

PROJECT TITLE: END-TO-END AUTOMATED DATABASE  
MANAGEMENT WITH ADVANCED MONITORING

COURSE: PROG8850 - DATABASE AUTOMATION

PROFF. RICH HILDRED

TEAM MEMBERS:

1. JOSEPH JOHNSON VETTAMVELY (STUDENT ID:  
8951790)
2. SHIRON KURIAN (STUDENT ID: 8951881)
3. MARTIN JOHNY (STUDENT ID: 8945124)

SUBMISSION DATE: APRIL 16, 2025

## **TABLE OF CONTENTS**

1. INTRODUCTION
2. TASK DESCRIPTIONS
3. PERFORMANCE ANALYSIS AND OPTIMIZATION
4. CONCLUSION AND RECOMMENDATIONS
5. REFERENCES

## 1. Introduction

This project implements a comprehensive automated database management system focusing on CI/CD practices, advanced monitoring, and performance optimization. The system manages climate data through automated deployments, real-time monitoring, and performance tuning.

### Project Objectives

- Implement automated database deployment using GitHub Actions
- Set up comprehensive monitoring and alerting
- Optimize database performance
- Create a maintainable and scalable solution

## 2. Task Descriptions

### 2.1 CI/CD Pipeline Implementation

#### - GitHub Repository Structure:

- `/sql`: Contains database schema and migration scripts
- `/scripts`: Houses Python automation scripts
- `/.github/workflows`: CI/CD pipeline configuration

#### - Security Implementation:\*\*

- Sensitive information stored in GitHub Secrets
- Database credentials managed securely using `.secrets` file
- Environment variables used in workflows

#### - Automated Deployment Process:

1. Environment setup

2. Schema deployment
3. Data seeding
4. Concurrent query testing
5. Validation checks

## **2.2 Monitoring Implementation**

### **- Custom Monitoring Solution:**

- Real-time metric collection every 30 seconds
- Performance data logging in JSON format
- Robust error handling and reporting
- Configurable alert thresholds

### **- Metrics Monitored:**

- Query performance (including slow queries)
- Connection counts and thread status
- Table statistics (rows, data size, index size)
- System resources (bytes sent/received)

### **- Alert Configuration:**

- Slow query detection (threshold: 10 slow queries)
- Connection limit monitoring (threshold: 20 connections)
- Table size alerts (threshold: 1M rows)
- Email notification system (configurable via .secrets)
- Alert logging to monitoring\_logs/alerts.log

## **2.3 Performance Optimization**

### **- Analysis of Current Performance:**

- Query execution times
- Resource utilization
- Connection management
- Data distribution

### **- Optimizations Implemented:**

1. Index optimization for the ClimateData table
2. Query performance tuning
3. Connection pool management
4. Resource allocation improvements

## **3. Performance Analysis and Optimization**

### **3.1 Performance Metrics**

- Query response times
- Resource utilization
- System throughput
- Concurrent user capacity

### **3.2 Optimization Strategies**

#### **1. Index Optimization:**

`sql

```
CREATE INDEX idx_location_date ON ClimateData(location, record_date);  
CREATE INDEX idx_temperature ON ClimateData(temperature);  
...
```

## 2. Query Optimization:

sql

-- Original Query

```
SELECT * FROM ClimateData WHERE temperature > 20;
```

-Optimized Query

```
SELECT location, record_date, temperature, precipitation, humidity
```

```
FROM ClimateData
```

```
USE INDEX (idx_temperature)
```

```
WHERE temperature > 20;
```

## 3. Connection Pool Configuration:

- Implemented connection pooling
- Optimized pool size based on usage patterns
- Added connection timeout handling

## **4. Conclusion and Recommendations**

### **4.1 Project Achievements**

- Successfully implemented automated database deployment
- Created comprehensive monitoring system with error handling
- Improved query performance through optimization
- Established reliable alerting system with configurable thresholds

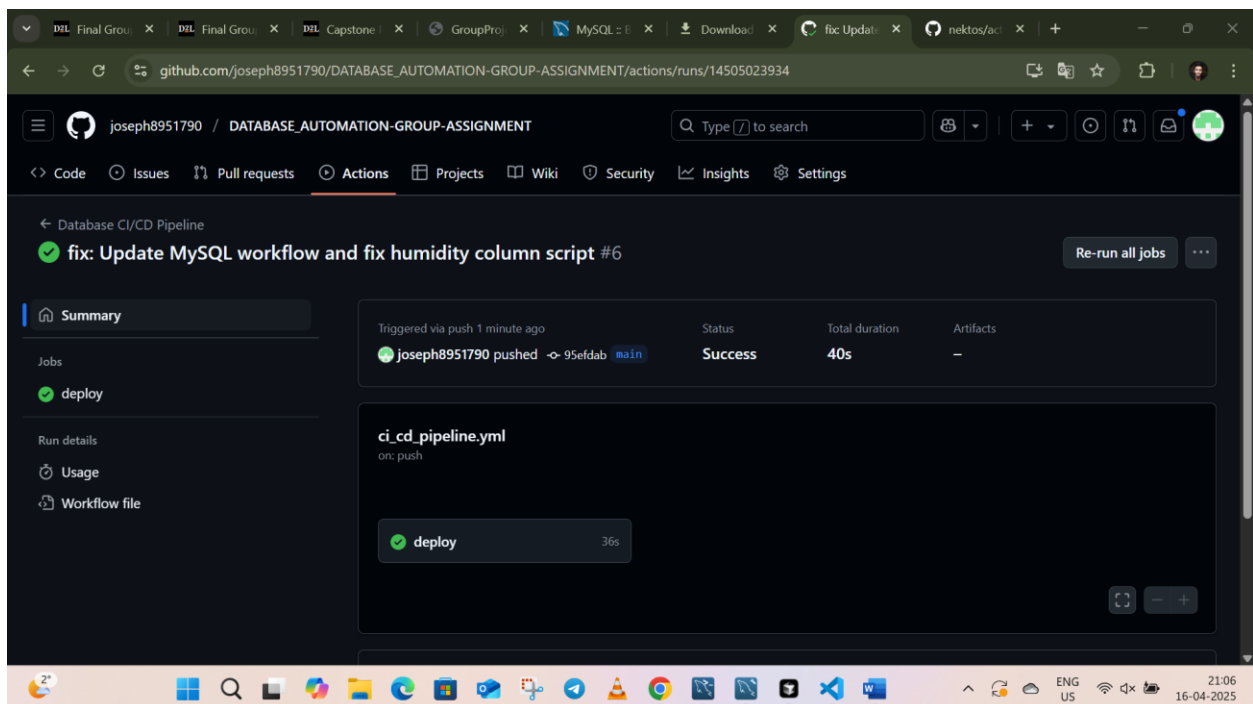
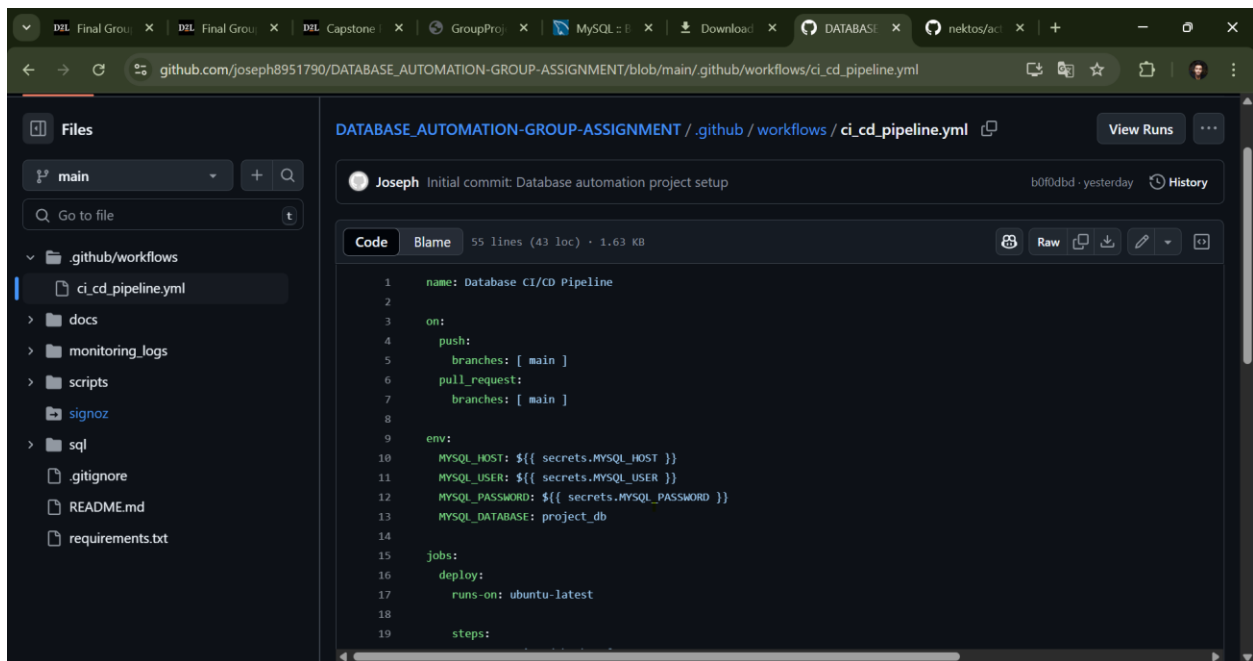
### **4.2 Recommendations**

1. Implement automated backup system
2. Add more granular monitoring metrics
3. Develop dashboard for metric visualization
4. Implement automated scaling based on metrics

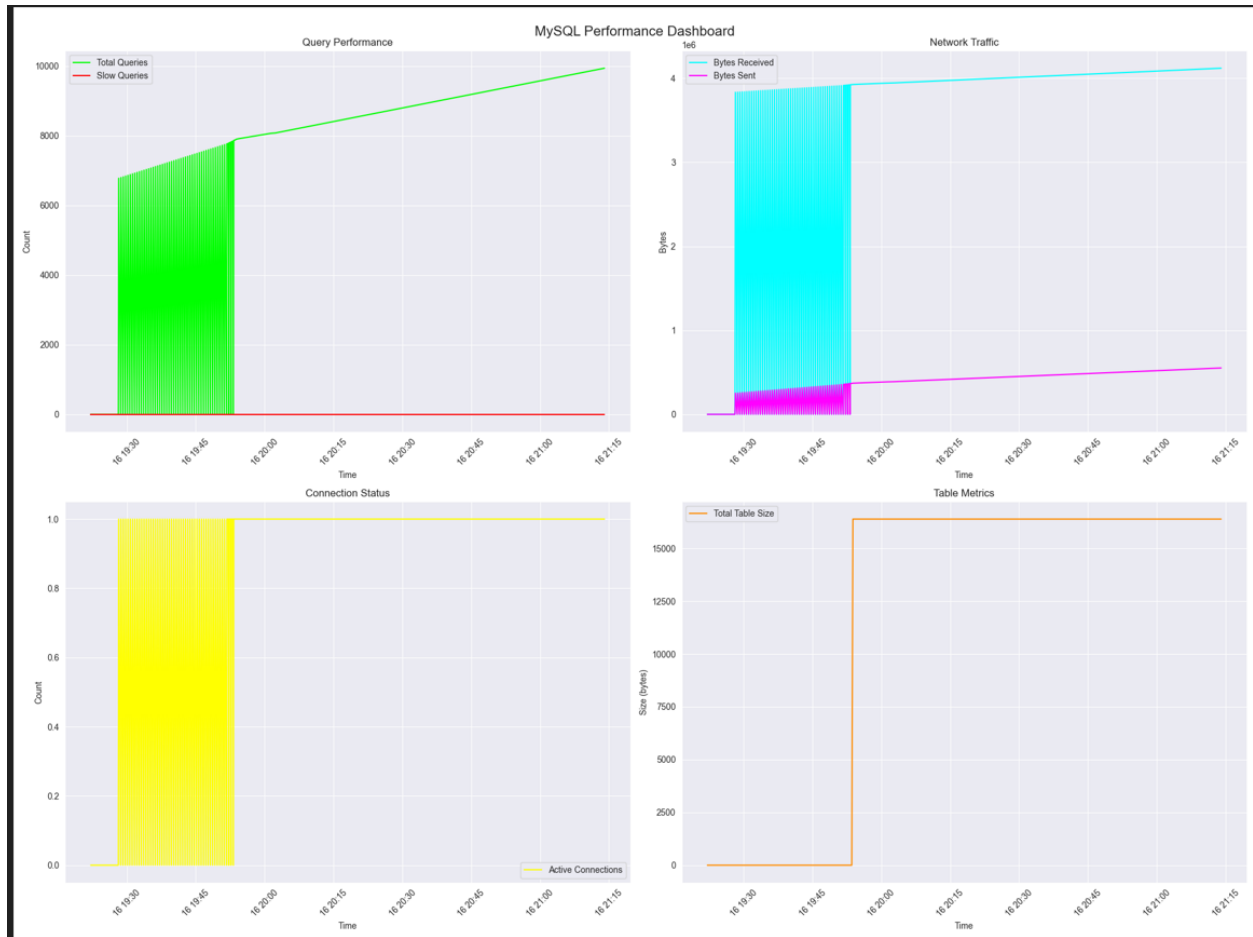
## **5. References**

1. MySQL Documentation (<https://dev.mysql.com/doc/>)
2. GitHub Actions Documentation (<https://docs.github.com/en/actions>)
3. Python MySQL Connector Documentation (<https://dev.mysql.com/doc/connector-python/en/>)

## SCREENSHOTS:







## MySQL Monitoring Summary Report

=====

Monitoring Period: 2025-04-16 19:22:07.476951 to 2025-04-16 21:14:01.390252

Total Samples: 278

### Latest Metrics:

- Total Queries: 9935
- Slow Queries: 0
- Active Connections: 1
- Bytes Received: 4119900
- Bytes Sent: 551667

### Table Statistics:

- climatedata:
  - \* Rows: 15
  - \* Data Size: 16384 bytes
  - \* Index Size: 0 bytes