Student Assistant API Dokümantasyonu - Tam Sürüm

Genel Bilgiler

Base URL: `http://localhost:8000`
API Sürümü: 1.0.0
Dokümantasyon Endpoints:

- `/docs` - Swagger UI
- `/redoc` - ReDoc
- `/openapi.json` - OpenAPI Schema

Kimlik Doğrulama

Tüm korumalı endpoint'ler Bearer Token kimlik doğrulaması kullanır:

```
Authorization: Bearer {token}
```

Token, `/login` endpoint'inden alınır ve kalıcıdır (expire olmaz).

---

1. AUTH (Kimlik Doğrulama)

1.1 Register (Kayıt Ol)

POST `/register`

Yeni kullanıcı kaydı oluşturur. Varsayılan rol: `student`

Request Body:

```
{
  "username": "string",  // Min 3 karakter, özel karaktersiz
  "password": "string"   // Min 4 karakter
}
```

Response (200):

```
{
  "ok": true,
  "username": "john_doe",
  "token": "abc123..."
}
```

Errors:

- 400: Geçersiz kullanıcı adı veya şifre
- 409: Kullanıcı adı zaten kayıtlı

1.2 Login (Giriş Yap)

POST `/login`

Kullanıcı girişi yapar ve token döndürür.

Request Body:

```
{
  "username": "string",
  "password": "string"
}
```

Response (200):

```
{
  "ok": true,
  "username": "john_doe",
  "token": "abc123...",
  "roles": ["student"],
  "awarded_badges": ["first_login"]
}
```

Errors:

- 401: Yanlış kullanıcı adı veya şifre

1.3 Assign Role (Rol Atama)

POST `/assign_role`

Yetki: Admin

Kullanıcıya rol atar.

Request Body:

```
{
  "username": "string",
  "role": "student|instructor|admin"
}
```

Response (200):

```
{
  "ok": true,
  "username": "john_doe",
  "roles": ["student", "instructor"]
}
```

---

2. PROFILE (Profil Yönetimi)

2.1 Get My Profile

GET `/profile/me`

Yetki: Giriş yapmış kullanıcı

Response (200):

```
{
  "username": "john_doe",
  "email": "john@example.com",
  "full_name": "John Doe",
  "phone": "+90 555 123 4567",
  "bio": "Computer Science Student",
  "avatar_url": "/static/profiles/images/avatar.jpg",

  // Öğretmen alanları
  "title": "Dr.",
  "department": "Computer Science",
  "specialization": ["AI", "Machine Learning"],
  "office_location": "Block A, Room 203",
  "office_hours": "Mon-Wed 14:00-16:00",

  // Öğrenci alanları
  "student_id": "2020123456",
  "grade_level": "3",
  "major": "Computer Engineering",
  "enrollment_year": 2020,

  // Ortak alanlar
  "social_links": {
    "linkedin": "https://linkedin.com/in/johndoe",
    "github": "https://github.com/johndoe"
  },
  "preferences": {
    "theme": "dark",
    "language": "tr"
  },
  "created_at": "2024-01-01T10:00:00Z",
  "updated_at": "2024-01-15T15:30:00Z"
}
```

2.2 Update My Profile

PATCH `/profile/me`

Yetki: Giriş yapmış kullanıcı

Request Body:

```
  {
    "email": "newemail@example.com",
    "full_name": "John Doe",
    "bio": "Updated bio",
    "department": "CS",
    "specialization": ["AI", "ML", "Deep Learning"]
  }
```

2.3 Get User Profile

GET `/profile/{username}`

Yetki: Giriş yapmış kullanıcı

Başka bir kullanıcının public profilini görüntüler.

2.4 Get Preferences

GET `/profile/preferences`

Kullanıcı tercihlerini getirir.

2.5 Update Preferences

PATCH `/profile/preferences`

Request Body:

```
{
  "theme": "dark",
  "notifications": true,
  "language": "tr"
}
```

2.6 Get Profile Stats

GET `/profile/stats`

Profil tamamlanma oranı ve istatistikleri.

Response:

```
{
  "completion_rate": 85.5,
  "has_avatar": true,
  "profile_views": 42,
  "last_updated": "2024-01-15T15:30:00Z"
}
```

2.7 Get Public Instructor Profiles

GET `/profile/instructors/public`

Tüm öğretmenlerin public profillerini listeler.

2.8 Upload Avatar

POST `/profile/avatar`

Request: `multipart/form-data`

- `file` : Resim dosyası (max 5MB, jpg/png/gif/webp)

Response:

```
  {
    "success": true,
    "avatar_url": "/static/profiles/images/abc123.jpg",
    "message": "Profil fotoğrafı başarıyla güncellendi"
  }
```

2.9 Delete Avatar

DELETE `/profile/avatar`

2.10 Change Email

POST `/profile/change-email`

Request Body:

```
  {
    "new_email": "newemail@example.com",
    "current_password": "password123"
  }
```

2.11 Change Password

POST `/profile/change-password`

Request Body:

```
{
  "current_password": "oldpass",
  "new_password": "newpass123"
}
```

2.12 Update Office Hours

PATCH `/profile/instructor/office-hours`

Yetki: Instructor/Admin

Request Body:

```
{
  "office_hours": "Mon-Wed 14:00-16:00, Fri 10:00-12:00"
}
```

---

3. NOTES (Not Sistemi)

3.1 Get Notes

GET `/notes/`

Query Parameters:

- `category` : Kategori filtresi
- `search` : Başlık ve içerik araması
- `is_pinned` : Sabitlenmiş notlar (boolean)

- `is_archived` : Arşivlenmiş notlar (boolean)
- `tags` : Etiket filtresi (virgülle ayrılmış)
- `order_by` : created_at|updated_at|title
- `order_desc` : Azalan sıralama (boolean)

Response:

```
[
  {
    "id": "uuid",
    "title": "Ders Notları",
    "content": "İçerik...",
    "category": "dersler",
    "color": "#fef3c7",
    "is_pinned": true,
    "is_archived": false,
    "tags": ["matematik", "calculus"],
    "created_at": "2024-01-01T10:00:00Z",
    "updated_at": "2024-01-02T10:00:00Z"
  }
]
```

3.2 Create Note

POST `/notes/`

Request Body:

```
{
  "title": "Yeni Not",
  "content": "Not içeriği",
  "category": "genel",
  "color": "#fef3c7",
  "is_pinned": false,
  "tags": ["önemli", "todo"]
}
```

3.3 Get Note

GET `/notes/{note_id}`

3.4 Update Note

PATCH `/notes/{note_id}`

3.5 Delete Note

DELETE `/notes/{note_id}`

3.6 Pin/Unpin Note

POST `/notes/{note_id}/pin`

3.7 Archive Note

POST `/notes/{note_id}/archive`

3.8 Get Categories

GET `/notes/categories/list`

3.9 Get Tags

GET `/notes/tags/list`

3.10 Get Notes Stats

GET `/notes/stats`

Response:

```
{
  "total_notes": 42,
  "pinned_notes": 5,
  "archived_notes": 10,
  "categories": [
    {"name": "dersler", "count": 15},
    {"name": "projeler", "count": 8}
  ],
  "recent_activity": []
}
```

3.11 Bulk Delete Notes

POST `/notes/bulk/delete`

Request Body:

```
{
  "note_ids": ["id1", "id2", "id3"]
}
```

3.12 Bulk Archive Notes

POST `/notes/bulk/archive`

Request Body:

```
{
  "note_ids": ["id1", "id2"],
  "archive": true
}
```

3.13 Export Notes

POST `/notes/export`

Request Body:

```
{
  "format": "json|txt|md",
  "include_archived": false
}
```

3.14 Import Notes

POST `/notes/import`

Request Body:

```
{
  "notes_data": [],
  "merge_strategy": "replace|merge|skip"
}
```

---

4. CALENDAR (Takvim)

4.1 Get Events

GET `/calendar/events`

Query Parameters:

- `start_date` : Başlangıç tarihi (ISO format)
- `end_date` : Bitiş tarihi
- `event_type` : personal|course|meeting|exam|assignment|other
- `course_id` : Kurs ID filtresi
- `include_cancelled` : İptal edilmiş etkinlikleri dahil et

4.2 Create Event

POST `/calendar/events`

Request Body:

```
{
  "title": "Matematik Sınavı",
  "description": "Final sınavı",
  "start_datetime": "2025-02-01T10:00:00",
  "end_datetime": "2025-02-01T12:00:00",
  "all_day": false,
  "event_type": "exam",
  "location": "Salon A",
  "color": "#3b82f6",
  "recurring": false,
  "reminder_minutes": [15, 60, 1440],
  "course_id": "math101",
  "participants": ["user1", "user2"]
}
```

4.3 Get Event

GET `/calendar/events/{event_id}`

4.4 Update Event

PATCH `/calendar/events/{event_id}`

4.5 Delete Event

DELETE `/calendar/events/{event_id}`

4.6 Cancel Event

POST `/calendar/events/{event_id}/cancel`

4.7 Today's Agenda

GET `/calendar/agenda/today`

4.8 Week Agenda

GET `/calendar/agenda/week`

4.9 Calendar Stats

GET `/calendar/stats`

4.10 Get Reminders

GET `/calendar/reminders`

Query Parameters:

- `hours_ahead` : Kaç saat öncesine kadar (default: 24)

4.11 Bulk Create Events

POST `/calendar/events/bulk`

4.12 Bulk Delete Events

DELETE `/calendar/events/bulk`

4.13 Check Conflicts

GET `/calendar/conflicts`

Query Parameters:

- `start_datetime` : Kontrol edilecek başlangıç
- `end_datetime` : Kontrol edilecek bitiş
- `exclude_event_id` : Hariç tutulacak etkinlik

4.14 Import Calendar

POST `/calendar/import`

4.15 Export Calendar

GET `/calendar/export`

Query Parameters:

- `format` : json|ical|csv

---

5. UPLOADS (Dosya Yönetimi)

5.1 Upload File

POST `/uploads/`

Request: `multipart/form-data`

- `file` : Dosya (max 50MB)
- `description` : Açıklama
- `tags` : Etiketler (virgülle ayrılmış)
- `course_id` : İlişkili kurs
- `assignment_id` : İlişkili ödev
- `access_level` : private|course|public

5.2 List Files

GET `/uploads/`

Query Parameters:

- `category` : image|document|spreadsheet|video|audio|code|other
- `course_id` : Kurs filtresi
- `search` : Dosya adı araması
- `tags` : Etiket filtresi
- `limit` : Sayfa başına dosya (max: 100)
- `offset` : Başlangıç noktası

5.3 Get File Info

GET `/uploads/{file_id}`

5.4 Update File Info

PATCH `/uploads/{file_id}`

5.5 Delete File

DELETE `/uploads/{file_id}`

5.6 Download File

GET `/uploads/download/{file_id}`

5.7 Get Categories Stats

GET `/uploads/categories/stats`

5.8 Bulk Delete Files

POST `/uploads/bulk/delete`

5.9 List Public Files

GET `/uploads/public/list`

5.10 Download Public File

GET `/uploads/public/download/{file_id}`

---

6. SESSIONS (Oturumlar)

6.1 List Sessions

GET `/sessions`

Response:

```
{
  "ok": true,
  "sessions": [
    {
      "session_id": "uuid",
      "title": "Python Soruları",
      "created_at": "2024-01-01T10:00:00Z",
      "messages_count": 15
    }
  ],
  "awarded_badges": []
}
```

6.2 Get Session

GET `/sessions/{session_id}`

6.3 Get Active Sessions

GET `/sessions/active`

6.4 Clear Session

DELETE `/sessions/{session_id}`

---

7. ASK (Sohbet Botu - Topic-Free)

7.1 Ask Question

POST `/ask`

Topic-free AI soru-cevap sistemi. Qdrant vektör DB ve OpenAI entegrasyonu ile çalışır.

Request Body:

```
{
  "question": "Python'da liste comprehension nedir?",
  "session_id": "optional-session-id",
  "username": "john_doe"
}
```

Response:

```
  {
    "ok": true,
    "session_id": "abc123",
    "message_id": "msg456",
    "answer": "Liste comprehension, Python'da tek satırda liste oluşturmanı
    "source": "qdrant|llm|cache",
    "score": 0.85,
    "from_cache": false,
    "llm_used": false,
    "cached_question": null,
    "awarded_badges": []
  }
```

7.2 Test Endpoint

GET `/test`

7.3 Get Stats

GET `/stats`

Sistem istatistiklerini getirir.

7.4 Ask Smart (Backward Compat)

POST `/ask/smart`

7.5 Ask Auto (Backward Compat)

POST `/ask/auto`

7.6 Analyze Session

GET `/session/analysis/{session_id}`

---

8. FEEDBACK

8.1 Submit Feedback

POST `/feedback`

Mesaj beğenisi. Dislike durumunda OpenAI ile alternatif cevap üretir.

Request Body:

```
{
  "session_id": "abc123",
  "message_id": "msg456",
  "like": false
}
```

Response:

```
{
  "ok": true,
  "regenerated": true,
  "source": "llm_feedback",
  "qa_file": "/data/generated/topic_qa.json",
  "answer": "Alternatif cevap...",
  "awarded_badges": []
}
```

---

9. CODE (Kod Asistanı)

9.1 Generate Code

POST `/code`

Request Body:

```
{
  "session_id": "optional",
  "prompt": "Python'da fibonacci fonksiyonu yaz",
  "topic": "code"
}
```

9.2 Stream Code

POST `/code_stream`

Token-by-token streaming response.

---

10. QUIZ (Sınav Sistemi - Topic-Free)

10.1 Generate Quiz

POST `/quiz/generate`

Oturumdaki sorulardan veya Qdrant'tan quiz oluşturur.

Request Body:

```
{
  "session_id": "abc123"
}
```

Response:

```
{
  "ok": true,
  "quiz": {
    "quiz_id": "quiz789",
    "session_id": "abc123",
    "size": 10,
    "time_per_question_sec": 60,
    "time_limit_sec": 600,
    "items": [
      {
        "qid": "q1",
        "question": "Python'da hangi veri tipi değiştirilemez (immutable)
        "options": ["Liste", "Tuple", "Dictionary", "Set"],
        "correct_index": 1
      }
    ]
  },
  "message": "10 soruluk quiz oluşturuldu",
  "awarded_badges": []
}
```

10.2 Start Attempt

POST `/quiz/attempt/start`

Request Body:

```
  {
    "session_id": "abc123",
    "quiz_id": "quiz789"
  }
```

10.3 Submit Quiz

POST `/quiz/submit`

Request Body:

```
  {
    "session_id": "abc123",
    "quiz_id": "quiz789",
    "answers": [0, 1, 2, 3, 1, 2, 0, 3, 2, 1],
    "attempt_id": "attempt456"
  }
```

Response:

```
  {
    "ok": true,
    "score": {
      "correct": 7,
      "total": 10
    },
    "results": [],
    "attempt_id": "attempt456",
    "time_exceeded": false,
    "finished_at": "2024-01-01T10:30:00Z",
    "awarded_badges": ["quiz_master"],
    "points_earned": 70
  }
```

10.4 Abandon Quiz

POST `/quiz/abandon`

---

11. TOPICS (Konu Yönetimi)

11.1 Create Topic

POST `/topics/create`

AI ile QA ve keyword üretimi yapar, Qdrant'a yükler.

Request Body:

```
  {
    "topic": "Machine Learning",
    "qa_count": 20,
    "dry_run": false
  }
```

11.2 Generate Topic Content

POST `/topics/topics/generate`

Backward compatibility endpoint.

11.3 List Topics

GET `/topics/list`

---

12. SORULAB (AI Destekli Soru Laboratuvarı)

12.1 Create SoruLab

POST `/sorulab/create`

Yetki: Instructor/Admin

AI ile soru seti oluşturur. Chatbot, quiz veya her ikisi için kullanılabilir.

Request Body:

```
{
  "topic": "Veri Yapıları",
  "description": "Stack ve Queue konuları",
  "qa_count": 20,
  "mode": "both",  // chatbot|quiz|both
  "difficulty": "medium",  // easy|medium|hard
  "question_type": "mixed",  // multiple_choice|open_ended|true_false|mi
  "course_id": "cs101",
  "auto_keywords": true,
  "keyword_count": 30,
  "add_to_chatbot": true,
  "language": "tr"
}
```

Response:

```
{
  "ok": true,
  "sorulab_id": "sl123",
  "topic": "Veri Yapıları",
  "slug": "veri-yapilari",
  "questions_count": 20,
  "keywords_count": 30,
  "in_qdrant": true,
  "message": "20 soru başarıyla oluşturuldu",
  "awarded_badges": []
}
```

12.2 List SoruLab Sets

GET `/sorulab/list`

Query Parameters:

- `course_id` : Kurs filtresi
- `search` : Arama metni

12.3 Get SoruLab Detail

GET `/sorulab/{set_id}`

Detaylı soru listesi ve metadata.

12.4 Update Question

PATCH `/sorulab/{set_id}/questions/{question_id}`

Request Body:

```
  {
    "question": "Güncellenmiş soru metni",
    "answer": "Güncellenmiş cevap",
    "options": ["A", "B", "C", "D"],
    "correct_option_index": 2,
    "difficulty": "hard",
    "tags": ["önemli", "final"],
    "explanation": "Açıklama..."
  }
```

12.5 Delete Question

DELETE `/sorulab/{set_id}/questions/{question_id}`

12.6 Assign to Students

POST `/sorulab/{set_id}/assign`

Request Body:

```
{
  "student_ids": ["user1", "user2", "user3"],
  "course_id": "cs101",
  "due_date": "2025-02-01T23:59:59",
  "as_quiz": true,
  "quiz_settings": {
    "time_limit": 30,
    "shuffle_questions": true
  },
  "message": "Lütfen quiz'i tamamlayın"
}
```

12.7 Export SoruLab

GET `/sorulab/{set_id}/export`

Query Parameters:

- `format` : json|txt

12.8 Delete SoruLab

DELETE `/sorulab/{set_id}`

---

13. FOCUS (Odaklanma Modülü)

13.1 Start Timer

POST `/focus/start`

Odaklanma zamanlayıcısını başlatır.

Request Body:

```
{
  "planned_minutes": 45,
  "note": "Matematik çalışması",
  "tasks": [
    {
      "label": "Calculus Chapter 1",
      "category": "study",
      "planned_minutes": 25
    },
    {
      "label": "Practice Problems",
      "category": "exercise",
      "planned_minutes": 20
    }
  ]
}
```

Response:

```
{
  "session_id": "focus123",
  "started_at": "2024-01-01T10:00:00Z",
  "planned_minutes": 45
}
```

13.2 Append Task

POST `/focus/append-task`

Request Body:

```
{
  "session_id": "focus123",
  "task": {
    "label": "Review notes",
    "category": "review",
    "planned_minutes": 10
  }
}
```

13.3 Finish Timer

POST `/focus/finish`

Request Body:

```
{
  "session_id": "focus123",
  "duration_minutes": 47,
  "note": "Tamamlandı",
  "tasks": [
    {
      "label": "Calculus Chapter 1",
      "category": "study",
      "actual_minutes": 27
    }
  ]
}
```

Response:

```
{
  "status": "ok",
  "added_minutes": 47,
  "tasks_recorded": 2,
  "awarded_badges": ["focus_master"],
  "points_earned": 10
}
```

13.4 Get Stats

GET `/focus/stats`

Response:

```
{
  "total_minutes": 450,
  "today_minutes": 90,
  "task_totals": {
    "study": 200,
    "exercise": 150,
    "review": 100
  }
}
```

---

14. BADGES (Rozet Sistemi)

14.1 My Badges

GET `/badges`

Response:

```
{
  "ok": true,
  "badges": ["explorer", "quiz_master", "focus_warrior"],
  "progress": {
    "quiz_attempts": 5,
    "sessions_count": 20,
    "focus_hours": 10
  },
  "all_badges": {
    "explorer": {
      "id": "explorer",
      "name": "Keşifçi",
      "description": "İlk keşif",
      "category": "exploration"
    }
  }
}
```

14.2 Badge Catalog

GET `/badges/catalog`

14.3 Weekly Performance

POST `/badges/weekly-performance`

Yetki: Admin

14.4 Leaderboard Position

POST `/badges/leaderboard-position`

Yetki: Admin

14.5 Award Teacher Badge

POST `/badges/teacher-badge`

Yetki: Instructor

14.6 Record Chatbot Error

POST `/badges/chatbot-error`

14.7 Record Study Session

POST `/badges/study-session`

14.8 Record Overtime Study

POST `/badges/overtime-study`

14.9 Session Abandoned

POST `/badges/session-abandoned`

14.10 Consecutive Chat

POST `/badges/consecutive-chat`

Yetki: Admin

15. RESOURCES (Kaynak Arama)

15.1 Search Resources

GET `/resources/search`

ArXiv ve Google Scholar'dan kaynak arar.

Query Parameters:

- `q` : Arama sorgusu (min 2 karakter)
- `limit` : Sonuç limiti (1-50)
- `sources` : Kaynak filtreleri (arxiv,scholar)

Response:

```
{
  "ok": true,
  "resources": [
    {
      "source": "arxiv",
      "id": "2401.12345",
      "title": "Deep Learning for NLP",
      "summary_first_line": "This paper presents...",
      "authors": ["John Doe", "Jane Smith"],
      "language": "en",
      "published": "2024-01-15",
      "url": "https://arxiv.org/abs/2401.12345"
    }
  ],
  "awarded_badges": []
}
```

15.2 Resource Read

POST `/resources/read`

Kaynak okunduğunda puan ve rozet kazandırır.

Request Body:

```
{
  "resource_id": "2401.12345",
  "topic": "machine_learning",
  "duration_minutes": 30
}
```

---

16. INSTRUCTORS (Eğitmen Modülü)

16.1 SORULAR

Send Question to Student

POST `/instructors/questions/send`

Yetki: Instructor/Admin

Öğrenciye soru gönderir.

Request Body:

```
{
  "student": "student_username",
  "course_id": "cs101",
  "title": "Ödev Hakkında",
  "detail": "Ödevin 3. sorusunu anlamadım...",
  "priority": "normal"
}
```

List Incoming Questions

GET `/instructors/questions/inbox`

Query Parameters:

- `status` : bekliyor|yanitlandi|onaylandi|reddedildi
- `course_id` : Kurs filtresi

List Outgoing Questions

GET `/instructors/questions/outgoing`

Respond to Question

POST `/instructors/questions/{qid}/respond`

Request Body:

```
{
  "response": "Cevap metni...",
  "status": "yanitlandi"
}
```

Update Question Status

PATCH `/instructors/questions/{qid}/status`

16.2 KURSLAR

Add Course

POST `/instructors/courses`

Request Body:

```
{
  "id": "cs101",
  "name": "Veri Yapıları ve Algoritmalar",
  "description": "Temel veri yapıları ve algoritma analizi",
  "days": ["Pazartesi", "Çarşamba", "Cuma"],
  "time": "10:00-12:00",
  "total_hours": 60,
  "start_date": "2025-02-01",
  "end_date": "2025-06-01",
  "max_students": 40,
  "objectives": "Öğrenciler temel veri yapılarını öğrenecek",
  "importance": "Yazılım geliştirme için temel",
  "outcomes": [
    "Veri yapılarını tanıma",
    "Algoritma karmaşıklığı analizi",
    "Problem çözme becerileri"
    ]
  }
```

List Courses

GET `/instructors/courses`

Query Parameters:

- `q` : Kurs adı araması
- `status` : yeni|devam|tamamlandi
- `order` : created_at|name|student_count

Get Course Detail

GET `/instructors/courses/{course_id}`

Update Course

PATCH `/instructors/courses/{course_id}`

Update Course Status

PATCH `/instructors/courses/{course_id}/status`

Request Body:

```
{
  "status": "devam"
}
```

Upload Course Image

POST `/instructors/courses/{course_id}/image`

Request: `multipart/form-data`

- `file` : Görsel dosyası (max 5MB)

Get Course Image

GET `/instructors/courses/{course_id}/image`

16.3 MÜFREDAT

Create Curriculum

POST `/instructors/courses/{course_id}/curriculum`

Request Body:

```
[
  {
    "title": "Hafta 1: Giriş",
    "description": "Dersin tanıtımı ve temel kavramlar",
    "order": 1
  },
  {
    "title": "Hafta 2: Arrays ve Linked Lists",
    "description": "Temel veri yapıları",
    "order": 2
  }
]
```

Get Curriculum

GET `/instructors/courses/{course_id}/curriculum`

Add Curriculum Item

POST `/instructors/courses/{course_id}/curriculum/item`

Update Curriculum Item

PATCH `/instructors/courses/{course_id}/curriculum/{item_id}`

Delete Curriculum Item

DELETE `/instructors/courses/{course_id}/curriculum/{item_id}`

Update Curriculum Status

PATCH `/instructors/courses/{course_id}/curriculum/{item_id}/status`

Request Body:

```
{
  "status": "tamamlandi"
}
```

16.4 İLERLEME

Get Course Progress

GET `/instructors/courses/{course_id}/progress`

Update Course Progress

PATCH `/instructors/courses/{course_id}/progress`

Request Body:

```
{
  "completed_hours": 30
}
```

16.5 ÖĞRENCİ YÖNETİMİ

Add Students to Course

POST `/instructors/courses/{course_id}/students`

Request Body:

```
[
  {
    "username": "student1",
    "full_name": "John Doe",
    "email": "john@example.com"
  }
]
```

List Course Students

GET `/instructors/courses/{course_id}/students`

Query Parameters:

- `q` : Öğrenci adı/username araması

Response:

```
[
  {
    "username": "student1",
    "full_name": "John Doe",
    "email": "john@example.com",
    "badges": 5,
    "time_spent_hours": 25,
    "solved_quizzes": 8,
    "last_activity": "2024-01-15T10:00:00Z",
    "status": "active"
  }
]
```

Remove Student from Course

DELETE `/instructors/courses/{course_id}/students/{username}`

List Student Sessions

GET `/instructors/students/{username}/sessions`

Get Student Performance

GET `/instructors/students/{username}/performance`

Get Student Weekly Activity

GET `/instructors/students/{username}/weekly-activity`

Query Parameters:

- `weeks` : Kaç haftalık veri (default: 4)

Get Student Badges

GET `/instructors/students/{username}/badges`

Get Student Comparative Stats

GET `/instructors/students/{username}/comparative-stats`

16.6 KURS DOKÜMANLARI

List Course Documents

GET `/instructors/courses/{course_id}/docs`

Query Parameters:

- `q` : Doküman adı araması
- `file_type` : Dosya türü filtresi
- `tag` : Etiket filtresi

Add Course Document

POST `/instructors/courses/{course_id}/docs`

Request Body:

```
{
  "title": "Ders Notları Hafta 1",
  "url": "https://example.com/notes.pdf",
  "description": "İlk hafta ders notları",
  "file_type": "pdf",
  "tags": ["week1", "introduction"]
}
```

Upload Course Document

POST `/instructors/courses/{course_id}/docs/upload`

Request: `multipart/form-data`

- `file` : Dosya (max 50MB)
- `title` : Başlık
- `description` : Açıklama
- `tags` : Etiketler (virgülle ayrılmış)

Download Course Document

GET `/instructors/courses/{course_id}/docs/{doc_id}/download`

Get Document Info

GET `/instructors/courses/{course_id}/docs/{doc_id}/info`

Delete Course Document

DELETE `/instructors/courses/{course_id}/docs/{doc_id}`

16.7 ÖDEVLENDİRME

Create Assignment

POST `/instructors/assignments`

Request Body:

```
{
  "course_id": "cs101",
  "title": "Ödev 1: Veri Yapıları",
  "description": "Stack ve Queue implementasyonu",
  "due_date": "2025-02-15",
  "max_score": 100,
  "instructions": "Python'da Stack ve Queue sınıfları yazın",
  "attachments": ["file_id_1", "file_id_2"]
}
```

List Assignments

GET `/instructors/assignments`

Query Parameters:

- `course_id` : Kurs filtresi
- `status` : draft|published|closed

Update Assignment Status

PATCH `/instructors/assignments/{assignment_id}/status`

16.8 DUYURULAR

Create Announcement

POST `/instructors/announcements`

Request Body:

```
{
  "title": "Sınav Tarihi Değişikliği",
  "content": "Final sınavı 20 Haziran'a ertelendi",
  "course_id": "cs101",
  "priority": "high",
  "expires_at": "2025-06-20T23:59:59",
  "is_pinned": true
}
```

List Announcements

GET `/instructors/announcements`

Delete Announcement

DELETE `/instructors/announcements/{announcement_id}`

16.9 KAYNAK KÜRASYONU

Curate Resource

POST `/instructors/resources/curate`

Request Body:

```
{
  "source": "manual",
  "title": "Python Tutorial",
  "url": "https://docs.python.org/3/tutorial/",
  "authors": ["Python Software Foundation"],
  "description": "Official Python tutorial",
  "tags": ["python", "programming", "beginner"],
  "difficulty": "beginner",
  "estimated_time": 120
}
```

List Curated Resources

GET `/instructors/resources/curated`

16.10 İSTATİSTİKLER

Instructor Dashboard

GET `/instructors/dashboard`

Response:

```
{
  "total_courses": 3,
  "total_students": 87,
  "active_questions": 12,
  "total_assignments": 15,
  "recent_activity": []
}
```

Bulk Import Students

POST `/instructors/bulk/students/import`

Export Course Students

GET `/instructors/export/students/{course_id}`

---

17. MESSAGES (Mesajlaşma Sistemi)

17.1 Send Message

POST `/messages/send`

Request Body:

```
{
  "recipient_username": "instructor1",
  "subject": "Ders Hakkında Soru",
  "content": "Merhaba hocam...",
  "priority": "normal",
  "parent_message_id": null
}
```

17.2 Get Inbox

GET `/messages/inbox`

Query Parameters:

- `page` : Sayfa numarası
- `limit` : Sayfa başına mesaj (max: 100)
- `unread_only` : Sadece okunmamış mesajlar

17.3 Get Sent Messages

GET `/messages/sent`

17.4 Search Messages

GET `/messages/search`

Query Parameters:

- `query` : Arama metni
- `message_type` : all|sent|received

17.5 Get Message Detail

GET `/messages/{message_id}`

17.6 Delete Message

DELETE `/messages/{message_id}`

17.7 Mark as Read

PATCH `/messages/mark-read`

Request Body:

```
{
  "message_ids": ["msg1", "msg2"]
}
```

17.8 Get Conversations

GET `/messages/conversations/list`

17.9 Get Available Recipients

GET `/messages/recipients/available`

17.10 Send with Attachment

POST `/messages/send-with-attachment`

Request: `multipart/form-data`

- `recipient_username` : Alıcı
- `subject` : Konu
- `content` : İçerik
- `priority` : Öncelik
- `file` : Dosya eki (max 10MB)

17.11 Download Attachment

GET `/messages/attachments/{file_id}/download`

17.12 Get Attachment Info

GET `/messages/attachments/{file_id}/info`

---

18. POINTS (Puan Sistemi)

18.1 Get My Points

GET `/points/me`

Response:

```
{
  "ok": true,
  "total": 250,
  "current_week": 45,
  "current_month": 180,
  "weekly_breakdown": {
    "sessions": 20,
    "quizzes": 15,
    "resources": 10
  },
  "weekly_progress": {
    "target": 100,
    "achieved": 45,
    "percentage": 45.0
  }
}
```

18.2 Get Leaderboard

GET `/points/leaderboard`

Query Parameters:

- `period` : week|month|all
- `limit` : Gösterilecek kullanıcı sayısı

18.3 Get Weekly Activity

GET `/points/weekly-activity`

18.4 Session Complete

POST `/points/session-complete`

18.5 Quiz Complete

POST `/points/quiz-complete`

18.6 Chat Complete

POST `/points/chat-complete`

18.7 Badge Earned

POST `/points/badge-earned`

18.8 Resource Read

POST `/points/resource-read`

18.9 Instructor Evaluation

POST `/points/instructor-evaluation`

Yetki: Instructor

18.10 Add Manual Points

POST `/points/admin/add-manual`

Yetki: Admin

18.11 Reset Points

POST `/points/admin/reset`

Yetki: Admin

18.12 Get All Users Points

GET `/points/admin/all-users`

Yetki: Admin

---

19. DASHBOARD

19.1 Dashboard Overview

GET `/dashboard/overview`

Response:

```
{
  "ok": true,
  "user": {
    "username": "john_doe",
    "full_name": "John Doe"
  },
  "stats": {
    "study_time": {
      "total": "45 saat 30 dakika",
      "week_change": "+8 saat bu hafta"
    },
    "quizzes": {
      "total": 12,
      "week_change": "+3 bu hafta"
    },
    "badges": {
      "total": 8,
      "week_change": "+2 bu hafta"
    },
    "chats": {
      "total": 35,
      "week_change": "+10 bu hafta"
    }
  },
  "points": {
    "current_week": 85,
    "current_month": 320,
    "total": 1250,
    "rank": 5
  },
  "leaderboard": [],
  "weekly_progress": {
    "Pzt": 100,
    "Sal": 80,
    "Çar": 60,
    "Per": 40
  }
}
```

19.2 Detailed Stats

GET `/dashboard/detailed-stats`

19.3 Weekly Performance

GET `/dashboard/weekly-performance`

19.4 Recent Activities

GET `/dashboard/recent-activities`

Query Parameters:

- `limit` : Aktivite sayısı (1-50)

19.5 User Goals

GET `/dashboard/goals`

---

20. DEBUG & HEALTH

20.1 List Routes

GET `/debug/routes`

Tüm route'ları listeler (geliştirme için).

20.2 Health Check

GET `/health`

Response:

```
{
  "status": "healthy",
  "service": "Student Assistant API"
}
```

Hata Kodları

Kod	Açıklama
200	Başarılı
201	Oluşturuldu
204	İçerik yok
400	Geçersiz istek
401	Yetkisiz (token gerekli)
403	Erişim reddedildi (yetersiz yetki)
404	Bulunamadı
409	Çakışma (örn: kullanıcı zaten var)
422	İşlenemeyen varlık
500	Sunucu hatası
502	Dış servis hatası

Limitler ve Kısıtlamalar

Dosya Yükleme Limitleri

- Profil fotoğrafı: 5MB
- Mesaj eki: 10MB
- Genel dosyalar: 50MB
- Kurs dokümanları: 50MB

Pagination Varsayılanları

- `page` : 1 (başlangıç)
- `limit` : 20 (varsayılan)
- `max_limit` : 100

Rate Limiting

Production ortamında uygulanması önerilir:

- Genel API: 100 req/dakika

- Auth endpoints: 10 req/dakika
- AI endpoints: 30 req/dakika

String Uzunlukları

- Username: 3-50 karakter
- Password: 4+ karakter
- Konu başlıkları: 2-200 karakter
- Quiz soruları: Max 50 adet/set
- Keywords: 5-50 adet

Teknolojiler

- **Framework:** FastAPI
- **Veritabanı:** JSON dosya sistemi
- **Vektör DB:** Qdrant
- **AI:** OpenAI GPT-4o-mini
- **Auth:** Bearer Token
- **CORS:** Tüm origin'lere açık

Ortam Değişkenleri

```
OPENAI_API_KEY=sk-...
QDRANT_HOST=localhost
QDRANT_PORT=6333
LOCAL_LLM=qwen2.5-coder:7b
DATA_DIR=./data
```

Docker Compose

```
services:
  api:
    build: .
    ports:
      - "8000:8000"
    environment:
      - OPENAI_API_KEY=${OPENAI_API_KEY}
    volumes:
      - ./data:/app/data

  qdrant:
    image: qdrant/qdrant
    ports:
      - "6333:6333"
    volumes:
      - ./qdrant_storage:/qdrant/storage
```

Notlar

1. **Token Yönetimi:** Token'lar kalıcıdır, expire olmaz. Production'da JWT ile değiştirilmeli.
2. **Veritabanı:** JSON dosya sistemi kullanılıyor. Production'da PostgreSQL önerilir.
3. **Dosya Depolama:** Lokal dosya sistemi. Production'da S3 veya benzeri kullanılmalı.
4. **CORS:** Tüm origin'lere açık. Production'da kısıtlanmalı.
5. **Rate Limiting:** Implementasyonu yok. Production'da eklenmeli.
6. **SSL/TLS:** HTTP kullanılıyor. Production'da HTTPS zorunlu olmalı.