AI-Based Automated STEM Evaluation System - API Specifications

1. API Overview

1.1 General Information

- **Base URL**: `https://api.stem-eval.example.com/api/v1`
- **Protocol**: HTTPS only (TLS 1.3)
- **Format**: JSON (Content-Type: application/json)
- **Authentication**: JWT Bearer tokens
- **API Version**: v1
- **Rate Limiting**: 100 requests per minute per user, 1000 per minute per IP

1.2 Common Response Formats

Success Response

```
{
  "success": true,
  "data": { ... },
  "meta": {
    "timestamp": "2025-10-18T10:30:00Z",
    "request_id": "abc-123-def-456"
  }
}
```

Error Response

```
{
  "success": false,
  "error": {
    "code": "VALIDATION_ERROR",
    "message": "Invalid input provided",
    "details": [
      {
        "field": "email",
        "message": "Email format is invalid"
      }
    ]
  },
  "meta": {
    "timestamp": "2025-10-18T10:30:00Z",
    "request_id": "abc-123-def-456"
  }
}
```

Paginated Response

```
{
  "success": true,
  "data": [ ... ],
  "pagination": {
    "total": 100,
    "page": 1,
    "per_page": 20,
    "total_pages": 5,
    "has_next": true,
    "has_prev": false
  },
  "meta": {
    "timestamp": "2025-10-18T10:30:00Z",
    "request_id": "abc-123-def-456"
  }
}
```

1.3 Common HTTP Status Codes

- **200 OK**: Successful GET, PUT, PATCH
- **201 Created**: Successful POST
- **204 No Content**: Successful DELETE
- **400 Bad Request**: Invalid input
- **401 Unauthorized**: Missing or invalid authentication
- **403 Forbidden**: Insufficient permissions
- **404 Not Found**: Resource not found
- **409 Conflict**: Resource conflict (e.g., duplicate)
- **422 Unprocessable Entity**: Validation errors
- **429 Too Many Requests**: Rate limit exceeded
- **500 Internal Server Error**: Server error
- **503 Service Unavailable**: Service temporarily unavailable

1.4 Common Query Parameters

- `page` (integer): Page number (1-indexed)
- `per_page` (integer): Items per page (default: 20, max: 100)
- `sort` (string): Sort field (e.g., `created_at`, `-score` for descending)
- `filter` (object): Filter criteria
- `fields` (array): Specific fields to return (sparse fieldsets)

2. Authentication API

2.1 Register User

POST `/auth/register`

Register a new user account.

Request Body:

```
{
  "email": "student@example.com",
  "password": "SecureP@ss123",
  "first_name": "John",
  "last_name": "Doe",
  "role": "student",
  "organization_id": "uuid-here"
}
```

Response (201 Created):

```
{
  "success": true,
  "data": {
    "user_id": "550e8400-e29b-41d4-a716-446655440000",
    "email": "student@example.com",
    "verification_sent": true,
    "message": "Please check your email to verify your account"
  }
}
```

Validation Rules: - `email`: Required, valid email format, unique - `password`: Required, min 8 characters, must include uppercase, lowercase, number, special char - `first_name`: Required, 1-50 characters - `last_name`: Required, 1-50 characters - `role`: Required, enum (student, coach, parent, admin) - `organization_id`: Required (UUID)

2.2 Login

POST `/auth/login`

Authenticate user and receive access tokens.

Request Body:

```
{
  "email": "student@example.com",
  "password": "SecureP@ss123"
}
```

Response (200 OK):

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

```
    "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
    "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
    "expires_in": 900,
    "token_type": "Bearer",
    "user": {
      "id": "550e8400-e29b-41d4-a716-446655440000",
      "email": "student@example.com",
      "first_name": "John",
      "last_name": "Doe",
      "role": "student",
      "organization_id": "org-uuid"
    }
  }
}
```

Error Response (401 Unauthorized):

```
{
  "success": false,
  "error": {
    "code": "INVALID_CREDENTIALS",
    "message": "Invalid email or password"
  }
}
```

2.3 Refresh Token

POST `/auth/refresh`

Obtain a new access token using refresh token.

Request Headers:

```
Authorization: Bearer <refresh_token>
```

Response (200 OK):

```
{
  "success": true,
  "data": {
    "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
    "expires_in": 900,
    "token_type": "Bearer"
  }
}
```

2.4 Logout

POST `/auth/logout`

Invalidate current session.

Request Headers:

`Authorization: Bearer <access_token>`

Response (200 OK):

```
{
  "success": true,
  "data": {
    "message": "Successfully logged out"
  }
}
```

2.5 Forgot Password

POST `/auth/forgot-password`

Request password reset email.

Request Body:

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

Response (200 OK):

```
{
  "success": true,
  "data": {
    "message": "If an account exists with this email, a reset link has been sent"
  }
}
```

2.6 Reset Password

POST `/auth/reset-password`

Reset password using token from email.

Request Body:

```
{
  "token": "reset-token-from-email",
```

```
  "new_password": "NewSecureP@ss123"
}
```

Response (200 OK):

```
{
  "success": true,
  "data": {
    "message": "Password successfully reset"
  }
}
```

2.7 Google OAuth Login

GET `/auth/google`

Redirect to Google OAuth consent screen.

Query Parameters: - `redirect_uri` (optional): Where to redirect after auth

Response: HTTP 302 redirect to Google

2.8 Google OAuth Callback

GET `/auth/google/callback`

Handle Google OAuth callback.

Query Parameters: - `code`: Authorization code from Google - `state`: CSRF token

Response: HTTP 302 redirect to frontend with tokens in URL or error

3. User Management API

3.1 Get Current User Profile

GET `/users/me`

Retrieve authenticated user's profile.

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "550e8400-e29b-41d4-a716-446655440000",
    "email": "student@example.com",
    "first_name": "John",
```

```
    "last_name": "Doe",
    "role": "student",
    "avatar_url": "https://storage.example.com/avatars/user.jpg",
    "organization": {
      "id": "org-uuid",
      "name": "Springfield High School"
    },
    "classes": [
      {
        "id": "class-uuid",
        "name": "Robotics 101"
      }
    ],
    "team": {
      "id": "team-uuid",
      "name": "Team 12345A",
      "role": "programmer"
    },
    "created_at": "2025-01-15T10:00:00Z",
    "updated_at": "2025-10-18T10:00:00Z"
  }
}
```

3.2 Update User Profile

PUT `/users/me`

Update authenticated user's profile.

Request Body:

```
{
  "first_name": "Jonathan",
  "last_name": "Doe",
  "bio": "Passionate robotics student",
  "notification_preferences": {
    "email_enabled": true,
    "submission_feedback": true,
    "forum_replies": true,
    "weekly_digest": false
  }
}
```

Response (200 OK):

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

```
  "data": {
    "id": "550e8400-e29b-41d4-a716-446655440000",
    "first_name": "Jonathan",
    "last_name": "Doe",
    "bio": "Passionate robotics student",
    "updated_at": "2025-10-18T10:30:00Z"
  }
}
```

3.3 Upload Avatar

POST `/users/me/avatar`

Upload profile picture.

Request: - Content-Type: `multipart/form-data` - Field name: `avatar` - Max size: 5MB - Formats: JPG, PNG, HEIC

Response (200 OK):

```
{
  "success": true,
  "data": {
    "avatar_url": "https://storage.example.com/avatars/user-uuid.jpg"
  }
}
```

3.4 Get User by ID

GET `/users/:id`

Retrieve user profile by ID.

Permissions: - Own profile: Always allowed - Other users: Coach/Admin only

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "550e8400-e29b-41d4-a716-446655440000",
    "first_name": "John",
    "last_name": "Doe",
    "role": "student",
    "avatar_url": "https://storage.example.com/avatars/user.jpg",
    "bio": "Passionate robotics student"
```

```
  }
}
```

3.5 List Users

GET `/users`

List users with filtering.

Permissions: Coach (own classes), Admin (all)

Query Parameters: - `role` (string): Filter by role - `class_id` (uuid): Filter by class - `search` (string): Search by name or email - `page`, `per_page`: Pagination

Response (200 OK):

```
{
  "success": true,
  "data": [
    {
      "id": "user-uuid-1",
      "first_name": "John",
      "last_name": "Doe",
      "email": "john@example.com",
      "role": "student",
      "avatar_url": "..."
    },
    ...
  ],
  "pagination": { ... }
}
```

3.6 Update User Role

PATCH `/users/:id/role`

Update user's role.

Permissions: Admin only

Request Body:

```
{
  "role": "coach"
}
```

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "user-uuid",
    "role": "coach",
    "updated_at": "2025-10-18T10:30:00Z"
  }
}
```

4. Class Management API

4.1 Create Class

POST `/classes`

Create a new class.

Permissions: Coach, Admin

Request Body:

```
{
  "name": "Robotics 101",
  "description": "Introduction to VEX robotics",
  "grade_level": "9-10",
  "school_year": "2025-2026",
  "organization_id": "org-uuid"
}
```

Response (201 Created):

```
{
  "success": true,
  "data": {
    "id": "class-uuid",
    "name": "Robotics 101",
    "description": "Introduction to VEX robotics",
    "grade_level": "9-10",
    "invite_code": "ABC123",
    "created_at": "2025-10-18T10:00:00Z"
  }
}
```

4.2 Get Class

GET `/classes/:id`

Retrieve class details.

Permissions: Students in class, Coach, Admin

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "class-uuid",
    "name": "Robotics 101",
    "description": "Introduction to VEX robotics",
    "grade_level": "9-10",
    "school_year": "2025-2026",
    "invite_code": "ABC123",
    "coaches": [
      {
        "id": "coach-uuid",
        "first_name": "Jane",
        "last_name": "Smith"
      }
    ],
    "student_count": 24,
    "created_at": "2025-09-01T10:00:00Z"
  }
}
```

4.3 Update Class

PUT `/classes/:id`

Update class information.

Permissions: Assigned coach, Admin

Request Body:

```
{
  "name": "Advanced Robotics",
  "description": "VEX V5 competition preparation"
}
```

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "class-uuid",
    "name": "Advanced Robotics",
    "description": "VEX V5 competition preparation",
```

```
    "updated_at": "2025-10-18T10:30:00Z"
  }
}
```

4.4 Add Students to Class

POST `/classes/:id/students`

Add one or more students to class.

Permissions: Assigned coach, Admin

Request Body:

```
{
  "student_ids": ["student-uuid-1", "student-uuid-2"]
}
```

Response (200 OK):

```
{
  "success": true,
  "data": {
    "added_count": 2,
    "students": [
      {
        "id": "student-uuid-1",
        "first_name": "John",
        "last_name": "Doe"
      },
      {
        "id": "student-uuid-2",
        "first_name": "Jane",
        "last_name": "Smith"
      }
    ]
  }
}
```

4.5 Remove Student from Class

DELETE `/classes/:id/students/:studentId`

Remove a student from class.

Permissions: Assigned coach, Admin

Response (204 No Content)

4.6 Get Class Students

GET `/classes/:id/students`

List all students in a class.

Permissions: Students in class, Assigned coach, Admin

Query Parameters: - `include_metrics` (boolean): Include performance metrics - `page`, `per_page`: Pagination

Response (200 OK):

```
{
  "success": true,
  "data": [
    {
      "id": "student-uuid",
      "first_name": "John",
      "last_name": "Doe",
      "email": "john@example.com",
      "avatar_url": "...",
      "team": {
        "id": "team-uuid",
        "name": "Team 12345A"
      },
      "metrics": {
        "overall_score": 85,
        "submission_count": 12,
        "assessment_completion_rate": 0.95
      }
    },
    ...
  ],
  "pagination": { ... }
}
```

4.7 Join Class with Invite Code

POST `/classes/join`

Student joins class using invite code.

Permissions: Student

Request Body:

```
{
  "invite_code": "ABC123"
}
```

Response (200 OK):

```
{
  "success": true,
  "data": {
    "class_id": "class-uuid",
    "name": "Robotics 101",
    "joined_at": "2025-10-18T10:30:00Z"
  }
}
```

5. Team Management API

5.1 Create Team

POST `/teams`

Create a VEX team.

Permissions: Coach, Student (with approval)

Request Body:

```
{
  "name": "Team 12345A",
  "team_number": "12345A",
  "class_id": "class-uuid",
  "competition_type": "VEX_V5",
  "description": "Our VEX V5 competition team"
}
```

Response (201 Created):

```
{
  "success": true,
  "data": {
    "id": "team-uuid",
    "name": "Team 12345A",
    "team_number": "12345A",
    "competition_type": "VEX_V5",
    "created_at": "2025-10-18T10:00:00Z"
  }
}
```

5.2 Add Team Member

POST `/teams/:id/members`

Add student to team.

Permissions: Team members, Coach

Request Body:

```
{
  "student_id": "student-uuid",
  "role": "programmer"
}
```

Roles: `driver`, `programmer`, `builder`, `designer`, `notebook`

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "member-uuid",
    "student": {
      "id": "student-uuid",
      "first_name": "John",
      "last_name": "Doe"
    },
    "role": "programmer",
    "joined_at": "2025-10-18T10:30:00Z"
  }
}
```

6. Assessment API

6.1 Create Assessment

POST `/assessments`

Create a new assessment.

Permissions: Coach, Admin

Request Body:

```
{
  "title": "Mid-term STEM Assessment",
  "description": "Evaluates programming and problem-solving skills",
  "question_pool_id": "pool-uuid",
  "question_count": 20,
  "difficulty_adaptation": true,
  "time_limit_minutes": 60,
```

```
  "passing_score": 70,
  "show_correct_answers": false
}
```

Response (201 Created):

```
{
  "success": true,
  "data": {
    "id": "assessment-uuid",
    "title": "Mid-term STEM Assessment",
    "question_count": 20,
    "created_at": "2025-10-18T10:00:00Z"
  }
}
```

6.2 Assign Assessment

POST `/assessments/:id/assign`

Assign assessment to students or classes.

Permissions: Coach, Admin

Request Body:

```
{
  "target_type": "class",
  "target_ids": ["class-uuid-1", "class-uuid-2"],
  "due_date": "2025-10-25T23:59:59Z",
  "available_from": "2025-10-18T00:00:00Z"
}
```

Target Types: `class`, `student`, `team`

Response (200 OK):

```
{
  "success": true,
  "data": {
    "assigned_count": 48,
    "due_date": "2025-10-25T23:59:59Z"
  }
}
```

6.3 Get Assigned Assessments

GET `/assessments/assigned`

List assessments assigned to current user.

Permissions: Student

Query Parameters: - `status` (string): `pending`, `in_progress`, `completed` - `page`, `per_page`: Pagination

Response (200 OK):

```
{
  "success": true,
  "data": [
    {
      "id": "assignment-uuid",
      "assessment": {
        "id": "assessment-uuid",
        "title": "Mid-term STEM Assessment",
        "question_count": 20,
        "time_limit_minutes": 60
      },
      "status": "pending",
      "due_date": "2025-10-25T23:59:59Z",
      "available_from": "2025-10-18T00:00:00Z",
      "attempts_remaining": 1
    },
    ...
  ],
  "pagination": { ... }
}
```

6.4 Start Assessment Session

POST `/assessment-sessions`

Begin taking an assessment.

Permissions: Student (if assigned)

Request Body:

```
{
  "assessment_id": "assessment-uuid"
}
```

Response (201 Created):

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

```
    "id": "session-uuid",
    "assessment_id": "assessment-uuid",
    "started_at": "2025-10-18T10:00:00Z",
    "expires_at": "2025-10-18T11:00:00Z",
    "current_question": {
      "id": "question-uuid",
      "sequence": 1,
      "type": "multiple_choice",
      "question_text": "What is the maximum number of motors allowed on a VEX V5 robot?",
      "options": [
        {"id": "opt-1", "text": "4"},
        {"id": "opt-2", "text": "8"},
        {"id": "opt-3", "text": "12"},
        {"id": "opt-4", "text": "16"}
      ]
    }
  }
}
```

6.5 Submit Answer

POST `/assessment-sessions/:id/submit-answer`

Submit answer for current question.

Permissions: Session owner

Request Body:

```
{
  "question_id": "question-uuid",
  "answer": "opt-2"
}
```

Response (200 OK):

```
{
  "success": true,
  "data": {
    "submitted": true,
    "next_question": {
      "id": "question-uuid-2",
      "sequence": 2,
      "type": "short_answer",
      "question_text": "Explain the advantage of using a gear ratio in a VEX drivetrain."
    }
```

```
  }
}
```

6.6 Complete Assessment

POST `/assessment-sessions/:id/complete`

Finish assessment and trigger scoring.

Permissions: Session owner

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "session-uuid",
    "completed_at": "2025-10-18T10:45:00Z",
    "status": "scoring",
    "message": "Your assessment is being evaluated. Results will be available soon."
  }
}
```

6.7 Get Assessment Results

GET `/assessment-sessions/:id/results`

Retrieve assessment results.

Permissions: Session owner, Coach, Admin

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "session-uuid",
    "assessment_title": "Mid-term STEM Assessment",
    "score": 85,
    "passing_score": 70,
    "passed": true,
    "total_questions": 20,
    "correct_answers": 17,
    "time_taken_minutes": 45,
    "completed_at": "2025-10-18T10:45:00Z",
    "breakdown": {
      "programming": 90,
```

```
      "problem_solving": 85,
      "design": 80
    },
    "questions": [
      {
        "id": "question-uuid-1",
        "question_text": "What is the maximum number of motors...",
        "student_answer": "8",
        "correct_answer": "8",
        "is_correct": true,
        "points": 5
      },
      ...
    ]
  }
}
```

7. Submission API

7.1 Create Submission

POST `/submissions`

Upload files for evaluation.

Permissions: Student

Request: - Content-Type: `multipart/form-data` - Fields: - `title` (string): Submission title - `description` (string): Optional description - `submission_type` (string): `robot_image`, `code`, `notebook`, `mixed` - `competition_type` (string): `VEX_IQ`, `VEX_V5` - `files[]` (files): Up to 20 files

Response (201 Created):

```
{
  "success": true,
  "data": {
    "id": "submission-uuid",
    "title": "Robot Design v3",
    "submission_type": "robot_image",
    "status": "uploaded",
    "files": [
      {
        "id": "file-uuid-1",
        "filename": "robot-front.jpg",
        "size": 2048576,
        "mime_type": "image/jpeg",
        "url": "https://storage.example.com/..."
```

```
      },
      ...
    ],
    "created_at": "2025-10-18T10:00:00Z"
  }
}
```

7.2 Get Submission

GET `/submissions/:id`

Retrieve submission details and evaluation.

Permissions: Submission owner, Coach (if in class), Parent (if linked), Admin

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "submission-uuid",
    "title": "Robot Design v3",
    "description": "Our latest robot design for competition",
    "submission_type": "robot_image",
    "competition_type": "VEX_V5",
    "status": "evaluated",
    "student": {
      "id": "student-uuid",
      "first_name": "John",
      "last_name": "Doe"
    },
    "files": [ ... ],
    "evaluation": {
      "overall_score": 88,
      "completion_date": "2025-10-18T10:15:00Z",
      "feedback": {
        "summary": "Excellent mechanical design with strong structural integrity...",
        "strengths": [
          "Well-balanced drivetrain",
          "Effective gear ratio selection",
          "Stable robot structure"
        ],
        "improvements": [
          "Consider adding sensors for autonomous",
          "Cable management could be improved"
        ],
```

```
        "detailed_feedback": "..."
      },
      "image_analysis": {
        "components_identified": ["8 motors", "4 wheels", "2 sensors"],
        "design_quality_score": 90
      }
    },
    "created_at": "2025-10-18T10:00:00Z"
  }
}
```

7.3 List Submissions

GET `/submissions`

List submissions with filtering.

Permissions: - Student: Own submissions - Coach: Class submissions - Parent: Child submissions - Admin: All submissions

Query Parameters: - `student_id` (uuid): Filter by student - `class_id` (uuid): Filter by class - `submission_type` (string): Filter by type - `status` (string): `uploaded`, `processing`, `evaluated`, `failed` - `from_date`, `to_date` (ISO date): Date range - `page`, `per_page`: Pagination - `sort`: `created_at`, `-score`, etc.

Response (200 OK):


```
{
  "success": true,
  "data": [
    {
      "id": "submission-uuid",
      "title": "Robot Design v3",
      "submission_type": "robot_image",
      "status": "evaluated",
      "overall_score": 88,
      "created_at": "2025-10-18T10:00:00Z",
      "student": {
        "id": "student-uuid",
        "first_name": "John",
        "last_name": "Doe"
      }
    },
    ...
  ],
  "pagination": { ... }
}
```

7.4 Delete Submission

DELETE `/submissions/:id`

Soft delete a submission.

Permissions: Submission owner (within 1 hour), Coach, Admin

Response (204 No Content)

8. Report API

8.1 Get Student Report

GET `/reports/student/:studentId`

Generate comprehensive student report.

Permissions: Student (self), Parent (linked child), Coach (class student), Admin

Query Parameters: - `date_from`, `date_to` (ISO date): Date range (default: all time) - `include_submissions` (boolean): Include submission details

Response (200 OK):

```
{
  "success": true,
  "data": {
    "student": {
      "id": "student-uuid",
      "first_name": "John",
      "last_name": "Doe"
    },
    "report_period": {
      "from": "2025-09-01T00:00:00Z",
      "to": "2025-10-18T23:59:59Z"
    },
    "stem_scores": {
      "overall": 85,
      "programming": 88,
      "engineering_design": 82,
      "problem_solving": 87,
      "documentation": 80
    },
    "progress_trend": [
      {"date": "2025-09-01", "score": 75},
      {"date": "2025-09-15", "score": 78},
      {"date": "2025-10-01", "score": 82},
```

```
      {"date": "2025-10-18", "score": 85}
    ],
    "statistics": {
      "total_submissions": 15,
      "assessments_completed": 8,
      "forum_participation": 12,
      "avg_submission_score": 85
    },
    "strengths": [
      "Strong programming fundamentals",
      "Consistent improvement trajectory",
      "Active forum participant"
    ],
    "growth_areas": [
      "Documentation quality",
      "Complex problem decomposition"
    ],
    "recommendations": [
      "Practice writing detailed engineering notebooks",
      "Explore advanced programming concepts like PID control"
    ],
    "recent_submissions": [ ... ]
  }
}
```

8.2 Get Student Benchmark

GET `/reports/benchmarks/student/:studentId`

Compare student performance to cohorts.

Permissions: Student (self), Parent (linked child), Coach (class student), Admin

Response (200 OK):

```
{
  "success": true,
  "data": {
    "student": {
      "id": "student-uuid",
      "first_name": "John",
      "last_name": "Doe"
    },
    "overall_score": 85,
    "performance_level": "Advanced",
    "benchmarks": {
```

```
      "class": {
        "percentile": 75,
        "average_score": 78,
        "std_deviation": 8,
        "student_deviation": "+7 points above average"
      },
      "grade_level": {
        "percentile": 72,
        "average_score": 76,
        "std_deviation": 10
      },
      "organization": {
        "percentile": 68,
        "average_score": 77,
        "std_deviation": 12
      }
    },
    "performance_levels": {
      "beginner": "0-60",
      "intermediate": "61-75",
      "advanced": "76-90",
      "expert": "91-100"
    },
    "score_distribution": {
      "0-60": 10,
      "61-75": 35,
      "76-90": 45,
      "91-100": 10
    }
  }
}
```

8.3 Export Report as PDF

GET `/reports/student/:studentId/pdf`

Generate and download PDF report.

Permissions: Student (self), Parent (linked child), Coach, Admin

Response: PDF file download (Content-Type: application/pdf)

8.4 Get Class Analytics

GET `/reports/class/:classId`

Class-level analytics for coaches.

Permissions: Assigned coach, Admin

Response (200 OK):

```
{
  "success": true,
  "data": {
    "class": {
      "id": "class-uuid",
      "name": "Robotics 101"
    },
    "summary": {
      "total_students": 24,
      "avg_overall_score": 78,
      "submission_rate": 0.92,
      "assessment_completion_rate": 0.88,
      "at_risk_students": 3
    },
    "score_distribution": {
      "0-60": 2,
      "61-75": 8,
      "76-90": 12,
      "91-100": 2
    },
    "engagement_metrics": {
      "avg_submissions_per_student": 12,
      "forum_participation_rate": 0.75,
      "weekly_active_rate": 0.85
    },
    "at_risk_students": [
      {
        "id": "student-uuid",
        "first_name": "Jane",
        "last_name": "Smith",
        "overall_score": 58,
        "reason": "Below passing threshold",
        "last_submission": "2025-09-25T10:00:00Z"
      },
      ...
    ],
    "top_performers": [ ... ]
  }
}
```

9. Forum API

9.1 Create Post

POST `/forum/posts`

Create a new forum post.

Permissions: Authenticated user

Request Body:

```
{
  "title": "How to optimize autonomous code?",
  "content": "I'm trying to improve my robot's autonomous performance...",
  "tags": ["programming", "autonomous", "vex-v5"],
  "is_question": true
}
```

Response (201 Created):

```
{
  "success": true,
  "data": {
    "id": "post-uuid",
    "title": "How to optimize autonomous code?",
    "content": "I'm trying to improve...",
    "author": {
      "id": "user-uuid",
      "first_name": "John",
      "last_name": "Doe",
      "avatar_url": "..."
    },
    "tags": ["programming", "autonomous", "vex-v5"],
    "is_question": true,
    "is_answered": false,
    "vote_count": 0,
    "comment_count": 0,
    "created_at": "2025-10-18T10:00:00Z"
  }
}
```

9.2 List Posts

GET `/forum/posts`

List forum posts with filtering and sorting.

Query Parameters: - `tags` (array): Filter by tags - `is_question` (boolean):

Questions only - `is_answered` (boolean): Answered/unanswered - `search` (string): Search in title and content - `sort` (string): `hot`, `new`, `top`, `unanswered` - `page`, `per_page`: Pagination

Response (200 OK):

```
{
  "success": true,
  "data": [
    {
      "id": "post-uuid",
      "title": "How to optimize autonomous code?",
      "content": "I'm trying to improve...",
      "author": {
        "id": "user-uuid",
        "first_name": "John",
        "last_name": "Doe"
      },
      "tags": ["programming", "autonomous"],
      "is_question": true,
      "is_answered": false,
      "vote_count": 5,
      "comment_count": 3,
      "created_at": "2025-10-18T10:00:00Z"
    },
    ...
  ],
  "pagination": { ... }
}
```

9.3 Get Post

GET `/forum/posts/:id`

Get post details with comments.

Response (200 OK):

```
{
  "success": true,
  "data": {
    "id": "post-uuid",
    "title": "How to optimize autonomous code?",
    "content": "Full content here...",
    "author": { ... },
    "tags": ["programming"],
    "is_question": true,
```

```
    "is_answered": true,
    "accepted_answer_id": "comment-uuid",
    "vote_count": 5,
    "user_vote": 1,
    "created_at": "2025-10-18T10:00:00Z",
    "updated_at": "2025-10-18T11:00:00Z",
    "comments": [
      {
        "id": "comment-uuid",
        "content": "You should try using PID control...",
        "author": {
          "id": "coach-uuid",
          "first_name": "Jane",
          "last_name": "Coach",
          "role": "coach"
        },
        "is_accepted_answer": true,
        "vote_count": 8,
        "user_vote": 0,
        "created_at": "2025-10-18T10:30:00Z",
        "replies": [
          {
            "id": "reply-uuid",
            "content": "Thanks! That helped a lot.",
            "author": { ... },
            "created_at": "2025-10-18T11:00:00Z"
          }
        ]
      },
      ...
    ]
  }
}
```

9.4 Create Comment

POST `/forum/posts/:postId/comments`

Add comment to a post.

Request Body:

```
{
  "content": "You should try using PID control for smoother autonomous movements.",
  "parent_id": null
}
```

Note: Set `parent_id` for nested replies (up to 5 levels)

Response (201 Created):

```
{
  "success": true,
  "data": {
    "id": "comment-uuid",
    "content": "You should try using PID control...",
    "author": { ... },
    "parent_id": null,
    "created_at": "2025-10-18T10:30:00Z"
  }
}
```

9.5 Vote on Post

POST `/forum/posts/:id/vote`

Upvote or downvote a post.

Request Body:

```
{
  "vote": 1
}
```

Vote Values: `1` (upvote), `0` (remove vote), `-1` (downvote)

Response (200 OK):

```
{
  "success": true,
  "data": {
    "post_id": "post-uuid",
    "vote_count": 6,
    "user_vote": 1
  }
}
```

9.6 Accept Answer

POST `/forum/comments/:id/accept-answer`

Mark comment as accepted answer (post author only).

Permissions: Post author, Coach, Admin

Response (200 OK):

```
{
  "success": true,
  "data": {
    "comment_id": "comment-uuid",
    "post_id": "post-uuid",
    "is_accepted": true
  }
}
```

9.7 Search Forum

GET `/forum/search`

Full-text search across posts and comments.

Query Parameters: - `q` (string): Search query - `tags` (array): Filter by tags - `author_id` (uuid): Filter by author - `date_from`, `date_to`: Date range - `page`, `per_page`: Pagination

Response (200 OK):

```
{
  "success": true,
  "data": {
    "posts": [ ... ],
    "total_results": 42
  },
  "pagination": { ... }
}
```

10. Notification API

10.1 Get Notifications

GET `/notifications`

List user's notifications.

Query Parameters: - `unread_only` (boolean): Only unread - `type` (string): Filter by notification type - `page`, `per_page`: Pagination

Response (200 OK):

```
{
  "success": true,
  "data": [
    {
      "id": "notif-uuid",
      "type": "submission_feedback",
```

```
      "title": "Feedback ready for Robot Design v3",
      "message": "Your submission has been evaluated",
      "is_read": false,
      "action_url": "/submissions/submission-uuid",
      "created_at": "2025-10-18T10:15:00Z",
      "metadata": {
        "submission_id": "submission-uuid",
        "score": 88
      }
    },
    ...
  ],
  "pagination": { ... }
}
```

10.2 Get Unread Count

GET `/notifications/unread-count`

Get count of unread notifications.

Response (200 OK):


```
{
  "success": true,
  "data": {
    "count": 5
  }
}
```

10.3 Mark as Read

POST `/notifications/:id/read`

Mark notification as read.

Response (200 OK):


```
{
  "success": true,
  "data": {
    "id": "notif-uuid",
    "is_read": true
  }
}
```

10.4 Mark All as Read

POST `/notifications/mark-all-read`

Mark all notifications as read.

Response (200 OK):

```
{
  "success": true,
  "data": {
    "marked_count": 12
  }
}
```

10.5 Update Notification Preferences

PUT `/notifications/preferences`

Configure notification settings.

Request Body:

```
{
  "email_enabled": true,
  "submission_feedback": true,
  "assessment_assigned": true,
  "forum_replies": true,
  "forum_mentions": true,
  "weekly_digest": false,
  "digest_day": "monday"
}
```

Response (200 OK):

```
{
  "success": true,
  "data": {
    "email_enabled": true,
    "submission_feedback": true,
    "assessment_assigned": true,
    "forum_replies": true,
    "forum_mentions": true,
    "weekly_digest": false,
    "digest_day": "monday",
    "updated_at": "2025-10-18T10:30:00Z"
  }
}
```

11. Admin API

11.1 Get System Statistics

GET `/admin/statistics`

System-wide metrics and statistics.

Permissions: Admin only

Response (200 OK):


```
{
  "success": true,
  "data": {
    "users": {
      "total": 1250,
      "students": 1000,
      "coaches": 50,
      "parents": 180,
      "admins": 20,
      "new_this_month": 150
    },
    "activity": {
      "dau": 750,
      "mau": 1100,
      "submissions_today": 85,
      "assessments_completed_today": 42
    },
    "system": {
      "ai_api_calls_today": 250,
      "ai_cost_today": 12.50,
      "storage_used_gb": 450,
      "avg_response_time_ms": 180
    }
  }
}
```

11.2 Manage Organizations

GET `/admin/organizations`

POST `/admin/organizations`

PUT `/admin/organizations/:id`

DELETE `/admin/organizations/:id`

CRUD operations for organizations (schools/districts).

11.3 System Health Check

GET `/health`

System health check (public endpoint).

Response (200 OK):

```
{
  "success": true,
  "data": {
    "status": "healthy",
    "services": {
      "api": "up",
      "database": "up",
      "redis": "up",
      "storage": "up",
      "ai_api": "up"
    },
    "version": "1.0.0",
    "timestamp": "2025-10-18T10:30:00Z"
  }
}
```

12. WebSocket API

12.1 Connection

WS `/ws`

Establish WebSocket connection for real-time updates.

Authentication: Send JWT token as first message after connection.

Message Format:

```
{
  "type": "auth",
  "token": "jwt-token-here"
}
```

12.2 Message Types

Incoming (Server → Client)

```
{
  "type": "notification",
```

```
  "data": {
    "id": "notif-uuid",
    "type": "submission_feedback",
    "title": "Feedback ready",
    "message": "Your submission has been evaluated",
    "created_at": "2025-10-18T10:15:00Z"
  }
}
```

```
{
  "type": "submission_status",
  "data": {
    "submission_id": "submission-uuid",
    "status": "evaluated",
    "overall_score": 88
  }
}
```

Outgoing (Client → Server)

```
{
  "type": "subscribe",
  "channel": "notifications"
}
```

```
{
  "type": "ping"
}
```

Response:

```
{
  "type": "pong",
  "timestamp": "2025-10-18T10:30:00Z"
}
```

13. Error Codes

Code	HTTP Status	Description
`VALIDATION_ERROR`	400	Invalid input data
`UNAUTHORIZED`	401	Missing or invalid authentication
`FORBIDDEN`	403	Insufficient permissions
`NOT_FOUND`	404	Resource not found
`CONFLICT`	409	Resource conflict (e.g., duplicate)
`UNPROCESSABLE_ENTITY`	422	Validation failed
`RATE_LIMIT_EXCEEDED`	429	Too many requests
`INTERNAL_ERROR`	500	Internal server error

Code	HTTP Status	Description
`SERVICE_UNAVAILABLE`	503	Service temporarily unavailable
`AI_API_ERROR`	500	Vertex AI API error
`STORAGE_ERROR`	500	File storage error

14. Rate Limiting

14.1 Rate Limit Headers

Every API response includes rate limit information:

```
X-RateLimit-Limit: 100
X-RateLimit-Remaining: 95
X-RateLimit-Reset: 1634567890
```

14.2 Rate Limit Rules

- **Global**: 1000 requests/minute per IP
- **Authenticated**: 100 requests/minute per user
- **File Upload**: 10 uploads/minute per user
- **AI Evaluation**: 20 submissions/hour per student
- **Forum**: 30 posts/hour per user

14.3 Rate Limit Response

When exceeded, returns 429 with:

```json
{
  "success": false,
  "error": {
    "code": "RATE_LIMIT_EXCEEDED",
    "message": "Rate limit exceeded. Please try again in 45 seconds.",
    "retry_after": 45
  }
}
```

Document Version: 1.0 **Last Updated**: 2025-10-18 **OpenAPI Specification**: Available at `/api/docs` (Swagger UI)