# Assignment 2: Automating Database Schema Changes and Implementing CI/CD for Database Deployment

Student: Jithin Jyothi  
Course: Database Automation and Scripting

**Github Repository:** <https://github.com/JithinJyothi95/A2-DBAutomation>

# 1. Introduction

This report details the steps undertaken for Assignment 2 of PROG8850. The objective was to automate database schema changes using SQL and Python and implement a CI/CD pipeline using GitHub Actions to deploy these changes to an Azure MySQL database.

# 2. Automating Database Schema Changes

A SQL script (`projects\_schema.sql`) was created to define a new table called `projects` with required columns, and another SQL command was used to conditionally add a `budget` column. A Python script (`database\_automation.py`) was developed using `mysql-connector-python` to automate the execution of this SQL script.

The script connects to the MySQL database, executes each statement from the SQL file, and commits the changes.

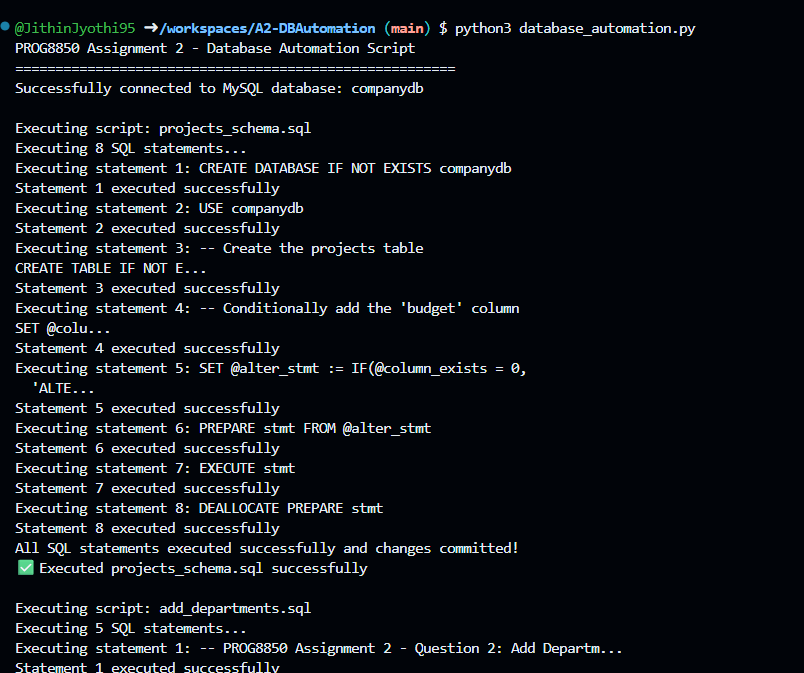


Figure 1: Successful execution of database\_automation.py script.

# 3. CI/CD Pipeline Implementation

A GitHub Actions workflow named `ci\_cd\_pipeline.yml` was created. It runs on every push to the `main` branch, sets up the MySQL container service, installs required dependencies, executes SQL scripts and the automation script, and verifies schema changes.

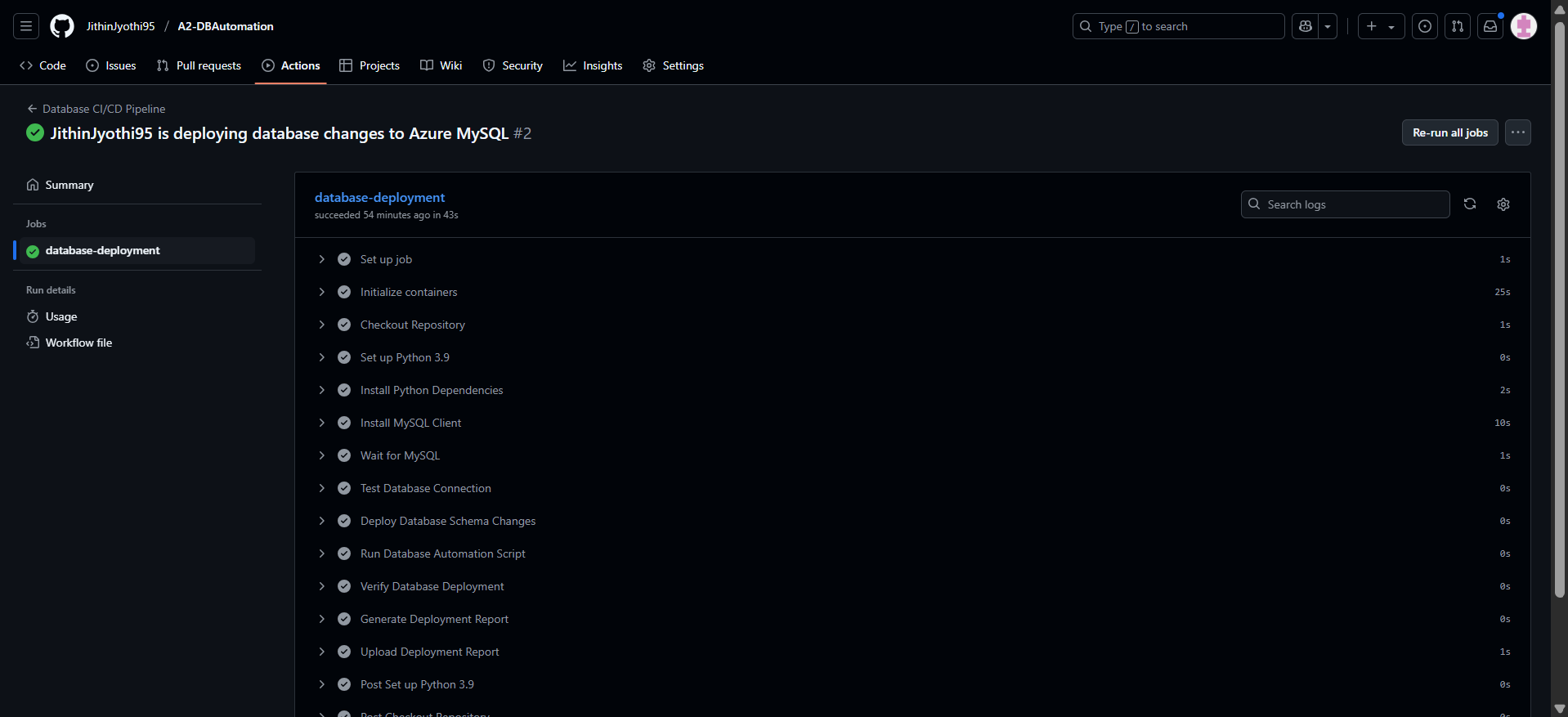


Figure 2: GitHub Actions pipeline execution succeeded.

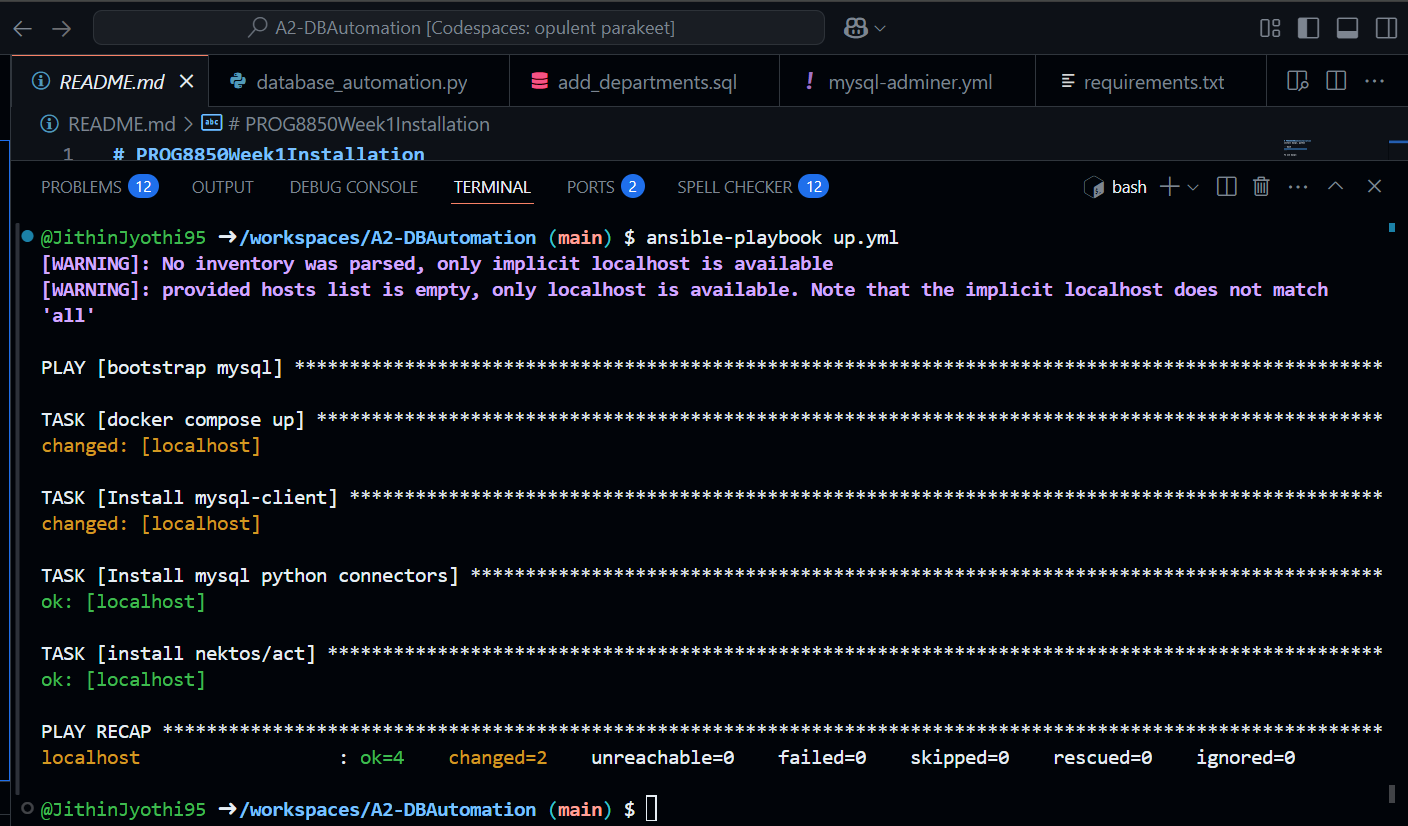


Figure 3: Ansible playbook successfully sets up the local MySQL environment.

# 4. Database Deployment Testing

A SQL script named `add\_departments.sql` was created to add a `departments` table to the `companydb` database. The GitHub Actions workflow was tested and verified successful creation and population of this table.

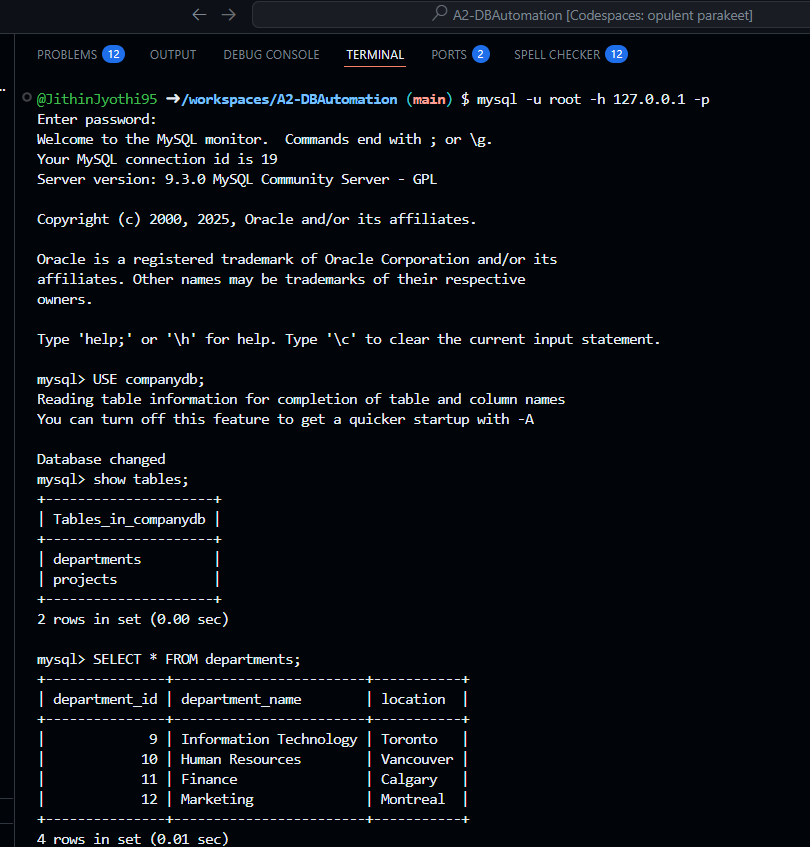


Figure 4: `projects` and `departments` tables created successfully.

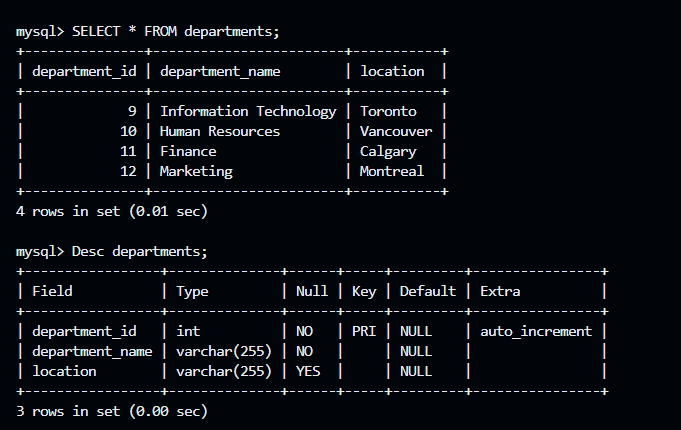


Figure 5: Schema of `projects` and `departments` tables.

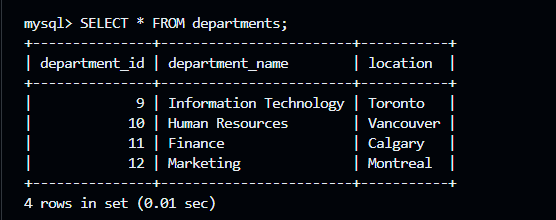


Figure 6: Sample data from `departments` table.

# 5. Project Folder Structure and Validation

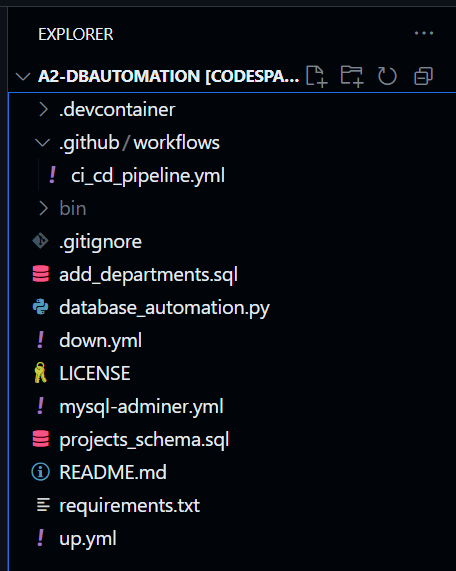


Figure 7: Organized project structure with SQL, Python, YAML files.

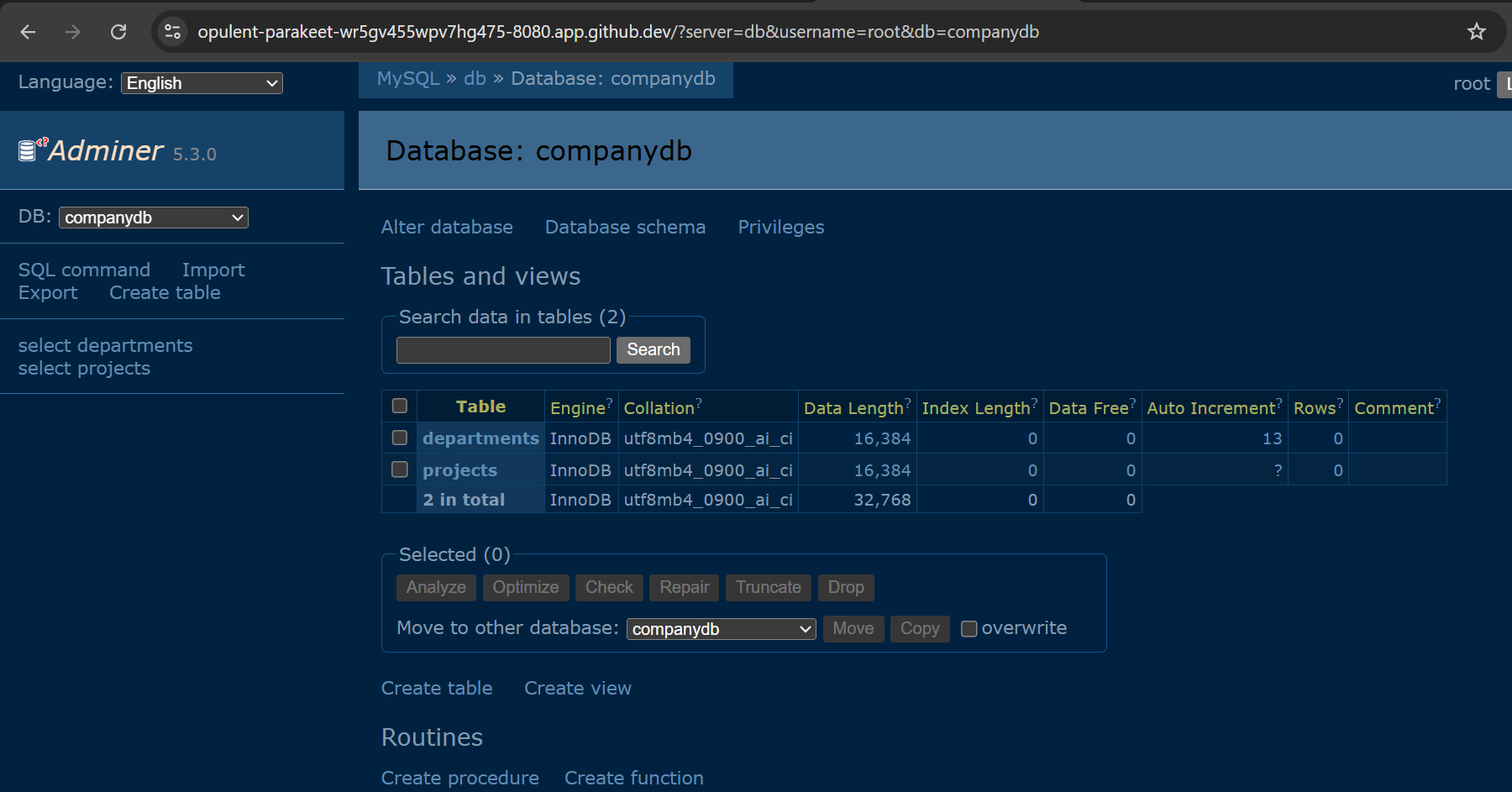
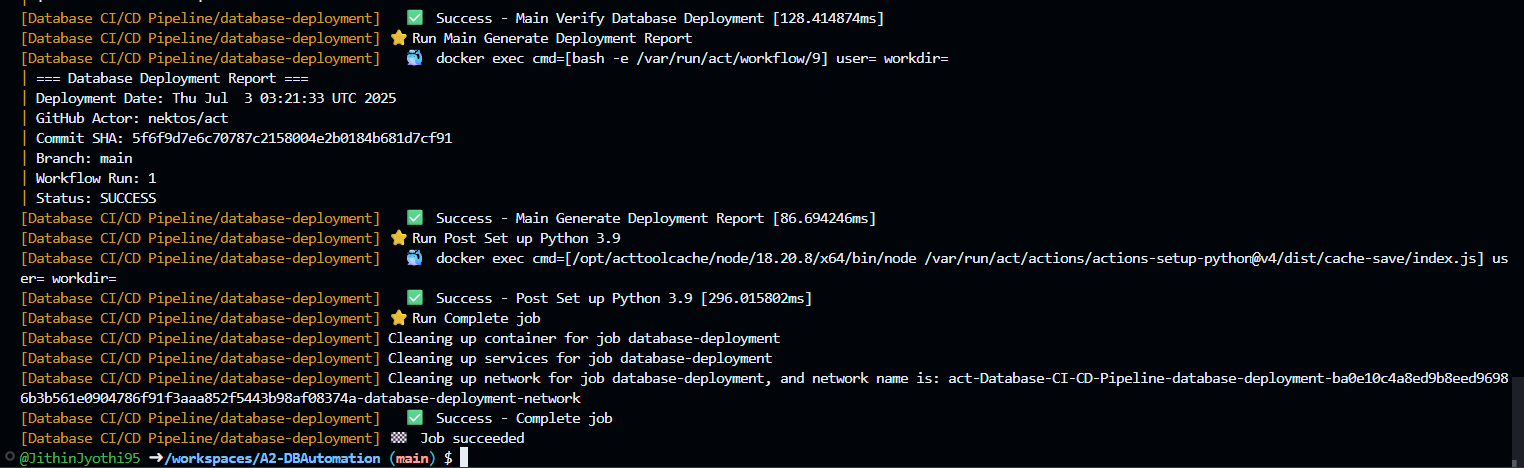


Figure 8: Adminer view showing table contents.

# 6. Deployment Report Artifact

The GitHub Actions workflow also generates a `deployment\_report.txt` that includes key metadata such as actor, commit SHA, branch name, and status. This serves as an audit trail for database deployments.



A screenshot of a computer

AI-generated content may be incorrect.

Figure 9: Deployment report artifact generated

# 7. Conclusion

This assignment successfully demonstrated the automation of database schema changes and implementation of a CI/CD pipeline using GitHub Actions. By integrating SQL, Python, and GitHub workflows, database deployments are now reproducible, verifiable, and fully automated.