

삭제

DELETE /api/v1/professor/contents/{contentId}

Course Notice API

> 강의 공지사항 API

목차

공지사항 관리

- 1. 공지사항 생성 (교수)
- 2. 공지사항 목록 조회
- 3. 공지사항 상세 조회
- 4. 공지사항 수정 (교수)
- 5. 공지사항 삭제 (교수)

댓글 관리

- 6. 댓글 작성
- 7. 대댓글 작성
- 8. 댓글 수정
- 9. 댓글 삭제

---

1. 공지사항 생성 (교수)

강의 공지사항을 생성합니다. 담당 교수만 가능합니다.

Request

POST /api/v1/courses/{courseId}/notices

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Request Body


```
{
  "title": "중간고사 안내",
  "content": "중간고사는 4월 15일에 진행됩니다.",
  "allowComments": true
}
```


필드	타입	필수	설명
title	string	O	제목 (최대 200자)
content	string	O	내용
allowComments	boolean	O	댓글 허용 여부

Response


```
{
  "success": true,
  "data": {
    "id": 1,
    "courseId": 24,
    "title": "중간고사 안내",
    "allowComments": true,
    "authorId": 20250101002,
    "authorName": "김교수",
    "createdAt": "2025-04-01T10:00:00",
```

```
    "updatedAt": "2025-04-01T10:00:00"
  },
  "message": "공지사항이 생성되었습니다."
}
```


Error Codes

HTTP Status	에러 코드	설명
400	INVALID_INPUT	입력값 유효성 검증 실패
401	UNAUTHORIZED	인증 필요
403	NOT_OWNER	담당 교수가 아님
404	COURSE_NOT_FOUND	강의를 찾을 수 없음

2. 공지사항 목록 조회

강의 공지사항 목록을 조회합니다. 수강생 및 담당 교수만 조회 가능합니다.

Request

GET /api/v1/courses/{courseId}/notices

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID

Query Parameters

파라미터	타입	필수	설명	기본값
page	int	X	페이지 번호	0
size	int	X	페이지 크기	10
sort	string	X	정렬 기준	createdAt,desc

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Response


```
{
  "success": true,
  "data": {
    "content": [
      {
        "id": 1,
        "courseId": 24,
        "title": "중간고사 안내",
        "allowComments": true,
        "authorId": 20250101002,
        "authorName": "김교수",
        "createdAt": "2025-04-01T10:00:00",
        "updatedAt": "2025-04-01T10:00:00"
      }
    ],
    "pageable": {
      "pageNumber": 0,
      "pageSize": 10,
      "sort": { "sorted": true, "direction": "DESC" }
    },
    "totalElements": 6,
    "totalPages": 1
  }
}
```


Error Codes

HTTP Status	에러 코드	설명
401	UNAUTHORIZED	인증 필요
403	NOT_ENROLLED	수강생/담당교수가 아님
404	COURSE_NOT_FOUND	강의를 찾을 수 없음

---

3. 공지사항 상세 조회

강의 공지사항 상세 정보를 조회합니다 (댓글 포함).

Request

GET /api/v1/courses/{courseId}/notices/{noticeId}

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID
noticeId	long	O	공지사항 ID

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Response


```
{
 "success": true,
 "data": {
  "id": 1,
  "courseId": 24,
  "title": "중간고사 안내",
  "content": "중간고사는 4월 15일에 진행됩니다.\n\n시험 범위: 1주차~7주차
        \n준비물: 학생증, 필기구",
  "allowComments": true,
  "authorId": 20250101002,
  "authorName": "김교수",
  "createdAt": "2025-04-01T10:00:00",
  "updatedAt": "2025-04-01T10:00:00",
  "comments": [
   {
    "id": 1,
    "noticeId": 1,
    "parentId": null,
    "content": "시험 범위가 어디까지인가요?",
    "authorId": 20250101012,
    "authorName": "홍길동",
    "createdAt": "2025-04-02T14:30:00",
    "updatedAt": "2025-04-02T14:30:00",
    "children": [
     {
      "id": 2,
      "noticeId": 1,
      "parentId": 1,
      "content": "1주차부터 7주차까지입니다.",
      "authorId": 20250101002,
      "authorName": "김교수",
      "createdAt": "2025-04-02T15:00:00",
      "updatedAt": "2025-04-02T15:00:00",
      "children": []
     }
    ]
```

```
    }
   ]
  }
}
```


Error Codes

HTTP Status	에러 코드	설명
401	UNAUTHORIZED	인증 필요
403	NOT_ENROLLED	수강생/담당교수가 아님
404	COURSE_NOT_FOUND	강의를 찾을 수 없음
404	NOTICE_NOT_FOUND	공지사항을 찾을 수 없음

---

4. 공지사항 수정 (교수)

강의 공지사항을 수정합니다. 담당 교수만 가능합니다.

Request

PUT /api/v1/courses/{courseId}/notices/{noticeId}

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID
noticeId	long	O	공지사항 ID

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Request Body


```
{
  "title": "중간고사 안내 (수정)",
  "content": "중간고사가 4월 20일로 연기되었습니다.",
```

```
  "allowComments": true
}
```

Response

```
{
  "success": true,
  "data": {
    "id": 1,
    "courseId": 24,
    "title": "중간고사 안내 (수정)",
    "allowComments": true,
    "authorId": 20250101002,
    "authorName": "김교수",
    "createdAt": "2025-04-01T10:00:00",
    "updatedAt": "2025-04-05T09:00:00"
  },
  "message": "공지사항이 수정되었습니다."
}
```

Error Codes

HTTP Status	에러 코드	설명
400	INVALID_INPUT	입력값 유효성 검증 실패
401	UNAUTHORIZED	인증 필요
403	NOT_OWNER	담당 교수가 아님
404	NOTICE_NOT_FOUND	공지사항을 찾을 수 없음

---

5. 공지사항 삭제 (교수)

강의 공지사항을 삭제합니다 (Soft Delete). 담당 교수만 가능합니다.

Request

DELETE /api/v1/courses/{courseId}/notices/{noticeId}

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID

파라미터	타입	필수	설명
noticeId	long	O	공지사항 ID

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Response

```
{
  "success": true,
  "message": "공지사항이 삭제되었습니다."
}
```

Error Codes

HTTP Status	에러 코드	설명
401	UNAUTHORIZED	인증 필요
403	NOT_OWNER	담당 교수가 아님
404	NOTICE_NOT_FOUND	공지사항을 찾을 수 없음

6. 댓글 작성

공지사항에 댓글을 작성합니다. 수강생 및 담당 교수 모두 가능합니다.

Request

POST /api/v1/courses/{courseId}/notices/{noticeId}/comments

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID
noticeId	long	O	공지사항 ID

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Request Body


```
{
  "content": "시험 범위가 어디까지인가요?"
}
```


필드	타입	필수	설명
content	string	O	댓글 내용

Response


```
{
  "success": true,
  "data": {
    "id": 1,
    "noticeId": 1,
    "parentId": null,
    "content": "시험 범위가 어디까지인가요?",
    "authorId": 20250101012,
    "authorName": "홍길동",
    "createdAt": "2025-04-02T14:30:00",
    "updatedAt": "2025-04-02T14:30:00",
    "children": []
  },
  "message": "댓글이 작성되었습니다."
}
```


Error Codes

HTTP Status	에러 코드	설명
400	INVALID_INPUT	입력값 유효성 검증 실패
400	COMMENTS_NOT_ALLOWED	댓글이 허용되지 않은 공지
401	UNAUTHORIZED	인증 필요
403	NOT_ENROLLED	수강생/담당교수가 아님
404	NOTICE_NOT_FOUND	공지사항을 찾을 수 없음

---

7. 대댓글 작성

공지사항 댓글에 대댓글을 작성합니다. 최대 2depth까지 지원합니다.

Request

POST /api/v1/courses/{courseId}/notices/{noticeId}/comments/{parentId}/replies

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID
noticeId	long	O	공지사항 ID
parentId	long	O	부모 댓글 ID

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Request Body


```
{
  "content": "1주차부터 7주차까지입니다."
}
```


Response


```
{
  "success": true,
  "data": {
    "id": 2,
    "noticeId": 1,
    "parentId": 1,
    "content": "1주차부터 7주차까지입니다.",
    "authorId": 20250101002,
    "authorName": "김교수",
    "createdAt": "2025-04-02T15:00:00",
    "updatedAt": "2025-04-02T15:00:00",
    "children": []
  },
  "message": "답글이 작성되었습니다."
}
```

Error Codes

HTTP Status	에러 코드	설명
400	INVALID_INPUT	입력값 유효성 검증 실패
400	COMMENTS_NOT_ALLOWED	댓글이 허용되지 않은 공지
401	UNAUTHORIZED	인증 필요
404	COMMENT_NOT_FOUND	부모 댓글을 찾을 수 없음

---

8. 댓글 수정

공지사항 댓글을 수정합니다. 작성자만 가능합니다.

Request

PUT /api/v1/courses/{courseId}/notices/{noticeId}/comments/{commentId}

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID
noticeId	long	O	공지사항 ID
commentId	long	O	댓글 ID

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Request Body

```
{
  "content": "시험 범위가 1~7주차라고 공지에 있네요!"
}
```

Response


```
{
 "success": true,
 "data": {
  "id": 1,
  "noticeId": 1,
  "parentId": null,
  "content": "시험 범위가 1~7주차라고 공지에 있네요!",
  "authorId": 20250101012,
  "authorName": "홍길동",
  "createdAt": "2025-04-02T14:30:00",
  "updatedAt": "2025-04-02T16:00:00",
  "children": []
 },
 "message": "댓글이 수정되었습니다."
}
```


Error Codes

HTTP Status	에러 코드	설명
400	INVALID_INPUT	입력값 유효성 검증 실패
401	UNAUTHORIZED	인증 필요
403	NOT_AUTHOR	작성자가 아님
404	COMMENT_NOT_FOUND	댓글을 찾을 수 없음

---

9. 댓글 삭제

공지사항 댓글을 삭제합니다 (Soft Delete). 작성자만 가능합니다.

Request

DELETE /api/v1/courses/{courseId}/notices/{noticeId}/comments/{commentId}

Path Parameters

파라미터	타입	필수	설명
courseId	long	O	강의 ID
noticeId	long	O	공지사항 ID
commentId	long	O	댓글 ID

Headers

헤더	필수	설명
Authorization	O	Bearer {accessToken}

Response

```
{
  "success": true,
  "message": "댓글이 삭제되었습니다."
}
```

Error Codes

HTTP Status	에러 코드	설명
401	UNAUTHORIZED	인증 필요
403	NOT_AUTHOR	작성자가 아님
404	COMMENT_NOT_FOUND	댓글을 찾을 수 없음

데이터 모델

CourseNoticeResponse

필드	타입	설명
id	number	공지사항 ID
courseId	number	강의 ID
title	string	제목
allowComments	boolean	댓글 허용 여부
authorId	number	작성자 ID
authorName	string	작성자 이름
createdAt	datetime	생성일시
updatedAt	datetime	수정일시

CourseNoticeDetailResponse

필드	타입	설명
id	number	공지사항 ID

필드	타입	설명
courseId	number	강의 ID
title	string	제목
content	string	내용
allowComments	boolean	댓글 허용 여부
authorId	number	작성자 ID
authorName	string	작성자 이름
createdAt	datetime	생성일시
updatedAt	datetime	수정일시
comments	CourseNoticeCommentResponse[]	댓글 목록

CourseNoticeCommentResponse

필드	타입	설명
id	number	댓글 ID
noticeId	number	공지사항 ID
parentId	number | null	부모 댓글 ID (대댓글인 경우)
content	string	댓글 내용
authorId	number	작성자 ID
authorName	string	작성자 이름
createdAt	datetime	생성일시
updatedAt	datetime	수정일시
children	CourseNoticeCommentResponse[]	대댓글 목록

권한 정리

API	교수 (담당)	교수 (비담당)	학생 (수강)	학생 (미수강)
공지 생성	O	X	X	X
공지 조회	O	X	O	X
공지 수정	O	X	X	X
공지 삭제	O	X	X	X
댓글 작성	O	X	O	X
댓글 수정	본인만	-	본인만	-
댓글 삭제	본인만	-	본인만	-

문서 작성일: 2025-12-23

Enrollment API

> 수강신청 API

목차

기간 조회

- 1. 현재 활성 기간 조회

강의 조회

- 2. 수강신청용 강의 목록 조회

수강신청

- 3. 일괄 수강신청
- 4. 일괄 수강신청 취소

- 5. 내 수강신청 목록 조회

장바구니

- 6. 장바구니 조회
- 7. 장바구니 일괄 추가
- 8. 장바구니 일괄 삭제
- 9. 장바구니 전체 비우기

---

1. 현재 활성 기간 조회

현재 활성화된 수강신청 기간을 조회합니다.

Request

GET /api/v1/enrollments/periods/current

Query Parameters

파라미터	타입	필수	설명
type	string	X	기간 타입 (기본: ENROLLMENT)

Period Types

타입	설명
ENROLLMENT	수강신청 기간
COURSE_REGISTRATION	강의 등록 기간
ADJUSTMENT	수강 정정 기간
CANCELLATION	수강 취소 기간

Response


```
{
  "success": true,
  "data": {
    "periodId": 1,
    "periodType": "ENROLLMENT",
    "academicTermId": 1,
    "academicTermName": "2025학년도 1학기",
    "startDate": "2025-02-01T09:00:00",
    "endDate": "2025-02-10T18:00:00",
    "isActive": true
```

```
  }
}
```

---

2. 수강신청용 강의 목록 조회

수강신청 가능한 강의 목록을 조회합니다.

Request

GET /api/v1/enrollments/courses

Query Parameters

파라미터	타입	필수	설명
enrollmentPeriodId	long	O	수강신청 기간 ID
keyword	string	X	검색어
departmentId	long	X	학과 ID
courseType	int	X	강의 유형
credits	int	X	학점
page	int	X	페이지 번호
size	int	X	페이지 크기
sort	string	X	정렬 기준

Response


```
{
  "success": true,
  "data": {
    "courses": [
      {
        "courseId": 1,
        "courseName": "자료구조",
        "professorName": "김교수",
        "credits": 3,
        "currentEnrollment": 25,
        "maxEnrollment": 40,
        "isInCart": false,
        "isEnrolled": false
      }
    ],
    "totalPages": 5,
    "totalElements": 100
  }
}
```

---

3. 일괄 수강신청

여러 강의를 한 번에 수강신청합니다.

Request

POST /api/v1/enrollments/bulk

Request Body

```
{
  "courseIds": [1, 2, 3]
}
```

Response

```
{
  "success": true,
  "data": {
    "results": [
      {
        "courseId": 1,
        "courseName": "자료구조",
        "success": true,
        "enrollmentId": 100
      },
      {
        "courseId": 2,
        "courseName": "운영체제",
        "success": false,
        "reason": "정원 초과"
      }
    ],
    "summary": {
      "totalAttempted": 3,
      "successCount": 2,
      "failedCount": 1
    }
  },
  "message": "2개 과목 수강신청 완료, 1개 과목 실패"
}
```

---

4. 일괄 수강신청 취소

수강신청을 일괄 취소합니다.

Request

DELETE /api/v1/enrollments/bulk

Request Body

```
{
  "enrollmentIds": [100, 101]
}
```

Response

```
{
  "success": true,
  "data": {
    "results": [
      {
        "enrollmentId": 100,
        "courseName": "자료구조",
        "success": true
      }
    ],
    "summary": {
      "totalAttempted": 2,
      "successCount": 2,
      "failedCount": 0
    }
  },
  "message": "2개 과목 취소 완료"
}
```

---

5. 내 수강신청 목록 조회

현재 수강신청한 강의 목록을 조회합니다.

Request

GET /api/v1/enrollments/my

Query Parameters

파라미터	타입	필수	설명
enrollmentPeriodId	long	X	수강신청 기간 ID

Response


```
{
  "success": true,
  "data": {
    "enrollments": [
      {
        "enrollmentId": 100,
        "courseId": 1,
        "courseName": "자료구조",
        "professorName": "김교수",
        "credits": 3,
        "schedule": "월 09:00-10:30",
        "enrolledAt": "2025-02-01T09:30:00"
      }
    ],
    "totalCredits": 15,
    "totalCourses": 5
  }
}
```


---

6. 장바구니 조회

장바구니에 담긴 강의 목록을 조회합니다.

Request

GET /api/v1/carts

Response


```
{
  "success": true,
  "data": {
    "cartItems": [
      {
        "cartId": 1,
        "courseId": 1,
        "courseName": "자료구조",
        "professorName": "김교수",
        "credits": 3,
        "currentEnrollment": 25,
        "maxEnrollment": 40,
        "addedAt": "2025-01-30T10:00:00"
      }
    ],
    "totalCredits": 9,
    "totalItems": 3
```

```
  }
}
```

---

7. 장바구니 일괄 추가

여러 강의를 장바구니에 추가합니다.

Request

POST /api/v1/carts/bulk

Request Body

```
{
  "courseIds": [1, 2, 3]
}
```

Response

```
{
  "success": true,
  "data": {
    "results": [
      {
        "courseId": 1,
        "courseName": "자료구조",
        "success": true,
        "cartId": 1
      }
    ],
    "summary": {
      "successCount": 3,
      "failedCount": 0
    }
  }
}
```

---

8. 장바구니 일괄 삭제

장바구니에서 여러 항목을 삭제합니다.

Request

DELETE /api/v1/carts/bulk

Request Body

```
{
  "cartIds": [1, 2]
}
```

Response

```
{
  "success": true,
  "data": {
    "results": [
      {
        "cartId": 1,
        "success": true
      }
    ],
    "summary": {
      "successCount": 2,
      "failedCount": 0
    }
  }
}
```

---

9. 장바구니 전체 비우기

장바구니를 전체 비웁니다.

Request

DELETE /api/v1/carts

Response

```
{
  "success": true,
  "data": {
    "summary": {
      "successCount": 5,
      "failedCount": 0
    }
  }
}
```

Attendance API

> 출석 관리 API

목차

학생용

- 1. 내 전체 출석 현황 조회
- 2. 특정 강의 출석 현황 조회
- 3. 주차별 출석 상세 조회

교수용

- 4. 강의 전체 출석 현황 조회
- 5. 학생별 출석 목록 조회
- 6. 주차별 학생 출석 현황 조회

학생용 API

1. 내 전체 출석 현황 조회

수강 중인 모든 강의의 출석 현황을 조회합니다.

Request

GET /api/v1/attendance/my

Headers

Authorization: Bearer {accessToken}

Response


```
{
  "success": true,
  "data": [
    {
      "courseId": 1,
      "courseName": "자료구조",
      "professorName": "김교수",
      "totalWeeks": 15,
      "attendedWeeks": 10,
```

```
    "attendanceRate": 66.7,
    "status": "NORMAL"
   },
   {
    "courseId": 2,
    "courseName": "운영체제",
    "professorName": "이교수",
    "totalWeeks": 15,
    "attendedWeeks": 8,
    "attendanceRate": 53.3,
    "status": "WARNING"
   }
  ]
}
```


Status Types

상태	설명
NORMAL	정상 출석
WARNING	경고 (출석률 낮음)
FAIL	출석 미달

---

2. 특정 강의 출석 현황 조회

특정 강의의 상세 출석 현황을 조회합니다.

Request

GET /api/v1/attendance/courses/{courseId}

Path Parameters

파라미터	타입	설명
courseId	long	강의 ID

Response


```
{
 "success": true,
 "data": {
   "courseId": 1,
   "courseName": "자료구조",
   "professorName": "김교수",
   "totalWeeks": 15,
   "attendedWeeks": 10,
```

```
    "attendanceRate": 66.7,
    "weekAttendances": [
      {
        "weekId": 1,
        "weekNumber": 1,
        "title": "오리엔테이션",
        "attended": true,
        "attendedAt": "2025-03-03T09:15:00",
        "totalContents": 3,
        "completedContents": 3,
        "completionRate": 100.0
      },
      {
        "weekId": 2,
        "weekNumber": 2,
        "title": "배열과 연결리스트",
        "attended": false,
        "totalContents": 4,
        "completedContents": 1,
        "completionRate": 25.0
      }
    ]
  }
}
```


---

3. 주차별 출석 상세 조회

특정 주차의 출석 상세 정보를 조회합니다.

Request

GET /api/v1/attendance/courses/{courseId}/weeks/{weekId}

Path Parameters

파라미터	타입	설명
courseId	long	강의 ID
weekId	long	주차 ID

Response


```
{
  "success": true,
  "data": {
    "weekId": 1,
    "weekNumber": 1,
    "title": "오리엔테이션",
    "attended": true,
```

```
    "attendedAt": "2025-03-03T09:15:00",
    "totalContents": 3,
    "completedContents": 3,
    "completionRate": 100.0,
    "contents": [
      {
        "contentId": 1,
        "contentType": "VIDEO",
        "title": "강의 소개",
        "completed": true,
        "completedAt": "2025-03-03T09:15:00",
        "watchedDuration": 600,
        "totalDuration": 600
      }
    ]
  }
}
```

교수용 API

4. 강의 전체 출석 현황 조회

강의의 전체 출석 통계를 조회합니다.

Request

GET /api/v1/professor/courses/{courseId}/attendance

Path Parameters

파라미터	타입	설명
courseId	long	강의 ID

Response


```
{
  "success": true,
  "data": {
    "courseId": 1,
    "courseName": "자료구조",
    "totalStudents": 40,
    "totalWeeks": 15,
    "averageAttendanceRate": 85.5,
    "weeklyStats": [
      {
        "weekId": 1,
        "weekNumber": 1,
```

```
      "title": "오리엔테이션",
      "attendedCount": 38,
      "attendanceRate": 95.0
    }
  ],
  "statusSummary": {
    "normal": 35,
    "warning": 3,
    "fail": 2
  }
 }
}
```


---

5. 학생별 출석 목록 조회

강의에 등록된 학생들의 출석 현황을 조회합니다.

Request

GET /api/v1/professor/courses/{courseId}/attendance/students

Response


```
{
 "success": true,
 "data": [
   {
     "studentId": 2025010001,
     "studentNumber": "2025010001",
     "studentName": "홍길동",
     "departmentName": "컴퓨터공학과",
     "attendedWeeks": 10,
     "totalWeeks": 15,
     "attendanceRate": 66.7,
     "status": "NORMAL"
   }
 ]
}
```


---

6. 주차별 학생 출석 현황 조회

특정 주차의 학생별 출석 현황을 조회합니다.

Request

GET /api/v1/professor/courses/{courseId}/weeks/{weekId}/
attendance

Path Parameters

파라미터	타입	설명
courseId	long	강의 ID
weekId	long	주차 ID

Response


```
{
  "success": true,
  "data": [
    {
      "studentId": 2025010001,
      "studentNumber": "2025010001",
      "studentName": "홍길동",
      "attended": true,
      "attendedAt": "2025-03-03T09:15:00",
      "completedContents": 3,
      "totalContents": 3,
      "completionRate": 100.0
    },
    {
      "studentId": 2025010002,
      "studentNumber": "2025010002",
      "studentName": "김철수",
      "attended": false,
      "completedContents": 0,
      "totalContents": 3,
      "completionRate": 0.0
    }
  ]
}
```


Assignment API

> 과제 관리 API

목차

교수용

- 1. 과제 등록
- 2. 과제 수정
- 3. 과제 삭제
- 4. 제출 목록 조회

- 5. 과제 채점
- 6. 재제출 허용

학생용

- 7. 과제 목록 조회
- 8. 과제 상세 조회
- 9. 과제 제출
- 10. 내 제출 조회

교수용 API

1. 과제 등록

새로운 과제를 등록합니다.

Request

POST /api/v1/assignments

Request Body

```
{
  "courseId": 1,
  "title": "1주차 과제",
  "content": "과제 설명입니다.",
  "dueDate": "2025-03-15T23:59:59",
  "maxScore": 100,
  "attachmentIds": [1, 2]
}
```

Response

```
{
  "id": 1,
  "postId": 100,
  "courseId": 1,
  "courseName": "자료구조",
  "title": "1주차 과제",
  "content": "과제 설명입니다.",
  "dueDate": "2025-03-15T23:59:59",
  "maxScore": 100,
  "attachments": [...],
  "submissionCount": 0,
  "createdAt": "2025-03-01T10:00:00",
  "createdBy": 2024010001
}
```

---

2. 과제 수정

과제 정보를 수정합니다.

Request

PUT /api/v1/assignments/{id}

Request Body

```
{
  "title": "수정된 과제 제목",
  "content": "수정된 과제 설명",
  "dueDate": "2025-03-20T23:59:59",
  "maxScore": 100
}
```

---

3. 과제 삭제

과제를 삭제합니다.

Request

DELETE /api/v1/assignments/{id}

Response

HTTP/1.1 204 No Content

---

4. 제출 목록 조회

과제에 대한 모든 제출물을 조회합니다.

Request

GET /api/v1/assignments/{id}/submissions

Response

```
[
  {
    "submissionId": 1,
    "assignmentId": 1,
```

```
    "studentId": 2025010001,
    "studentNumber": "2025010001",
    "studentName": "홍길동",
    "content": "제출 내용",
    "attachments": [...],
    "submittedAt": "2025-03-10T15:30:00",
    "score": null,
    "feedback": null,
    "status": "SUBMITTED"
  },
  {
    "submissionId": 2,
    "assignmentId": 1,
    "studentId": 2025010002,
    "studentNumber": "2025010002",
    "studentName": "김철수",
    "submittedAt": "2025-03-12T10:00:00",
    "score": 85,
    "feedback": "잘 했습니다.",
    "status": "GRADED"
  }
]
```


Status Types

상태	설명
SUBMITTED	제출됨
GRADED	채점 완료
RESUBMISSION_ALLOWED	재제출 허용됨

---

5. 과제 채점

제출된 과제를 채점합니다.

Request

PUT /api/v1/assignments/submissions/{submissionId}/grade

Request Body


```
{
  "score": 85,
  "feedback": "잘 했습니다. 다음에는 코드 주석을 더 추가해주세요."
}
```

Response


```
{
  "submissionId": 1,
  "assignmentId": 1,
  "studentName": "홍길동",
  "score": 85,
  "maxScore": 100,
  "feedback": "잘 했습니다. 다음에는 코드 주석을 더 추가해주세요.",
  "gradedAt": "2025-03-15T14:00:00",
  "gradedBy": 2024010001,
  "status": "GRADED"
}
```


---

6. 재제출 허용

학생에게 과제 재제출을 허용합니다.

Request

POST /api/v1/assignments/submissions/{submissionId}/allow-resubmission

Query Parameters

파라미터	타입	필수	설명
deadline	datetime	X	재제출 마감일 (미지정 시 원래 마감일)

Response


```
{
  "submissionId": 1,
  "status": "RESUBMISSION_ALLOWED",
  "resubmissionDeadline": "2025-03-20T23:59:59",
  "message": "재제출이 허용되었습니다."
}
```

학생용 API

7. 과제 목록 조회

과제 목록을 조회합니다.

Request

GET /api/v1/assignments

Query Parameters

파라미터	타입	필수	설명
courseId	long	X	강의 ID (미지정 시 전체)

Response


```
[
  {
    "id": 1,
    "postId": 100,
    "courseId": 1,
    "courseName": "자료구조",
    "title": "1주차 과제",
    "dueDate": "2025-03-15T23:59:59",
    "maxScore": 100,
    "submissionCount": 25,
    "createdAt": "2025-03-01T10:00:00"
  }
]
```


---

8. 과제 상세 조회

과제 상세 정보를 조회합니다.

Request

GET /api/v1/assignments/{id}

Response


```
{
  "id": 1,
  "postId": 100,
  "courseId": 1,
  "courseName": "자료구조",
  "title": "1주차 과제",
  "content": "과제 설명입니다.",
  "dueDate": "2025-03-15T23:59:59",
  "maxScore": 100,
  "attachments": [
    {
      "attachmentId": 1,
```

```
    "originalName": "과제안내.pdf",
    "downloadUrl": "/api/v1/attachments/1/download"
   }
  ],
  "createdAt": "2025-03-01T10:00:00"
}
```


---

9. 과제 제출

과제를 제출합니다.

Request

POST /api/v1/assignments/{id}/submit

Request Body


```
{
  "content": "과제 제출 내용입니다.",
  "attachmentIds": [10, 11]
}
```


Response


```
{
  "submissionId": 1,
  "assignmentId": 1,
  "studentId": 2025010001,
  "content": "과제 제출 내용입니다.",
  "attachments": [...],
  "submittedAt": "2025-03-10T15:30:00",
  "status": "SUBMITTED"
}
```


Error Response

상태 코드	메시지
400	마감 기한이 지났습니다.
400	이미 제출한 과제입니다.

---

10. 내 제출 조회

내가 제출한 과제를 조회합니다.

Request

GET /api/v1/assignments/{id}/my-submission

Response


```
{
  "submissionId": 1,
  "assignmentId": 1,
  "content": "과제 제출 내용입니다.",
  "attachments": [...],
  "submittedAt": "2025-03-10T15:30:00",
  "score": 85,
  "maxScore": 100,
  "feedback": "잘 했습니다.",
  "status": "GRADED",
  "gradedAt": "2025-03-15T14:00:00"
}
```


Response (미제출 시)


```
{
  "submissionId": null,
  "status": "NOT_SUBMITTED",
  "message": "아직 제출하지 않았습니다."
}
```


Dashboard API

> 대시보드 API (학생용)

목차

- 1. 미제출 과제 목록 조회
- 2. 오늘의 강의 목록 조회
- 3. 최신 공지사항 목록 조회
- 4. 수강 현황 요약 조회

---

1. 미제출 과제 목록 조회

마감일이 임박한 미제출 과제 목록을 조회합니다.

Request

GET /api/v1/dashboard/student/pending-assignments

Headers

Authorization: Bearer {accessToken}

Query Parameters

파라미터	타입	필수	기본값	설명
days	int	X	7	마감일 기준 일수 (최대 30)

Response


```
{
  "success": true,
  "data": [
    {
      "assignmentId": 1,
      "courseId": 1,
      "courseName": "자료구조",
      "title": "1주차 과제",
      "dueDate": "2025-01-20T23:59:59",
      "daysRemaining": 3,
      "isUrgent": true
    },
    {
      "assignmentId": 2,
      "courseId": 2,
      "courseName": "운영체제",
      "title": "프로세스 스케줄링 레포트",
      "dueDate": "2025-01-25T23:59:59",
      "daysRemaining": 8,
      "isUrgent": false
    }
  ],
  "count": 2
}
```


Notes

- isUrgent: 마감까지 3일 이하이면 true
- daysRemaining: 마감까지 남은 일수 (음수면 마감 지남)

---

2. 오늘의 강의 목록 조회

오늘 수강해야 할 강의 목록을 조회합니다.

Request

GET /api/v1/dashboard/student/today-courses

Headers

Authorization: Bearer {accessToken}

Response


```
{
  "success": true,
  "data": [
    {
      "courseId": 1,
      "courseName": "자료구조",
      "professorName": "김교수",
      "schedule": "09:00-10:30",
      "classroom": "공학관 301호",
      "weekNumber": 3,
      "weekTitle": "스택과 큐",
      "hasNewContent": true,
      "attendanceStatus": "NOT_ATTENDED"
    },
    {
      "courseId": 2,
      "courseName": "운영체제",
      "professorName": "이교수",
      "schedule": "13:00-14:30",
      "classroom": "공학관 401호",
      "weekNumber": 3,
      "weekTitle": "프로세스 관리",
      "hasNewContent": false,
      "attendanceStatus": "ATTENDED"
    }
  ],
  "count": 2
}
```


Attendance Status

상태	설명
NOT_ATTENDED	미출석
ATTENDED	출석 완료

상태	설명
IN_PROGRESS	진행 중

---

3. 최신 공지사항 목록 조회

최신 공지사항을 조회합니다.

Request

GET /api/v1/dashboard/student/notices

Query Parameters

파라미터	타입	필수	기본값	설명
limit	int	X	5	조회할 개수 (최대 10)

Response


```
{
  "success": true,
  "data": [
    {
      "postId": 100,
      "title": "2025학년도 1학기 수강신청 안내",
      "authorName": "학사팀",
      "createdAt": "2025-01-10T10:00:00",
      "isImportant": true,
      "boardType": "notice"
    },
    {
      "postId": 99,
      "title": "도서관 운영시간 변경 안내",
      "authorName": "도서관",
      "createdAt": "2025-01-08T14:00:00",
      "isImportant": false,
      "boardType": "notice"
    }
  ],
  "count": 2
}
```


---

4. 수강 현황 요약 조회

현재 수강 중인 과목 수와 총 학점을 조회합니다.

Request

GET /api/v1/dashboard/student/enrollment-summary

Headers

Authorization: Bearer {accessToken}

Response


```
{
  "success": true,
  "data": {
    "totalCourses": 5,
    "totalCredits": 15,
    "maxCredits": 21,
    "coursesByType": {
      "major": 3,
      "general": 2
    },
    "creditsByType": {
      "major": 9,
      "general": 6
    },
    "averageAttendanceRate": 92.5,
    "completedAssignments": 8,
    "pendingAssignments": 2
  }
}
```


Response Fields

필드	설명
totalCourses	수강 중인 과목 수
totalCredits	현재 수강 학점
maxCredits	최대 수강 가능 학점
coursesByType	유형별 과목 수
creditsByType	유형별 학점
averageAttendanceRate	평균 출석률 (%)
completedAssignments	완료한 과제 수
pendingAssignments	미완료 과제 수

Board API (Community Boards)

> 커뮤니티 게시판 API (게시글, 댓글, 첨부파일)

목차

게시글

- 1. 게시글 생성
- 2. 게시글 상세 조회
- 3. 게시글 목록 조회
- 4. 게시글 수정
- 5. 게시글 삭제
- 6. 게시글 좋아요 토글
- 7. 게시글 좋아요 여부 조회

댓글

- 8. 댓글 생성
- 9. 댓글 목록 조회
- 10. 댓글 수정
- 11. 댓글 삭제

첨부파일

- 12. 단일 파일 업로드
- 13. 다중 파일 업로드
- 14. 파일 다운로드
- 15. 첨부파일 삭제

해시태그

- 16. 해시태그 검색/자동완성

게시판 분류

1. 기본 게시판 (Basic Boards)

일반적인 커뮤니티 기능을 제공하는 게시판

타입	설명	허용 게시글 타입	특징
NOTICE	학교 공지사항	NOTICE, URGENT	댓글 불가, 관리자만 작성
FREE	자유게시판	NORMAL, URGENT	해시태그 지원
QUESTION	질문게시판	NORMAL	해시태그 지원(과목별)
DISCUSSION	토론게시판	NORMAL	해시태그 지원(주제별)
DEPARTMENT	학과게시판	NORMAL, NOTICE, URGENT	학과별 자동 필터링

2. 역할별 제한 게시판 (Role-Restricted Boards)

특정 역할만 접근 가능한 게시판

타입	설명	접근 권한	허용 게시글 타입
PROFESSOR	교수 게시판	교수만	NORMAL, NOTICE
STUDENT	학생 게시판	학생만	NORMAL

3. 특수목적 게시판 (Special Purpose Boards)

특별한 목적을 위한 게시판

타입	설명	허용 게시글 타입	특징
CONTEST	공모전게시판	NORMAL	해시태그(분야별 필터링)
CAREER	취업정보게시판	NORMAL	해시태그(카테고리별 필터링)

게시판 공통 기능

해시태그 지원

- **FREE**: 대학생활, 맛집추천, 동아리, 취미, 일상, 고민상담
- **QUESTION**: 과제질문, 시험준비, 개념이해, 문제풀이, 학습자료, 프로젝트
- **DISCUSSION**: AI윤리토론, 기후변화대응, 교육개혁, 메타버스 미래 등

- **CONTEST**: it/소프트웨어, 디자인, 마케팅, 아이디어, 창업, 사회혁신
- **CAREER**: 채용공고, 면접후기, 인턴, 자소서첨삭, 포트폴리오, 이력서

권한 기반 접근 제어 (RBAC)

- **PROFESSOR 게시판**: 교수 역할만 접근 가능
- **STUDENT 게시판**: 학생 역할만 접근 가능
- **DEPARTMENT 게시판**: 사용자 소속 학과 게시글만 필터링

학과별 자동 필터링 (DEPARTMENT 게시판)

- 학생: 소속 학과 게시글만 조회/작성
- 교수: 소속 학과 게시글만 조회/작성
- 게시글 작성 시 자동으로 사용자 학과 ID 설정

게시글 API

1. 게시글 생성

새로운 게시글을 작성합니다.

Request

POST /api/v1/boards/{boardType}/posts

Path Parameters

파라미터	타입	설명
boardType	string	게시판 타입

Request Body

```
{
  "title": "게시글 제목",
  "content": "게시글 내용",
  "hashtags": ["태그1", "태그2"],
  "attachmentIds": [1, 2, 3]
}
```

Response


```
{
 "postId": 1,
 "title": "게시글 제목",
 "content": "게시글 내용",
 "authorId": 2025010001,
 "authorName": "홍길동",
 "boardType": "free",
 "hashtags": ["태그1", "태그2"],
 "attachments": [...],
 "viewCount": 0,
 "likeCount": 0,
 "commentCount": 0,
 "createdAt": "2025-01-15T10:00:00"
}
```


---

2. 게시글 상세 조회

게시글 상세 정보를 조회합니다.

Request

GET /api/v1/boards/{boardType}/posts/{id}

Response


```
{
 "postId": 1,
 "title": "게시글 제목",
 "content": "게시글 내용",
 "authorId": 2025010001,
 "authorName": "홍길동",
 "authorProfileImageUrl": "...",
 "boardType": "free",
 "hashtags": ["태그1", "태그2"],
 "attachments": [
  {
   "attachmentId": 1,
   "originalName": "file.pdf",
   "fileSize": 1024,
   "downloadUrl": "/api/v1/attachments/1/download"
  }
 ],
 "viewCount": 100,
 "likeCount": 10,
 "commentCount": 5,
 "isLiked": false,
 "isAuthor": true,
 "createdAt": "2025-01-15T10:00:00",
```

```
  "updatedAt": "2025-01-15T10:00:00"
}
```


---

3. 게시글 목록 조회

게시글 목록을 조회합니다. (검색, 해시태그 필터링 지원)

Request

GET /api/v1/boards/{boardType}/posts

Query Parameters

파라미터	타입	필수	기본값	설명
search	string	X	-	검색어 (제목, 내용)
hashtag	string	X	-	해시태그 필터
page	int	X	0	페이지 번호
size	int	X	20	페이지 크기
sort	string	X	createdAt,DESC	정렬 기준

Response


```
{
  "content": [
    {
      "postId": 1,
      "title": "게시글 제목",
      "authorName": "홍길동",
      "boardType": "free",
      "hashtags": ["태그1"],
      "viewCount": 100,
      "likeCount": 10,
      "commentCount": 5,
      "hasAttachment": true,
      "createdAt": "2025-01-15T10:00:00"
    }
  ],
  "totalElements": 100,
  "totalPages": 5,
  "number": 0,
  "size": 20
}
```


---

4. 게시글 수정

게시글을 수정합니다.

Request (JSON)

PUT /api/v1/boards/posts/{id}
Content-Type: application/json

Request (Multipart - 파일 포함)

PUT /api/v1/boards/posts/{id}
Content-Type: multipart/form-data

Request Body

```
{
  "title": "수정된 제목",
  "content": "수정된 내용",
  "hashtags": ["태그1", "태그3"],
  "attachmentIds": [1, 4],
  "deleteAttachmentIds": [2, 3]
}
```

---

5. 게시글 삭제

게시글을 삭제합니다.

Request

DELETE /api/v1/boards/posts/{id}

Response

HTTP/1.1 204 No Content

---

6. 게시글 좋아요 토글

게시글 좋아요를 토글합니다.

Request

POST /api/v1/boards/posts/{id}/like?userId={userId}

Response

```
{
  "liked": true,
  "message": "좋아요가 등록되었습니다."
}
```

---

7. 게시글 좋아요 여부 조회

게시글 좋아요 여부를 확인합니다.

Request

GET /api/v1/boards/posts/{id}/liked?userId={userId}

Response

```
{
  "liked": true
}
```

댓글 API

8. 댓글 생성

댓글을 작성합니다. (대댓글 지원)

Request

POST /api/v1/board/comments

Request Body

```
{
  "postId": 1,
  "content": "댓글 내용",
  "parentCommentId": null
}
```

> parentCommentId를 지정하면 대댓글로 작성됩니다.

Response


```
{
  "commentId": 1,
  "postId": 1,
  "content": "댓글 내용",
  "authorId": 2025010001,
  "authorName": "홍길동",
  "authorProfileImageUrl": "...",
  "parentCommentId": null,
  "depth": 0,
  "createdAt": "2025-01-15T10:30:00"
}
```


---

9. 댓글 목록 조회

게시글의 댓글 목록을 조회합니다.

Request

GET /api/v1/board/comments?postId={postId}

Response


```
[
  {
    "commentId": 1,
    "content": "댓글 내용",
    "authorName": "홍길동",
    "authorProfileImageUrl": "...",
    "depth": 0,
    "createdAt": "2025-01-15T10:30:00",
    "replies": [
      {
        "commentId": 2,
        "content": "대댓글 내용",
        "authorName": "김철수",
        "depth": 1,
        "createdAt": "2025-01-15T10:35:00"
      }
    ]
  }
]
```


---

10. 댓글 수정

댓글을 수정합니다.

Request

PUT /api/v1/board/comments/{id}

Request Body

```
{
  "content": "수정된 댓글 내용"
}
```

---

11. 댓글 삭제

댓글을 삭제합니다.

Request

DELETE /api/v1/board/comments/{id}

Response

HTTP/1.1 204 No Content

첨부파일 API

12. 단일 파일 업로드

파일을 업로드합니다.

Request

POST /api/v1/attachments/upload
Content-Type: multipart/form-data

Form Data

필드	타입	필수	설명
file	file	O	업로드할 파일
attachmentType	string	O	첨부 유형 (POST, ASSIGNMENT, PROFILE)

Response


```
{
  "status": "SUCCESS",
  "message": "파일이 업로드되었습니다.",
  "data": {
    "attachmentId": 1,
    "originalName": "document.pdf",
    "storedName": "uuid-document.pdf",
    "fileSize": 1024000,
    "mimeType": "application/pdf",
    "downloadUrl": "/api/v1/attachments/1/download"
  }
}
```


---

13. 다중 파일 업로드

여러 파일을 한 번에 업로드합니다.

Request

POST /api/v1/attachments/upload/multiple
Content-Type: multipart/form-data

Form Data

필드	타입	필수	설명
files	file[]	O	업로드할 파일들
attachmentType	string	O	첨부 유형

Response


```
{
  "status": "SUCCESS",
  "message": "3개 파일이 업로드되었습니다.",
  "data": [
    {
      "attachmentId": 1,
      "originalName": "file1.pdf",
      ...
    }
  ]
}
```


---

14. 파일 다운로드

첨부파일을 다운로드합니다.

Request

GET /api/v1/attachments/{attachmentId}/download

Response

Content-Type: application/pdf
Content-Disposition: attachment; filename="document.pdf"

[Binary Data]

---

15. 첨부파일 삭제

첨부파일을 삭제합니다.

Request

DELETE /api/v1/attachments/{attachmentId}

Response


```
{
  "status": "SUCCESS",
  "message": "파일이 삭제되었습니다."
}

{
  "status": "SUCCESS",
  "message": "파일이 삭제되었습니다."
}
```

해시태그 API

16. 해시태그 검색/자동완성

키워드를 기반으로 해시태그를 검색합니다. (자동완성용)

Request

GET /api/v1/hashtags/search

Query Parameters

파라미터	타입	필수	기본값	설명
keyword	string	X	""	검색 키워드

Response


```
[
  {
    "id": 1,
    "name": "대학생활",
    "displayName": "대학생활",
    "description": "대학 생활 관련 이야기",
    "postCount": 45,
    "isActive": true,
    "createdAt": "2025-01-15T10:00:00"
  },
  {
    "id": 2,
    "name": "맛집추천",
    "displayName": "맛집추천",
    "description": "맛집 정보 및 추천",
    "postCount": 32,
    "isActive": true,
    "createdAt": "2025-01-15T10:00:00"
  }
]
```


특징

- 최대 10개 결과 반환
- name과 displayName 모두에서 LIKE 검색
- 활성 상태(isActive=true) 해시태그만 조회
- 키워드가 비어있으면 빈 배열 반환

사용 예시

```
# 해시태그 자동완성
GET /api/v1/hashtags/search?keyword=대학

# 모든 해시태그 조회 (빈 키워드)
GET /api/v1/hashtags/search

# 특정 키워드로 검색
GET /api/v1/hashtags/search?keyword=프로젝트
```

📝 참고사항

게시글 목록 조회 시 해시태그 필터링

```
# 해시태그로 게시글 필터링 (기존 API 활용)
GET /api/v1/boards/{boardType}/posts?hashtag=대학생활
```

미구현 기능 (향후 개발 필요)

- 게시판별 해시태그 목록 조회
- AI 기반 해시태그 추천
- 인기 해시태그 조회

Message API

> 메시지/대화 API

목차

대화방

- 1. 대화방 목록 조회
- 2. 대화방 생성/조회
- 3. 대화방 상세 조회
- 4. 대화방 삭제
- 5. 대화방 읽음 처리
- 6. 전체 읽지 않은 메시지 수 조회

메시지

- 7. 메시지 발송
- 8. 메시지 일괄 발송
- 9. 메시지 목록 조회
- 10. 메시지 삭제
- 11. 메시지 읽음 처리

실시간 알림

- 12. SSE 구독

대화방 API

1. 대화방 목록 조회

내 대화방 목록을 조회합니다.

Request

GET /api/v1/conversations

Headers

Authorization: Bearer {accessToken}

Response

```
[
  {
    "conversationId": 1,
    "otherUserId": 2025010002,
    "otherUserName": "김철수",
    "otherUserProfileImageUrl": "...",
    "otherUserType": "STUDENT",
    "lastMessage": "네, 알겠습니다!",
    "lastMessageAt": "2025-01-15T14:30:00",
    "unreadCount": 3
  }
]
```

---

2. 대화방 생성/조회

특정 사용자와의 대화방을 생성하거나 기존 대화방을 조회합니다.

Request

POST /api/v1/conversations/with/{otherUserId}

Path Parameters

파라미터	타입	설명
otherUserId	long	상대방 사용자 ID

Response


```
{
  "conversationId": 1,
  "otherUserId": 2025010002,
  "otherUserName": "김철수",
  "otherUserProfileImageUrl": "...",
  "otherUserType": "STUDENT",
  "createdAt": "2025-01-15T10:00:00"
}
```


---

3. 대화방 상세 조회

대화방 상세 정보를 조회합니다.

Request

GET /api/v1/conversations/{conversationId}

Response


```
{
  "conversationId": 1,
  "otherUserId": 2025010002,
  "otherUserName": "김철수",
  "otherUserProfileImageUrl": "...",
  "otherUserType": "STUDENT",
  "createdAt": "2025-01-15T10:00:00"
}
```


---

4. 대화방 삭제

대화방을 삭제합니다. (본인 기준으로만 삭제)

Request

DELETE /api/v1/conversations/{conversationId}

Response

HTTP/1.1 204 No Content

---

5. 대화방 읽음 처리

대화방의 모든 메시지를 읽음 처리합니다.

Request

POST /api/v1/conversations/{conversationId}/read

Response

HTTP/1.1 200 OK

---

6. 전체 읽지 않은 메시지 수 조회

모든 대화방의 읽지 않은 메시지 총 개수를 조회합니다.

Request

GET /api/v1/conversations/unread-count

Response

5

메시지 API

7. 메시지 발송

메시지를 발송합니다.

Request

POST /api/v1/messages

Request Body

```
{
  "conversationId": 1,
  "content": "안녕하세요!"
}
```

Response

```
{
  "messageId": 100,
  "conversationId": 1,
  "senderId": 2025010001,
```

```
  "senderName": "홍길동",
  "content": "안녕하세요!",
  "sentAt": "2025-01-15T14:30:00",
  "isRead": false
}
```


---

8. 메시지 일괄 발송

여러 사용자에게 동일한 메시지를 발송합니다.

Request

POST /api/v1/messages/bulk

Request Body


```
{
  "receiverIds": [2025010002, 2025010003, 2025010004],
  "content": "공지사항입니다."
}
```


Response


```
[
  {
    "messageId": 101,
    "conversationId": 1,
    "receiverId": 2025010002,
    "content": "공지사항입니다.",
    "sentAt": "2025-01-15T14:30:00"
  },
  {
    "messageId": 102,
    "conversationId": 2,
    "receiverId": 2025010003,
    "content": "공지사항입니다.",
    "sentAt": "2025-01-15T14:30:00"
  }
]
```


---

9. 메시지 목록 조회

대화방의 메시지 목록을 조회합니다. (커서 기반 페이징)

Request

GET /api/v1/messages/conversations/{conversationId}

Query Parameters

파라미터	타입	필수	기본값	설명
cursor	long	X	-	커서 (이전 응답의 nextCursor)
size	int	X	20	페이지 크기

Response


```
{
 "messages": [
   {
     "messageId": 100,
     "senderId": 2025010001,
     "senderName": "홍길동",
     "senderProfileImageUrl": "...",
     "content": "안녕하세요!",
     "sentAt": "2025-01-15T14:30:00",
     "isRead": true,
     "isMine": true
   },
   {
     "messageId": 99,
     "senderId": 2025010002,
     "senderName": "김철수",
     "content": "네, 안녕하세요!",
     "sentAt": "2025-01-15T14:25:00",
     "isRead": true,
     "isMine": false
   }
 ],
 "nextCursor": 98,
 "hasNext": true
}
```


---

10. 메시지 삭제

메시지를 삭제합니다. (본인이 보낸 메시지만 삭제 가능)

Request

DELETE /api/v1/messages/{messageId}

Response

HTTP/1.1 204 No Content

---

11. 메시지 읽음 처리

대화방의 메시지들을 읽음 처리합니다.

Request

POST /api/v1/messages/conversations/{conversationId}/read

Response

HTTP/1.1 200 OK

실시간 알림 API

12. SSE 구독

Server-Sent Events를 구독하여 실시간 알림을 받습니다.

Request

GET /api/v1/sse/subscribe

Headers

Authorization: Bearer {accessToken}
Accept: text/event-stream

Response (Event Stream)

event: message
data:
{"type":"NEW_MESSAGE","conversationId":1,"messageId":100,"sender
길동","content":"안녕하세요!"}

event: notification
data: {"type":"ASSIGNMENT_DUE","title":"과제 마감 알림","content":"자
료구조 과제 마감이 1시간 남았습니다."}

Event Types

이벤트	설명
message	새 메시지 도착
notification	시스템 알림

이벤트	설명
read	메시지 읽음 처리됨

연결 유지

- 기본 타임아웃: 30분
- 클라이언트는 연결이 끊어지면 자동 재연결 필요
- 주기적으로 heartbeat 이벤트 발송

Notification API

> 알림 API (커서 기반 페이징)

목차

- 1. 알림 목록 조회
- 2. 알림 상세 조회
- 3. 읽지 않은 알림 개수 조회
- 4. 알림 읽음 처리
- 5. 모든 알림 읽음 처리
- 6. 알림 삭제
- 7. 읽은 알림 일괄 삭제
- 8. 모든 알림 삭제

---

1. 알림 목록 조회

알림 목록을 조회합니다. (커서 기반 페이징)

Request

GET /api/notifications

Headers

Authorization: Bearer {accessToken}

Query Parameters

파라미터	타입	필수	기본값	설명
cursor	long	X	-	커서 (이전 응답의 nextCursor)
size	int	X	20	페이지 크기 (최대 100)
unreadOnly	boolean	X	false	읽지 않은 알림만 조회

Response


```
{
  "notifications": [
    {
      "notificationId": 100,
      "type": "ASSIGNMENT_DUE",
      "title": "과제 마감 알림",
      "content": "자료구조 1주차 과제 마감이 1시간 남았습니다.",
      "isRead": false,
      "createdAt": "2025-01-15T13:00:00",
      "link": "/courses/1/assignments/1",
      "metadata": {
        "courseId": 1,
        "assignmentId": 1
      }
    },
    {
      "notificationId": 99,
      "type": "NEW_MESSAGE",
      "title": "새 메시지",
      "content": "김교수님이 메시지를 보냈습니다.",
      "isRead": true,
      "createdAt": "2025-01-15T12:30:00",
      "link": "/messages/conversations/5"
    }
  ],
  "nextCursor": 98,
  "hasNext": true,
  "totalUnread": 5
}
```


Notification Types

타입	설명
ASSIGNMENT_DUE	과제 마감 알림
ASSIGNMENT_GRADED	과제 채점 완료
NEW_MESSAGE	새 메시지

타입	설명
COURSE_NOTICE	강의 공지
ENROLLMENT_SUCCESS	수강신청 성공
ENROLLMENT_FAILED	수강신청 실패
SYSTEM	시스템 알림

---

2. 알림 상세 조회

알림 상세 정보를 조회합니다.

Request

GET /api/notifications/{notificationId}

Response


```
{
  "notificationId": 100,
  "type": "ASSIGNMENT_DUE",
  "title": "과제 마감 알림",
  "content": "자료구조 1주차 과제 마감이 1시간 남았습니다.",
  "isRead": false,
  "createdAt": "2025-01-15T13:00:00",
  "link": "/courses/1/assignments/1",
  "metadata": {
    "courseId": 1,
    "courseName": "자료구조",
    "assignmentId": 1,
    "assignmentTitle": "1주차 과제"
  }
}
```


---

3. 읽지 않은 알림 개수 조회

읽지 않은 알림의 개수를 조회합니다.

Request

GET /api/notifications/unread-count

Response


```
{
  "unreadCount": 5
}
```


---

4. 알림 읽음 처리

특정 알림을 읽음 처리합니다.

Request

PATCH /api/notifications/{notificationId}/read

Response


```
{
  "notificationId": 100,
  "type": "ASSIGNMENT_DUE",
  "title": "과제 마감 알림",
  "isRead": true,
  "readAt": "2025-01-15T14:00:00"
}
```


---

5. 모든 알림 읽음 처리

모든 알림을 읽음 처리합니다.

Request

PATCH /api/notifications/read-all

Response


```
{
  "success": true,
  "message": "모든 알림이 읽음 처리되었습니다.",
  "updatedCount": 5
}
```


---

6. 알림 삭제

특정 알림을 삭제합니다.

Request

DELETE /api/notifications/{notificationId}

Response

```
{
  "success": true,
  "message": "알림이 삭제되었습니다."
}
```

---

7. 읽은 알림 일괄 삭제

읽은 알림을 일괄 삭제합니다.

Request

DELETE /api/notifications/read

Response

```
{
  "success": true,
  "message": "읽은 알림이 삭제되었습니다.",
  "deletedCount": 10
}
```

---

8. 모든 알림 삭제

모든 알림을 삭제합니다.

Request

DELETE /api/notifications/all

Response

```
{
  "success": true,
```

```
  "message": "모든 알림이 삭제되었습니다.",
  "deletedCount": 25
}
```


Subject API

> 과목 관리 API

목차

- 1. 과목 목록 조회
- 2. 과목 상세 조회
- 3. 과목 검색

---

1. 과목 목록 조회

과목 목록을 조회합니다.

Request

GET /api/v1/subjects

Query Parameters

파라미터	타입	필수	기본값	설명
page	int	X	0	페이지 번호
size	int	X	20	페이지 크기
keyword	string	X	-	검색어 (과목명, 과목코드)
departmentId	long	X	-	학과 ID
showAllDepartments	boolean	X	false	전체 학과 표시
courseType	string	X	-	강의 유형
credits	int	X	-	학점
isActive	boolean	X	true	활성 과목만 조회

Response


```
{
  "success": true,
  "data": {
    "content": [
      {
        "subjectId": 1,
        "subjectCode": "CS101",
        "subjectName": "자료구조",
        "departmentName": "컴퓨터공학과",
        "credits": 3,
        "courseType": "전공필수",
        "description": "자료구조의 기본 개념"
      }
    ],
    "totalElements": 50,
    "totalPages": 3,
    "currentPage": 0,
    "size": 20,
    "hasNext": true,
    "hasPrevious": false
  }
}
```


---

2. 과목 상세 조회

과목 상세 정보를 조회합니다.

Request

GET /api/v1/subjects/{subjectId}

Path Parameters

파라미터	타입	설명
subjectId	long	과목 ID

Response


```
{
  "success": true,
  "data": {
    "subjectId": 1,
    "subjectCode": "CS101",
    "subjectName": "자료구조",
    "departmentId": 1,
```

```
    "departmentName": "컴퓨터공학과",
    "credits": 3,
    "courseType": "전공필수",
    "description": "자료구조의 기본 개념과 알고리즘을 학습합니다.",
    "prerequisiteSubjects": [
      {
        "subjectId": 2,
        "subjectName": "프로그래밍 기초"
      }
    ],
    "isActive": true
  }
}
```


---

3. 과목 검색

과목을 검색합니다. (페이징 지원)

Request

GET /api/v1/subjects/search

Query Parameters

파라미터	타입	필수	기본값	설명
q	string	O	-	검색어
page	int	X	0	페이지 번호
size	int	X	20	페이지 크기 (최대 50)

Response


```
{
  "success": true,
  "data": {
    "content": [
      {
        "subjectId": 1,
        "subjectCode": "CS101",
        "subjectName": "자료구조",
        "departmentName": "컴퓨터공학과",
        "credits": 3
      }
    ],
    "totalElements": 10,
    "totalPages": 1,
    "currentPage": 0,
```

```
    "size": 20,
    "hasNext": false,
    "hasPrevious": false
  }
}
```