Company Database Management System

CS 306 – PROJECT PHASE 2

YASİN BARKIN BAŞARAN

STUDENT ID: 28890

Trigger Implementation

1. Introduction

In this section, we implemented three triggers to automate and log specific tasks within the database. These triggers enhance the database's functionality by maintaining logs for critical Events like salary updates, employee transfers, and project hours updates.

2. Salary Update Trigger

- Purpose:

  The SalaryUpdateTrigger was created to log any changes to an employee's salary in the Salary_Audit table. This ensures that salary modifications are tracked for audit purposes.

- SQL Script

```
mysql> CREATE TABLE Salary_Audit (
    ->     Audit_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Ssn VARCHAR(9),
    ->     Old_Salary DECIMAL(10, 2),
    ->     New_Salary DECIMAL(10, 2),
    ->     Change_Date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql>
```

```
mysql> DELIMITER $$
mysql> CREATE TRIGGER SalaryUpdateTrigger
    -> AFTER UPDATE ON Employee
    -> FOR EACH ROW
    -> BEGIN
    ->     IF OLD.Salary <> NEW.Salary THEN
    ->         INSERT INTO Salary_Audit (Ssn, Old_Salary, New_Salary)
    ->         VALUES (OLD.Ssn, OLD.Salary, NEW.Salary);
    ->     END IF;
    -> END$$
Query OK, 0 rows affected (0.05 sec)

mysql> DELIMITER ;
mysql>
```

- Execution

  - **Before Update**

The Employee and Salary_Audit tables before executing the update command are shown below:

```
mysql> SELECT * FROM Employee;
+-----------+---------+-------+---------+----------+------+
| Ssn       | FName   | MName | LName   | Salary   | Sex  |
+-----------+---------+-------+---------+----------+------+
| 123456789 | John    | A     | Doe     | 55000.00 | M    |
| 147258369 | William | G     | Miller  | 50000.00 | M    |
| 258963147 | James   | I     | Moore   | 62000.00 | M    |
| 321654987 | Michael | E     | White   | 58000.00 | M    |
| 456789123 | Robert  | C     | Brown   | 48000.00 | M    |
| 654321789 | Emma    | F     | Davis   | 70000.00 | F    |
| 789456123 | Emily   | D     | Johnson | 72000.00 | F    |
| 852741963 | Sophia  | J     | Taylor  | 67000.00 | F    |
| 963852741 | Olivia  | H     | Wilson  | 80000.00 | F    |
| 987654321 | Jane    | B     | Smith   | 65000.00 | F    |
+-----------+---------+-------+---------+----------+------+
10 rows in set (0.01 sec)

mysql> SELECT * FROM Salary_Audit;
Empty set (0.01 sec)

mysql>
```

  - Action

We updated the salary of the employee with (Ssn = ‘123456789’) to test the trigger. The command used is:

```
mysql> SELECT * FROM Salary_Audit;
Empty set (0.01 sec)

mysql> UPDATE Employee SET Salary = 75000 WHERE Ssn = '123456789';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

- After Update

After executing the update command, the Salary_Audit table successfully logged the change. The update states of the tables are shown below:

```
mysql> SELECT * FROM Employee;
+-----------+---------+-------+---------+----------+------+
| Ssn       | FName   | MName | LName   | Salary   | Sex  |
+-----------+---------+-------+---------+----------+------+
| 123456789 | John    | A     | Doe     | 75000.00 | M    |
| 147258369 | William | G     | Miller  | 50000.00 | M    |
| 258963147 | James   | I     | Moore   | 62000.00 | M    |
| 321654987 | Michael | E     | White   | 58000.00 | M    |
| 456789123 | Robert  | C     | Brown   | 48000.00 | M    |
| 654321789 | Emma    | F     | Davis   | 70000.00 | F    |
| 789456123 | Emily   | D     | Johnson | 72000.00 | F    |
| 852741963 | Sophia  | J     | Taylor  | 67000.00 | F    |
| 963852741 | Olivia  | H     | Wilson  | 80000.00 | F    |
| 987654321 | Jane    | B     | Smith   | 65000.00 | F    |
+-----------+---------+-------+---------+----------+------+
10 rows in set (0.00 sec)

mysql> SELECT * FROM Salary_Audit;
+----------+-----------+------------+------------+---------------------+
| Audit_ID | Ssn       | Old_Salary | New_Salary | Change_Date         |
+----------+-----------+------------+------------+---------------------+
|        1 | 123456789 |   55000.00 |   75000.00 | 2024-12-04 18:33:43 |
+----------+-----------+------------+------------+---------------------+
1 row in set (0.00 sec)
```

3. Employee Transfer Trigger

- Purpose

  The EmplyeeTransferTrigger was created to log any changes to an employee'S department in the Works_For table. The changes are logged in the Transfer_Audit table for tracking purposes.

- SQL Script

```
mysql> CREATE TABLE Transfer_Audit (
    ->     Audit_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Ssn VARCHAR(9),
    ->     Old_Department INT,
    ->     New_Department INT,
    ->     Transfer_Date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER $$
mysql> CREATE TRIGGER EmployeeTransferTrigger
    -> AFTER UPDATE ON Works_For
    -> FOR EACH ROW
    -> BEGIN
    ->     IF OLD.Department_Number <> NEW.Department_Number THEN
    ->         INSERT INTO Transfer_Audit (Ssn, Old_Department, New_Department)
    ->         VALUES (OLD.Ssn, OLD.Department_Number, NEW.Department_Number);
    ->     END IF;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
```

- Before Update

```
mysql> SELECT * FROM Works_For;
+-----------+-------------------+
| Ssn       | Department_Number |
+-----------+-------------------+
| 123456789 |                 1 |
| 987654321 |                 2 |
| 456789123 |                 3 |
| 789456123 |                 4 |
| 321654987 |                 5 |
| 654321789 |                 6 |
| 147258369 |                 7 |
| 963852741 |                 8 |
| 258963147 |                 9 |
| 852741963 |                10 |
+-----------+-------------------+
10 rows in set (0.01 sec)

mysql> SELECT * FROM Transfer_Audit;
Empty set (0.00 sec)

mysql>
```

- Action

```
mysql> SELECT * FROM Works_For;
+-----------+-----------------+
| Ssn       | Department_Number |
+-----------+-----------------+
| 987654321 |               2 |
| 456789123 |               3 |
| 123456789 |               4 |
| 789456123 |               4 |
| 321654987 |               5 |
| 654321789 |               6 |
| 147258369 |               7 |
| 963852741 |               8 |
| 258963147 |               9 |
| 852741963 |              10 |
+-----------+-----------------+
10 rows in set (0.00 sec)

mysql> SELECT * FROM Transfer_Audit;
+----------+-----------+----------------+----------------+---------------------+
| Audit_ID | Ssn       | Old_Department | New_Department | Transfer_Date       |
+----------+-----------+----------------+----------------+---------------------+
|        1 | 123456789 |              1 |              4 | 2024-12-04 18:41:09 |
+----------+-----------+----------------+----------------+---------------------+
1 row in set (0.00 sec)
```

4. Project Hours Trigger

- Purpose

  The HoursUpdateTrigger was created to log any changes to the hours worked by an employee on a Project in the Works_On table. The changes are logged in the Hours_Audit table for accountability.

- SQL Script

```
mysql> CREATE TABLE Hours_Audit (
    ->     Audit_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Ssn VARCHAR(9),
    ->     Project_Number INT,
    ->     Old_Hours DECIMAL(5, 2),
    ->     New_Hours DECIMAL(5, 2),
    ->     Change_Date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql>
```

```
mysql> DELIMITER $$
mysql> CREATE TRIGGER HoursUpdateTrigger
    -> AFTER UPDATE ON Works_On
    -> FOR EACH ROW
    -> BEGIN
    ->     IF OLD.Hours <> NEW.Hours THEN
    ->         INSERT INTO Hours_Audit (Ssn, Project_Number, Old_Hours, New_Hour
s)
    ->         VALUES (OLD.Ssn, OLD.Project_Number, OLD.Hours, NEW.Hours);
    ->     END IF;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
```

- Execution
    - Before Update

```
mysql> DELIMITER ;
mysql> SELECT * FROM Works_On;
+-----------+----------------+-------+
| Ssn       | Project_Number | Hours |
+-----------+----------------+-------+
| 123456789 |            101 | 35.50 |
| 147258369 |            107 | 50.00 |
| 258963147 |            109 | 25.00 |
| 321654987 |            105 | 30.00 |
| 456789123 |            103 | 28.00 |
| 654321789 |            106 | 37.50 |
| 789456123 |            104 | 42.50 |
| 852741963 |            110 | 45.00 |
| 963852741 |            108 | 33.00 |
| 987654321 |            102 | 40.00 |
+-----------+----------------+-------+
10 rows in set (0.00 sec)

mysql> SELECT * FROM Hours_Audit;
Empty set (0.00 sec)

mysql>
```

- Action and After Updation

```
mysql> UPDATE Works_On SET Hours = 42.5 WHERE Ssn = '123456789' AND Project_Number = 101;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM Works_On;
+-----------+----------------+-------+
| Ssn       | Project_Number | Hours |
+-----------+----------------+-------+
| 123456789 |            101 | 42.50 |
| 147258369 |            107 | 50.00 |
| 258963147 |            109 | 25.00 |
| 321654987 |            105 | 30.00 |
| 456789123 |            103 | 28.00 |
| 654321789 |            106 | 37.50 |
| 789456123 |            104 | 42.50 |
| 852741963 |            110 | 45.00 |
| 963852741 |            108 | 33.00 |
| 987654321 |            102 | 40.00 |
+-----------+----------------+-------+
10 rows in set (0.00 sec)

mysql> SELECT * FROM Hours_Audit;
+----------+-----------+----------------+-----------+-----------+---------------------+
| Audit_ID | Ssn       | Project_Number | Old_Hours | New_Hours | Change_Date         |
+----------+-----------+----------------+-----------+-----------+---------------------+
|        1 | 123456789 |            101 |     35.50 |     42.50 | 2024-12-04 18:48:05 |
+----------+-----------+----------------+-----------+-----------+---------------------+
1 row in set (0.00 sec)

mysql>
```

5. Conclusion

The three triggers implemented in this section successfully automated the logging of salary updates, employee transfers, and Project hours changes. The triggers were thoroughly tested, and their execution and results were verified with screenshots.

Stored Prodecure Development

- Purpose

  The GetProjectEmployees stored procedure retrieves the names and hours of employees working on a specific project, using the Project number as a parameter.

- SQL Script

```
mysql> DELIMITER $$
mysql> CREATE PROCEDURE GetProjectEmployees(IN projectNum INT)
    -> BEGIN
    ->     SELECT E.FName, E.LName, W.Hours
    ->     FROM Employee E
    ->     JOIN Works_On W ON E.Ssn = W.Ssn
    ->     WHERE W.Project_Number = projectNum;
    -> END$$
Query OK, 0 rows affected (0.05 sec)

mysql> DELIMITER ;
mysql>
```

- Execution

  - Before Execution

```
mysql> SELECT * FROM Works_On;
+-----------+----------------+-------+
| Ssn       | Project_Number | Hours |
+-----------+----------------+-------+
| 123456789 |            101 | 42.50 |
| 147258369 |            107 | 50.00 |
| 258963147 |            109 | 25.00 |
| 321654987 |            105 | 30.00 |
| 456789123 |            103 | 28.00 |
| 654321789 |            106 | 37.50 |
| 789456123 |            104 | 42.50 |
| 852741963 |            110 | 45.00 |
| 963852741 |            108 | 33.00 |
| 987654321 |            102 | 40.00 |
+-----------+----------------+-------+
10 rows in set (0.00 sec)

mysql>
```

```
mysql> SELECT * FROM Employee;
+-----------+---------+-------+---------+----------+------+
| Ssn       | FName   | MName | LName   | Salary   | Sex  |
+-----------+---------+-------+---------+----------+------+
| 123456789 | John    | A     | Doe     | 75000.00 | M    |
| 147258369 | William | G     | Miller  | 50000.00 | M    |
| 258963147 | James   | I     | Moore   | 62000.00 | M    |
| 321654987 | Michael | E     | White   | 58000.00 | M    |
| 456789123 | Robert  | C     | Brown   | 48000.00 | M    |
| 654321789 | Emma    | F     | Davis   | 70000.00 | F    |
| 789456123 | Emily   | D     | Johnson | 72000.00 | F    |
| 852741963 | Sophia  | J     | Taylor  | 67000.00 | F    |
| 963852741 | Olivia  | H     | Wilson  | 80000.00 | F    |
| 987654321 | Jane    | B     | Smith   | 65000.00 | F    |
+-----------+---------+-------+---------+----------+------+
10 rows in set (0.00 sec)
```

- Procedure Call

```
mysql> CALL GetProjectEmployees(101);
+-------+-------+-------+
| FName | LName | Hours |
+-------+-------+-------+
| John  | Doe   | 42.50 |
+-------+-------+-------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

mysql>
```

- After Execution

No changes were made to the tables. This section is not applicable.

Web Access Module

- Purpose
  - The Web Access Module allows users to retrieve and view employee details for specific projects through a responsive web interface. Users input a **Project Number**, and the system fetches the following details:
    - First Name

    - Last Name
    - Hours Worked
  - The modüle is built using **Flask** for backend processing, **MySQL** for data retrieval, and **HTML** for user interface.

- Scripts
  - Flask Script which is app.py
    - Below is the code script :

```
  UW PICO 5.09                                                                File: app.py

from flask import Flask, request, render_template
import mysql.connector

app = Flask(__name__)

# MySQL bağlantı ayarları
db_config = {
    'host': 'localhost',
    'user': 'root',        # MySQL kullanıcı adı
    'password': '',        # MySQL şifrenizi buraya yazın
    'database': 'my_project_db'  # Veritabanı adı
}

# MySQL'den Çalışan Verilerini Getir
def get_project_employees(project_num):
    try:
        conn = mysql.connector.connect(**db_config)
        cursor = conn.cursor(dictionary=True)
        query = "CALL GetProjectEmployees(%s)"
        cursor.execute(query, (project_num,))
        results = cursor.fetchall()
        cursor.close()
        conn.close()
        return results
    except mysql.connector.Error as err:
        print(f"MySQL Hatası: {err}")
        return []

# Ana sayfa: Formu göster
@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        project_num = request.form['projectNum']
        employees = get_project_employees(project_num)
        print(f"Flask - Employees Data: {employees}")  # Debugging için terminale yazdır
        return render_template('result.html', project_num=project_num, employees=employees)
    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=True)
```

- HTML Scripts:

```
  UW PICO 5.09                                                                File: result.html

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Results</title>
</head>
<body>
    <h1>Employees for Project #{{ project_num }}</h1>
    {% if employees %}
        <table border="1">
            <tr>
                <th>First Name</th>
                <th>Last Name</th>
                <th>Hours</th>
            </tr>
            {% for employee in employees %}
            <tr>
                <td>{{ employee.FName }}</td>
                <td>{{ employee.LName }}</td>
                <td>{{ employee.Hours }}</td>
            </tr>
            {% endfor %}
        </table>
    {% else %}
        <p>No employees found for this project.</p>
    {% endif %}
</body>
</html>
```

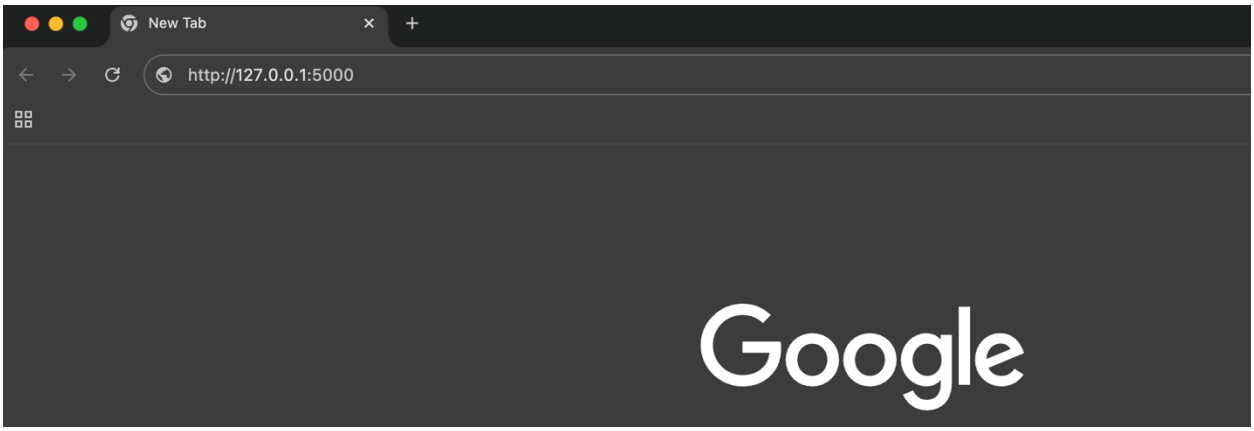
```
  UW PICO 5.09                                          File: index.html

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Get Employees</title>
</head>
<body>
    <h1>Get Employees Working on a Project</h1>
    <form method="POST">
        <label for="projectNum">Enter Project Number:</label>
        <input type="number" id="projectNum" name="projectNum" required>
        <button type="submit">Submit</button>
    </form>
</body>
</html>
```

- Execution

  - Here are screenshots of the steps of execution :

```
barkinbasaran@Barkn-MacBook-Pro WebAccessModule % python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 225-123-901
```

```
barkinbasaran@Barkn-MacBook-Pro WebAccessModule % python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 225-123-901
```

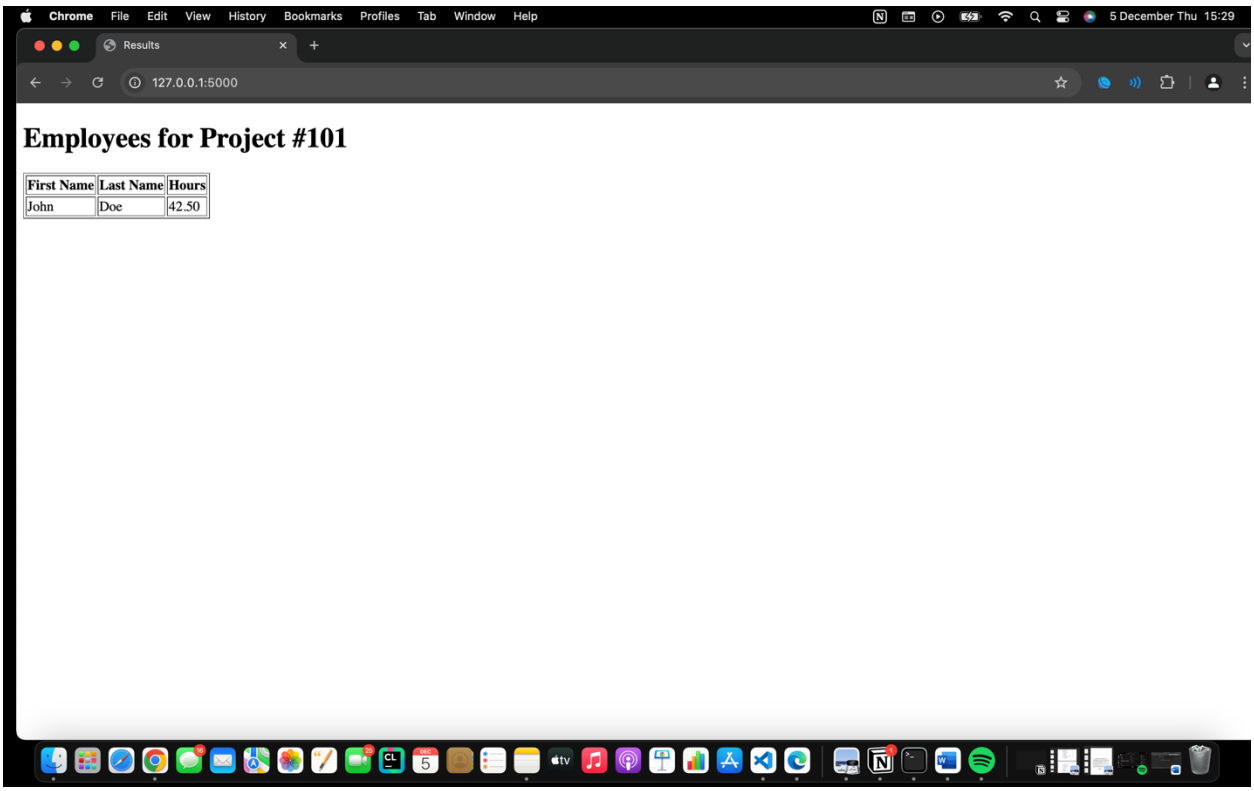


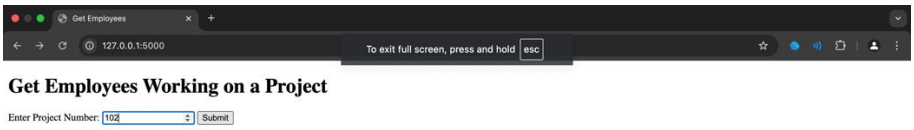
Get Employees Working on a Project

Enter Project Number: Submit

Get Employees Working on a Project

Enter Project Number: 101 Submit


Chrome File Edit View History Bookmarks Profiles Tab Window Help
5 December Thu 15:29
Results
127.0.0.1:5000
Employees for Project #101
First Name
Last Name
Hours
John
Doe
42.50


Get Employees
127.0.0.1:5000
To exit full screen, press and hold esc
Get Employees Working on a Project
Enter Project Number: 102
Submit

Employees for Project #102

First Name	Last Name	Hours
Jane	Smith	40.00

```
barkinbasaran@Barkn-MacBook-Pro WebAccessModule % python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 225-123-901
127.0.0.1 - - [05/Dec/2024 15:27:49] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [05/Dec/2024 15:27:49] "GET /favicon.ico HTTP/1.1" 404 -
Flask - Employees Data: [{'FName': 'John', 'LName': 'Doe', 'Hours': Decimal('42.50')}]
127.0.0.1 - - [05/Dec/2024 15:29:13] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [05/Dec/2024 15:29:26] "GET / HTTP/1.1" 200 -
Flask - Employees Data: [{'FName': 'Jane', 'LName': 'Smith', 'Hours': Decimal('40.00')}]
127.0.0.1 - - [05/Dec/2024 15:29:45] "POST / HTTP/1.1" 200 -
```