Implementasi Class Diagram

|  |
| --- |
| NPM-Nama lengkap : 5240411230-Muhammad Zaki As Shidiqi |
| Judul Tugas Akhir : Rancang Bangun Chatbot sebagai Pusat Informasi Lembaga Kursus dan Pelatihan Menggunakan Pendekatan Natural Language Processing |
| Nama pemilik karya : AGRA LAKSMI MAITRI |
| Use Case Diagram :    Penjelasan :  Berdasarkan usecase diagram tersebut dapat disimpulkan bahwa admin dapat melakukan 6 kegiatan, yaitu mengelola user, mengelola data mata pelajaran, mengelola data jadwal, mengelola data biaya, mengelola data pertanyaan, dan mengelola data stopword. Pelanggan hanya dapat melakukan 2 kegiatan, yaitu membuka chat telegram dan mengajukan pertanyaan. |
| Class Diagram :    Penjelasan :  Berdasarkan usecase diagram tersebut terdapat 6 class yang memiliki atribut dan method sebagai berikut:   1. User 2. Atribut  * id bertipe integer. * username bertipe varchar. * password bertipe varchar.  1. Method  * getUser(), untuk mendapatkan / request data user. * process\_edit(), untuk melakukan edit. * createUser(), dapat melakukan registrasi. * deleteUser(), dapat menghapus akun.  1. Mapel 2. Atribut  * id bertipe integer. * mata\_pelajaran bertipe varchar.  1. Method  * tambahMapel(), untuk menambahkan mata pelajaran. * tampilData(), untuk menampilkan data. * updateData(), untuk memperbarui data. * hapus(), untuk menghapus data.  1. Jadwal 2. Atribut  * id bertipe integer * id\_mapel bertipe integer * hari bertipe varchar * waktu tidak diebutkan  1. Method  * tambahJadwal(), untuk menambahkan jadwal. * tampilData(), untuk menampilkan data. * updateData(), untuk memperbarui data. * hapus(), untuk menghapus data.  1. Biaya 2. Atribut  * id bertipe integer. * id\_mapel bertipe integer * biaya bertipe varchar  1. Method  * tambahBiaya (), untuk menambahkan biaya. * tampilData(), untuk menampilkan data. * updateData(), untuk memperbarui data. * hapus(), untuk menghapus data.  1. Pertanyaan 2. Atribut  * id bertipe integer * mata\_pelajaran bertipe varchar  1. Method  * tambahPertanyaan (), untuk menambahkan pertanyaan. * updateData(), untuk memperbarui data. * hapus(), untuk menghapus data.  1. Stopword 2. Atribut  * id bertipe integer * stopword bertipe varchar  1. Method  * tambahStopword (), untuk menambahkan stopword. * tampilData(), untuk menampilkan data. * updateData(), untuk memperbarui data. * hapus(), untuk menghapus data. |
| Implementasi Class Diagram :  class Stopword:      def \_\_init\_\_(self, id, stopword):          self.id = id          self.stopword = stopword      def tambah\_stopword(self, kata):          self.stopword = kata      def tampil\_data(self):          return f"{self.id}. {self.stopword}"      def update\_data(self, new\_word):          self.stopword = new\_word      def hapus(self):          del self.stopword  class User:      def \_\_init\_\_(self, id, username, password):          self.id = id          self.username = username          self.password = password      def get\_user(self):          return {"id": self.id, "username": self.username}      def process\_edit(self, username=None, password=None):          if username:              self.username = username          if password:              self.password = password      def create\_user(self, id, username, password):          return User(id, username, password)      def delete\_user(self):          del self.username          del self.password  class Mapel:      def \_\_init\_\_(self, id, mata\_pelajaran):          self.id = id          self.mata\_pelajaran = mata\_pelajaran      def tambah\_mapel(self, nama):          self.mata\_pelajaran = nama      def tampil\_data(self):          return f"[{self.id}] {self.mata\_pelajaran}"      def update\_data(self, nama\_baru):          self.mata\_pelajaran = nama\_baru      def hapus(self):          del self.mata\_pelajaran  class Jadwal:      def \_\_init\_\_(self, id, id\_mapel, waktu):          self.id = id          self.id\_mapel = id\_mapel          self.waktu = waktu      def tambah\_jadwal(self, waktu):          self.waktu = waktu      def tampil\_data(self):          return f"Mapel dengan ID {self.id\_mapel} dijadwalkan pada {self.waktu}"      def update\_data(self, waktu\_baru):          self.waktu = waktu\_baru      def hapus(self):          del self.waktu  class Biaya:      def \_\_init\_\_(self, id, id\_mapel, biaya):          self.id = id          self.id\_mapel = id\_mapel          self.biaya = biaya      def tampil\_data(self):          return f"Biaya untuk mapel dengan ID {self.id\_mapel}: Rp {self.biaya}"      def update\_data(self, biaya\_baru):          self.biaya = biaya\_baru      def delete(self):          del self.biaya      def tambah\_biaya(self, biaya):          self.biaya = biaya  class Pertanyaan:      def \_\_init\_\_(self, id, mata\_pelajaran, pertanyaan):          self.id = id          self.mata\_pelajaran = mata\_pelajaran          self.pertanyaan = pertanyaan      def tambah\_pertanyaan(self, pertanyaan):          self.pertanyaan = pertanyaan      def tampil\_data(self):          return f"[{self.mata\_pelajaran}] {self.pertanyaan}"      def hapus(self):          del self.pertanyaan  # Implementasi  if \_\_name\_\_ == "\_\_main\_\_":      user1 = User(1, "admin", "1234")      mapel1 = Mapel(1, "Bahasa Inggris")      jadwal1 = Jadwal(1, mapel1.id, "Senin 09:00")      biaya1 = Biaya(1, mapel1.id, "500000")      print(user1.get\_user())      print(mapel1.tampil\_data())      print(jadwal1.tampil\_data())      print(biaya1.tampil\_data())  Penjelasan :  Program ini menggunakan konsep OOP, artinya terdapat class, atribut, dan method. Program ini dibuat berdasarkan class diagram sebelumnya telah dibahas. Setelah sintaks class terdapat nama class yang sudah sesuai dengan diagram. Pada awal baris setiap class terdapat \_\_init\_\_, ini akan menginisialisasi class tersebut akan memiliki atribut yang letaknya ada di dalam \_\_init\_\_. Setelah \_\_init\_\_ akan berisi method-method, artinya setiap class dapat memiliki method lebih dari satu. Pada akhir program ini terdapat contoh implementasi untuk menjalankan program, berikut adalah output dari kode tersebut. |