Assignment 4 - Question 1: Analysis and Integration of Database Automation Tools

1.1 Overview and Key Features

GitHub Actions

GitHub Actions is a powerful CI/CD tool provided by GitHub. It allows developers to automate their workflows with simple YAML configuration files directly in their repositories.

Key Features:
- Integrated with GitHub repositories
- Supports matrix builds across different OS and languages
- Triggers based on Git events (push, pull request, etc.)
- Marketplace for reusable Actions
- Native support for secrets and environment variables

Flyway

Flyway is a lightweight, open-source database migration tool. It is ideal for implementing version control in database development and integrating database changes in CI/CD pipelines.

Key Features:
- Version control for database schemas
- Supports SQL and Java-based migrations
- Works with many databases (MySQL, PostgreSQL, Oracle, etc.)
- Command-line interface and integration with build tools
- Repeatable, undo, and baseline migrations support

Comparison Table

Criteria	GitHub Actions	Flyway
Ease of Use	Easy with YAML workflow syntax	Straightforward CLI and SQL-based
CI/CD Integration	Native GitHub support	Requires scripting or manual trigger
Supported Databases	N/A (not a DB tool)	MySQL, PostgreSQL, Oracle, SQL Server, etc.

1.2 Integration Strategy

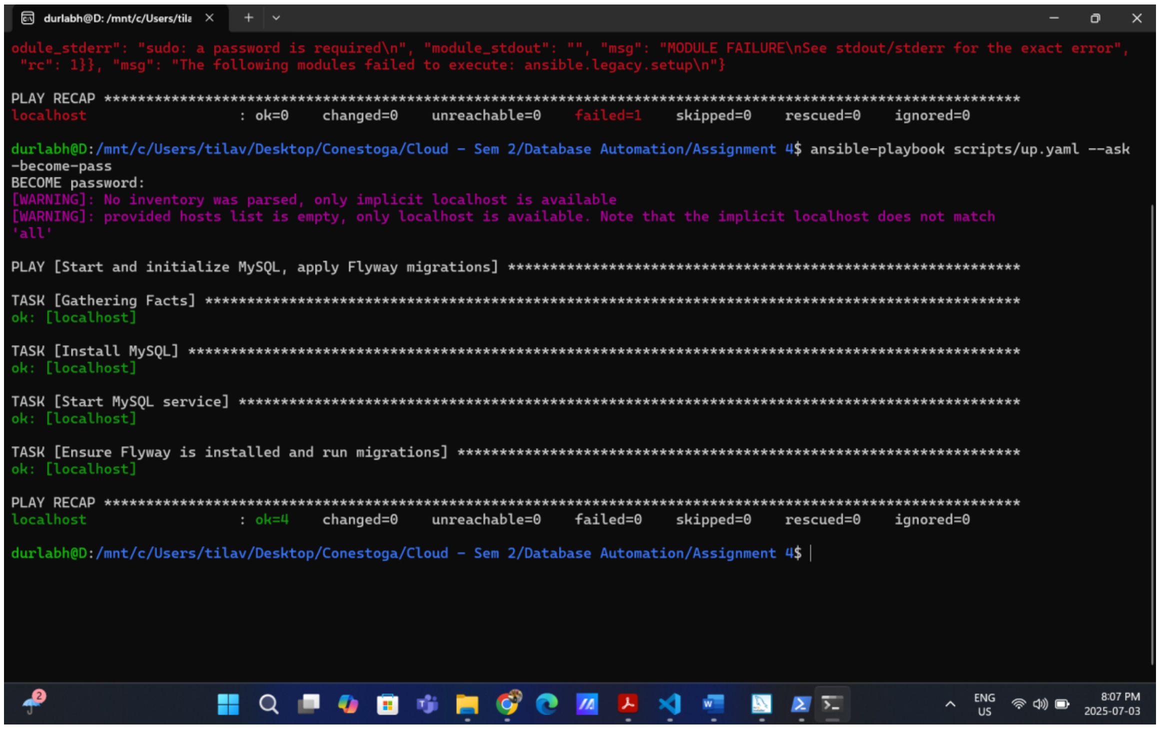
To integrate GitHub Actions and Flyway into a CI/CD pipeline:

1. Define a GitHub Actions workflow file (e.g., `.github/workflows/db-deploy.yml`).
2. Configure the job to run on push/PR to the main branch.
3. Include a step that installs Flyway CLI.
4. Set up Flyway with environment variables for DB credentials.
5. Use a Flyway command to apply migrations (`flyway migrate`).
6. Add a step to run `dbtests.py` to validate schema changes.

This ensures database changes are automated, tested, and version-controlled along with code deployments.

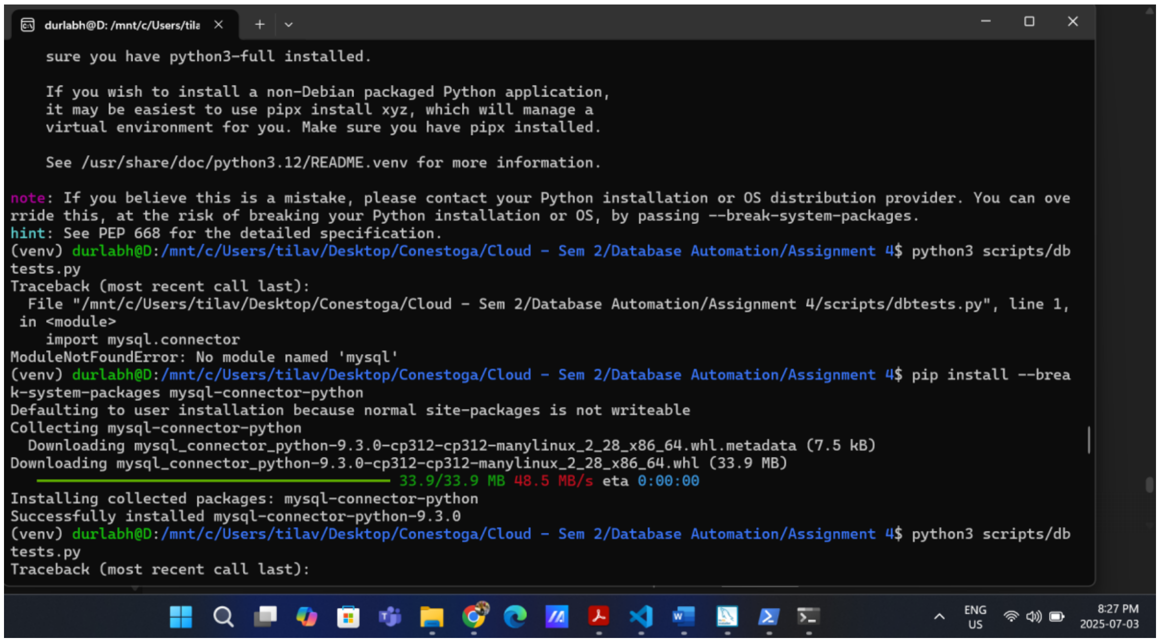
- ❖ **Screenshots:**

- Run Ansible Playbook:



```
durlabh@D: /mnt/c/Users/tila

odule_stderr": "sudo: a password is required\n", "module_stdout": "", "msg": "MODULE FAILURE\nSee stdout/stderr for the exact error",
 "rc": 1}}, "msg": "The following modules failed to execute: ansible.legacy.setup\n"}

PLAY RECAP *************************************************************************************************
localhost                  : ok=0    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0

durlabh@D:/mnt/c/Users/tilav/Desktop/Conestoga/Cloud - Sem 2/Database Automation/Assignment 4$ ansible-playbook scripts/up.yaml --ask
-become-pass
BECOME password:
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match
'all'

PLAY [Start and initialize MySQL, apply Flyway migrations] *************************************************

TASK [Gathering Facts] *************************************************************************************
ok: [localhost]

TASK [Install MySQL] ***************************************************************************************
ok: [localhost]

TASK [Start MySQL service] *********************************************************************************
ok: [localhost]

TASK [Ensure Flyway is installed and run migrations] *******************************************************
ok: [localhost]

PLAY RECAP *************************************************************************************************
localhost                  : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

durlabh@D:/mnt/c/Users/tilav/Desktop/Conestoga/Cloud - Sem 2/Database Automation/Assignment 4$ |
```

- dbtests.py showing validation passed:



```
    sure you have python3-full installed.

    If you wish to install a non-Debian packaged Python application,
    it may be easiest to use pipx install xyz, which will manage a
    virtual environment for you. Make sure you have pipx installed.

    See /usr/share/doc/python3.12/README.venv for more information.

note: If you believe this is a mistake, please contact your Python installation or OS distribution provider. You can ove
rride this, at the risk of breaking your Python installation or OS, by passing --break-system-packages.
hint: See PEP 668 for the detailed specification.
(venv) durlabh@D:/mnt/c/Users/tilav/Desktop/Conestoga/Cloud - Sem 2/Database Automation/Assignment 4$ python3 scripts/db
tests.py
Traceback (most recent call last):
  File "/mnt/c/Users/tilav/Desktop/Conestoga/Cloud - Sem 2/Database Automation/Assignment 4/scripts/dbtests.py", line 1,
 in <module>
    import mysql.connector
ModuleNotFoundError: No module named 'mysql'
(venv) durlabh@D:/mnt/c/Users/tilav/Desktop/Conestoga/Cloud - Sem 2/Database Automation/Assignment 4$ pip install --brea
k-system-packages mysql-connector-python
Defaulting to user installation because normal site-packages is not writeable
Collecting mysql-connector-python
  Downloading mysql_connector_python-9.3.0-cp312-cp312-manylinux_2_28_x86_64.whl.metadata (7.5 kB)
Downloading mysql_connector_python-9.3.0-cp312-cp312-manylinux_2_28_x86_64.whl (33.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 33.9/33.9 MB 48.5 MB/s eta 0:00:00
Installing collected packages: mysql-connector-python
Successfully installed mysql-connector-python-9.3.0
(venv) durlabh@D:/mnt/c/Users/tilav/Desktop/Conestoga/Cloud - Sem 2/Database Automation/Assignment 4$ python3 scripts/db
tests.py
Traceback (most recent call last):
```