ELO Learning

SERVICE CONTRACTS V1.3

ZERO DAY

Last updated: 19 August 2025

University of Pretoria

Name	Student number
RM (Rene) Brancon	u22556771
NF (Nigel) Mofati	u22528084
TM (Tukelo) Mokwena	u22536800
S (Saskia) Steyn	u17267162
NG (Ntokozo) Tonga	u22506773

Team contact:
ZeroDay0D4y@gmail.com

Contents

1. Introduction

The ELO Learning platform provides a comprehensive set of REST APIs and WebSocket services for managing users, questions, answers, gameplay, and mathematical validation. All endpoints return JSON responses and follow standard HTTP status codes.

Base URL:

```
http://localhost:3000 (development) / https://your-domain.com
(production)
```

2. Authentication

Most endpoints require Bearer token authentication. Include the token in the Authorization header:

```
Authorization: Bearer <jwt_token>
```

Tokens are obtained through the login endpoint and have a 1-hour expiration time.

3. User Management APIs

3.1 GET /users

Retrieve all users in the system

Authentication: Not required

3.2 GET /user/:id

Retrieve a specific user by ID

Authentication: Required (Bearer token)

- id (path): User UUID

3.3 GET /users/:id/achievements

Retrieve all achievements for a specific user

Authentication: Required (Bearer token)

- id (path): User UUID

3.4 POST /user/:id/xp

Update a user's XP points

Authentication: Required (Bearer token)

- id (path): User UUID

3.5 POST /user/:id/avatar

Update a user's avatar

Authentication: Not required

- id (path): User UUID

- Request Body: { "pfpURL": "string" }

- Response: Updated user object

3.6 GET /users/rank/:rank

Retrieve all users with a specific rank

Authentication: Not required

- rank (path): Rank name

3.7 POST /register

Register a new user account

Authentication: Not required

3.8 POST /login

Authenticate user and get access token

Authentication: Not required

3.9 POST /forgot-password

Request password reset email

Authentication: Not required

3.10 POST /reset-password

Reset password using reset token

Authentication: Not required

3.11 GET /verify-reset-token/:token

Verify if a password reset token is valid

Authentication: Not required

- token (path): Reset token

4. Question Management APIs

4.1 GET /questions

Retrieve all questions in the system

Authentication: Not required

4.2 GET /questionsById/:id

Retrieve a specific question by ID

Authentication: Not required

- id (path): Question UUID

4.3 GET /question/:level

Retrieve all questions for a specific level

Authentication: Required (Bearer token)

- level (path): Level number

4.4 GET /question/:id/answer

Retrieve the correct answer for a specific question

Authentication: Required (Bearer token)

- id (path): Question UUID

4.5 GET /questions/topic

Retrieve questions by topic

Authentication: Not required

- topic (query): Topic name

4.6 GET /questions/level/topic

Retrieve questions filtered by both level and topic

Authentication: Not required

- level (query): Level number

- topic (query): Topic ID

4.7 GET /topics

Retrieve all available topics

Authentication: Not required

4.8 GET /questions/random

Retrieve random questions for a specific level

Authentication: Not required

- level (query): Level number (required)

4.9 POST /question/:id/submit

Submit and validate an answer for a specific question

Authentication: Not required

- id (path): Question UUID

4.10 GET /questions/type/:type

Retrieve questions by type

Authentication: Not required

- type (path): Question type

4.11 GET /questions/mixed

Retrieve a mixed set of questions

Authentication: Not required

5. Answer Management APIs

5.1 GET /answers/:id

Retrieve all answers for a specific question

Authentication: Not required

- id (path): Question UUID

5.2 POST /submit-answer

Submit an answer (alternative to /question/:id/submit)

Authentication: Not required

- Request Body:

```
{ "question_id": "uuid", "studentAnswer": "string", "userId": "uuid (optional)" }
```

- Response:

```
{ "isCorrect": "boolean", "studentAnswer": "string", "correctAnswer": "string", "message": "string", "xpAwarded": "number", "updatedUser": { "id": "uuid", "xp": "number" } }
```

6. Practice Mode APIs

6.1 GET /practice

Retrieve 10 practice questions with answers

Authentication: Not required

6.2 GET /practice/type/:questionType

Retrieve practice questions by type

Authentication: Not required

- questionType (path): Type of question

7. Single Player Game APIs

7.1 POST /singleplayer

Process single player game attempt and update user statistics

Authentication: Not required

8. Multiplayer Game APIs

8.1 POST /multiplayer

Process multiplayer game results and update both players' statistics

Authentication: Not required

9. OAuth APIs

9.1 POST /oauth/user

Create or retrieve OAuth authenticated user

Authentication: Not required

10. Math Validation APIs

10.1 POST /validate-answer

Validate a mathematical answer

Authentication: Not required

10.2 POST /quick-validate

Quick validation for real-time feedback

Authentication: Not required

10.3 POST /validate-expression

Validate mathematical expression format

Authentication: Not required

11. WebSocket Events

The system uses Socket.IO for real-time multiplayer functionality. Connect to the same base URL with Socket.IO client.

11.1 Client Events (Sent to Server)

11.1.1 queue-for-game

Join the multiplayer game queue

```
{ "id": "uuid", "name": "string", "username": "string", "xp": "number" }
```

11.1.2 leave-queue

Leave the multiplayer game queue

None

11.1.3 disconnect

Handle client disconnection

None

11.2 Server Events (Sent to Client)

11.2.1 startGame

Game has started with matched opponent

```
{ "gameId": "uuid", "opponent": { "name": "string", "username":
"string", "xp": "number" } }
```

11.2.2 queueUpdate

Update on queue status

```
{ "position": "number", "totalInQueue": "number" }
```

11.2.3 gameEnded

Game has ended

```
{ "gameId": "uuid", "result": "string", "stats": "object" }
```

11.2.4 achievementsUnlocked

Achievements unlocked while in queue

```
{ "achievements": [ { "achievement_id": "uuid", "achievement_name":
"string", "description": "string", "earned_date": "ISO date string" } ]
}
```

12. Error Handling

12.1 Standard HTTP Status Codes

- **200 OK**: Request successful
- **201 Created**: Resource created successfully
- **400 Bad Request**: Invalid request data
- **401 Unauthorized**: Authentication required or invalid
- **404 Not Found**: Resource not found
- **409 Conflict**: Resource already exists (e.g., email during registration)
- **500 Internal Server Error**: Server error

12.2 Error Response Format

```
{ "error": "string (error message)" }
```

12.3 Common Error Scenarios

12.3.1. Missing Authorization Header:

```
{ "error": "You are unauthorized to make this request." }
```

12.3.2. Invalid Token:

```
{ "error": "Invalid or expired token" }
```

12.3.3. Missing Required Fields:

```
{ "error": "All fields are required" }
```

12.3.4. User Not Found:

```
{ "error": "User doesn't exist" }
```

12.3.5. Database Error:

```
{ "error": "Failed to fetch/update/create resource" }
```

13. Response Formats

13.1 Success Response

All successful responses return data in JSON format with appropriate HTTP status codes.

13.2 Date Formats

All dates are returned in ISO 8601 format: YYYY-MM-DDTHH:mm:ss.sssZ

13.3 UUID Format

All IDs use UUID v4 format: xxxxxxxx-xxxx-4xxx-yxxx-xxxxxxxxxxxx

13.4 Pagination

Currently, pagination is not implemented. The /practice endpoint limits results to 10 items, and /questions/random limits to 5 items.

13.5 Data Validation

- Email addresses must be valid format

- Passwords must be at least 8 characters for registration/reset

- Level parameters must be positive integers

- XP values must be numbers

- Boolean fields only accept true or false