Entities:

1. **Calculator**: This represents the main entity, which is a mathematical calculator. It can have operations to perform various arithmetic calculations.
2. **Operation**: This entity represents a mathematical operation to be performed by the calculator. It includes information like the type of operation (addition, subtraction, multiplication, division) and the operands (numbers on which the operation will be performed).
3. **Result**: This entity represents the result of a mathematical operation. It includes information like the result value, a message (for errors or success), and an error flag.

Operations:

1. **Perform Addition Operation**:
    - HTTP Method: POST
    - Endpoint: /calculator/add
    - Request: An operation object containing two operands for addition.
    - Response: The result of the addition operation.
2. **Perform Subtraction Operation**:
    - HTTP Method: POST
    - Endpoint: /calculator/subtract
    - Request: An operation object containing two operands for subtraction.
    - Response: The result of the subtraction operation.
3. **Perform Multiplication Operation**:
    - HTTP Method: POST
    - Endpoint: /calculator/multiply
    - Request: An operation object containing two operands for multiplication.
    - Response: The result of the multiplication operation.
4. **Perform Division Operation**:
    - HTTP Method: POST
    - Endpoint: /calculator/divide

- Request: An operation object containing two operands for division.
- Response: The result of the division operation.

REST API Description: The REST API provided by this application allows users to perform basic arithmetic operations (addition, subtraction, multiplication, and division) using a mathematical calculator. The API provides the following endpoints:

- **/calculator/add**: Allows users to perform addition by sending a POST request with two operands. The result of the addition operation is returned in the response.
- **/calculator/subtract**: Allows users to perform subtraction by sending a POST request with two operands. The result of the subtraction operation is returned in the response.
- **/calculator/multiply**: Allows users to perform multiplication by sending a POST request with two operands. The result of the multiplication operation is returned in the response.
- **/calculator/divide**: Allows users to perform division by sending a POST request with two operands. The result of the division operation is returned in the response.

Functional Requirements:

- The API should handle basic arithmetic operations.
- It should validate the operands and provide meaningful error messages.
- It should return the result of the operation.

Non-Functional Requirements:

- Error messages should be descriptive and meaningful.
- HTTP status codes should be used appropriately (e.g., 200 for success, 400 for bad request, 500 for server errors).
- Logging should be implemented for error handling and debugging.
- The API should be secure and protect against input validation vulnerabilities.

Richardson Maturity Model (Optional):

- The API follows the Level 2 of the Richardson Maturity Model, as it uses HTTP methods and resources (endpoints) for performing operations.

Errors:

- Errors are described using HTTP status codes and meaningful error messages.
- Common errors include division by zero, invalid operands, and general server errors.

Authentication Method:

- The provided code does not include authentication. However, for a production API, appropriate authentication and authorization mechanisms should be implemented.