Rule Engine API Documentation

This document provides an overview of the Rule Engine API developed using Flask, which allows dynamic creation, evaluation, and combination of rules using an Abstract Syntax Tree (AST). The API accepts rule strings that describe conditions and expressions, stores them in a SQL Server database, and evaluates them against given data.

API Overview

The Rule Engine API consists of three primary endpoints:

1. **/create_rule** (POST): This endpoint accepts a rule string, converts it to an AST, and stores it in the database.
2. **/evaluate_rule** (POST): This endpoint evaluates a given AST against input data to determine the outcome of the rule.
3. **/combine_rules** (POST): This endpoint combines multiple rules into a single AST structure and returns the combined result.

Database Setup

The application uses SQL Server for storing rule definitions. The database schema includes a 'rules' table with fields for storing the rule name and rule string. The table is created if it does not already exist when the application starts.

```SQL
CREATE TABLE dbo.rules (
```

```
    id INT PRIMARY KEY IDENTITY(1,1),
    rule_name VARCHAR(255) NOT NULL,
    rule_string TEXT NOT NULL
)
```

API Endpoints and Usage

1. /create_rule (POST)

This endpoint accepts a rule string and stores it in the database.

Request Payload:

```JSON
{
    "rule_string": "((age > 30 AND department == 'Sales') OR (age < 25 AND department == 'Marketing'))"
}
```

2. /evaluate_rule (POST)

This endpoint evaluates a rule represented as an AST against input data.

Request Payload:

```JSON
{
    "ast": {
        "type": "operator",
        "value": "AND",
        "left": {"type": "operand", "value": "age > 30"},
```

```
        "right": {"type": "operand", "value": "department == 'Sales'"}
    },
    "data": {"age": 35, "department": "Sales"}
}
```

Sample Requests and Responses

3. /combine_rules (POST)

This endpoint combines multiple rules and returns the combined AST.

Request Payload:

```JSON
{
    "rules": [
        "age > 30 AND department == 'Sales'",
        "age < 25 AND department == 'Marketing'"
    ]
}
```

Response:

```JSON
{
    "result": True
}
```

Conclusion

This API provides a flexible and dynamic rule evaluation system using Abstract Syntax Trees (ASTs). It supports rule creation, combination, and evaluation based on user-provided data. Further enhancements can include additional operators, more complex rules, and improved error handling.