**Dokumentasi Pengujian Unit untuk Kelas HistoryManager**

1. **Persyaratan Sistem**

* Python 3.10.
* Library Python berikut harus terinstal:
  + unittest
  + unittest.mock

1. **Struktur Kode yang Diuji**

Kelas HistoryManager bertanggung jawab untuk mengelola riwayat perhitungan dalam sebuah database. Kelas ini memiliki beberapa metode utama:

* fetch\_history: Mengambil riwayat perhitungan dari database.
* insert\_history: Menyimpan ekspresi dan hasil perhitungan ke database.
* delete\_all\_history: Menghapus semua riwayat perhitungan dari database.

1. **Struktur Pengujian**

Pengujian dilakukan menggunakan modul unittest dan unittest.mock untuk membuat mock objek database dan cursor.

1. **Implementasi Pengujian**

|  |
| --- |
| **import unittest**  **from unittest.mock import MagicMock**  **class HistoryManager:**  **def \_\_init\_\_(self, db):**  **self.db = db**  **self.cursor = self.db.cursor()**    **def fetch\_history(self):**  **self.cursor.execute("SELECT expression, result FROM history ORDER BY id DESC")**  **return self.cursor.fetchall()**  **def insert\_history(self, expression, result):**  **self.cursor.execute("INSERT INTO history (expression, result) VALUES (%s, %s)", (expression, result))**  **self.db.commit()**  **def delete\_all\_history(self):**  **self.cursor.execute("DELETE FROM history")**  **self.db.commit()**  **self.histories = []**  **self.update\_history\_expression('')**  **self.update\_expression('')**  **def update\_history\_expression(self, expression):**  **pass**  **def update\_expression(self, expression):**  **pass**  **class TestHistoryManager(unittest.TestCase):**  **def setUp(self):**  **self.mock\_db = MagicMock()**  **self.mock\_cursor = MagicMock()**  **self.mock\_db.cursor.return\_value = self.mock\_cursor**  **self.history\_manager = HistoryManager(self.mock\_db)**  **def test\_fetch\_history(self):**  **expected\_result = [("1+1", "2"), ("2+2", "4")]**  **self.mock\_cursor.fetchall.return\_value = expected\_result**  **result = self.history\_manager.fetch\_history()**  **self.mock\_cursor.execute.assert\_called\_once\_with("SELECT expression, result FROM history ORDER BY id DESC")**  **self.assertEqual(result, expected\_result)**  **def test\_insert\_history(self):**  **expression = "3+3"**  **result = "6"**  **self.history\_manager.insert\_history(expression, result)**  **self.mock\_cursor.execute.assert\_called\_once\_with("INSERT INTO history (expression, result) VALUES (%s, %s)", (expression, result))**  **self.mock\_db.commit.assert\_called\_once()**  **def test\_delete\_all\_history(self):**  **self.history\_manager.delete\_all\_history()**  **self.mock\_cursor.execute.assert\_called\_once\_with("DELETE FROM history")**  **self.mock\_db.commit.assert\_called\_once()**  **self.assertEqual(self.history\_manager.histories, [])**  **if \_\_name\_\_ == '\_\_main\_\_':**  **unittest.main()** |

1. **Penjelasan Pengujian**

**setUp Method**

* **Tujuan:** Menginisialisasi mock database dan cursor sebelum setiap pengujian.
* **Implementasi:** Membuat objek MagicMock untuk database dan cursor, kemudian menginisialisasi objek HistoryManager dengan mock database.

**test\_fetch\_history Method**

* **Tujuan:** Menguji apakah metode fetch\_history mengambil data yang benar dari database.
* **Implementasi:** Mengatur return\_value dari fetchall pada mock cursor, lalu memanggil fetch\_history dan memverifikasi apakah execute dipanggil dengan query yang benar dan hasil yang dikembalikan sesuai dengan yang diharapkan.

**test\_insert\_history Method**

* **Tujuan:** Menguji apakah metode insert\_history menyimpan data yang benar ke database.
* **Implementasi:** Memanggil insert\_history dengan ekspresi dan hasil, lalu memverifikasi apakah execute dipanggil dengan query yang benar dan commit dipanggil sekali.

**test\_delete\_all\_history Method**

* **Tujuan:** Menguji apakah metode delete\_all\_history menghapus semua riwayat dari database dan memperbarui atribut internal histories.
* **Implementasi:** Memanggil delete\_all\_history, lalu memverifikasi apakah execute dipanggil dengan query yang benar, commit dipanggil sekali, dan atribut histories kosong.

1. **Menjalankan Pengujian**

Simpan kode pengujian ke dalam file, misalnya test\_history\_manager.py, dan jalankan pengujian dengan perintah:

*python test\_history\_manager.py*