**LAPORAN**

**TUGAS AKHIR**

Mata Kuliah Pemrograman Berorientasi Objek



Disusun Oleh:

1. Alfin Afriza M (2213020173)
2. Mohammad Ubaidillah Ridlo (2213020215)
3. Raya Osgibran (2213020139)

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK DAN ILMU KOMPUTER

UNIVERSITAS NUSANTARA PGRI KEDIRI

TAHUN 2023

Kata Pengantar

Dengan rasa syukur dan penuh kegembiraan, penulis menyampaikan puji syukur kepada Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya yang telah melimpah selama proses pembuatan tugas akhir ini. Tugas akhir ini merupakan hasil perjuangan dan dedikasi penulis dalam mengembangkan aplikasi sederhana berbasis PyQt ini yang bertujuan untuk memudahkan manajemen acara.

Tidak lupa, penulis mengucapkan terima kasih kepada dosen pembimbing, teman-teman sejawat, dan semua pihak yang telah memberikan dukungan, bimbingan, serta masukan konstruktif selama perjalanan pembuatan tugas akhir ini.

Pada kesempatan ini, penulis ingin menegaskan bahwa tugas akhir ini bukanlah hasil akhir yang sempurna. Oleh karena itu, setiap kritik, saran, dan masukan membangun dari pembaca akan sangat dihargai untuk pengembangan selanjutnya.

Semoga tugas akhir ini dapat memberikan manfaat, inspirasi, dan kontribusi positif terhadap dunia pemrograman berorientasi objek, khususnya dalam pengembangan aplikasi menggunakan PyQt.

Akhir kata, penulis berharap semoga tugas akhir ini dapat menjadi pijakan awal untuk mengeksplorasi dan mengembangkan lebih lanjut di masa mendatang. Terima kasih.

Daftar Isi

**Kata Pengantar...................................................................................................................... i**

**Daftar Isi..........................................................................................................................…... ii**

**BAB I PENDAHULUAN…………………………………………………………………. 1**

1.1  Latar Belakang ............................................................................................. **1**

**BAB II PEMBAHASAN………………………………………………………………….. 2**

2.1 Flowchart Sistem………............................................................................... **2**

2.2 Class Diagram................................................................................................ **5**

2.3 Source code…………………........................................................................ **7**

2.4 Hasil Program Dan Penjelasan…………………………………………….. 27

**BAB III PENUTUP………………………………………………………………………... 29**

3.1 Kesimpulan..................................................................................................... **29**

**Daftar Pustaka…………………………………………………………………………….. 30**

BAB I

PENDAHULUAN

* 1. Latar Belakang

Dalam era perkembangan teknologi informasi yang begitu pesat, penggunaan aplikasi perangkat lunak semakin meluas dan mendalam ke berbagai sektor kehidupan. Seiring dengan itu, tuntutan akan aplikasi yang efisien, user-friendly, dan dapat memenuhi kebutuhan pengguna menjadi suatu keharusan. Dalam konteks tersebut, pemrograman berorientasi objek (PBO) menjadi pendekatan yang relevan dan banyak digunakan untuk merancang aplikasi yang modular dan mudah dimengerti.

Dalam mata kuliah pemrograman berorientasi objek, penulis merasa perlu untuk mengembangkan suatu aplikasi sebagai bentuk implementasi dari konsep-konsep yang telah dipelajari. Dalam hal ini, pemilihan PyQt sebagai framework pengembangan aplikasi GUI (Graphical User Interface) memberikan keleluasaan dan kemudahan dalam merancang antarmuka pengguna yang menarik.

Latar belakang pembuatan aplikasi ini didasarkan pada kebutuhan untuk menyederhanakan manajemen acara, baik acara pribadi maupun acara di tingkat organisasi. Dengan adanya aplikasi ini, diharapkan pengguna dapat dengan mudah melihat, mengedit, dan menghapus informasi terkait acara tanpa harus bersusah payah melakukan proses manual yang cenderung rumit dan memakan waktu.

Melalui tugas akhir ini, penulis berharap dapat memberikan kontribusi kecil dalam pengembangan aplikasi berbasis PyQt dan membuka peluang untuk eksplorasi lebih lanjut terkait pemrograman berorientasi objek. Selain itu, aplikasi ini juga diharapkan dapat memberikan manfaat nyata bagi pengguna dalam memudahkan pengelolaan jadwal acara sehari-hari.

Dengan latar belakang tersebut, tugas akhir ini bertujuan untuk memberikan solusi praktis melalui implementasi konsep-konsep PBO dalam pembuatan aplikasi manajemen acara berbasis desktop.

BAB II

PEMBAHASAN

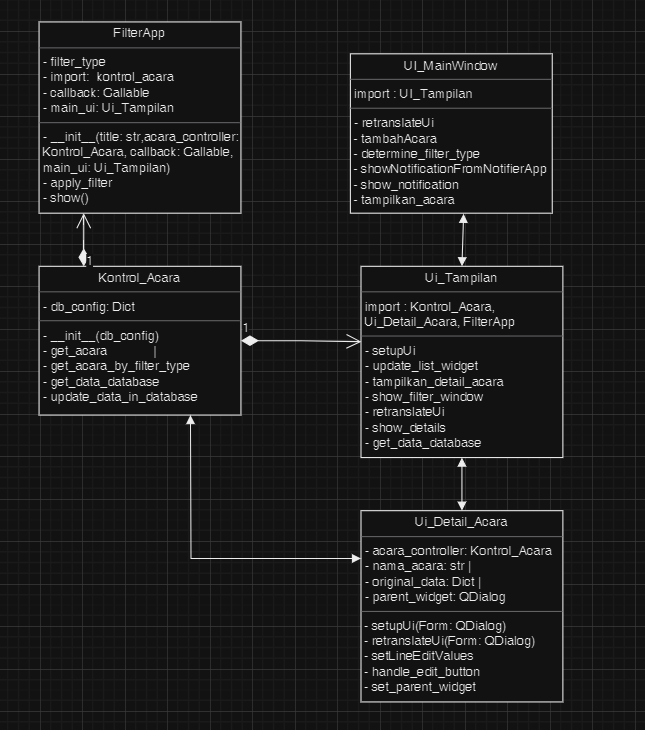
2.1 Flowchart Sistem

A diagram of a company

Description automatically generated

1. **Main Program:**
   * **Start:**
     + Program dimulai dari file **main\_program.py**.
     + File ini akan membuat instance dari kelas **Ui\_Tampilan** dan menampilkan antarmuka pengguna utama.
     + Menginisialisasi instance **Kontrol\_Acara** dan **FilterApp**.
     + **Event Loop:**
       - Program terus berjalan dan menanggapi interaksi pengguna.
       - Menampilkan daftar acara menggunakan antarmuka pengguna utama.
       - Pengguna dapat mengklik tombol filter untuk memfilter acara.
       - Pengguna juga dapat mengklik acara untuk melihat detailnya.
2. **Tampilan Utama (Ui\_Tampilan):**
   * **Inisialisasi:**
     + Membuat instance dari **Ui\_Tampilan**.
     + Menginisialisasi elemen antarmuka pengguna.
     + Membuat instance dari **Kontrol\_Acara** dan **FilterApp**.
   * **Menampilkan Daftar Acara:**
     + Menggunakan **Kontrol\_Acara** untuk mendapatkan data acara.
     + Menampilkan data acara dalam QListWidget.
   * **Menanggapi Klik Filter:**
     + Memanggil fungsi **show\_filter\_window** dari **FilterApp**.
   * **Menanggapi Klik Acara:**
     + Memanggil fungsi **show\_details** saat pengguna mengklik acara.
     + Menampilkan detail acara menggunakan **Ui\_Detail\_Acara**.
3. **Kontrol Acara (Kontrol\_Acara):**
   * **Inisialisasi:**
     + Menerima konfigurasi database saat dibuat.
     + Membuat koneksi ke database.
   * **Operasi Database:**
     + Menyediakan fungsi untuk melakukan operasi pada tabel acara.
     + Menyediakan metode untuk mendapatkan acara berdasarkan filter tertentu.
     + Menyediakan metode untuk mendapatkan data acara dari database.
     + Menyediakan metode untuk mengupdate dan menghapus data acara.
4. **FilterApp (Filter.py):**
   * **Inisialisasi:**
     + Menerima judul, kontrol acara, fungsi pembaruan, dan antarmuka pengguna utama.
     + Membuat elemen-elemen antarmuka pengguna untuk filter.
   * **Filter Acara:**
     + Memanggil metode **get\_acara\_by\_filter\_type** dari **Kontrol\_Acara**.
     + Mengupdate antarmuka pengguna utama menggunakan fungsi pembaruan.
5. **Detail Acara (Ui\_Detail\_Acara):**
   * **Inisialisasi:**
     + Menerima kontrol acara dan nama acara saat dibuat.
     + Menginisialisasi elemen-elemen antarmuka pengguna untuk detail acara.
   * **Menampilkan Detail Acara:**
     + Memanggil fungsi **setLineEditValues** untuk mengisi elemen antarmuka pengguna dengan data acara.
     + Menerima input pengguna untuk mengedit atau menghapus acara.
     + Menangani tombol edit dan hapus untuk memperbarui atau menghapus acara dari database.

2.2 Class Diagram



1. **Main Program (jadwal.py):**
   * Classes:
     + UI\_MainWindow
   * Associations:
     + **UI\_Tampilan** (Association)
2. **Ui\_Tampilan (Tampilan.py):**
   * Classes:
     + **Ui\_Tampilan**
   * Associations:
     + **Kontrol\_Acara** (Composition)
     + **FilterApp** (Composition)
     + **Ui\_Detail\_Acara** (Association)
3. **Kontrol Acara (kontrol\_acara.py):**
   * Classes:
     + **Kontrol\_Acara**
   * Associations:
     + Tidak ada dalam file ini.
4. **FilterApp (filter.py):**
   * Classes:
     + **FilterApp**
   * Associations:
     + **Kontrol\_Acara** (Composition)
     + **Ui\_Tampilan** (Association)
5. **Ui\_Detail\_Acara (detail\_acara.py):**
   * Classes:
     + **Ui\_Detail\_Acara**
   * Associations:
     + **Kontrol\_Acara** (Association)

2.3 Source code

source code disini kita bagi menjadi 5 fileyang berbeda: yaitu jadwal.py, kontrol\_acara.py, detail\_acara.py, tampilan.py, filter.py. berikut adalah full dari kode nya:

A). Jadwal.py

**from** PyQt5**.**QtWidgets **import** QApplication**,** QMainWindow**,** QWidget**,** QLabel**,** QLineEdit**,** QPushButton**,** QVBoxLayout**,** QDateTimeEdit

**from** PyQt5**.**QtGui **import** QPixmap

**from** PyQt5**.**QtCore **import** QTimer

**from** plyer **import** notification

**from** datetime **import** datetime**,** timedelta

**from** PyQt5 **import** QtCore**,** QtGui**,** QtWidgets

**from** PyQt5**.**QtWidgets **import** QApplication**,** QMessageBox

**from** Tampilan **import** Ui\_Tampilan

**import** calendar

**import** mysql**.**connector

**import** sys

class **Ui\_MainWindow**(*object*)**:**

**def** setupUi(*self***,** **MainWindow**)**:**

        MainWindow**.**setObjectName("MainWindow")

        MainWindow**.**resize(754**,** 602)

*self***.**centralwidget **=** QtWidgets**.**QWidget(MainWindow)

*self***.**centralwidget**.**setObjectName("centralwidget")

*self***.**label **=** QtWidgets**.**QLabel(*self***.**centralwidget)

*self***.**label**.**setGeometry(QtCore**.**QRect(40**,** 30**,** 331**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label**.**setFont(font)

*self***.**label**.**setObjectName("label")

*self***.**lineEdit **=** QtWidgets**.**QLineEdit(*self***.**centralwidget)

*self***.**lineEdit**.**setGeometry(QtCore**.**QRect(390**,** 30**,** 321**,** 41))

*self***.**lineEdit**.**setObjectName("lineEdit")

*self***.**label\_2 **=** QtWidgets**.**QLabel(*self***.**centralwidget)

*self***.**label\_2**.**setGeometry(QtCore**.**QRect(40**,** 100**,** 201**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_2**.**setFont(font)

*self***.**label\_2**.**setObjectName("label\_2")

*self***.**label\_3 **=** QtWidgets**.**QLabel(*self***.**centralwidget)

*self***.**label\_3**.**setGeometry(QtCore**.**QRect(40**,** 170**,** 201**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_3**.**setFont(font)

*self***.**label\_3**.**setObjectName("label\_3")

*self***.**dateEdit **=** QtWidgets**.**QDateEdit(*self***.**centralwidget)

*self***.**dateEdit**.**setGeometry(QtCore**.**QRect(390**,** 100**,** 161**,** 41))

*self***.**dateEdit**.**setObjectName("dateEdit")

*self***.**timeEdit **=** QtWidgets**.**QTimeEdit(*self***.**centralwidget)

*self***.**timeEdit**.**setGeometry(QtCore**.**QRect(390**,** 170**,** 161**,** 41))

*self***.**timeEdit**.**setObjectName("timeEdit")

*self***.**label\_4 **=** QtWidgets**.**QLabel(*self***.**centralwidget)

*self***.**label\_4**.**setGeometry(QtCore**.**QRect(140**,** 220**,** 161**,** 31))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(24)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_4**.**setFont(font)

*self***.**label\_4**.**setText("")

*self***.**label\_4**.**setObjectName("label\_4")

*self***.**label\_6 **=** QtWidgets**.**QLabel(*self***.**centralwidget)

*self***.**label\_6**.**setGeometry(QtCore**.**QRect(40**,** 310**,** 331**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_6**.**setFont(font)

*self***.**label\_6**.**setObjectName("label\_6")

*self***.**textEdit **=** QtWidgets**.**QTextEdit(*self***.**centralwidget)

*self***.**textEdit**.**setGeometry(QtCore**.**QRect(390**,** 310**,** 321**,** 141))

*self***.**textEdit**.**setUndoRedoEnabled(True)

*self***.**textEdit**.**setObjectName("textEdit")

*self***.**label\_5 **=** QtWidgets**.**QLabel(*self***.**centralwidget)

*self***.**label\_5**.**setGeometry(QtCore**.**QRect(40**,** 240**,** 201**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_5**.**setFont(font)

*self***.**label\_5**.**setObjectName("label\_5")

*self***.**pushButton **=** QtWidgets**.**QPushButton(*self***.**centralwidget)

*self***.**pushButton**.**setGeometry(QtCore**.**QRect(390**,** 240**,** 81**,** 41))

*self***.**pushButton**.**setObjectName("pushButton")

*self***.**pushButton\_2 **=** QtWidgets**.**QPushButton(*self***.**centralwidget)

*self***.**pushButton\_2**.**setGeometry(QtCore**.**QRect(500**,** 240**,** 81**,** 41))

*self***.**pushButton\_2**.**setObjectName("pushButton\_2")

*self***.**pushButton\_4 **=** QtWidgets**.**QPushButton(*self***.**centralwidget)

*self***.**pushButton\_4**.**setGeometry(QtCore**.**QRect(170**,** 480**,** 151**,** 61))

*self***.**pushButton\_4**.**setObjectName("pushButton\_4")

*self***.**pushButton\_6 **=** QtWidgets**.**QPushButton(*self***.**centralwidget)

*self***.**pushButton\_6**.**setGeometry(QtCore**.**QRect(430**,** 480**,** 151**,** 61))

*self***.**pushButton\_6**.**setObjectName("pushButton\_6")

*self***.**pushButton\_4**.**clicked**.**connect(*self***.**tambahAcara)

*self***.**pushButton\_6**.**clicked**.**connect(*self***.**tampilkan\_acara)

        MainWindow**.**setCentralWidget(*self***.**centralwidget)

*self***.**menubar **=** QtWidgets**.**QMenuBar(MainWindow)

*self***.**menubar**.**setGeometry(QtCore**.**QRect(0**,** 0**,** 754**,** 18))

*self***.**menubar**.**setObjectName("menubar")

        MainWindow**.**setMenuBar(*self***.**menubar)

*self***.**statusbar **=** QtWidgets**.**QStatusBar(MainWindow)

*self***.**statusbar**.**setObjectName("statusbar")

        MainWindow**.**setStatusBar(*self***.**statusbar)

*self***.**notification\_timer **=** QTimer(*self***.**centralwidget)

*self***.**notification\_timer**.**timeout**.**connect(*self***.**show\_notification)

*self***.**retranslateUi(MainWindow)

        QtCore**.**QMetaObject**.**connectSlotsByName(MainWindow)

**def** retranslateUi(*self***,** **MainWindow**)**:**

        \_translate **=** QtCore**.**QCoreApplication**.**translate

        MainWindow**.**setWindowTitle(\_translate(

            "MainWindow"**,** "Penjadwalan Acara"))

*self***.**label**.**setText(\_translate("MainWindow"**,** "Nama Acara:"))

*self***.**label\_2**.**setText(\_translate("MainWindow"**,** "Tanggal:"))

*self***.**label\_3**.**setText(\_translate("MainWindow"**,** "Waktu:"))

*self***.**label\_6**.**setText(\_translate("MainWindow"**,** "Deskripsi Acara:"))

*self***.**label\_5**.**setText(\_translate("MainWindow"**,** "Pengingat:"))

*self***.**pushButton**.**setText(\_translate("MainWindow"**,** "Ya"))

*self***.**pushButton\_2**.**setText(\_translate("MainWindow"**,** "Tidak"))

*self***.**pushButton\_4**.**setText(\_translate("MainWindow"**,** "Tambah Acara"))

*self***.**pushButton\_6**.**setText(\_translate("MainWindow"**,** "Tampilkan Acara"))

**def** tambahAcara(*self*)**:**

*# Ambil data dari UI*

        event\_name **=** *self***.**lineEdit**.**text()

        event\_date **=** *self***.**dateEdit**.**date()**.**toString("yyyy-MM-dd")

        event\_time **=** *self***.**timeEdit**.**time()**.**toString("hh:mm:ss")

        event\_description **=** *self***.**textEdit**.**toPlainText()

*# Validasi data*

**if** **not** event\_date **or** **not** event\_time**:**

            QMessageBox**.**critical(*self***.**centralwidget**,** "Error"**,** "Tanggal dan Waktu harus diisi.")

**return**

**try:**

*# Konfigurasi koneksi ke database MySQL*

            db\_config **=** {

                'host'**:** 'localhost'**,**

                'user'**:** 'root'**,**

                'password'**:** 'valhalla123'**,**

                'database'**:** 'jadwal\_acara'**,**

            }

*# Membuat koneksi*

            connection **=** mysql**.**connector**.**connect(**\*\***db\_config)

*# Membuat kursor*

            cursor **=** connection**.**cursor()

*# Tentukan Filter Type berdasarkan tanggal acara*

            filter\_type **=** *self***.**determine\_filter\_type(event\_date)

*# Query untuk menyimpan data*

            query **=** "INSERT INTO acara (Nama\_Acara, Tanggal, Waktu, Deskripsi\_Acara, Filter\_Type) VALUES (%s, %s, %s, %s, %s)"

            values **=** (event\_name**,** event\_date**,** event\_time**,** event\_description**,** filter\_type)

*# Eksekusi query*

            cursor**.**execute(query**,** values)

*# Commit perubahan ke database*

            connection**.**commit()

*# Menutup kursor dan koneksi*

            cursor**.**close()

            connection**.**close()

*# Mengatur nilai default setelah acara ditambahkan*

*self***.**lineEdit**.**clear()

*self***.**dateEdit**.**setDate(QtCore**.**QDate**.**currentDate())

*self***.**timeEdit**.**setTime(QtCore**.**QTime**.**currentTime())

*self***.**textEdit**.**clear()

*# Memberikan notifikasi bahwa acara berhasil ditambahkan*

            QMessageBox**.**information(

*self***.**centralwidget**,** "Sukses"**,** "Acara berhasil ditambahkan ke database.")

**except** Exception **as** e**:**

*# Menampilkan pesan error jika terjadi masalah*

            QMessageBox**.**critical(*self***.**centralwidget**,** "Error"**,** f"Terjadi kesalahan: {str(e)}")

**def** determine\_filter\_type(*self***,** **event\_date**)**:**

*# Tentukan Filter Type berdasarkan tanggal acara*

        selected\_date **=** datetime**.**strptime(event\_date**,** "%Y-%m-%d")**.**date()

        today **=** datetime**.**now()**.**date()

        \_**,** last\_day\_of\_month **=** calendar**.**monthrange(today**.**year**,** today**.**month)

        \_**,** last\_day\_of\_selected\_month **=** calendar**.**monthrange(selected\_date**.**year**,** selected\_date**.**month)

        end\_of\_week **=** today **+** timedelta(**days=**(calendar**.**SATURDAY **-** today**.**weekday()))

        end\_of\_month **=** datetime(today**.**year**,** today**.**month**,** last\_day\_of\_month)**.**date()

**if** selected\_date **==** today**:**

**return** "Harian"

**elif** today **<=** selected\_date **<=** end\_of\_week**:**

**return** "Mingguan"

**elif** today **<=** selected\_date **<=** end\_of\_month**:**

**return** "Bulanan"

**else:**

**return** "Semua"

**def** showNotificationFromNotifierApp(*self*)**:**

        print("Entering showNotificationFromNotifierApp")

*# Extract relevant data from Ui\_MainWindow*

        event\_name **=** *self***.**lineEdit**.**text()

        event\_date **=** *self***.**dateEdit**.**date()**.**toString("yyyy-MM-dd")

        event\_time **=** *self***.**timeEdit**.**time()**.**toString("hh:mm:ss")

        event\_description **=** *self***.**textEdit**.**toPlainText()

*# Validasi: Pastikan pengingat dijadwalkan dengan acara*

**if** **not** (event\_date **and** event\_time)**:**

            print("Pengingat dijadwalkan tanpa acara")

*self***.**notification\_timer**.**start(0)  *# Menjalankan timer tanpa waktu tunggu*

**return**

*# Validasi: Pastikan nama acara diisi*

**if** **not** event\_name**:**

            print("Nama Acara harus diisi")

            QMessageBox**.**critical(*self***.**centralwidget**,** "Error"**,** "Nama Acara harus diisi.")

**return**

*# Validasi: Jangan tampilkan notifikasi jika waktu yang dipilih sudah berlalu*

        selected\_datetime\_str **=** f"{event\_date} {event\_time}"

        selected\_datetime **=** datetime**.**strptime(selected\_datetime\_str**,** "%Y-%m-%d %H:%M:%S")

        current\_datetime **=** datetime**.**now()

**if** selected\_datetime **<=** current\_datetime**:**

            print("Waktu yang dipilih sudah berlalu")

            QMessageBox**.**critical(*self***.**centralwidget**,** "Error"**,** "Waktu yang dipilih sudah berlalu.")

**return**

*# Validasi: Pastikan deskripsi acara diisi*

**if** **not** event\_description**:**

            print("Deskripsi Acara harus diisi")

            QMessageBox**.**critical(*self***.**centralwidget**,** "Error"**,** "Deskripsi Acara harus diisi.")

**return**

*# Calculate time difference in seconds*

        time\_difference **=** (selected\_datetime **-** current\_datetime)**.**total\_seconds()

*# Schedule the notification using a QTimer*

*self***.**notification\_timer**.**start(int(time\_difference **\*** 1000))

        print("Notification scheduled")

**def** show\_notification(*self*)**:**

*# Stop the timer to avoid repeated notifications*

*self***.**notification\_timer**.**stop()

*# Extract relevant data from Ui\_MainWindow*

        event\_name **=** *self***.**lineEdit**.**text()

        event\_description **=** *self***.**textEdit**.**toPlainText()

*# Set data for notification*

        get\_title **=** event\_name

        get\_msg **=** event\_description

*# Use plyer's notification with a longer timeout*

**try:**

            notification**.**notify(

**title=**get\_title**,**

**message=**get\_msg**,**

**app\_name=**"Notifier"**,**

**timeout=**20

            )

            print("Notification shown")

**except** Exception **as** e**:**

            print(f"Error showing notification: {e}")

**def** tampilkan\_acara(*self*)**:**

*# Membuat instance dari QWidget untuk menampilkan antarmuka pengguna*

*self***.**form\_widget **=** QtWidgets**.**QWidget()

*# Membuat instance dari Ui\_Form dan menampilkan form*

        ui\_form **=** Ui\_Tampilan()

        ui\_form**.**setupUi(*self***.**form\_widget)

*# Menyimpan rujukan ke Ui\_Form*

*self***.**form\_ui **=** ui\_form

*# Mengambil data dari database*

**try:**

*# Konfigurasi koneksi ke database MySQL*

            db\_config **=** {

                'host'**:** 'localhost'**,**

                'user'**:** 'root'**,**

                'password'**:** 'valhalla123'**,**

                'database'**:** 'jadwal\_acara'**,**

            }

*# Membuat koneksi*

            connection **=** mysql**.**connector**.**connect(**\*\***db\_config)

*# Membuat kursor*

            cursor **=** connection**.**cursor()

*# Query untuk mengambil semua data acara*

            query **=** "SELECT Nama\_Acara FROM acara"

            cursor**.**execute(query)

*# Mendapatkan semua data*

            all\_events **=** cursor**.**fetchall()

*# Menutup kursor dan koneksi*

            cursor**.**close()

            connection**.**close()

*# Menampilkan nama acara pada QListWidget*

**for** event **in** all\_events**:**

                ui\_form**.**listWidget**.**addItem(event[0])

*# Tampilkan form*

*self***.**form\_widget**.**show()

**except** Exception **as** e**:**

*# Menampilkan pesan error jika terjadi masalah*

            QMessageBox**.**critical(*self***.**centralwidget**,** "Error"**,**f"Terjadi kesalahan: {str(e)}")

**if** \_\_name\_\_ **==** "\_\_main\_\_"**:**

    app **=** QApplication([])

    MainWindow **=** QMainWindow()

    ui\_main **=** Ui\_MainWindow()

    ui\_main**.**setupUi(MainWindow)

*# Hanya membuat koneksi di \_\_main\_\_*

    ui\_main**.**pushButton**.**clicked**.**connect(ui\_main**.**showNotificationFromNotifierApp)

    ui\_main**.**pushButton\_6**.**clicked**.**connect(ui\_main**.**tampilkan\_acara)

    MainWindow**.**show()

    sys**.**exit(app**.**exec())

B). Tampilan.py

**from** PyQt5 **import** QtCore**,** QtGui**,** QtWidgets

**from** PyQt5**.**QtCore **import** Qt

**from** PyQt5**.**QtWidgets **import** QMessageBox

**from** datetime **import** datetime**,** timedelta**,** date

**from** Detail\_Acara **import** Ui\_Detail\_Acara

**from** Kontrol\_Acara **import** Kontrol\_Acara

**from** Filter **import** FilterApp

**import** mysql**.**connector

class **Ui\_Tampilan**(*object*)**:**

**def** setupUi(*self***,** **Form**)**:**

        Form**.**setObjectName("Form")

        Form**.**resize(481**,** 358)

*self***.**label **=** QtWidgets**.**QLabel(Form)

*self***.**label**.**setGeometry(QtCore**.**QRect(50**,** 20**,** 301**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(18)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label**.**setFont(font)

*self***.**label**.**setObjectName("label")

*self***.**listWidget **=** QtWidgets**.**QListWidget(Form)

*self***.**listWidget**.**setGeometry(QtCore**.**QRect(50**,** 80**,** 381**,** 251))

*self***.**listWidget**.**setObjectName("listWidget")

*self***.**pushButton **=** QtWidgets**.**QPushButton(Form)

*self***.**pushButton**.**setGeometry(QtCore**.**QRect(340**,** 20**,** 91**,** 41))

*self***.**pushButton**.**setObjectName("pushButton")

*self***.**pushButton**.**clicked**.**connect(*self***.**show\_filter\_window)

*# Menghubungkan sinyal doubleClicked ke fungsi show\_details*

*self***.**listWidget**.**doubleClicked**.**connect(*self***.**show\_details)

*self***.**retranslateUi(Form)

*# Buat instance AcaraController dengan konfigurasi database yang sesuai*

*self***.**acara\_controller **=** Kontrol\_Acara({

            'host'**:** 'localhost'**,**

            'user'**:** 'root'**,**

            'password'**:** 'valhalla123'**,**

            'database'**:** 'jadwal\_acara'**,**

        })

*# Buat instance FilterApp dengan kontrol\_acara dan callback yang sesuai*

*self***.**filter\_app **=** FilterApp("Filter Acara"**,** *self***.**acara\_controller**,** *self***.**update\_list\_widget**,** *self*)

*self***.**filter\_app**.**main\_ui **=** *self*  *# Atur atribut main\_ui pada objek FilterApp*

**def** filter\_acara(*self*)**:**

*# Mendapatkan tipe filter dari ComboBox*

        filter\_type **=** *self***.**comboBox**.**currentText()

*# Set filter type di objek FilterApp*

*self***.**filter\_app**.**filter\_type **=** filter\_type

*# Panggil metode apply\_filter dari objek FilterApp*

*self***.**filter\_app**.**apply\_filter()

**def** update\_list\_widget(*self***,** **data**)**:**

*# Hapus semua item di listWidget sebelum memperbarui*

*self***.**listWidget**.**clear()

*# Check if filter\_type is set*

**if** hasattr(*self***.**filter\_app**,** 'filter\_type') **and** *self***.**filter\_app**.**filter\_type **is** **not** None**:**

*# Filter data based on filter type*

            filtered\_data **=** [item **for** item **in** data **if** item[1] **==** *self***.**filter\_app**.**filter\_type]

**for** item **in** filtered\_data**:**

*self***.**listWidget**.**addItem(str(item[1]))

**else:**

*# Display all data if no filter type is selected*

**for** item **in** data**:**

*self***.**listWidget**.**addItem(str(item[1]))

**def** tampilkan\_detail\_acara(*self***,** **original\_data**)**:**

        detail\_acara\_dialog **=** QtWidgets**.**QDialog()

        detail\_acara\_ui **=** Ui\_Detail\_Acara(*self***.**acara\_controller**,** original\_data)

        detail\_acara\_ui**.**setupUi(detail\_acara\_dialog)

        result **=** detail\_acara\_dialog**.**exec\_()

**def** show\_filter\_window(*self*)**:**

*# Panggil metode show\_filter\_window dari objek FilterApp*

*self***.**filter\_app**.**show()

**def** retranslateUi(*self***,** **Form**)**:**

        \_translate **=** QtCore**.**QCoreApplication**.**translate

        Form**.**setWindowTitle(\_translate("Form"**,** "Form"))

*self***.**label**.**setText(\_translate("Form"**,** "Daftar Acara:"))

*self***.**pushButton**.**setText(\_translate("Form"**,** "Filter"))

**def** show\_details(*self*)**:**

*# Mendapatkan item yang dipilih dari QListWidget*

        selected\_item **=** *self***.**listWidget**.**currentItem()

*# Pernyataan print tambahan*

        print("Nama Acara yang Dipilih:"**,** selected\_item**.**text())

*# Memastikan ada item yang dipilih sebelum melanjutkan*

**if** selected\_item**:**

*# Menyimpan data awal sebelum diedit*

            data\_item **=** *self***.**acara\_controller**.**get\_data\_database(selected\_item**.**text())

*# Pernyataan print tambahan*

            print("Data Item dari Database:"**,** data\_item)

*# Memastikan data\_item tidak None dan memiliki panjang yang cukup*

**if** data\_item **is** **not** None **and** len(data\_item) **>** 0**:**

                data\_item **=** data\_item[0]  *# Ambil elemen pertama dari tuple*

*self***.**original\_data **=** {

                    "Nama Acara"**:** data\_item[1]**,**

                    "Tanggal"**:** QtCore**.**QDate**.**fromString(str(data\_item[2])**,** "yyyy-MM-dd")**,**

                    "Waktu"**:** QtCore**.**QTime**.**fromString(str(data\_item[3])**,** "hh:mm:ss")**,**

                    "Deskripsi Acara"**:** data\_item[4]**,**

                }

*# Membuat instance dari dialog Detail\_Acara*

                detail\_acara\_dialog **=** QtWidgets**.**QDialog()

                detail\_acara\_ui **=** Ui\_Detail\_Acara()

*# Creating an instance of Ui\_Detail\_Acara and initializing it*

                detail\_acara\_ui**.**initialize(*self***.**acara\_controller**,** selected\_item**.**text())

*# Setting up the UI for the detail view*

                detail\_acara\_ui**.**setupUi(detail\_acara\_dialog)

*# Menampilkan data pada UI Detail\_Acara*

                detail\_acara\_ui**.**setLineEditValues(*self***.**original\_data)

*# Menampilkan dialog Detail\_Acara*

**if** detail\_acara\_dialog**.**exec\_() **==** QtWidgets**.**QDialog**.**Accepted**:**

*# Update QListWidget*

*self***.**update\_list\_widget(*self***.**acara\_controller**.**get\_acara\_by\_filter\_type(*self***.**filter\_app**.**filter\_type))

**else:**

*# Menampilkan pesan kesalahan jika data\_item tidak lengkap atau tidak tersedia*

                error\_message **=** f"Data untuk item dengan nama acara '{selected\_item**.**text()}' tidak tersedia atau tidak lengkap."

                QtWidgets**.**QMessageBox**.**critical(*self***.**listWidget**,** "Error"**,** error\_message)

                print(error\_message)  *# Tambahkan ini untuk mencetak informasi tambahan ke terminal*

**else:**

*# Menampilkan pesan kesalahan jika tidak ada item yang dipilih*

            QtWidgets**.**QMessageBox**.**critical(*self***.**listWidget**,** "Error"**,** "Tidak ada item yang dipilih.")

**def** get\_data\_database(*self***,** **nama\_acara**)**:**

*# Assuming 'acara\_controller' is an instance of Kontrol\_Acara*

**return** *self***.**acara\_controller**.**get\_data\_database(nama\_acara)

**def** tampilkan\_detail\_acara(*self***,** **original\_data**)**:**

        detail\_acara\_dialog **=** QtWidgets**.**QDialog()

        detail\_acara\_ui **=** Ui\_Detail\_Acara()

        detail\_acara\_ui**.**initialize(*self***.**acara\_controller**,** original\_data)

        detail\_acara\_ui**.**setupUi(detail\_acara\_dialog)

        result **=** detail\_acara\_dialog**.**exec\_()

C). Kontrol\_acara.py

**import** mysql**.**connector

class **Kontrol\_Acara:**

**def** \_\_init\_\_(*self***,** **db\_config**)**:**

*self***.**db\_config **=** db\_config

*self***.**connection **=** None

**def** connect\_to\_database(*self*)**:**

**try:**

*self***.**connection **=** mysql**.**connector**.**connect(**\*\****self***.**db\_config)

**except** mysql**.**connector**.**Error **as** err**:**

            print(f"Error connecting to the database: {err}")

**def** close\_database\_connection(*self*)**:**

**if** *self***.**connection**:**

*self***.**connection**.**close()

**def** execute\_query(*self***,** **query,** **params=**None)**:**

**try:**

**if** **not** *self***.**connection **or** **not** *self***.**connection**.**is\_connected()**:**

*self***.**connect\_to\_database()

**with** *self***.**connection**.**cursor() **as** cursor**:**

**if** params **is** **not** None**:**

                    cursor**.**execute(query**,** params)

**else:**

                    cursor**.**execute(query)

                data **=** cursor**.**fetchall()

**if** **not** data**:**

                    print("No data matching the filter criteria.")

**return** data

**except** mysql**.**connector**.**Error **as** err**:**

            print(f"MySQL error: {err}")

            print(f"Failed query: {query}")

**return** None

**except** Exception **as** e**:**

            print(f"Error: {str(e)}")

**return** None

**finally:**

*self***.**close\_database\_connection()

**def** get\_acara\_by\_filter\_type(*self***,** **filter\_type**)**:**

**if** filter\_type **==** "Harian"**:**

            query **=** "SELECT \* FROM acara WHERE Tanggal = CURDATE()"

**elif** filter\_type **==** "Mingguan"**:**

            query **=** "SELECT \* FROM acara WHERE Tanggal BETWEEN CURDATE() AND CURDATE() + INTERVAL 6 DAY"

**elif** filter\_type **==** "Bulanan"**:**

            query **=** "SELECT \* FROM acara WHERE Tanggal BETWEEN CURDATE() AND CURDATE() + INTERVAL 29 DAY"

**else:**

            query **=** "SELECT \* FROM acara"

**return** *self***.**execute\_query(query)

**def** get\_data\_database(*self***,** **nama\_acara**)**:**

        query **=** "SELECT \* FROM acara WHERE Nama\_Acara = %s"

**return** *self***.**execute\_query(query**,** (str(nama\_acara)**,**))

**def** update\_data\_in\_database(*self***,** **nama\_acara,** **updated\_data**)**:**

        query **=** "UPDATE acara SET Nama\_Acara = %s, Tanggal = %s, Waktu = %s, Deskripsi\_Acara = %s WHERE Nama\_Acara = %s"

*# Ensure that the values are properly formatted for SQL*

        updated\_values **=** (

            updated\_data**.**get("Nama\_Acara"**,** "")**,**

            updated\_data**.**get("Tanggal"**,** "")**,**

            updated\_data**.**get("Waktu"**,** "")**,**

            updated\_data**.**get("Deskripsi\_Acara"**,** "")

        )

*# Print the data before the update*

        select\_query **=** "SELECT \* FROM acara WHERE Nama\_Acara = %s"

        selected\_data **=** *self***.**get\_data\_from\_database(select\_query**,** (str(nama\_acara)**,**))

        print(f"Selected data before update: {selected\_data}")

**try:**

**if** **not** *self***.**connection **or** **not** *self***.**connection**.**is\_connected()**:**

*self***.**connect\_to\_database()

**with** *self***.**connection**.**cursor() **as** cursor**:**

                print(f"Executing query: {query} with values: {updated\_values!r}")

                cursor**.**execute(query**,** updated\_values **+** (str(nama\_acara)**,**))  *# Concatenate the nama\_acara to the tuple*

*self***.**connection**.**commit()

            print("Data updated successfully.")

**except** mysql**.**connector**.**Error **as** err**:**

            print(f"MySQL error: {err}")

            print(f"Failed query: {query} with values: {updated\_values!r}")

**raise**  *# Reraise the exception to see the full traceback*

**finally:**

*# Always close the connection in the finally block*

*self***.**close\_database\_connection()

**def** delete\_data\_in\_database(*self***,** **nama\_acara**)**:**

        query **=** "DELETE FROM acara WHERE Nama\_Acara = %s"

**try:**

**if** **not** *self***.**connection **or** **not** *self***.**connection**.**is\_connected()**:**

*self***.**connect\_to\_database()

**with** *self***.**connection**.**cursor() **as** cursor**:**

                cursor**.**execute(query**,** (str(nama\_acara)**,**))

*self***.**connection**.**commit()

            print("Data deleted successfully.")

**except** mysql**.**connector**.**Error **as** err**:**

            print(f"MySQL error: {err}")

            print(f"Failed query: {query} with value: {nama\_acara!r}")

**raise**  *# Reraise the exception to see the full traceback*

**finally:**

*self***.**close\_database\_connection()

**def** get\_data\_for\_detail\_acara(*self***,** **event\_name**)**:**

*# Fungsi khusus untuk mendapatkan data untuk Detail Acara*

        query **=** "SELECT \* FROM acara WHERE Nama\_Acara = %s"

**return** *self***.**execute\_query(query**,** (event\_name**,**))

**def** get\_data\_from\_database(*self***,** **query,** **query\_params=**None)**:**

**try:**

**if** **not** *self***.**connection **or** **not** *self***.**connection**.**is\_connected()**:**

*self***.**connect\_to\_database()

**with** *self***.**connection**.**cursor() **as** cursor**:**

                print(f"Executing query: {query} with params: {query\_params}")

**if** query\_params **is** **not** None**:**

                    cursor**.**execute(query**,** query\_params)

**else:**

                    cursor**.**execute(query)

                data **=** cursor**.**fetchall()

**if** **not** data**:**

                    print("No data matching the filter criteria.")

**return** data

**except** mysql**.**connector**.**Error **as** err**:**

            print(f"MySQL error: {err}")

            print(f"Failed query: {query}")

**return** None

**except** Exception **as** e**:**

            print(f"Error: {str(e)}")

**return** None

**finally:**

*self***.**close\_database\_connection()

D). Filter.py

**from** PyQt5**.**QtWidgets **import** QDialog**,** QLabel**,** QPushButton**,** QVBoxLayout**,** QFormLayout**,** QComboBox

**from** Kontrol\_Acara **import** Kontrol\_Acara

**import** mysql**.**connector

class **FilterApp**(*QDialog*)**:**

**def** \_\_init\_\_(*self***,** **title,** **kontrol\_acara,** **update\_list\_widget\_callback,** **main\_ui**)**:**

        super()**.**\_\_init\_\_()

*self***.**setWindowTitle(title)

*self***.**kontrol\_acara **=** kontrol\_acara

*self***.**update\_list\_widget\_callback **=** update\_list\_widget\_callback

*self***.**filter\_type **=** None  *# Add a new attribute for filter type*

*self***.**main\_ui **=** main\_ui  *# Set the main\_ui attribute*

*self***.**create\_widgets()

**def** create\_widgets(*self*)**:**

        layout **=** QFormLayout()

*# Combo Box for filter types*

*self***.**filter\_type\_combo **=** QComboBox()

*self***.**filter\_type\_combo**.**addItems(["Harian"**,** "Mingguan"**,** "Bulanan"])

        layout**.**addRow(QLabel("Filter Type:")**,** *self***.**filter\_type\_combo)

*# Filter Button*

        filter\_button **=** QPushButton("Filter")

        filter\_button**.**clicked**.**connect(*self***.**apply\_filter)

        layout**.**addRow(filter\_button)

*# Status Label*

*self***.**status\_label **=** QLabel()

        layout**.**addRow(*self***.**status\_label)

*self***.**setLayout(layout)

*# Dictionary to store filter values*

*self***.**filters **=** {"Filter Type"**:** None}

**def** apply\_filter(*self*)**:**

*# Set filter values based on user input*

        filter\_type **=** *self***.**filter\_type\_combo**.**currentText()

*self***.**filters["Filter Type"] **=** filter\_type

*self***.**main\_ui**.**filter\_type **=** filter\_type  *# Set filter type in the main UI*

*# Call the get\_acara\_by\_filter\_type function with the set filter*

        result **=** *self***.**kontrol\_acara**.**get\_acara\_by\_filter\_type(filter\_type)

*# Update the view if there are filter results*

**if** result **is** **not** None**:**

*self***.**update\_list\_widget\_callback(result)

*self***.**status\_label**.**setText(f"Filter Applied: {*self***.**filters}")

E). Detail\_acara.py

**from** PyQt5 **import** QtCore**,** QtGui**,** QtWidgets

**from** PyQt5**.**QtWidgets **import** QLineEdit**,** QDateEdit**,** QTimeEdit**,** QTextEdit

class **Ui\_Detail\_Acara**(*object*)**:**

**def** initialize(*self***,** **acara\_controller,** **nama\_acara**)**:**

*self***.**acara\_controller **=** acara\_controller

*self***.**nama\_acara **=** nama\_acara

*self***.**original\_data **=** None

**def** setupUi(*self***,** **Form**)**:**

        Form**.**setObjectName("Form")

        Form**.**resize(731**,** 531)

*self***.**label **=** QtWidgets**.**QLabel(Form)

*self***.**label**.**setGeometry(QtCore**.**QRect(30**,** 20**,** 311**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(20)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label**.**setFont(font)

*self***.**label**.**setObjectName("label")

*self***.**lineEdit\_2 **=** QtWidgets**.**QLineEdit(Form)

*self***.**lineEdit\_2**.**setGeometry(QtCore**.**QRect(380**,** 80**,** 321**,** 41))

*self***.**lineEdit\_2**.**setObjectName("lineEdit\_2")

*self***.**label\_7 **=** QtWidgets**.**QLabel(Form)

*self***.**label\_7**.**setGeometry(QtCore**.**QRect(30**,** 80**,** 331**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_7**.**setFont(font)

*self***.**label\_7**.**setObjectName("label\_7")

*self***.**dateEdit\_2 **=** QtWidgets**.**QDateEdit(Form)

*self***.**dateEdit\_2**.**setGeometry(QtCore**.**QRect(380**,** 150**,** 161**,** 41))

*self***.**dateEdit\_2**.**setObjectName("dateEdit\_2")

*self***.**label\_8 **=** QtWidgets**.**QLabel(Form)

*self***.**label\_8**.**setGeometry(QtCore**.**QRect(30**,** 150**,** 201**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_8**.**setFont(font)

*self***.**label\_8**.**setObjectName("label\_8")

*self***.**timeEdit\_2 **=** QtWidgets**.**QTimeEdit(Form)

*self***.**timeEdit\_2**.**setGeometry(QtCore**.**QRect(380**,** 220**,** 161**,** 41))

*self***.**timeEdit\_2**.**setObjectName("timeEdit\_2")

*self***.**label\_4 **=** QtWidgets**.**QLabel(Form)

*self***.**label\_4**.**setGeometry(QtCore**.**QRect(30**,** 220**,** 201**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_4**.**setFont(font)

*self***.**label\_4**.**setObjectName("label\_4")

*self***.**label\_9 **=** QtWidgets**.**QLabel(Form)

*self***.**label\_9**.**setGeometry(QtCore**.**QRect(30**,** 290**,** 331**,** 41))

        font **=** QtGui**.**QFont()

        font**.**setPointSize(16)

        font**.**setBold(True)

        font**.**setWeight(75)

*self***.**label\_9**.**setFont(font)

*self***.**label\_9**.**setObjectName("label\_9")

*self***.**textEdit\_2 **=** QtWidgets**.**QTextEdit(Form)

*self***.**textEdit\_2**.**setGeometry(QtCore**.**QRect(380**,** 290**,** 321**,** 141))

*self***.**textEdit\_2**.**setUndoRedoEnabled(True)

*self***.**textEdit\_2**.**setObjectName("textEdit\_2")

*self***.**pushButton **=** QtWidgets**.**QPushButton(Form)

*self***.**pushButton**.**setGeometry(QtCore**.**QRect(150**,** 460**,** 141**,** 51))

*self***.**pushButton**.**setObjectName("pushButton")

*self***.**pushButton\_2 **=** QtWidgets**.**QPushButton(Form)

*self***.**pushButton\_2**.**setGeometry(QtCore**.**QRect(440**,** 460**,** 141**,** 51))

*self***.**pushButton\_2**.**setObjectName("pushButton\_2")

*self***.**pushButton**.**clicked**.**connect(*self***.**handle\_edit\_button)

*self***.**pushButton\_2**.**clicked**.**connect(*self***.**handle\_delete\_button)

        print("LineEdit\_2 found:"**,** hasattr(*self***,** 'lineEdit\_2'))

        print("DateEdit\_2 found:"**,** hasattr(*self***,** 'dateEdit\_2'))

        print("TimeEdit\_2 found:"**,** hasattr(*self***,** 'timeEdit\_2'))

        print("TextEdit\_2 found:"**,** hasattr(*self***,** 'textEdit\_2'))

*self***.**retranslateUi(Form)

        QtCore**.**QMetaObject**.**connectSlotsByName(Form)

**def** initialize(*self***,** **acara\_controller,** **nama\_acara**)**:**

*self***.**acara\_controller **=** acara\_controller

*self***.**nama\_acara **=** nama\_acara

*self***.**original\_data **=** None

**def** retranslateUi(*self***,** **Form**)**:**

        \_translate **=** QtCore**.**QCoreApplication**.**translate

        Form**.**setWindowTitle(\_translate("Form"**,** "Detail Acara"))

*self***.**label**.**setText(\_translate("Form"**,** "Detail Acara:"))

*self***.**label\_7**.**setText(\_translate("Form"**,** "Nama Acara:"))

*self***.**label\_8**.**setText(\_translate("Form"**,** "Tanggal:"))

*self***.**label\_4**.**setText(\_translate("Form"**,** "Waktu:"))

*self***.**label\_9**.**setText(\_translate("Form"**,** "Deskripsi Acara:"))

*self***.**pushButton**.**setText(\_translate("Form"**,** "Edit Acara"))

*self***.**pushButton\_2**.**setText(\_translate("Form"**,** "Hapus Acara"))

*# Perubahan pada metode setLineEditValues*

**def** setLineEditValues(*self***,** **data**)**:**

        print("Data diatur ke Detail\_Acara:"**,** data)  *# Debug line*

*self***.**lineEdit\_2**.**setText(data**.**get("Nama Acara"**,** ""))

*self***.**dateEdit\_2**.**setDate(data**.**get("Tanggal"**,** QtCore**.**QDate**.**currentDate()))

*self***.**timeEdit\_2**.**setTime(data**.**get("Waktu"**,** QtCore**.**QTime**.**currentTime()))

*self***.**textEdit\_2**.**setPlainText(data**.**get("Deskripsi Acara"**,** ""))

*# Tambahkan fungsi untuk menangani tombol Edit Acara*

**def** handle\_edit\_button(*self*)**:**

*# Mengambil nilai dari elemen UI*

        updated\_data **=** {

            "Nama\_Acara"**:** *self***.**lineEdit\_2**.**text()**,**

            "Tanggal"**:** *self***.**dateEdit\_2**.**date()**.**toString("yyyy-MM-dd")**,**

            "Waktu"**:** *self***.**timeEdit\_2**.**time()**.**toString("hh:mm:ss")**,**

            "Deskripsi\_Acara"**:** *self***.**textEdit\_2**.**toPlainText()**,**

        }

*# Update data di database menggunakan AcaraController*

*self***.**acara\_controller**.**update\_data\_in\_database(*self***.**nama\_acara**,** updated\_data)

*# Tutup dialog setelah selesai mengedit (if applicable)*

**if** hasattr(*self***,** 'parent\_widget') **and** *self***.**parent\_widget**:**

*self***.**parent\_widget**.**accept()

**def** set\_parent\_widget(*self***,** **parent\_widget**)**:**

*self***.**parent\_widget **=** parent\_widget

*# New method for handling the "Hapus Acara" button click*

**def** handle\_delete\_button(*self*)**:**

*# Confirmation dialog before deleting the event*

        reply **=** QtWidgets**.**QMessageBox**.**question(

            None**,**

            'Hapus Acara'**,**

            'Apakah Anda yakin ingin menghapus acara ini?'**,**

            QtWidgets**.**QMessageBox**.**Yes **|** QtWidgets**.**QMessageBox**.**No**,**

            QtWidgets**.**QMessageBox**.**No

        )

**if** reply **==** QtWidgets**.**QMessageBox**.**Yes**:**

*# Call the delete method from the controller*

*self***.**acara\_controller**.**delete\_data\_in\_database(*self***.**nama\_acara)

*# Close the detail window or perform any other actions needed*

**if** hasattr(*self***,** 'parent\_widget') **and** *self***.**parent\_widget**:**

*self***.**parent\_widget**.**accept()

2.4 Hasil Program dan Penjelasan

**Alur Program:**

1. **Pembukaan Aplikasi:**
   * Program dimulai dengan eksekusi file **jadwal.py**.
   * File tersebut menyertakan definisi kelas dan berfungsi sebagai titik masuk utama.
2. **Inisialisasi UI dan Koneksi Database:**
   * Program membuat instance dari **Ui\_Tampilan** (kelas yang bertanggung jawab untuk tampilan utama) dan menginisialisasi antarmuka pengguna (UI) menggunakan PyQt5.
   * **Ui\_Tampilan** melakukan inisialisasi objek **Kontrol\_Acara** yang bertindak sebagai penghubung antara antarmuka pengguna dan database MySQL.
3. **Tampilan Utama dan Filter:**
   * UI menampilkan daftar acara pada **listWidget**.
   * Pengguna dapat memfilter acara berdasarkan tipe (Harian, Mingguan, Bulanan) dengan mengklik tombol "Filter".
   * Saat tombol "Filter" ditekan, muncul jendela filter (**FilterApp**) yang memungkinkan pengguna memilih tipe filter.
4. **Pembaruan Tampilan Berdasarkan Filter:**
   * Hasil filter diterapkan pada daftar acara, dan UI diperbarui untuk mencerminkan hasil tersebut.
   * **FilterApp** mengubah filter type pada instance **Ui\_Tampilan** dan memanggil **update\_list\_widget** untuk memperbarui tampilan.
5. **Menampilkan Detail Acara:**
   * Pengguna dapat melihat detail acara dengan mengklik dua kali pada salah satu acara dalam daftar.
   * **Ui\_Tampilan** menanggapi sinyal double-click dengan memanggil metode **show\_details**.
6. **Jendela Detail Acara (Ui\_Detail\_Acara):**
   * Jendela detail acara menampilkan informasi lengkap tentang acara yang dipilih.
   * Pengguna memiliki opsi untuk mengedit atau menghapus acara.
7. **Edit Acara:**
   * Pengguna dapat mengklik tombol "Edit Acara" di jendela detail acara.
   * **Ui\_Detail\_Acara** menanggapi klik tombol dan memanggil metode **handle\_edit\_button**.
8. **Memperbarui Database:**
   * Data acara yang diubah dimasukkan kembali ke database melalui **Kontrol\_Acara**.
9. **Hapus Acara:**
   * Pengguna dapat mengklik tombol "Hapus Acara" di jendela detail acara.
   * **Ui\_Detail\_Acara** menanggapi klik tombol dan memanggil metode **handle\_delete\_button**.
10. **Menghapus Data dari Database:**
    * Data acara dihapus dari database melalui **Kontrol\_Acara**.
11. **Penutupan Aplikasi:**
    * Aplikasi berjalan sampai pengguna memutuskan untuk menutupnya.

**Penjelasan:**

* Aplikasi ini menggunakan konsep MVC (Model-View-Controller), di mana **Kontrol\_Acara** berperan sebagai pengontrol logika bisnis dan interaksi dengan database, **Ui\_Tampilan** sebagai antarmuka pengguna, dan **FilterApp** sebagai bagian dari antarmuka pengguna yang mengelola filter.
* **Ui\_Detail\_Acara** mengelola tampilan dan interaksi pada jendela detail acara.
* Data diambil dari database melalui **Kontrol\_Acara** dan ditampilkan dalam UI menggunakan PyQt5.
* Pengguna dapat memanipulasi data acara melalui antarmuka pengguna, dan perubahan tersebut diterapkan kembali ke database melalui **Kontrol\_Acara**.
* Program ini memanfaatkan modularitas dengan memisahkan fungsi-fungsi ke dalam kelas-kelas terpisah untuk memudahkan pemeliharaan dan pengembangan.

BAB III

PENUTUP

3.1 Kesimpulan

Dalam pengembangan aplikasi reminder ini, kami dapat menyimpulkan beberapa hal penting. Pertama, dari latar belakang yang telah dijelaskan pada Bab I Pendahuluan, aplikasi reminder ini dibangun untuk menjawab kebutuhan akan alat pengingat yang dapat membantu pengguna dalam mengelola waktu dan tugas sehari-hari.

Pembahasan di Bab II mengenai flowchart sistem memberikan pemahaman visual terhadap langkah-langkah kerja aplikasi. Class diagram menunjukkan struktur kelas dan hubungan antar kelas yang mendasari implementasi program. Dengan adanya diagram ini, kami dapat memastikan bahwa desain aplikasi reminder ini telah dikerjakan secara sistematis dan efisien.

Hasil program yang berhasil dijelaskan dalam bagian 2.3 menunjukkan bahwa aplikasi reminder ini dapat diimplementasikan dengan baik dan sesuai dengan kebutuhan pengguna. Fitur-fitur yang dihadirkan, seperti pengaturan waktu, notifikasi, dan manajemen tugas, memberikan solusi yang praktis dan efektif dalam membantu pengguna menjalani kegiatan sehari-hari.

Melalui praktik ini, kami mendapatkan pengalaman berharga dalam pengembangan perangkat lunak, mulai dari perencanaan, desain, hingga implementasi. Meskipun terdapat beberapa tantangan, seperti pengoptimalan performa dan penanganan perubahan kebutuhan, namun kami berhasil mengatasinya dengan mengandalkan kerjasama tim dan kreativitas.

Secara keseluruhan, aplikasi reminder ini diharapkan dapat memberikan nilai tambah bagi penggunanya dan menjadi solusi yang efisien dalam manajemen waktu. Ke depannya, kami akan terus mempertimbangkan masukan dan feedback dari pengguna untuk meningkatkan kualitas dan fungsionalitas aplikasi ini.

Demikianlah kesimpulan dari laporan praktik pengembangan aplikasi reminder ini. Terima kasih dan maaf jika ada kurangnya.

DAFTAR PUSTAKA

Membuat Class diagram. (n.d). Class Diagram. <https://app.diagrams.net/>

Membuat Flowchart. (n.d). Flowchart Sistem. <https://whimsical.com/>

PyQt5 Tutorial. (n.d). tutorialspoint <https://www.tutorialspoint.com/pyqt5/index.htm>

Panduan Class Diagram. (n.d.). Class Diagram Relationships. <https://creately.com/guides/class-diagram-relationships/>