Student Registration API

Endpoint

```
POST /api/register
```

Description

Registers a new student account with personal information, trading level, and learning preferences.

Request Headers

```
Content-Type: application/json
```

Request Body Parameters

Field	Type	Required	Validation	Description
`firstName`	string	Yes	1-50 chars, trimmed, escaped	Student's first name
`lastName`	string	Yes	1-50 chars, trimmed, escaped	Student's last name
`email`	string	Yes	Valid email format	Student's email address
`phoneNumber`	string	Yes	Valid mobile phone	Student's phone number
`dob`	string	Yes	ISO8601 date format	Date of birth (YYYY-MM-DD)
`address`	string	Yes	1-500 chars, trimmed	Street address
`city`	string	Yes	1-100 chars, trimmed	City name
`province`	string	Yes	1-100 chars, trimmed	Province/State
`gender`	string	Yes	'Male', 'Female', or 'Other'	Gender identity
`education`	string	Yes	1-100 chars, trimmed	Educational background
`tradingLevel`	string	Yes	1-50 chars, trimmed	Trading experience level
`username`	string	Yes	3-15 chars, trimmed	Unique username
`password`	string	Yes	Min 6 chars	Account password
`device`	string/array	No	-	Preferred device(s) for learning
`learningStyle`	string/array	No	-	Preferred learning style(s)

Sample Request

json

```
{
  "firstName": "John",
  "lastName": "Doe",
  "email": "john.doe@example.com",
  "phoneNumber": "+1234567890",
  "dob": "1990-05-15",
  "address": "123 Main Street",
  "city": "New York",
  "province": "NY",
  "gender": "Male",
  "education": "Bachelor's Degree",
  "tradingLevel": "Beginner",
  "username": "johndoe123",
  "password": "securepass123",
  "device": ["Desktop", "Mobile"],
  "learningStyle": ["Visual", "Interactive"]
}
```

Sample Response

Success (201 Created)

json

```
{
  "message": "Student successfully registered.",
  "student_id": "S1703123456789_12345"
}
```

Error Responses

Validation Error (400 Bad Request)

json

```
{
  "errors": [
    {
      "field": "email",
      "message": "Invalid email format"
    },
    {
      "field": "password",
      "message": "Password must be at least 6 characters"
    }
  ]
}
```

Duplicate Entry (409 Conflict)

json

```
{
  "error": "Username or email already exists"
}
```

Server Error (500 Internal Server Error)

json

```
{
  "error": "Failed to register student"
}
```

cURL Example

bash

```bash
curl -X POST http://localhost:3000/api/register \
  -H "Content-Type: application/json" \
  -d '{
    "firstName": "John",
    "lastName": "Doe",
    "email": "john.doe@example.com",
    "phoneNumber": "+1234567890",
    "dob": "1990-05-15",
    "address": "123 Main Street",
    "city": "New York",
    "province": "NY",
    "gender": "Male",
    "education": "Bachelor'\''s Degree",
    "tradingLevel": "Beginner",
    "username": "johndoe123",
    "password": "securepass123",
    "device": ["Desktop", "Mobile"],
    "learningStyle": ["Visual", "Interactive"]
  }'
```

JavaScript/Fetch Example

javascript

```javascript
const registerStudent = async (studentData) => {
  try {
    const response = await fetch('/api/register', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(studentData)
    });

    const result = await response.json();

    if (response.ok) {
      console.log('Registration successful:', result);
      return result;
    } else {
      console.error('Registration failed:', result);
      throw new Error(result.error);
    }
  } catch (error) {
    console.error('Network error:', error);
    throw error;
  }
};

// Usage
const newStudent = {
  firstName: "Jane",
  lastName: "Smith",
  email: "jane.smith@example.com",
  phoneNumber: "+0987654321",
  dob: "1995-08-20",
  address: "456 Oak Avenue",
  city: "Los Angeles",
  province: "CA",
  gender: "Female",
  education: "Master's Degree",
  tradingLevel: "Intermediate",
  username: "janesmith",
  password: "mypassword456",
  device: "Mobile",
  learningStyle: "Hands-on"
};
```

```
registerStudent(newStudent);
```

Notes

- All string fields are automatically trimmed of whitespace
- Password is hashed using bcrypt with salt rounds of 12
- Student ID is auto-generated in format: `S{timestamp}_{person_id}`
- The endpoint uses database transactions to ensure data consistency
- Device and learningStyle can be single values or arrays
- Phone numbers must be in valid mobile phone format
- Date of birth must be in ISO8601 format (YYYY-MM-DD)
- All operations are wrapped in a database transaction for data integrity